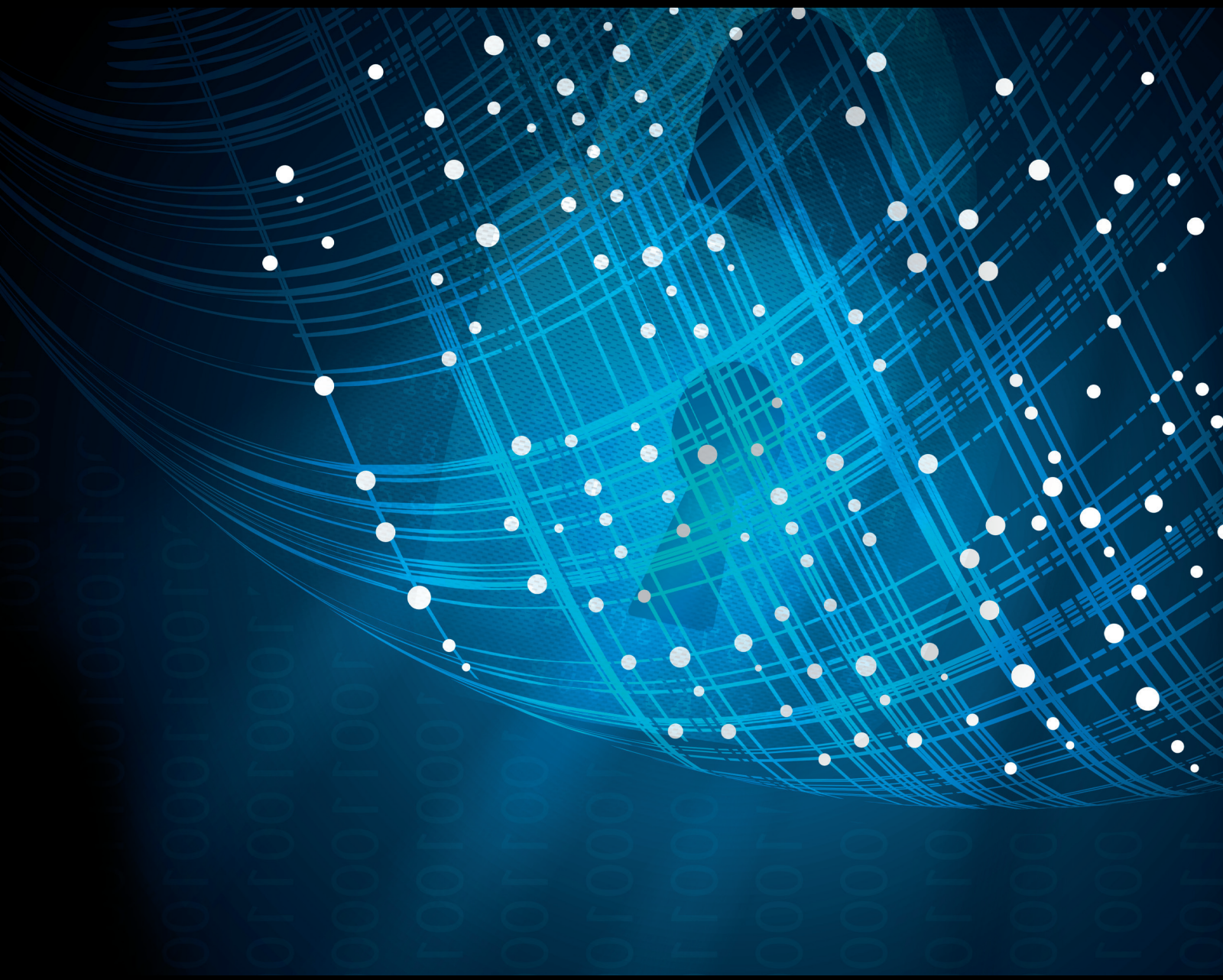
Security and Communication Networks

Theory and Engineering Practice for Security and Privacy of Edge Computing

Lead Guest Editor: Honghao Gao
Guest Editors: Zhiyuan Tan, Wenbing Zhao, and Yuyu Yin



Theory and Engineering Practice for Security and Privacy of Edge Computing

Security and Communication Networks

Theory and Engineering Practice for Security and Privacy of Edge Computing

Lead Guest Editor: Honghao Gao
Guest Editors: Zhiyuan Tan, Wenbing Zhao, and Yuyu Yin

Contents

Research on Cross-Company Defect Prediction Method to Improve Software Security
Yanli Shao, Jingru Zhao, Xingqi Wang, Weiwei Wu, and Jinglong Fang
Research Article (19 pages), Article ID 5558561, Volume 2021 (2021)

CF Model: A Coarse-to-Fine Model Based on Two-Level Local Search for Image Copy-Move Forgery Detection
Fang Mei, Tianchang Gao, and Yingda Lyu
Research Article (13 pages), Article ID 6688393, Volume 2021 (2021)

Blockchain-Enhanced Fair Task Scheduling for Cloud-Fog-Edge Coordination Environments: Model and Algorithm
Wenjuan Li, Shihua Cao, Keyong Hu, Jian Cao, and Rajkumar Buyya
Research Article (18 pages), Article ID 5563312, Volume 2021 (2021)

Research on Coordinated Scheduling Strategy of Heat Storage Thermoelectric Units Based on Wind Power Data Acquisition System Using Edge Computing
D. J. Guan, X. Y. Qian, S. X. Lu, N. An, P. Ye, M. L. Zhang, and N. Zhang
Research Article (11 pages), Article ID 8864552, Volume 2021 (2021)

An Intelligent Offloading System Based on Multiagent Reinforcement Learning
Yu Weng, Haozhen Chu, and Zhaoyi Shi
Research Article (13 pages), Article ID 8830879, Volume 2021 (2021)

Reverse Auction-Based Services Optimization in Cloud Computing Environments
Hongkun Zhang and Xinmin Liu
Research Article (10 pages), Article ID 6666628, Volume 2021 (2021)

PB: A Product-Bitmatrix Construction to Reduce the Complexity of XOR Operations of PM-MSR and PM-MBR Codes over $GF(2^w)$
Chuqiao Xiao, Xueqing Gong, Yefeng Xia, and Qian Zhang
Research Article (18 pages), Article ID 6642121, Volume 2021 (2021)

Deep Learning-Based Security Behaviour Analysis in IoT Environments: A Survey
Yawei Yue, Shancang Li, Phil Legg, and Fuzhong Li
Research Article (13 pages), Article ID 8873195, Volume 2021 (2021)

Improving Topic-Based Data Exchanges among IoT Devices
Fu Chen, Peng Liu, Jianming Zhu, Sheng Gao, Yanmei Zhang, Meijiao Duan, Youwei Wang, and Kai Hwang
Research Article (14 pages), Article ID 8884924, Volume 2020 (2020)

Seam-Carved Image Tampering Detection Based on the Cooccurrence of Adjacent LBPs
Dengyong Zhang, Xiao Chen, Feng Li, Arun Kumar Sangaiah, and Xiangling Ding
Research Article (12 pages), Article ID 8830310, Volume 2020 (2020)

Visual Object Multimodality Tracking Based on Correlation Filters for Edge Computing
Guosheng Yang and Qisheng Wei
Research Article (13 pages), Article ID 8891035, Volume 2020 (2020)

A Privacy-Protection Model for Patients
Wenzhi Cheng, Wei Ou, Xiangdong Yin, Wanqin Yan, Dingwan Liu, and Chunyan Liu
Research Article (12 pages), Article ID 6647562, Volume 2020 (2020)

A Novel Coevolutionary Approach to Reliability Guaranteed Multi-Workflow Scheduling upon Edge Computing Infrastructures
Zhenxing Wang, Wanbo Zheng, Peng Chen, Yong Ma, Yunni Xia, Wei Liu, Xiaobo Li, and Kunyin Guo
Research Article (11 pages), Article ID 6697640, Volume 2020 (2020)

A Novel Approach to Transform Bitmap to Vector Image for Data Reliable Visualization considering the Image Triangulation and Color Selection
Zhike Han, Xiuchao Wu, Meng Chi, Jun Tang, and Lijing Yang
Research Article (13 pages), Article ID 8871588, Volume 2020 (2020)

Cyborgan OS: A Lightweight Real-Time Operating System for Artificial Organ
Pan Lv, Hong Li, Jinsong Qiu, Yiqi Li, and Gang Pan
Research Article (9 pages), Article ID 8871626, Volume 2020 (2020)

An Improved Broadcast Authentication Protocol for Wireless Sensor Networks Based on the Self-Reinitializable Hash Chains
Haiping Huang, Qinglong Huang, Fu Xiao, Wenming Wang, Qi Li, and Ting Dai
Research Article (17 pages), Article ID 8897282, Volume 2020 (2020)

Hindawi
Security and Communication Networks
Volume 2021, Article ID 5558561, 19 pages
https://doi.org/10.1155/2021/5558561

Research Article

Research on Cross-Company Defect Prediction Method to Improve Software Security

Yanli Shao, Jingru Zhao, Xingqi Wang, Weiwei Wu, and Jinglong Fang

Key Laboratory of Complex Systems Modeling and Simulation, School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China

Correspondence should be addressed to Jinglong Fang; fjl@hdu.edu.cn

Received 18 January 2021; Revised 9 April 2021; Accepted 6 August 2021; Published 24 August 2021

Academic Editor: Honghao Gao

As the scale and complexity of software increase, software security issues have become the focus of society. Software defect prediction (SDP) is an important means to assist developers in discovering and repairing potential defects that may endanger software security in advance and improving software security and reliability. Currently, cross-project defect prediction (CPDP) and cross-company defect prediction (CCDP) are widely studied to improve the defect prediction performance, but there are still problems such as inconsistent metrics and large differences in data distribution between source and target projects. Therefore, a new CCDP method based on metric matching and sample weight setting is proposed in this study. First, a clustering-based metric matching method is proposed. The multigranularity metric feature vector is extracted to unify the metric dimension while maximally retaining the information contained in the metrics. Then use metric clustering to eliminate metric redundancy and extract representative metrics through principal component analysis (PCA) to support one-to-one metric matching. This strategy not only solves the metric inconsistent and redundancy problem but also transforms the cross-company heterogeneous defect prediction problem into a homogeneous problem. Second, a sample weight setting method is proposed to transform the source data distribution. Wherein the statistical source sample frequency information is set as an impact factor to increase the weight of source samples that are more similar to the target samples, which improves the data distribution similarity between the source and target projects, thereby building a more accurate prediction model. Finally, after the above two-step processing, some classical machine learning methods are applied to build the prediction model, and 12 project datasets in NASA and PROMISE are used for performance comparison. Experimental results prove that the proposed method has superior prediction performance over other mainstream CCDP methods.

1. Introduction

With the increasing scale and complexity of software, software security, and quality issues are becoming more and more important. Generally, it is difficult for developers to directly develop a safe and reliable software system all at once. Due to the influence of many factors, such as irregular development process and excessive code complexity, it is inevitable that there are defects in the software system; attackers can use them to destroy the software confidentiality, damage the software integrity, and cause serious security problems and economic losses. The software system needs to be fully tested to ensure its security and reliability. In the software development process, the later the defect is discovered, the higher the repair cost. However, test resources such as test time and personnel are limited, and modules with greater probability of defects should be prioritized to improve software repair efficiency and shorten software development cycle. Software defect prediction (SDP) is a common method to detect potential defects that may endanger software security and reliability. Currently, SDP has become a hot issue in software engineering for a long time [1–3]. Traditional SDP aims to use historical defect dataset within or across projects with the same metrics to build a prediction model to predict potential defects in new projects [4, 5]. The general process of SDP is as follows: first, analyze the software code or development process, extract the metrics [6, 7] related to the software defect information,

and combine the defect label information to constitute the training dataset. Then, construct the defect prediction model based on the above defect dataset. Finally, use the prediction model to predict whether there are defects in the other modules to optimize the allocation of test resources. This can assist developers to discover and repair potential defects in the software code of new projects as early as possible, thereby reducing software testing costs and improving software security, reliability, and maintainability.

Generally, large-scale projects like Java projects often contain many classes or methods. Each class is taken as a sample, and the number of lines of code, the number of operands, the coupling and cohesion between objects, cyclomatic complexity, etc., are selected as the metrics. Moreover, the defect label is obtained through manually review of each class code, and it also needs to be verified by domain expert to ensure its correctness. However, due to the extremely low efficiency and long cycle of manually determining module defects, the historical defect dataset for a new project is usually insufficient to build an accurate prediction model. More importantly, accurately marking module defect information is a prerequisite to ensure the effectiveness and accuracy of the constructed model, but ensuring the accuracy of defect labels requires repeated verification by a large number of professionals. This work is very costly for some small- and medium-sized enterprises. Since there are often fewer labeled defect datasets for a new project, it is difficult to build an effective prediction model to predict potential defects to improve software quality. As a result, many researchers began to study the cross-project defect prediction (CPDP) and cross-company defect prediction (CCDP) with the aim to make full use of the existing defect datasets of other projects to build an effective and accurate prediction model. The difference is that the former uses cross-project datasets with consistent metrics but different data distributions as the source dataset, while the latter uses cross-company project datasets with both different metrics and different data distributions as the source dataset for prediction model construction.

CPDP generally refers to constructing a defect prediction model to predict the defect information in the target project, under the condition that the metric number and meaning of the source and target projects are the same. The main research work is how to solve the problem of inconsistent data distribution. Since the dataset for constructing the model and using the model comes from two projects with different data distributions, it may reduce the accuracy of defect prediction to a certain extent. Thus, the data distribution transformation method is studied to increase the data distribution similarity between projects to improve the prediction accuracy. However, most companies are only willing to provide researchers with extracted metrics and defect information, rather than complete source code to ensure that the project will not be maliciously attacked. This leads to the fact that the metric number and meaning between the source and target projects for different companies are quite different, CPDP will no longer be applicable and heterogeneity will occur. In fact, the meaning and number of metrics extracted by the same company are also very different from before as more and more defect-related metrics are proposed to describe the project information more completely [8]. Heterogeneous problems are common, especially when the dataset used for defect prediction comes from multiple companies. For a target project, there is often no or few source project datasets that have exactly the same metrics as the target project, and the samples from different projects are very different because of their diversity in size and metrics. This makes it difficult to build a prediction model directly based on the existing historical defect dataset to achieve the accurate defect prediction of target project.

Therefore, CCDP is more practical and can be applied to cross-company projects with different metrics, different dataset sizes, and different data distributions. It mainly studied how to build a more accurate defect prediction model by unifying metrics and adjusting data distribution. Currently, some researches build CCDP models by extracting common metrics of the source and target projects, but these methods are not informative when the source and target projects from different companies have few or no common metrics. Another part of the research first uses the feature selection method to filter out some metrics of the source project and then matches with the metrics of target project. After the metrics are unified, the homogeneous defect prediction method can be used to solve the cross-company problems. However, the selected metrics may not fully describe the defect characteristics of the software module. This method not only ignores the impact of relevant metrics of source project on the defect prediction model construction but also does not consider the degree of differences between the source and target samples. Furthermore, the data distribution spaces from different projects are different, resulting in the prediction model built directly using the source dataset is not suitable for the target project, thereby reducing the prediction accuracy to a certain extent.

To address the above issues, this study proposes a CCDP method based on metric matching and sample weight setting to improve the security and reliability of software. The innovations lie in：on the one hand, a clustering-based metric matching method is proposed to solve the metric inconsistent and redundancy problem. The multigranularity metric feature vector is extracted to unify the metric dimension between the source and target projects. Moreover, metric clustering is applied to eliminate metric redundancy, and representative metrics are extracted to facilitate subsequent one-to-one metric matching. In this way, the cross-company heterogeneous defect prediction problem is transformed into a homogeneous problem. On the other hand, a sample selection-based weight setting method is applied to adjust the source data distribution to make it as consistent as possible with target project dataset. It uses sample selection frequency information as the impact factor to increase the weight of source samples that are more similar to the target samples, so as to improve the data distribution similarity between projects to further improve the prediction accuracy. The rest of this study is organized as follows: Section 2 is the related work. Section 3 introduces the core work in detail, including the clustering-based metric matching method and sample selection-based weight setting

algorithm. Section 4 is the related experimental verification and performance analysis. Finally, the conclusions and future work are given in Section 5.

2. Related Work

Traditional SDP aims to construct a defect prediction model using historical defect dataset of within project to predict the defects in the same projects. This can accurately and timely predict whether the software module is defective or not in the early development stage, thereby improving the software security, reliability, and maintainability. However, this method has great limitations due to the lack of labeling defect dataset for new projects, and it is often difficult to use small amount of historical dataset to build an effective and accurate prediction model. Therefore, many researchers have begun to study CPDP and CCDP to make full use of historical within-project, cross-project, and cross-company defect dataset to construct an effective prediction model to improve the prediction accuracy of target project. The following will introduce related work from the perspective of CPDP and CCDP.

2.1. CPDP. As mentioned above, CPDP mainly refers to the defect prediction when the metric number and meaning of the source and target projects are basically the same. In recent years, many studies have been conducted on CPDP [9]. Zimmermann et al. [10] deeply analyzed the feasibility of CPDP through 28 datasets from 12 real-world large software projects and conducted 622 groups of cross-project prediction experiments and found that only 3.4% achieved satisfactory results. For example, although the defect prediction model based on the Firefox project can achieve good results in the IE project, but not vice versa. Many factors need to be considered, such as the development process, the data itself, and the belonged domain. Generally, the data distribution between the source and target projects is quite different, so the model trained directly based on the source project normally does not perform well in the target project. Even if the metrics are consistent, the defect dataset itself still has problems such as metric redundancy and class imbalance, which may affect the prediction accuracy to a certain extent. Therefore, the CPDP research work is roughly carried out from the above three aspects.

Rahman [11] analyzed 12 different open source projects from the aspects of defect prediction performance, prediction stability, and dataset characteristics and concluded that there are potential commonalities between different software projects. Moreover, the data distribution transformation can be used to increase the similarity between projects to improve the prediction accuracy. Some researchers studied data distribution transformation method to reduce data distribution differences to improve prediction accuracy. Pan et al. [12] proposed a transfer component analysis (TCA) algorithm to map the datasets of different projects to the latent feature space to minimize the distance between the source domain and target domain. Although this method reduces the data distribution differences between different projects, the selection of data normalization method has a great impact on the final prediction performance. After that, Nam et al. [13] proposed the TCA + method based on TCA, which analyzed the characteristics of the source and target projects and adaptively selected the most suitable data normalization method. This method further increases the data distribution similarity between different projects and improves the accuracy of prediction models. The performance of TCA + varies greatly when using different source projects to build prediction models. To find the source project that is most similar data distribution to the target project to construct the model, Liu et al. [14] proposed a two-phase transfer learning model (TPTL) for CPDP. A source project estimator (SPE) is proposed to automatically choose two source projects with the highest distribution similarity to a target project from candidates and leverage TCA + to build two prediction models based on the two selected projects and combine their prediction results to further improve the prediction performance. Jinyin et al. [15] proposed a collective training mechanism consists of two-phase source data expansion phase and adaptive weighting phase for defect prediction, which makes the feature distributions of source and target projects similar to each other by transfer learning, and uses the particle swarm optimization algorithm to comprehensively consider the multiple source projects to predict the target project. Sun et al.[16] proposed a Collaborative Filtering-based source Projects Selection (CFPS) method, which used collaborative filtering algorithm to recommend the appropriate source projects to filter out the project that is most similar to the target project. Jin et al. [17] used the kernel twin support vector machines (KTSVMs) to implement domain adaptation (DA) to match the distributions of training data for different projects. These methods solve the problem of inconsistent data distribution to a certain extent and improve the prediction accuracy.

Some researchers have studied the problem of metric redundancy. Menzies et al. [18] solved the feature redundancy problem through feature subsets selection, but they found that the feature subsets selected from different datasets are inconsistent, which means that it is meaningless to search for feature subsets that are applicable to all projects. Liu et al. [19] proposed a feature selection method based on clustering, which selected a specified number of features in each class by calculating the correlation with another feature in same cluster, and avoided selecting irrelevant features. Similarly, Wang et al. [20] applied genetic algorithms to select feature subsets and simultaneously detected outliers in the dataset to further improve the quality of feature subsets. These methods considered the impact of feature redundancy on model construction and improve the prediction accuracy to a certain extent. However, the selection criterion is to discard the metrics with relatively small relevance, which will cause the amount of information contained in the metrics to be insufficient to represent the entire project in some cases, making it difficult to build a better model.

In addition, some researchers have conducted class imbalance studies, which is a common problem in CPDP and can affect the performance of prediction model. Sun et al. [21] proposed a coding-based ensemble learning (CEL)

method to solve the class imbalance problem. Jing et al. [22] proposed an improved subclass discriminant analysis (ISDA) method. The premise of the above methods is that the metrics between the source project and target project are the same or coincident in most cases. However, there is often no or few source projects that have exactly the same metrics as a certain target project, and CPDP method may not be applicable. In this case, CCDP is more important in practical applications due to the relatively limited dataset in the same projects and the different metrics between projects.

2.2. CCDP. CCDP aims to achieve accurate defect prediction between cross-company projects with different metrics, different dataset sizes, and different data distributions. Many researchers have done a lot of research based on this difference.

Some research works build the prediction model by extracting common metrics of the source and target projects. Turhan et al. [23] proposed a nearest neighbor filtering (NNFilter) algorithm. This method extracts the common metrics in the source and target projects and then uses a clustering algorithm to filter source samples similar to the target samples as the training dataset for prediction model construction. After that, Peters et al. [24] further proposed the Peter Filter method, which also first extracts the source samples similar to each target sample, and further improves the similarity between projects by filtering the training dataset. Ma et al. [25] proposed a transfer Naïve Bayes (TNB) algorithm. This method extracts the common metrics between the source and target projects, calculates their similarity by Euclidean distance, uses the gravitational formula to convert it into the weight of the training sample, and constructs a prediction model that is more suitable for the target datasets. The above methods are not universal since there are rarely the same metrics between the source and target projects. Furthermore, it may ignore unused metrics that may be useful for defect prediction. For example, there are 39 and 20 metrics in the NASA and PROMISE defect datasets, respectively, while their common metrics is only one. Therefore, the problem of few or no common metrics between the source and target projects has become a bottleneck for CCDP.

To solve this problem, some researchers have improved the data distribution similarity of the source and target projects from the perspective of transforming the data distribution, thereby improving the prediction accuracy. Jing et al. [26] proposed a canonical correlation analysis-based transfer learning (CCA+) algorithm. It uses the unified metric representation (UMR) method and typical association analysis to make the data distribution between source and target projects more similar, so as to improve the prediction accuracy through considering the linear separability of defect datasets. Ying et al. [27] proposed a kernel function-based typical association analysis method (Kernel CCA), which maps the defect dataset to the high-dimensional Hilbert space and then applies the CCA method to the projected transformation matrix of the source and target projects.

Another part of the research studies how to unify the metrics to convert the heterogeneous problem into homogeneous problem and use homogeneous defect prediction methods to solve the cross-company problem. Nam et al. [28] proposed a heterogeneous defect prediction (HDP) method, which not only solves the problem of feature redundancy during metrics selection but also realizes the metric matching between the source and target projects. However, this method only selects 15% of the source metrics, which may ignore the metrics that are strongly related to defect prediction in the source project and reduce the defect prediction accuracy. Similarly, feature matching and transfer method (FMT) proposed by Yu et al. [29] also performs metric matching based on feature similarity. But this method is only applicable to the filtered source project metrics that are less than the target project metrics, and the scale of the source and target projects depends on the companies with fewer dataset, so it cannot make full use of the richer source dataset.

Based on the above analysis, most studies only extracted part of the original metrics to build the defect prediction model without considering the impacts of unselected metrics on the prediction model. Meanwhile, the sample scale of the source and target projects from different companies was limited to the smaller of the two. Moreover, inconsistent data distribution has a certain impact on the prediction accuracy, and the different effects of samples with different degrees of similarity are not taken into account in most cases, which should also be considered to further improve the similarity of data distribution, thereby building a more accurate prediction model.

3. CCDP Method Based on Metric Matching and Sample Weight Setting

3.1. Method Overview. As a method to improve software security and reliability, SDP has always been a research hotspot in the field of software engineering. Building an accurate defect prediction model often requires sufficient historical defect dataset with defect labels, but there is usually not enough labeled defect dataset for a new project. Although the CPDP and CCDP methods are widely used to enrich defect dataset, there are still some problems in practical applications such as inconsistency of metrics and differences in the data distribution between source and target projects, which reduces the prediction performance. Thus, a CCDP method based on metric matching and sample weight setting is proposed in this study to address above issues. The method overview is shown in Figure 1. The specific process is as follows:

(1) Clustering-based metric matching: since the metric meaning and number between the source and target projects are quite different in the CCDP, it is necessary to match metrics between projects to facilitate the defect prediction model construction. Therefore, a clustering-based metric matching method is proposed here. First, extract multigranularity metric feature vectors from the source and target projects

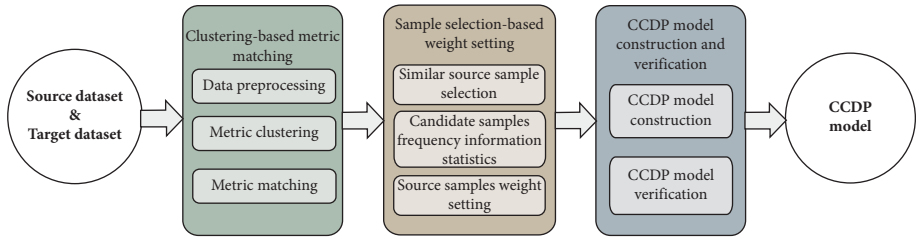


Figure 1: Method overview.

and unify them into the same dimension to calculate the similarity between metrics in the data preprocessing step. Then, apply metric clustering to obtain the representative feature vectors. Finally, perform one-to-one metric to solve the problem of inconsistent metrics. In this way, the cross-company heterogeneous defect prediction problem can be transformed into a homogeneous problem.

(2) Sample selection-based weight setting: even if the metrics are unified, the source and target datasets still have differences in data distribution, which reduces the effectiveness and accuracy of the defect prediction model. Hence, a sample weight setting method based on sample selection is applied to adjust the source data distribution. The key is to improve the data distribution similarity by increasing the weight of source samples similar to the target sample, thereby improving the defect prediction accuracy.

(3) CCDP model construction and verification: through the above metric matching and data distribution adjustment, the training dataset is now obtained. Finally, common machine learning methods are used to build the prediction model, and the proposed method and the mainstream CCDP algorithms are applied to 12 projects in NASA and PROMISE for experimental comparison and performance analysis to verify its feasibility and superiority. Accurate defect prediction results can be used to optimize new software testing resources, thereby reducing testing costs and improving product quality.

3.2. Clustering-Based Metric Matching. In the CCDP, the original source project dataset cannot be directly used as training dataset to build a prediction model for target defect prediction due to the inconsistent metrics of source and target projects. Even if the metrics are unified, the metric redundancy problem will also increase the training time and affect the prediction accuracy. Therefore, a clustering-based metric matching method is proposed to solve the above problems, as shown in Figure 2, which is mainly carried out in the following three steps:

(1) Data preprocessing: data preprocessing is performed first to unify the metric dimensions. Here, after data normalization, a multigranularity metric feature vector representation method is proposed to convert the source and target metrics into the same dimension, while retaining the metric information as much as possible to describe the data characteristics of source and target projects.

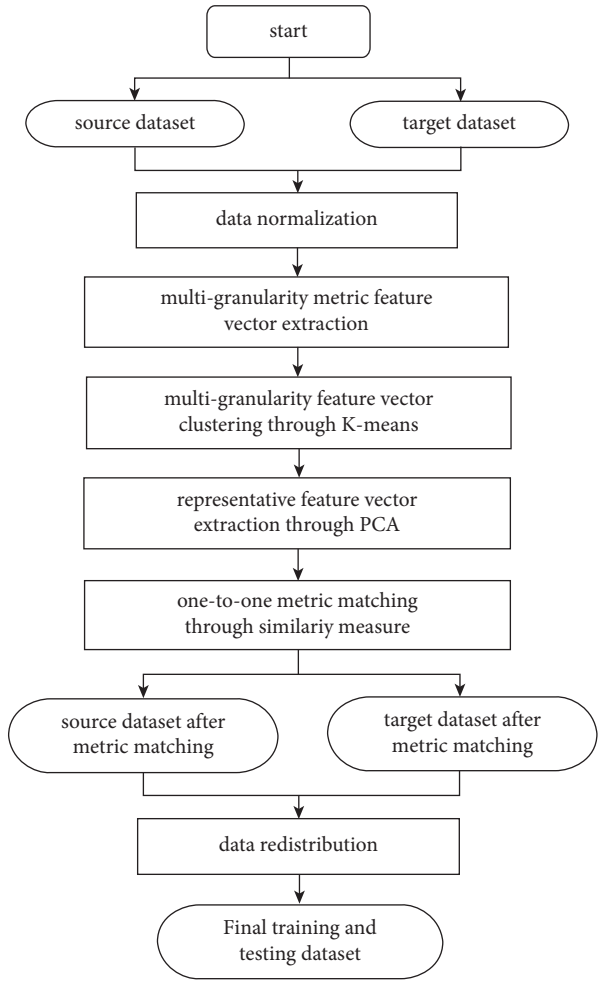


Figure 2: Flowchart of clustering-based metric matching.

(2) Metric clustering: after the metric dimensions are unified, the K-means clustering method with the same number of clusters is applied to the source and target projects, respectively. The Euclidean distance that can describe the similarity between metrics is used as a clustering measure. Meanwhile, the principal component analysis [30] (PCA) method is applied to extract the representative feature vector of each cluster in preparation for the following one-to-one metric matching.

(3) Metric matching: after unifying the metric dimension and number through above two steps, the metric pairs with highest similarity are matched in turn based on the Euclidean distance between the source and target representative metrics. This achieves one-to-one metric matching and eliminates the problem of inconsistent metrics. Even though the metrics have been unified through above operations, the extraction of multigranularity metric feature vector results in the loss of defect information of each sample in the dataset while the metric dimensions are unified. Thus, the datasets after metric matching cannot be used as training and testing dataset directly. Data redistribution should be performed based on the above clustering and matching results to obtain the final datasets for prediction model construction.

3.2.1. Data Preprocessing. To better understand the subsequent algorithm, some basic symbols are defined firstly, as shown in Table 1; the table shows the relevant information of the source and target project samples and metrics. Initially, the metric feature vectors of the source project and target project are represented as $S_{\text{original}} \in \{a_s^j|_{j=1}^{ls}\}$ and $T_{\text{original}} \in \{a_t^j|_{j=1}^{lt}\}$, respectively, where $a_s^j \in \Re^{n_s \times 1}$ and $a_t^j \in \Re^{n_t \times 1}$. There are three problems if the original metrics are directly used for metric matching. First, the dimensions of source and target metric vectors are quite different, but the premise of metric matching is that the metric dimensions of the source and target projects are consistent. For example, the sample number of source project CM1 in NASA is 327, and one of its metric vector dimensions is $a_s^{327\times 1}$. The sample number of target project Ant-1.7 is 745, so the metric vector dimension is $a_t^{745\times 1}$. It is difficult to directly calculate the metric similarity due to the different number of samples. Second, even if the same sample numbers of the source and target projects are selected to make the dimensions the same, the similarity between metrics is also affected by the order of the selected samples. The source metric vector varies with the sample positions. Thus, the similarity between metrics calculated in this way has great randomness. Third, if the metric feature vector is directly extracted, the data information may be greatly lost. For example, the metric dimension of the CM1 project in NASA is $a_s^{327\times 1}$. After extracting the metric feature vector, it becomes $\text{Feavecs}^{5\times 1}$, which may greatly reduce the data information contained in the metrics.

Due to the different scale of source code between different projects and different modules, the metric dimension is quite different, which makes it difficult to build prediction model directly. The core of data preprocessing in Step (1) is multigranularity metric feature vector extraction, which aims to unify the metric dimension between source and target projects. The MaxMinNormalization method is used to first standardize the sample metrics to minimize the impact of the dataset itself on the model performance. Generally, the numerical statistical attributes, such as minimum, maximum, average, median, and standard deviation, can briefly describe the data characteristics. But only using these values to describe the metric is too simple. To fully represent the metric, here each metric a_s^j is sorted from small to large according to their values and divided into $m = 5$ parts evenly. Supposing that x^{ij} represents the jth metric value of the ith sample, the jth metric of S can be described as $a_s^j = \{x_s^{1j}, x_s^{2j}, \ldots, x_s^{ij}, \ldots, x_s^{n_s j}\}$. The corresponding minimum, maximum, average, median, and standard deviation in each part are calculated (as shown in equation (1)) and combined to form the final metric feature vector $(\text{Fea vec}_s^j)_{25\times 1}$.

$$\text{Fea vec}_s^{j-m} = \left\{\min\left(a_s^{j-m}\right),\ \max\left(a_s^{j-m}\right), \text{avg}\left(a_s^{j-m}\right), \text{median}\left(a_s^{j-m}\right), \text{std}\left(a_s^{j-m}\right)\right\} m \in [1, 5]. \tag{1}$$

Now the original dimension of source metrics $(n_s * 1)$ and target metrics $(n_t * 1)$ are transformed into the same dimension $(25 * 1)$. Repeat the above steps for all the metrics, the multigranularity metric feature vectors of S and T can finally be expressed as $S_{\text{feavec}} \in \{\text{Feavec}_s^j|_{j=1}^{ls}\}$ and $T_{\text{fea vec}} \in \{\text{Fea vec}_t^j|_{j=1}^{lt}\}$, respectively, where $\text{Fea vec}^j \in \Re^{25\times 1}$.

Multigranularity metric feature vector can not only represent the metrics more comprehensively by characterizing the original metric information as much as possible but also unify the dimensions of the source and target metrics. After that, the Euclidean distance of the above two feature vectors can be directly calculated as a similarity measure between metrics to facilitate the subsequent metric clustering and matching. The Euclidean distance between the ith source metric (Fea vec_s^i) and jth target metric (Fea vec_t^j) is

$$\text{Dist}\left(\text{Fea vec}_s^i, \text{Fea vec}_t^j\right) = \sqrt{\sum_{k=1}^{25}\left(\text{Fea vec}_s^{ik} - \text{Fea vec}_t^{jk}\right)^2}. \tag{2}$$

3.2.2. Metric Clustering. Through the above data preprocessing step, the multigranularity metric feature vectors of S and T are $S_{\text{fea vec}} \in \{\text{Fea vec}_s^j|_{j=1}^{ls}\}$ and $T_{\text{fea vec}} \in \{\text{Fea vec}_t^j|_{j=1}^{lt}\}$, respectively, where each metric $\text{Fea vec}^j \in \Re^{25\times 1}$. Data preprocessing unifies the metric dimensions of the source and target projects, but the number and meaning of their metrics are still different, making it difficult to perform one-to-one metric matching directly. Therefore, metric clustering and PCA in Step (2) are performed on the source and target projects to eliminate the

Table 1: Basic symbol definition.

Symbol	Definition
S	Source project dataset
T	Target project dataset
x_s^i, x_t^i	ith sample of source and target projects
a_s^j, a_t^j	jth metric of source and target projects
n_s, n_t	Number of samples in the source and target projects
l_s, l_t	Number of metrics in the source and target projects

redundancy between metrics, while unifying the number of metrics to facilitate subsequent metric matching. First, the K-means clustering method divides all the metrics in the $S_{\text{fea vec}}$ and $T_{\text{fea vec}}$ into K ($l_s<K$ and $l_t<K$) clusters ($C = \{C_1, C_2, \ldots, C_K\}$, respectively, which makes the metrics in the same cluster have a high correlation, and the metric correlation between different clusters is not large. Fea vec_s^{ij} and Fea vec_t^{ij} represent the ith multigranularity metric of cluster C_j of project S and T. During the clustering process, the Euclidean distance is used as clustering measure. The Euclidean distance between the pairwise metrics of projects is calculated based on equation (2).

$$\begin{aligned} &\text{Fea vec}_s^{ij} \in C_{sj} \text{ while} i \in [1, l_s], \quad j \in [1, K], \\ &\text{Fea vec}_t^{ij} \in C_{tj} \text{ while} i \in [1, l_t], \quad j \in [1, K]. \end{aligned} \tag{3}$$

During the clustering process, the Euclidean distance is used as a clustering measure. The Euclidean distance between the pairwise metrics of projects is calculated based on equation (2). Moreover, the PCA method is used to extract the principal component of each cluster as the representative metrics $S_{\text{rep vec}} \in \{\text{Rep vec}_s^i | _{i=1}^K\}$ and $T_{\text{rep vec}} \in \{\text{Rep vec}_t^i | _{i=1}^K\}$, where $\text{Rep vec}^i \in \mathfrak{R}^{25\times 1}$. This step aims to eliminate metric redundancy and retain clustering information as much as possible since the highly correlated metrics are clustered in a cluster. The specific flow is shown in Algorithm 1.

Note that the number of clusters of the source and target metric needs to be set the same to facilitate the following one-to-one metric matching. Furthermore, when the clustering number is different, the clustering results on the same dataset will be different, and it is difficult to define the optimal cluster number directly. If the parameter setting is too small, it will be difficult to construct an effective model due to the loss of information. But if the parameter setting is too large, the clustering result is meaningless, which makes it impossible to eliminate strong correlation features. Since different cluster numbers will greatly affect the final results, in the experiment part, some parameters within the appropriate range will be selected as reference parameters first, and then inappropriate parameter will be excluded by filtering out abnormal predicted values. Refer to Section 4.2 for more details.

3.2.3. Metric Matching. After metric clustering in Step (2), the source and target project dataset features are now represented as $S_{\text{rep vec}} \in \{\text{Rep vec}_s^i | _{i=1}^K\}$ and $T_{\text{rep vec}} \in \{\text{Rep vec}_s^i | _{i=1}^K\}$ (K is the number of metrics after clustering with the same metric dimension). Metric matching in Step (3) refers to matching K metrics of project S and T in pairs to improve the data distribution similarity of the source and target projects, so as to further improve the versatility and accuracy of defect prediction model constructed. The metric matching process is shown in Figure 3.

Firstly, the similarity between the source and target metrics is measured through the Euclidean distance based on equation (2), which can be represented by a K^*K dimensional matrix W, as shown in Figure 3(a). Each element W_{ij} represents the similarity measure between Rep vec_s^i and Rep vec_t^i. Then, the subscript of the smallest value in the matrix W would be selected as the matching pair between the source and target metric, i.e., if W_{ij} is the minimum value in W, it means that the metric Rep vec_s^i and the metric Rep vec_t^j are the most similar, so they are matched and the corresponding rows and columns in the matrix are deleted, as shown in Figure 3(b), where gray means deleted. Similarly, next if W_{12} is the minimum value after the first matching, the metric Rep vec_s^1 and Rep vec_t^2 are matched, and the 1st row and 2nd column will be deleted as well, as shown in Figure 3(c). According to the matching order, all the metrics in the project S and T are, respectively, matched in pairs until the matching process is completed.

Figure 4 shows the entire clustering-based metric matching process, and the details are as follows.

As shown in Figure 4(a), there are the original source project S and target project T contains n_s and n_t samples, respectively. The corresponding number of metrics is also different, namely l_s and l_t. Initially, the original source and target metrics are represented as $S_{\text{original}} \in \{a_s^j | _{j=1}^{ls}\}$ and $T_{\text{original}} \in \{a_s^j | _{j=1}^{ls}\}$, respectively, where $a_s^j \in \mathfrak{R}^{n_s\times 1}$ and $a_t^j \in \mathfrak{R}^{n_t\times 1}$. Lines of the same color in the same project represent the metrics with similar meaning. The metrics cannot be matched directly because the metric dimensions from different projects are different and the similarity between metrics cannot be directly calculated. In other words, there are metric inconsistencies and redundancy problems in cross-company defect prediction. Thus, metric matching must be performed between the original source and target project dataset to facilitate the construction of subsequent prediction model. As shown in Figure 4(b), after MaxMinNormalization, the multigranularity metric feature vectors of SNorm and TNorm, represented as $S_{\text{fea vec}} \in \{\text{Fea vec}_s^j | _{j=1}^{ls}\}$ and $T_{\text{fea vec}} \in \{\text{Fea vec}_t^j | _{j=1}^{lt}\}$, are extracted in the data preprocessing step. In this way, although the number of metrics remains unchanged, the metric dimension is unified to 25 ($\text{Fea vec}^j \in \mathfrak{R}^{25\times 1}$), while retaining the metric information as much as possible. Then, as shown in Figure 4(c), the K-means clustering method is applied to unify the metric number of different projects by setting the same number of clusters (here K is 3). Metrics with similar meaning (same color) are clustered together, and then the PCA method is performed on each cluster to extract the representative metrics $S_{\text{rep vec}} \in \{\text{Rep vec}_s^i | _{i=1}^K\}$ and $T_{\text{rep vec}} \in \{\text{Rep vec}_t^i | _{i=1}^K\}$, where $\text{Repvec}^i \in \mathfrak{R}^{25\times 1}$. This step solves the metric redundancy problem and prepares for the next step of metric matching. As shown in Figure 4(d), after calculating the Euclidean distance of different metrics

Input: source and target multigranularity metric feature vector $S_{\text{fea vec}} \in \left\{\text{Fea vec}_s^j \middle|_{j=1}^{ls}\right\}$ and $T_{\text{feavec}} \in \left\{\text{Fea vec}_t^j \middle|_{j=1}^{lt}\right\}$; number of clusters K;
Output: source and target representative vector $S_{\text{rep vec}} \in \{\text{Rep vec}_s^i|_{i=1}^K\}$ and $T_{\text{rep vec}} \in \{\text{Rep vec}_t^i|_{i=1}^K\}$;
(1) For each project $\in$ {Source project S, Target project T}
(2) Randomly select K metrics as the starting centroid of the project $C = \{C_1, C_2, \ldots, C_K\}$;
(3) Repeat the following process until convergence
(4) For each metric Fea $\text{vec}^i \in \Re^{25\times 1}$:
(5) Calculate the Euclidean distance to each starting centroid based on eq.(2);
(6) Assign the metric to its nearest cluster with minimum distance;
(7) End for
(8) For each cluster C_j ($j \in [1, K]$):
(9) Calculate the mean value of the cluster C_j and update its centroid;
(10) End for
(11) For each cluster C_j ($j \in [1, K]$):
(12) Use PCA method to extract the corresponding representative vector Rep vec^i;
(13) End for
(14) End for
(15) Output $S_{\text{rep vec}} \in \{\text{Rep vec}_s^i|_{i=1}^K\}$ and $T_{\text{rep vec}} \in \{\text{Rep vec}_s^i|_{i=1}^K\}$;

Algorithm 1: Metric clustering algorithm.

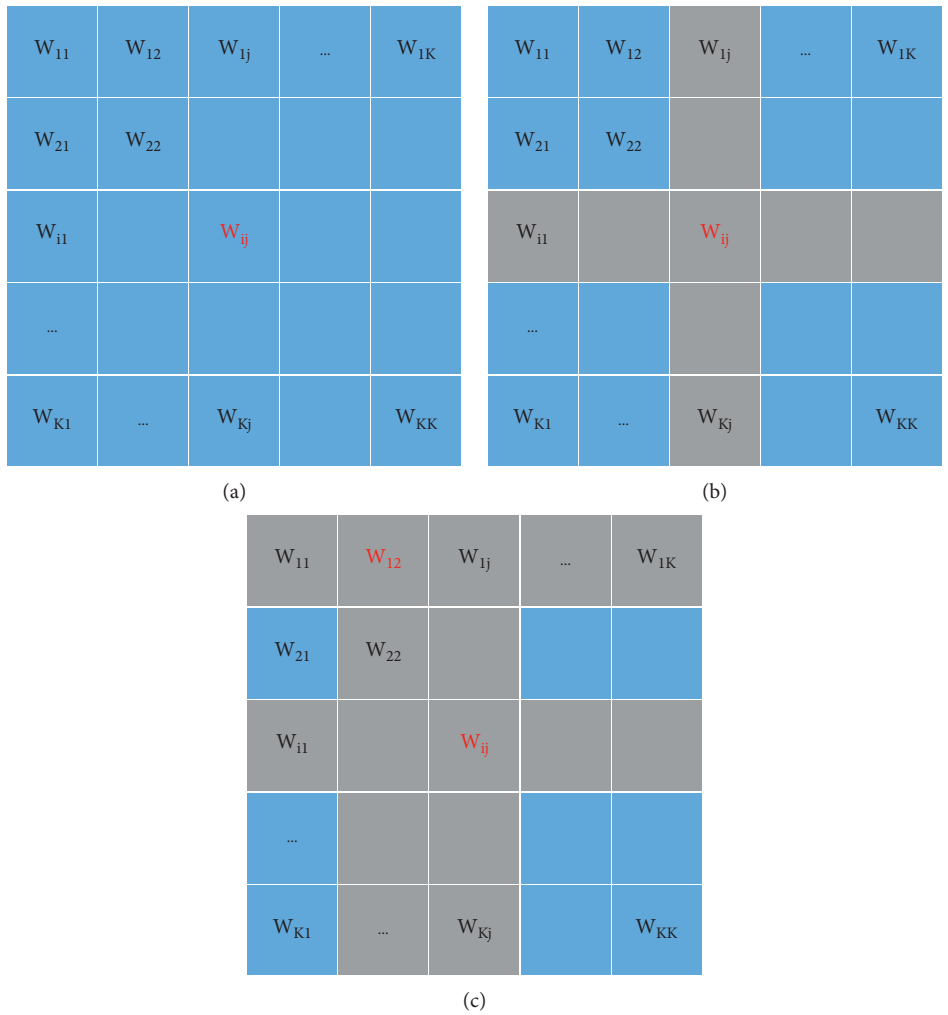


Figure 3: Metric matching process. (a) Original matrix, (b) matrix after first matching, and (c) matrix after second matching.

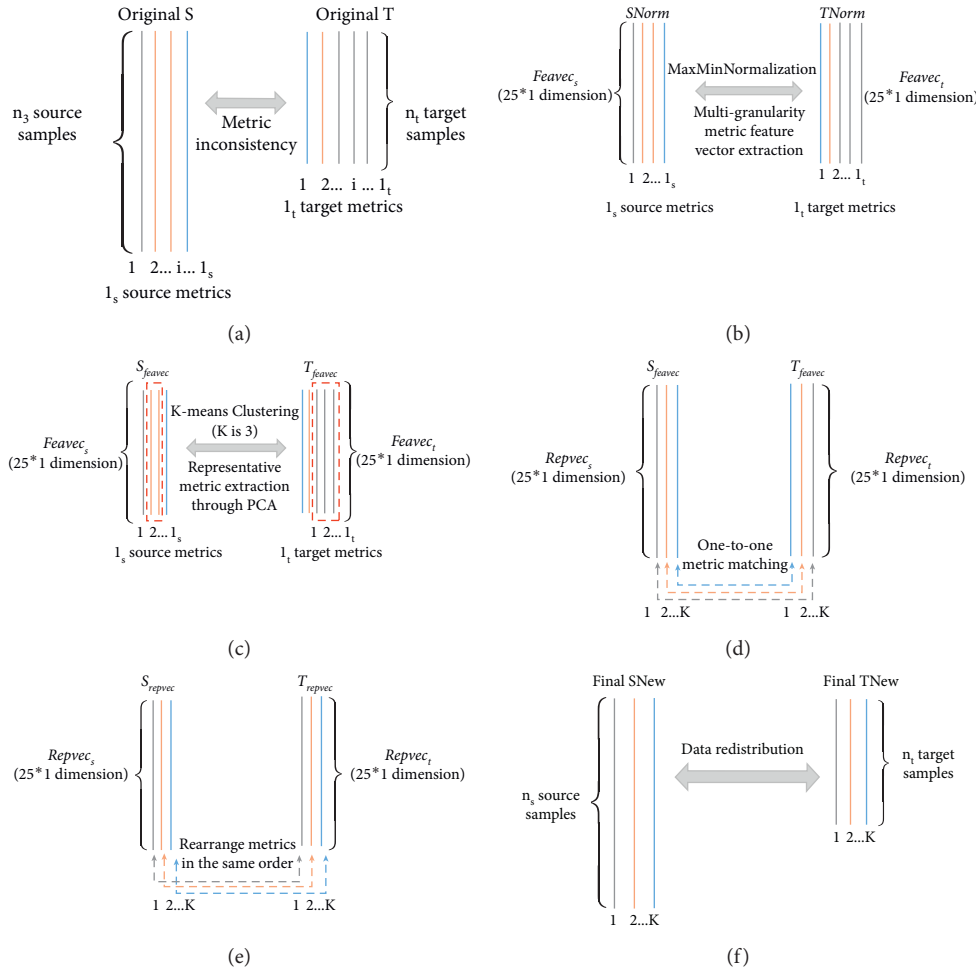


FIGURE 4: Schematic diagram of clustering-based metric matching. (a) Original datasets, (b) data preprocessing, (c) metric clustering, (d) metric matching, (e) metric arrangement, and (f) final datasets.

between different projects, one-to-one metric matching is performed based on the metric similarity information. The specific metric matching process is shown in Figure 3. Next, as shown in Figure 4(e), the matched metrics $S_{\text{rep vec}}$ and $T_{\text{rep vec}}$ are rearranged in the same order to make the data distribution of the source and target project as similar as possible. Finally, the original datasets are redistributed based on the above clustering results and metric order to obtain the final training and testing dataset, expressed as $SNew \in \mathfrak{R}^{n_s \times K} TNew \in \mathfrak{R}^{n_t \times K}$, as shown in Figure 4(f). The features of the source and target project datasets are restored to $S_{\text{rep vec}}{}' \in \left\{a_s^j \middle| {}_{j=1}^{K}\right\}$ and $T_{\text{rep vec}}{}' \in \left\{a_s^j \middle| {}_{j=1}^{K}\right\}$, respectively, where $a_s^j \in \mathfrak{R}^{n_s \times 1}$ and $a_t^j \in \mathfrak{R}^{n_t \times 1}$. This step is to restore the lost defect information of each sample in the dataset due to the extraction of multigranularity metric feature vector, so as to ensure the authenticity of the dataset used for prediction model construction and the fairness of comparison with other mainstream CCDP methods. The above operations have successfully unified the metrics and enhanced the data distribution similarity between source and target projects, so that the defect prediction model constructed using such source project dataset can achieve satisfactory results.

3.3. Sample Selection-Based Weight Setting. Although the clustering-based metric matching largely unified the source and target metrics, the data distribution difference between the source and target datasets makes the built defect prediction model still insufficient to achieve the accurate defect prediction of the target project. Therefore, how to select source data samples similar to the target ones and make the two data distributions as similar as possible to further improve the prediction accuracy is the focus of this section. Here, a sample selection-based weight setting algorithm, as shown in Algorithm2, is proposed to adjust the source data distribution to make it more similar to the target data distribution to build a more accurate defect prediction model.

The specific method steps are as follows:

(1) First, sample selection is performed to select the source samples similar to target samples. Similar to NNFilter [23], for each target sample, the Euclidean distance of all source samples is calculated and sorted, and then top N source samples with the smallest distance are selected as the candidate samples for each target one. In this study, N is set to 10 based on the practical experience [31].

Input: source and target $SNew \in \Re^{n_s \times K} TNew \in \Re^{n_t \times K}$, number of candidate samples N;
Output: the final training dataset $SNew\prime \in \Re^{n_{S\prime} \times K}$
(1) SCandidate = ∅ //array, which is used to store all the candidate training source samples
(2) For each sample x_t^j of target project T New:
(3) Advanced = ∅;//Array, which is used to store the selected samples in each loop
(4) For each sample x_s^i of source project SNew:
(5) Calculate the Euclidean distance between x_t^j and x_s^i based on equation (2);
(6) Sort source samples according to the above Euclidean distance information;
(7) Select the Top N source samples with the smallest distance and store them in advanced;
(8) SCandidate ← SCandidate + Advanced;
(9) End
(10) End
(11) Array Of Weight$^i \in \Re^{n_s \times 1}$ //array, which is used to store sample weight
(12) For each sample x_s^i of source project SNew:
(13) Statistic the sample frequency of x_s^i in SCandidate and update the Array Of Weight^i
(14) End
(15) Use MaxMinNormalization method to normalize ArrayOfWeight;
(16) Set the source sample weight based on ArrayOfWeight to obtain the final $SNew\prime \in \Re^{n_{S\prime} \times K}$

Algorithm 2: Sample selection-based weight setting algorithm.

(2) Second, the frequency information of each candidate sample is counted based on the aforementioned top N selection information. For the datasets with n target samples, $n * N$ source samples will be selected as candidate training samples, and some of source samples will be selected multiple times. It is assumed that the samples that are repeatedly selected are more similar to the target samples. Therefore, the frequency of each source sample being selected is counted as the basis for sample weight setting.

(3) Finally, the frequency information is used as a reference for sample weight setting. The maximum and minimum frequency value of entire candidate source samples are counted, and MaxMinNormalization is used to normalized the frequency information first, and then the weighted source samples with defect label information constitute the training dataset.

The above sample weight setting operation can accurately filter the source samples that are similar to the target sample and then use the calculated frequency information to set the source sample weight to further improve the data distribution similarity between the source and target projects, thereby improving the defect prediction performance.

3.4. CCDP Model Construction and Verification. Through the above metric matching and sample weight setting processing, the source and target project datasets with consistent metrics and similar data distribution are obtained, and the heterogeneity problem in CCDP is transformed into a homogeneous issue. Next, the commonly machine learning methods are used in the source dataset to build the defect prediction model to predict the defect information of target dataset. Extensive experiments should be performed on the experimental datasets for performance verification based on the evaluation indicators to prove the superiority of the proposed method.

3.4.1. Model Construction. Generally, there are many excellent machine learning models that perform well in defect prediction field. Here, the commonly machine learning methods such as logistic regression model (LR) [32], Naïve Bayes (NB) [33], and K-Nearest Neighbor (KNN) are used to build the defect prediction model. The model details are as follows:

(1) LR: when the dependent variable is dichotomous, LR is more suitable [34]. The method avoids the Gaussian assumption used in standard Naive Bayes.

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k, \quad (4)$$

where p is the probability that the defective module was found and x_1, x_2, ..., x_k are the independent variables. $\beta_0, \beta_1, \ldots, \beta_k$ are the regression coefficients estimated using maximum likelihood.

(2) NB: a statistical learning scheme that assumes that metrics are equally important and statistically independent.

$$P(c_k|x) = P(c_k) \times \frac{P(x|c_k)}{P(x)}, \quad (5)$$

where c_k is a member of the set of values for the dependent metric and x represents unknown sample. NB finds the conditional probability of that sample being labeled c_k to classify the test samples. The c_k with the highest probability is chosen as the label for x.

(3) KNN: KNN is a classic nonparametric decision procedure that classifies x, an unknown sample in the category of its nearest neighbor. As one of the simplest defect prediction models, it is usually used as baseline.

3.4.2. Model Verification. After the construction of CCDP models, extensive experiments are performed for performance verification and analysis. Some evaluation indicators that are commonly used in software defect prediction performance verification are selected to evaluate the prediction performance of the CCDP models, and the specific definitions are as follows, where TP, TN, FP, and FN refer to the number of true positive, true negative, false positive, false negative, respectively.

(1) Pd (probability of detection or recall) is the percentage of defects that are predicted correctly within the defect class. A higher Pd means more defects are detected, so the ideal case is Pd = 1, where all the defects are detected:

$$\text{Pd} = \frac{\text{TP}}{(\text{TP} + \text{FN})}. \tag{6}$$

(2) Pf (probability of false alarm) refers to the percentage of nondefective samples that are incorrectly predicted within the nondefect class. The higher the Pf, the more time and cost are wasted to predict the true defect:

$$\text{Pf} = \frac{\text{FP}}{(\text{FP} + \text{TN})}. \tag{7}$$

(3) Precision is what percentage of samples predicted as defective are actually such. However, it does not tell us anything about the number of samples that the classifier mislabeled:

$$\text{precision} = \frac{\text{TP}}{(\text{TP} + \text{FP})}. \tag{8}$$

(4) F-measure ($F1$) is the harmonic mean of precision and recall. Precision is a measure of exactness, whereas recall is a measure of completeness. But there tends to increase one at the cost of reducing the other. F-measure is an alternative way to use precision and recall by combining them into a single measure:

$$\text{F1} = \frac{2 \times \text{recall} \times \text{precision}}{\text{recall} + \text{precision}}. \tag{9}$$

(5) AUC (area under curve) is the area under the receiver operating characteristic curve. It is a useful measure for comparing different models because it is unaffected by class imbalance and is independent of the prediction threshold.

Generally, Pd and Pf will be affected by issues such as threshold setting and class imbalance and cannot effectively and accurately evaluate the prediction performance. In fact, a good classifier should both have higher Pd and lower Pf. F-measure and AUC are the trade-off measure that balances the performance between Pd and Pf. A higher F-measure means a better prediction performance. Especially, AUC is recognized by many researchers and is widely used in SDP [35]. The performance verification experiment based on the NASA datasets carried out by Jiang et al. [36] also proved that AUC is superior to other evaluation criteria in stability by comparing the variance of different evaluation indicators. The value of AUC is between 0 and 1. The larger the value of AUC, the better the performance of the prediction model.

Based on the above analysis, the overall algorithm flow is shown in Algorithm 3.

4. Experimental Verification and Performance Analysis

In this section, after the introduction of experimental datasets, three groups of experiments are designed to verify the feasibility and effectiveness of clustering-based metric matching method, sample selection-based weight setting method, and the overall proposed method. The threats to validity are given finally.

4.1. Experimental Dataset Introduction. All the experimental datasets are from NASA [37] and PROMISE, which are widely used in the field of SDP and are generally authoritative and recognized. The specific dataset information is shown in Table 2.

The number of metrics of NASA and PROMISE is 37 and 20, respectively, and the specific metric meaning is shown in Table 3 and 4, respectively. Generally, a project consists of multiple software modules, and a module is a sample in software defect prediction. The features that describe the software module mainly consist of Mccabe [38], Halstead, CK, etc. Mccabe mainly designs the metric from the structural point of view, such as 7-CYCLOMATIC_COMPLEX and 14-ESSENTIAL_COMPLEXITY in Table 3. Halstead mainly designs metrics from the aspect of code size, such as 22-HALSTEAD_LENGTH and 25-HALSTEAD_VOLUME. CK mainly designs the metrics from the object-oriented perspective, such as 1-wmc and 2-dit, in Table 4. Especially, complexity, coupling, and cohesion metrics [39, 40], i.e., the 7-CYCLOMATIC_COMPLEX, 11-DESIGN_COMPLEXITY , 14-DESIGN_COMPLEXITY, and 26-MAINTENANCE_SEVERITY in Table 3 and 4-cbo, 6-lcom, 7-ca, 8-ce, 13-moa, and 18-amc in Table 4, etc., are all important metrics related to software security. The metrics are extracted and defined from different views by different experts in different companies, covering all aspects of software quality, security and reliability, and can comprehensively represent software defect information to build more accurate prediction models. Meanwhile, the metric number and meaning of each project datasets of the two companies are quite different, which are very suitable for verifying the effectiveness of the proposed CCDP method in this study.

4.2. Clustering-Based Metric Matching Performance Analysis. The first experiment is to verify the performance of the clustering-based metric matching strategy, which is mainly divided into two parts, metric clustering performance analysis and metric matching performance analysis.

Input: source company project datasets $S \in \{x_s^i|_{i=1}^{n_s}, y_s^i|_{i=1}^{n_s}\}$ as training datasets;
Target company project datasets $T \in \{x_t^i|_{i=1}^{n_t}\}$ as test datasets;
Output: trained defect prediction model
(1) Clustering-based metric matching:
(a) Use MaxMinNormalization method to normalize S and T to get SNorm and TNorm;
(b) Extract the multigranularity metric feature vector of S and T, expressed as $S_{\text{fea vec}} \in \{\text{Fea vec}_s^j|_{j=1}^{ls}\}$ and $T_{\text{fea vec}} \in \{\text{Fea vec}_t^j|_{j=1}^{lt}\}$, where $\text{Fea vec}^j \in \Re^{25\times 1}$;
(c) Apply K-means clustering algorithm to cluster $S_{\text{fea vec}}$ and $T_{\text{fea vec}}$ into K clusters, respectively;
(d) Extract the principal component of each cluster through PCA as the representative vector $S_{\text{rep vec}} \in \{\text{Rep vec}_s^i|_{i=1}^{K}\}$ and $T_{\text{rep vec}} \in \{\text{Rep vec}_t^i|_{i=1}^{K}\}$, where $\text{Rep vec}^i \in \Re^{25\times 1}$;
(e) Perform one-to-one metric matching on $S_{\text{fea vec}}$ and $T_{\text{fea vec}}$ through metric matching;
(f) Redistribute SNorm and TNorm based on above steps, expressed as $S\text{New} \in \Re^{n_s\times K}$, $T\text{New} \in \Re^{n_t\times K}$.
(2) Sample selection-based weight setting:
(a) Use NNFilter to select N source samples that similar to each target sample based on Euclidean distance as candidate training data samples $S\text{Candidate} \in \{x_s^i|_{i=1}^{N\times n_t}, y_s^i|_{i=1}^{N\times n_t}\}$;
(b) Statistic the frequency of selected source samples in the SCandidate;
(c) Use samples frequency information in SCandidate as the basis for sample weight setting.
(3) CCDP model construction and verification:
(a) Use the weighted source samples as the training dataset and apply common machine learning methods including LR, NB, and KNN to construct the predict model;
(b) Perform experiments on multiple defect datasets and evaluate the performance of the proposed method.

Algorithm 3: Metric matching and sample weight setting based CCDP algorithm.

Table 2: Defect datasets.

Company	Dataset	Feature num	Samples num	Defective samples num	Nondefective samples num	Defect rate (%)	Imbalance ratio
NASA	CM1	37	327	42	285	12.8	6
	MW1	37	253	27	226	10.7	8
	PC1	37	705	61	644	8.7	10
	PC3	37	1077	134	943	12.4	7
	PC4	37	1287	177	1110	13.8	6
PROMISE	Ant-1.7	20	745	166	579	22.3	3
	Camel-1.6	20	965	188	777	19.5	4
	Ivy-2.0	20	352	40	312	11.4	7
	Jedit-4.3	20	492	11	481	2.2	43
	Synapse-1.2	20	256	86	170	33.6	1
	Velocity-1.6	20	229	78	151	34.1	1
	Xalan-2.4	20	723	110	613	15.2	5

4.2.1. Metric Clustering Performance Analysis. Generally, there are metric inconsistencies and redundancy problems in the cross-company defect prediction, so after extracting multigranularity metric feature vectors, K-means clustering, and PCA methods are applied to cluster the metrics and extract the representative metric to address above problems. As mentioned earlier, the parameter setting (i.e., clustering number) is very important, too large or too small may affect the accuracy of final prediction results. Therefore, the selection of the number of clusters will be discussed here. Five source projects in NASA are selected as training dataset to construct the prediction model to predict the defects of the target project Ant-1.7 in PROMISE. Each source project will conduct 6 experiments, and the number of clusters varies from 4 to 9, a total of 30 experiments. The experimental results are shown in Table 5.

In Table 5, K represents the clustering number and ratio is the imbalance rate of predicted labels in target projects, that is, the percentage of samples predicted to be nondefective divided by the percentage of samples predicted to be defective. Pd, Pf, F1, and AUC are used to evaluate the prediction performance. Among them, F1 and AUC are considered mainly due to their accuracy and robustness of performance evaluation. It can be observed from the table that

(1) The proposed method performs well in the heterogeneous cross-project defect prediction. The evaluation indicators such as Pd, Pf, F1, and AUC can achieve satisfactory results to a certain extent even with different K values. The reason is that the metric clustering can alleviate the metric redundancy while retaining the metric information maximally, thereby

Table 3: The metrics in NASA datasets.

ID	Feature
1	LOC_BLANK
2	BRANCH_COUNT
3	CALL_PAIRS
4	LOC_CODE_AND_COMMENT
5	LOC_COMMENTS
6	CONDITION_COUNT
7	CYCLOMATIC_COMPLEXITY
8	CYCLOMATIC_DENSITY
9	DECISION_COUNT
10	DECISION_DENSITY
11	DESIGN_COMPLEXITY
12	DESIGN_DENSITY
13	EDGE_COUNT
14	ESSENTIAL_COMPLEXITY
15	ESSENTIAL_DENSITY
16	LOC_EXECUTABLE
17	PARAMETER_COUNT
18	HALSTEAD_CONTENT
19	HALSTEAD_DIFFICULTY
20	HALSTEAD_EFFORT
21	HALSTEAD_ERROR_EST
22	HALSTEAD_LENGTH
23	HALSTEAD_LEVEL
24	HALSTEAD_PROG_TIME
25	HALSTEAD_VOLUME
26	MAINTENANCE_SEVERITY
27	MODIFIED_CONDITION_COUNT
28	MULTIPLE_CONDITION_COUNT
29	NODE_COUNT
30	NORMALIZED_CYLOMATIC_COMPLEXITY
31	NUM_OPERANDS
32	NUM_OPERATORS
33	NUM_UNIQUE_OPERANDS
34	NUM_UNIQUE_OPERATORS
35	NUMBER_OF_LINES
36	PERCENT_COMMENTS
37	LOC_TOTAL

Table 4: The metrics in PROMISE datasets.

ID	Feature	Description
1	wmc	Weighted methods per class
2	dit	Depth of inheritance tree
3	noc	Number of children
4	cbo	Coupling between object classes
5	rfc	Response for a class
6	lcom	Lack of cohesion in methods
7	ca	Afferent couplings
8	ce	Efferent couplings
9	npm	Number of public methods
10	Lcom3	Lack of cohesion in methods, different from LCOM
11	loc	Lines of code
12	dam	Data access metric
13	moa	Measure of aggregation
14	mfa	Measure of functional abstraction
15	cam	Cohesion among methods of class
16	ic	Inheritance coupling
17	cbm	Coupling between methods
18	amc	Average method complexity
19	max_cc	Maximum McCabe's cyclomatic complexity
20	avg_cc	Average McCabe's cyclomatic complexity

building a more accurate prediction model to further improve the prediction performance.

(2) For the same source project, different K values show different prediction performance, and in some cases, the difference is quite large. For example, when CM1 is used as the source project, the value of F1 falls in the area between 0.462 and 0.58, and AUC is in the range of 0.654 and 0.746, correspondingly that means the performance difference is about 10%. For an inappropriate clustering parameter, the performance may be very poor in some source projects, i.e., when MW1 is used as the source project and $K = 6$, all of the evaluation indicators are lower than the average level. As the number of clusters affects the clustering performance, when it is not set properly, the extracted principal component feature vector cannot effectively represent the metrics, thereby reducing the prediction accuracy.

(3) The imbalance ratio can be used as a reference for selecting effective clustering parameters. For example, when K is 6 for MW1, the prediction performance is poor and the ratio is also an abnormal value, so the imbalance ratio can be used to exclude an inappropriate cluster number. For a specific target project, the above 30 experiments are all aimed at this target project. Therefore, the imbalance ratio of the prediction results of 30 models is calculated for each target project, the related results are shown in Figure 5. The visualization results are the box plots corresponding to 7 target projects in PROMISE.

Here, we use the imbalance ratio of the predicted result as an indicator and then filter outliers by calculating the quartiles of the boxplot and finally choose an appropriate clustering number for the following metric clustering. As can be seen from Table 6, the prediction result class imbalance ratio box plot of Ant-1.7 has $Q1 = 2.027$, $Q3 = 2.555$. Here, the ratio between $Q1$ and $Q3$ is taken as candidate parameters to filter out inappropriate value of K. For the results of MW1, the ratio is 0.815 when $K = 6$ and the predicted result is obviously not within the normal range of ratio, thus the abnormal value is not involved to calculate the overall result. This strategy excludes the inappropriate parameters, such as $K = 4$, 5, 6, 8, 9, and the remaining parameter $K = 7$ is set to the number of clusters. When the ratio of a target project is not within the range to be selected, the K closest to the range will be selected to perform metric clustering. Generally, after excluding inappropriate K values, the median of remaining K values is chosen as the number of clusters to obtain the final prediction result.

4.2.2. Metric Matching Performance Analysis. To verify the effectiveness of metric matching method, we use the original defect dataset in NASA without and with metric matching as training dataset for comparison experiments, and build a LR model as defect prediction model. Here, Euclidean distance

Table 5: Performance results of different cluster numbers for target project Ant-1.7

Source	K	Ratio	Pd	Pf	F1	AUC	K	Ratio	Pd	Pf	F1	AUC
CM1	4	2.477	0.584	0.202	0.511	0.691	7	2.543	0.578	0.197	0.511	0.691
	5	2.278	0.687	0.196	0.58	0.746	8	3.022	0.488	0.18	0.462	0.654
	6	2.307	0.675	0.196	0.573	0.74	9	3.482	0.476	0.151	0.476	0.663
MW1	4	2.701	0.639	0.164	0.578	0.737	7	2.153	0.639	0.225	0.527	0.707
	5	2.56	0.633	0.18	0.56	0.726	8	2.012	0.645	0.242	0.518	0.701
	6	**0.815**	**0.307**	**0.621**	**0.177**	**0.343**	9	2.012	0.62	0.249	0.499	0.686
PC1	4	2.207	0.669	0.209	0.558	0.73	7	1.522	0.458	0.379	0.33	0.539
	5	1.918	0.681	0.246	0.537	0.718	8	2.153	0.536	0.254	0.443	0.641
	6	1.725	0.681	0.277	0.515	0.702	9	2.113	0.578	0.247	0.474	0.665
PC3	4	2.221	0.669	0.208	0.559	0.731	7	2.153	0.452	0.279	0.373	0.587
	5	2.413	0.645	0.192	0.557	0.726	8	2.074	0.705	0.216	0.574	0.744
	6	2.526	0.669	0.173	0.589	0.748	9	1.988	0.723	0.223	0.578	0.75
PC4	4	2.979	0.596	0.152	0.561	0.722	7	3.376	0.398	0.18	0.393	0.609
	5	2.012	0.289	0.344	0.232	0.472	8	2.351	0.663	0.194	0.567	0.734
	6	2.895	0.434	0.206	0.403	0.614	9	2.796	0.536	0.185	0.492	0.676

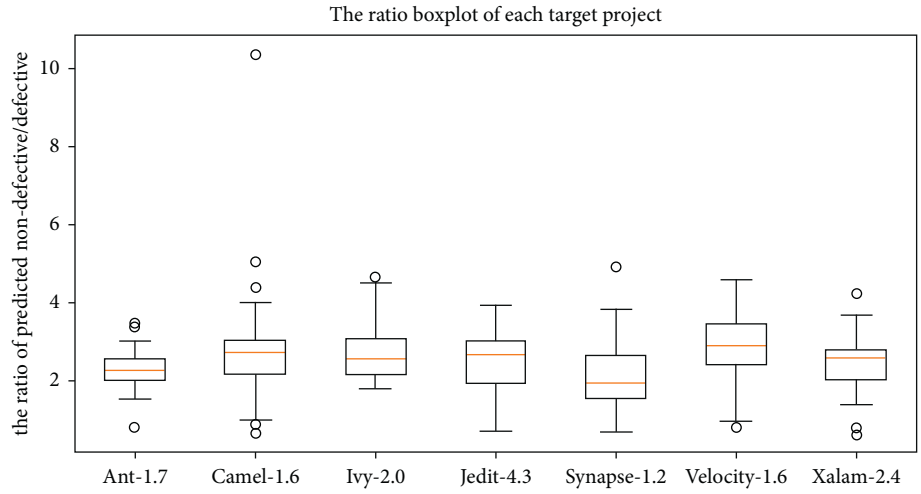


Figure 5: The imbalance ratio boxplots for each target project.

Table 6: Performance results of different K values for target project Ant-1.7

Source	Target	K	Ratio	Pd	Pf	F1	AUC
MW1	Ant-1.7	4	2.701	0.639	0.164	0.578	0.737
		5	2.56	0.633	0.18	0.56	0.726
		6	0.815	0.307	0.621	0.177	0.343
		7	**2.153**	0.639	0.225	0.527	0.707
		8	2.012	0.645	0.242	0.518	0.701
		9	2.012	0.62	0.249	0.499	0.686
	Median		2.153	0.639	0.225	0.527	0.707

is used as a measure to support metric matching. The experimental results are shown in Table 7, AUC is selected as the evaluation indicator here. The larger the AUC value, the better the prediction model, and the bold value in the table indicates better performance. In the table, the AUC-Original refers to the AUC values obtained when the source project dataset without metric matching is used as training dataset. AUC-MM refers to the AUC values obtained when the source project dataset with metric matching is used as training dataset for prediction model construction. It can be seen from the table that, in most cases, the prediction model constructed based on the dataset after metric matching shows better prediction performance, higher prediction stability, and accuracy. Under the same machine learning model, the final average AUC of the source project dataset after metric matching in 35 groups of experiments is 31.05% = (0.65–0.496)/0.496 higher than the source project dataset without metric matching. The reason is that metric matching not only unifies the metrics and eliminates the metric redundancy problem, and transforms the heterogeneous

Table 7: Metric matching performance analysis results.

No.	Source	Target	AUC-original	AUC-MM
1	CM1	Ant-1.7	0.569	**0.746**
2	MW1	Ant-1.7	0.431	**0.707**
3	PC1	Ant-1.7	0.427	**0.641**
4	PC3	Ant-1.7	0.687	**0.748**
5	PC4	Ant-1.7	0.424	**0.734**
6	CM1	Camel-1.6	**0.611**	0.549
7	MW1	Camel-1.6	0.457	**0.536**
8	PC1	Camel-1.6	0.419	**0.55**
9	PC3	Camel-1.6	**0.574**	0.549
10	PC4	Camel-1.6	0.472	**0.589**
11	CM1	Ivy-2.0	0.569	**0.761**
12	MW1	Ivy-2.0	0.382	**0.71**
13	PC1	Ivy-2.0	0.457	**0.603**
14	PC3	Ivy-2.0	0.681	**0.756**
15	PC4	Ivy-2.0	0.467	**0.729**
16	CM1	Jedit-4.3	0.555	**0.689**
17	MW1	Jedit-4.3	0.405	**0.693**
18	PC1	Jedit-4.3	0.333	**0.674**
19	PC3	Jedit-4.3	0.656	**0.678**
20	PC4	Jedit-4.3	0.454	**0.7**
21	CM1	Synapse-1.2	0.556	**0.635**
22	MW1	Synapse-1.2	0.398	**0.676**
23	PC1	Synapse-1.2	0.45	**0.656**
24	PC3	Synapse-1.2	**0.69**	0.665
25	PC4	Synapse-1.2	0.45	**0.65**
26	CM1	Velocity-1.6	0.495	**0.68**
27	MW1	Velocity-1.6	0.371	**0.593**
28	PC1	Velocity-1.6	0.362	**0.599**
29	PC3	Velocity-1.6	**0.613**	0.596
30	PC4	Velocity-1.6	0.496	**0.579**
31	CM1	Xalan-2.4	0.583	**0.602**
32	MW1	Xalan-2.4	0.42	**0.611**
33	PC1	Xalan-2.4	0.394	**0.6**
34	PC3	Xalan-2.4	0.623	**0.663**
35	PC4	Xalan-2.4	0.437	**0.607**
	Average		0.496	**0.65**

prediction problem into a more common homogeneous prediction problem, but also improves the data distribution similarity between the source and target project. Therefore, metric matching is effective and can improve the performance of the defect prediction model to a certain extent.

4.3. Sample Weight Setting Performance Analysis. The second experiment is to verify the impact of data distribution adjustments based on sample weight setting on performance improvement. After performing metric matching, the datasets with and without sample weight setting were used to train the LR model for performance comparison, respectively. The experimental results are shown in Table 8. Here, AUC-MM refers to the AUC value obtained by training the prediction model using the source project dataset with only metric matching. AUC-MMWS refers to the AUC value obtained by training the prediction model after sample weight setting. Similarly, the bold in the table indicates a relatively high AUC value, which has better prediction performance. It can be seen that, in most cases, the prediction model constructed after setting the sample weights performs better. Even if the former (AUC-MM) performs better, the performance of the two is not of much different, i.e., for the 4th, 10th, 19th, 22th, 26th experiment, and so on. Under the same machine learning model, the source project dataset after sample weight setting in the 35 experiments is 5.54% = (0.686–0.65)/0.65 higher than the source project dataset with only metric matching. Therefore, the prediction performance can be optimized by improving the data distribution similarity between the source and target projects by sample weight setting strategy.

4.4. Overall Prediction Performance Verification. The last experiment is to verify the overall prediction performance of the proposed method (abbreviated as MMWS) in this study. Comparative experiments are made with the mainstream CCDP algorithms (FMT [29], HDP [28], and RM [29]) and the proposed method to demonstrate its applicability and effectiveness. Five datasets of NASA are used as the training dataset in the CCDP to build the defect prediction model, and seven datasets of PROMISE are used as the test dataset, a total of 35 groups of experiments. Three commonly used

Table 8: Sample weight setting performance analysis results.

No.	Source	Target	AUC-MM	AUC-MMWS
1	CM1	Ant-1.7	**0.746**	0.69
2	MW1	Ant-1.7	0.707	**0.74**
3	PC1	Ant-1.7	0.641	**0.708**
4	PC3	Ant-1.7	**0.748**	0.743
5	PC4	Ant-1.7	**0.734**	0.715
6	CM1	Camel-1.6	0.549	**0.558**
7	MW1	Camel-1.6	0.536	**0.629**
8	PC1	Camel-1.6	0.55	**0.617**
9	PC3	Camel-1.6	0.549	**0.622**
10	PC4	Camel-1.6	**0.589**	0.544
11	CM1	Ivy-2.0	**0.761**	0.708
12	MW1	Ivy-2.0	0.71	**0.772**
13	PC1	Ivy-2.0	0.603	**0.769**
14	PC3	Ivy-2.0	**0.756**	0.723
15	PC4	Ivy-2.0	**0.729**	0.687
16	CM1	Jedit-4.3	0.689	**0.711**
17	MW1	Jedit-4.3	0.693	**0.707**
18	PC1	Jedit-4.3	0.674	**0.743**
19	PC3	Jedit-4.3	**0.678**	0.67
20	PC4	Jedit-4.3	0.7	**0.703**
21	CM1	Synapse-1.2	0.635	**0.648**
22	MW1	Synapse-1.2	**0.676**	0.667
23	PC1	Synapse-1.2	0.656	**0.685**
24	PC3	Synapse-1.2	**0.665**	0.632
25	PC4	Synapse-1.2	0.65	**0.711**
26	CM1	Velocity-1.6	**0.68**	0.663
27	MW1	Velocity-1.6	0.593	**0.68**
28	PC1	Velocity-1.6	0.599	**0.679**
29	PC3	Velocity-1.6	0.596	**0.686**
30	PC4	Velocity-1.6	0.579	**0.67**
31	CM1	Xalan-2.4	0.602	**0.684**
32	MW1	Xalan-2.4	0.611	**0.726**
33	PC1	Xalan-2.4	0.6	**0.717**
34	PC3	Xalan-2.4	0.663	**0.716**
35	PC4	Xalan-2.4	0.607	**0.703**
	Average		0.65	**0.686**

machine learning models LR, NB, and KNN are used as the prediction models, and the experimental results are shown in Table 9. The standard deviations and average values of AUC for different experiments are shown in the last two rows. Compared with the three mainstream defect prediction algorithms, the standard deviation value of MMWS for all the target projects is the lowest, which shows better stability in defect prediction. In addition, the experimental results show that the prediction performance of MMWS is also the best. As can be seen from the last row, although the prediction results of the proposed method are not always the best when using different machine learning models. However, its best performance on the LR model is 0.686, which is 1.9% (0.686–0.667), 4.4% (0.686–0.642), and 3.9% (0.686–0.647) higher than the average best AUC values of all the comparison algorithms. The reason is that the other mainstream methods only extract part of the metric or samples to adjust the data distribution of the source and target projects. For the proposed method (MMWS), on the one hand, metric matching strategy can characterize the metric information maximally while solving the problem of metric inconsistency and redundancy, so as to transform the heterogeneous prediction problem into a homogeneous problem. On the other hand, the weight setting strategy further improves the data distribution similarity between the source and target project, which facilitates the construction of a more general and accurate prediction model. The experimental results of the first two groups also verify the feasibility and effectiveness of these two strategies. Overall, the proposed method shows better performance in the CCDP, with higher prediction accuracy and stronger prediction stability.

4.5. Threats to Validity. Experimental results prove that the method proposed in this study performs better in the cross-company defect prediction, but there are still some factors and potential threats that affect the method validity:

First, it is difficult to obtain large-scale project datasets with defect labels, here only some datasets in NASA and PROMISE are used for comparison experiments. More cross-company software defect datasets should be used in the future to further verify the availability and stability of the cross-company defect model.

Table 9: Performance verification results of mainstream methods.

Source	Target	KNN model				NB model				LR model			
		FMT	HDP	RM	MMWS	FMT	HDP	RM	MMWS	FMT	HDP	RM	MMWS
CM1	Ant-1.7	**0.713**	0.654	0.664	0.555	**0.756**	0.646	0.688	0.663	0.703	**0.826**	0.660	0.690
MW1	Ant-1.7	0.728	0.722	**0.756**	0.701	0.812	**0.816**	0.685	0.630	0.710	**0.768**	0.641	0.740
PC1	Ant-1.7	**0.743**	0.675	0.717	0.618	**0.778**	0.582	0.598	0.612	**0.772**	0.651	0.494	0.708
PC3	Ant-1.7	**0.745**	0.708	0.704	0.676	**0.751**	0.732	0.604	0.702	0.715	**0.743**	0.392	0.743
PC4	Ant-1.7	0.616	0.560	**0.670**	0.659	**0.701**	0.576	0.553	0.584	0.634	0.579	0.506	**0.715**
CM1	Camel-1.6	**0.606**	0.588	0.547	0.589	**0.611**	0.585	0.588	0.538	**0.636**	0.568	0.573	0.558
MW1	Camel-1.6	0.584	0.546	**0.594**	0.582	0.592	0.597	0.596	**0.600**	0.624	0.610	0.555	**0.629**
PC1	Camel-1.6	0.529	0.542	**0.576**	0.567	**0.596**	0.489	0.542	0.583	0.572	0.535	0.498	**0.617**
PC3	Camel-1.6	0.580	0.571	0.584	**0.586**	**0.606**	0.587	0.547	0.573	0.615	0.594	0.455	**0.622**
PC4	Camel-1.6	0.552	0.467	**0.569**	0.567	0.519	0.422	0.534	**0.571**	0.498	0.414	0.517	**0.544**
CM1	Ivy-2.0	**0.691**	0.649	0.665	0.641	**0.792**	0.706	0.694	0.643	0.578	**0.738**	0.661	0.708
MW1	Ivy-2.0	0.749	0.607	**0.764**	0.706	**0.807**	0.700	0.695	0.688	0.496	0.717	0.619	**0.772**
PC1	Ivy-2.0	0.690	0.556	**0.719**	0.641	0.682	0.518	0.602	**0.723**	0.506	0.514	0.517	**0.769**
PC3	Ivy-2.0	0.684	0.658	**0.715**	0.682	0.603	0.669	0.649	**0.740**	**0.765**	0.732	0.405	0.723
PC4	Ivy-2.0	0.625	0.605	**0.655**	0.612	0.636	0.540	0.595	**0.669**	0.637	0.534	0.516	**0.687**
CM1	Jedit-4.3	0.594	0.538	0.628	**0.717**	**0.667**	0.639	0.561	0.545	0.569	0.578	0.552	**0.711**
MW1	Jedit-4.3	0.612	**0.719**	0.626	0.613	0.611	**0.626**	0.543	0.606	0.614	0.616	0.537	**0.707**
PC1	Jedit-4.3	0.594	0.547	0.601	**0.602**	0.563	0.537	0.567	**0.661**	0.592	0.460	0.524	**0.743**
PC3	Jedit-4.3	0.598	0.591	0.580	**0.647**	0.585	0.603	0.514	**0.636**	0.593	0.637	0.477	**0.670**
PC4	Jedit-4.3	0.520	0.545	0.540	**0.579**	0.514	0.551	0.521	**0.555**	0.522	0.555	0.511	**0.703**
CM1	Synapse-1.2	**0.646**	**0.646**	0.605	0.586	**0.666**	0.653	0.638	0.619	**0.740**	0.719	0.655	0.648
MW1	Synapse-1.2	0.655	0.655	**0.693**	0.603	0.715	**0.729**	0.651	0.633	0.634	**0.728**	0.583	0.667
PC1	Synapse-1.2	0.656	0.660	**0.661**	0.628	0.582	**0.646**	0.571	0.639	0.446	**0.703**	0.494	0.685
PC3	Synapse-1.2	0.666	**0.680**	0.649	0.658	0.580	**0.691**	0.600	0.575	0.573	**0.700**	0.410	0.632
PC4	Synapse-1.2	**0.633**	—	0.630	0.617	**0.674**	—	0.563	0.618	0.591	—	0.542	**0.711**
CM1	Velocity-1.6	0.647	0.605	0.600	**0.657**	**0.714**	0.621	0.624	0.569	**0.703**	0.627	0.611	0.663
MW1	Velocity-1.6	0.643	0.623	**0.669**	0.629	**0.736**	0.734	0.669	0.626	0.722	**0.729**	0.566	0.680
PC1	Velocity-1.6	0.577	**0.650**	0.642	0.614	**0.748**	0.576	0.579	0.570	**0.767**	0.643	0.495	0.679
PC3	Velocity-1.6	0.663	**0.721**	0.634	0.657	0.699	**0.719**	0.599	0.656	0.729	**0.736**	0.431	0.686
PC4	Velocity-1.6	0.562	0.546	0.584	**0.628**	**0.615**	**0.615**	0.551	0.545	0.585	0.604	0.515	**0.670**
CM1	Xalan-2.4	0.662	0.621	0.657	**0.687**	0.666	0.520	**0.671**	0.648	0.608	0.665	0.635	**0.684**
MW1	Xalan-2.4	0.683	0.639	**0.726**	0.644	**0.741**	0.723	0.677	0.634	0.623	0.721	0.607	**0.726**
PC1	Xalan-2.4	0.675	0.627	**0.687**	0.651	**0.717**	0.601	0.600	0.682	0.713	0.625	0.513	**0.717**
PC3	Xalan-2.4	0.670	0.671	**0.679**	0.590	**0.667**	0.667	0.605	0.591	0.637	0.679	0.392	**0.716**
PC4	Xalan-2.4	0.593	0.563	**0.648**	0.507	**0.651**	0.584	0.560	0.566	0.562	0.589	0.525	**0.703**
Std		0.084	0.093	0.076	**0.051**	0.060	0.063	0.057	**0.046**	0.080	0.081	0.053	**0.052**
Average		0.640	0.616	**0.647**	0.626	**0.667**	0.624	0.601	0.620	0.628	0.642	0.531	**0.686**

Second, although the proposed method maintains the AUC value above 0.62 on different models, different machine learning models will have a certain impact on the prediction performance. Hence, more machine learning methods will be applied to choose a more suitable prediction model construction method. Moreover, it is necessary to explore different metric matching and data distribution transformation strategies to further improve the versatility and accuracy of the prediction model.

Finally, in many cases, there are fewer defective samples than nondefective samples in defect dataset. This phenomenon will lead to class imbalance problem and may affect the performance of the prediction model, so the next step will try to improve the overall prediction performance from the perspective of how to solve the class imbalance problem.

5. Conclusions and Future Work

In this study, a new CCDP method based on metric matching and sample weight setting is proposed to further improve the defect prediction performance, thereby improving the software security and reliability. The main contributions are as follows:

(1) A clustering-based metric matching algorithm is proposed first. The multigranularity metric feature vector is extracted to unify metric dimension. Moreover, metric clustering is applied to eliminate the metric redundancy problem, and the representative metric is extracted to facilitate the subsequent one-to-one metric matching. This method not only unifies the metrics and eliminates the impact of redundant metrics but also has no restrictions on the scale of the source and target projects. Furthermore, this method approximately converted the heterogeneity problem in CCDP into a homogeneous issue, which has certain reference value for solving heterogeneous situations in other fields.

(2) A sample selection-based weight setting algorithm is proposed to reduce the differences in data

distribution of different projects. Based on the metric matching results, the selection frequency information of source samples is obtained through the metric similarity measure as an influence factor to increase the weight of source samples that are more similar to target samples. This can further improve the data distribution similarity between the source and target projects, thereby improving the prediction accuracy.

(3) Based on the above key technologies, extensive experiments are conducted to demonstrate the feasibility and effectiveness of the proposed strategies and overall method. Experimental results prove that the proposed method has superior prediction performance over other mainstream CCDP methods.

However, there are still some open problems, such as only part of NASA and PROMISE datasets, are used for performance verification. In the future, more defect datasets will be collected to verify effectiveness of this method. Moreover, the metric clustering operation does not consider the impact of irrelevant metrics in the project when constructing the prediction model, which may affect the defect prediction accuracy to a certain extent. Those issues will also be solved in the future work to further improve the security and reliability of large-scale software.

Data Availability

Data will be available in the following link: https://www.researchgate.net/search/publication?q=NASA%20MDP.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors appreciate the support from the Zhejiang Provincial Natural Science Foundation of China (LY20F020015 and LY21F020015), the National Science Foundation of China (61702517, 61972121, 61902345, and 61772525), the Defense Industrial Technology Development Program (no. JCKY2019415C001), the Open Project Program of the State Key Lab of CAD&CG (Grant no. 2109), and Zhejiang University.

References

[1] N. J. Pizzi and J. Nick, "A fuzzy classifier approach to estimating software quality," *Information Sciences*, vol. 241, pp. 1–11, 2013.

[2] L. He, Q. B. Song, and J. Y. Shen, "Boosting-based k-NN learning for software defect prediction," *Moshi Shibie yu Rengong Zhineng/Pattern Recognition and Artificial Intelligence*, vol. 25, pp. 792–802, 2012.

[3] Z. Xu, S. Li, J. Liu et al., "LDFR: learning deep feature representation for software defect prediction," *Journal of Systems and Software*, vol. 158, 2019.

[4] S. K. Pandey, R. B. Mishra, and A. K. Tripathi, "BPDET: an effective software bug prediction model using deep representation and ensemble learning techniques," *Expert Systems with Applications*, vol. 144, Article ID 113085, 2019.

[5] L. Zhao, Z. Shang, L. Zhao, T. Zhang, and Y. Y. Tang, "Software defect prediction via cost-sensitive siamese parallel fully-connected neural networks," *Neurocomputing*, vol. 352, pp. 64–74, 2019.

[6] M. Tim, M. Zach, T. Burhak, C. Bojan, J. Yue, and B. Ayse, "Defect prediction from static code features: current results, limitations, new approaches," *Automated Software Engineering*, vol. 17, pp. 375–407, 2010.

[7] M. D'Ambros, M. Lanza, and R. Robbes, "Evaluating defect prediction approaches: a benchmark and an extensive comparison," *Empirical Software Engineering*, vol. 17, pp. 531–577, 2012.

[8] X. Yang, S. Zhou, and M. Cao, "An approach to alleviate the sparsity problem of hybrid collaborative filtering based recommendations: the product-attribute perspective from user reviews," *Mobile Networks and Applications*, vol. 25, pp. 376–390, 2019.

[9] S. Herbold, A. Trautsch, and J. Grabowski, "A comparative study to benchmark cross-project defect prediction approaches," *IEEE Transactions on Software Engineering*, vol. 44, no. 9, pp. 811–833, 2017.

[10] T. Zimmermann, N. Nagappan, H. Gall, E. Giger, and B. Murphy, "Cross-project defect prediction: a large scale experiment on data vs. domain vs. process," in *Proceedings of the Joint Meeting of the European Software Engineering Conference & the Acm Sigsoft Symposium on the Foundations of Software Engineering*, Amsterdam, Netherlands, August 2009.

[11] F. Rahman et al., "How, and why, process metrics are better," in *Proceedings of the 35th International Conference on Software Engineering*, pp. 432–441, ICSE), Francisco, CA, USA, May 2013.

[12] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Transactions on Neural Networks*, vol. 22, no. 2, pp. 199–210, 2011.

[13] J. Nam, S. J. Pan, and S. Kim, "Transfer defect learning," in *Proceedings of the 2013 35th International Conference Software Engineering (ICSE)*, May 2013.

[14] C. Liu, D. Yang, X. Xia, M. Yan, and X. Zheng, "A two-phase transfer learning model for cross-project defect prediction," *Information and Software Technology*, vol. 107, 2018.

[15] C. Jinyin, K. Hu, Y. Yang, Y. Liu, and Q. Xuan, "Collective transfer learning for defect rediction," *Neurocomputing*, vol. 416, pp. 103–116, 2020.

[16] Z. Sun, J. Li, H. Sun, and L. He, "CFPS: collaborative filtering based source projects selection for cross-project defect prediction," *Applied Soft Computing*, vol. 99, Article ID 106940, 2020.

[17] C. Jin, "Cross-project software defect prediction based on domain adaptation learning and optimization," *Expert Systems with Applications*, vol. 117, Article ID 114637, 2021.

[18] T. Menzies, J. Greenwald, and A. Frank, "Data mining static code attributes to learn defect predictors," *IEEE Transactions on Software Engineering*, vol. 33, no. 1, pp. 2–13, 2007.

[19] S. L. Liu, X. Chen, and W. S. Liu, "FECAR : A feature selection framework for software defect prediction," in *Proceedings of the Annual Computer Software and Applications Conference*, pp. 426–435, Vasteras, Sweden, July 2014.

[20] Q Wang, J. Zhu, and B. Yu, *Feature Selection and Clustering in Software Quality Prediction*, Ease, East Sussex, United Kingdom, 2007.

[21] Z. Sun, Q. Song, and X. Zhu, "Using coding-based ensemble learning to improve software defect prediction," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1806–1817, 2012.

[22] X. Y. Jing, F. Wu, X. Dong, and B. Xu, "An improved SDA based defect prediction framework for both within-project and cross-project class-imbalance problems," *IEEE Transactions on Software Engineering*, vol. 43, no. 4, 2016.

[23] B. Turhan, T. Menzies, A. B. Bener, and J. Di Stefano, "On the relative value of cross-company and within-company data for defect prediction," *Empirical Software Engineering*, vol. 14, no. 5, pp. 540–578, 2009.

[24] F. Peters, T. Menzies, and A. Marcus, "Better cross company defect prediction," in *Proceedings of the Mining Software Repositories (MSR), 2013 10th IEEE Working Conference on IEEE*, San Francisco, CA, USA, May 2013.

[25] Y. Ma, G. Luo, X. Zeng, and A. Chen, "Transfer learning for cross-company software defect prediction," *Information and Software Technology*, vol. 54, no. 3, pp. 248–256, 2012.

[26] X. Jing, W. Fei, D. Xiwei, Q. Fumin, and X. Baowen, "ACM Press the 2015 10th Joint Meeting - Bergamo, Italy (2015.08.30-2015.09.04)] Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering - ESEC/FSE 2015 - Heterogeneous cross-company defect prediction by unified metric representation and CCA-based transfer learning," *Joint Meeting*, pp. 496–507, 2015.

[27] M. Ying, Z. Shunzhi, C. Yumin, and L. Jingjing, "Kernel CCA based transfer learning for software defect prediction," *IEICE Transactions on Information and Systems*, vol. 8, pp. 1903–1906, 2017.

[28] J. Nam, W. Fu, S. Kim, T. Menzies, and L. Tan, "Heterogeneous defect prediction," *IEEE Transactions on Software Engineering*, vol. 44, no. 9, pp. 874–896, 2017.

[29] Q. Yu, S. Jiang, and Y. Zhang, "A feature matching and transfer approach for cross-company defect prediction," *Journal of Systems and Software*, Article ID S0164121217301346, 2017.

[30] D. Whitlark and G. H. Dunteman, "Principal components analysis," *Journal of Marketing Research*, vol. 27, no. 2, p. 243, 1990.

[31] L. Chen, B. Fang, Z. Shang, and Y. Tang, "Negative samples reduction in cross-company software defects prediction," *Information and Software Technology*, vol. 62, pp. 67–77, 2015.

[32] L. C. Briand, W. L. Melo, and J. Wust, "Assessing the applicability of fault-proneness models across object-oriented software projects," *IEEE Transactions on Software Engineering*, vol. 28, no. 7, pp. 706–720, 2002.

[33] D. Lewis, *Naive (Bayes) at Forty: The Independence Assumption in Information Retrieval*, Springer, New York, NY, USA, 1998.

[34] W. Afzal, "Using faults-slip-through metric as a predictor of fault-proneness," in *Proceedings of the 2010 Asia Pacific Software Engineering Conference*, Sydney, Australia, December 2011.

[35] G. Czibula, Z. Marian, and I. G. Czibula, "Software defect prediction using relational association rule mining," *Information Sciences*, vol. 264, pp. 260–278, 2014.

[36] Y. Jiang, J. Lin, B. Cukic, and T. Menzies, "Variance analysis in software fault prediction models," in *Proceedings of the 2009. ISSRE '09. 20th International Symposium on Software Reliability Engineering*, Bengaluru-Mysuru, India, November 2009.

[37] M. Shepperd, Q. Song, Z. Sun, and C. Mair, "Data quality: some comments on the NASA software defect datasets," *IEEE Transactions on Software Engineering*, vol. 39, no. 9, pp. 1208–1215, 2013.

[38] T. J. Mccabe, "A complexity measure," *IEEE Transactions on Software Engineering*, vol. 2, pp. 308–320, 2006.

[39] I. Chowdhury and M. Zulkernine, *Using Complexity, Coupling, and Cohesion Metrics as Early Indicators of Vulnerabilities*, Queen's University, Doctoral dissertation, Kingston, Canada.

[40] S. Moshtari, A. Sami, and M. Azimi, "Using complexity metrics to improve software security," *Computer Fraud & Security*, vol. 2013, no. 5, pp. 8–17, 2013.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6688393, 13 pages
https://doi.org/10.1155/2021/6688393

Research Article

CF Model: A Coarse-to-Fine Model Based on Two-Level Local Search for Image Copy-Move Forgery Detection

Fang Mei,[1,2] **Tianchang Gao**,[2,3] **and Yingda Lyu**[2,4]

[1]*College of Computer Science and Technology, Jilin University, Changchun 130012, China*
[2]*Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun 130012, China*
[3]*College of Software, Jilin University, Changchun 130012, China*
[4]*Public Computer Education and Research Center, Jilin University, Changchun 130012, China*

Correspondence should be addressed to Yingda Lyu; ydlv@jlu.edu.cn

Received 10 December 2020; Revised 27 March 2021; Accepted 12 April 2021; Published 4 May 2021

Academic Editor: Honghao Gao

Copy-move forgery is the most predominant forgery technique in the field of digital image forgery. Block-based and interest-based are currently the two mainstream categories for copy-move forgery detection methods. However, block-based algorithm lacks the ability to resist affine transformation attacks, and interest point-based algorithm is limited to accurately locate the tampered region. To tackle these challenges, a coarse-to-fine model (CFM) is proposed. By extracting features, affine transformation matrix and detecting forgery regions, the localization of tampered areas from sparse to precise is realized. Specifically, in order to further exactly extract the forged regions and improve performance of the model, a two-level local search algorithm is designed in the refinement stage. In the first level, the image blocks are used as search units for feature matching, and the second level is to refine the edge of the region at pixel level. The method maintains a good balance between the complexity and effectiveness of forgery detection, and the experimental results show that it has a better detection effect than the traditional interest-based copy and move forgery detection method. In addition, CFM method has high robustness on postprocessing operations, such as scaling, rotation, noise, and JPEG compression.

1. Introduction

With the rapid development of technology worldwide, there are many ways to obtain and process images [1]. Evolutions in computer technology, the Internet, and image applications have allowed individuals to tamper easily with image content. Copy-move is the most common means of image forgery, in which a copy of a region is inserted into the same image. Two examples are shown in Figure 1, where the copy-move forgeries are used to enrich image content. Considering scenarios involving the court, news, and so on, it is of paramount importance to determine whether an image is tampered. The purpose of digital image forensics is to verify the authenticity of an image.

As one of the most common means of image tampering, copy-move forgeries may be accompanied by certain postprocessing, including JPEG compression, noise addition, and blurring, to change the image content and confuse the information recipient [2]. In particular, the copied area is often geometrically transformed (rotated, scaled, etc.). Therefore, the passive forensics of copy-move tampered images faces great technical challenges and has a strong practical application value. This paper studies the corresponding passive forensic techniques for copy-move operations.

Our main contributions can be summarized as follows:

(1) This paper proposes a coarse-to-fine model for detecting forged regions by the affine transformation matrix (CFM). The localization of the forged regions from sparse to accurate is achieved.

(2) To further extract the forgery region accurately, a two-stage local search algorithm is designed in the refinement stage to better maintain the balance

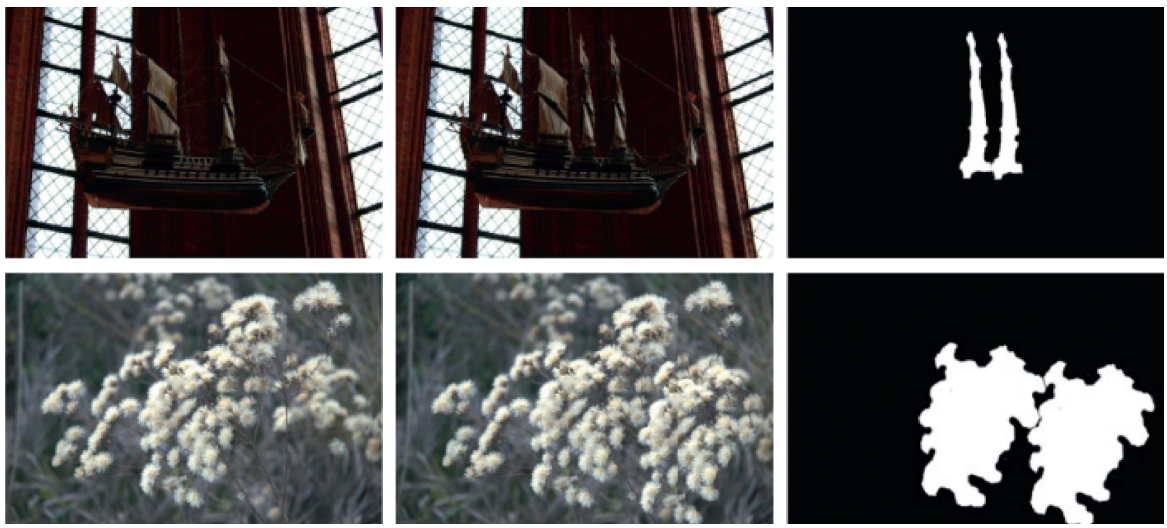


Figure 1: Two examples of the copy-move forgery. Left to right: original images, forged images through copy-move operations, and copy-move regions.

between complexity and effectiveness of forgery detection.

(3) The method has better detection results and higher robustness to postprocessing operations such as scaling, rotation, noise, and JPEG compression.

2. Related Work

Numerous methods for copy-move forgery detection (CMFD) have been proposed in the last decade, which are traditionally categorized into two classes: block-based and interest point-based methods.

2.1. Block-Based CMFD. In 2003, Fridrich [3] proposed the first CMFD algorithm which divided an input image into overlapping blocks to yield similar block pairs and used discrete cosine transform (DCT) to describe image blocks. LBP is a grey-scale texture operator which is used to describe the spatial structure of the image texture. Wang et al. [4] extracted Quaternion Exponent Moment (QEM) moduli from each overlapped circular color block. The main limitation of this method is the higher computational complexity, which can be reduced by applying super pixel theory. Chen et al. [5] proposed a scheme to detect copy-move regions through the invariant features extracted from each block, and each block was only compared with other blocks under the intersection of closed mean and variance features. Mahmood et al. [6] divided the approximation sub-band of the shift invariant stationary wavelet transform into overlapping blocks. Distinct features extracted from the overlapping blocks were used to expose tampered regions forged in digital images. The features of these algorithms can be classified as follows: invariant moments, dimension reduction, textural features, and polar transform. Matching techniques include dictionary sorting and Euclidean distance [7]. However, most algorithms based on image blocks do not perform well in resisting affine transformation attacks.

2.2. Interest Point-Based CMFD. Different from block-based algorithms, interest point-based CMFD algorithms are more robust against affine transformations. Unlike dividing an image, this method extracts interest points on the image, and image features are then extracted around the interest points. He et al. [8] used PCA on the feature vector to reduce computational complexity. Mohamadian and Pouyan [9] combined SIFT and Zernike moments to reduce the potential of being unable to detect tampered regions in flat regions. Pun et al. [10] proposed a novel CMFD scheme using adaptive oversegmentation and feature point matching, which integrates block-based and interest point-based forgery detection methods. Pandey et al. [11] proposed a fast and effective copy-move forgery detection algorithm through hierarchical feature point matching. Due to the high stability of intermediate and postprocessing operations, the SIFT method has been widely used in CMFD. To improve SIFT performance, Bay et al. [12] initially proposed the speeded-up robust features (SURF) technique. The SURF operator maintains the excellent performance of the SIFT operator but addresses the shortcomings of high computational complexity and time consumption. Bo et al. [13] proposed a CMFD technique based on SURF and extended the dimensions of Bay's techniques to 128 to reduce false matching. Many scholars have only used this technique in interest point detection to produce feature points, after which local features were employed to describe an interest point to achieve satisfactory results [14, 15]. Mishra et al. [16] presented a detection method based on the combination between speeded-up robust features (SURF) and hierarchical agglomerative clustering (HAC). Zandi et al. [17] proposed a new interest point detector that leverages the advantages of block-based and traditional interest point-based methods and uses improved strategies to implement

the algorithm. However, because the interest points are comparatively few and scattered, interest point-based detection methods can encounter difficulties in locating a precise forged region.

The block-based CMFD algorithm and interest point-based CMFD algorithm each have a similar framework as depicted in Figure 2 [18].

(i) Preprocessing: its main purpose is to eliminate irrelevant information in the image and restore useful real information; the most common approach is to convert the image from an RGB version to a grayscale image

(ii) Feature extraction: local image information is extracted from an image block or interest point represented by a feature descriptor

(iii) Matching: similar pairs of image blocks or points are determined during the matching process

Most existing algorithms based on image blocks suffer from some attacks, such as scaling, rotation, and noise addition, and interest point-based methods cannot locate the tampered region precisely. To solve these problems, a hybrid two-level method combining image blocks and interest points is proposed in this paper. We chose the SIFT as the feature descriptor to represent the interest point. Then, the adaptive oversegmentation method is used to improve the matching process and calculate the affine transformation matrix. Finally, the proposed local search algorithm is applied to image block level and pixel level, respectively, to locate the tampered region accurately.

3. Proposed Detection Algorithm

In this paper, an accurate CMFD method based on interest point and local search algorithm is proposed. The process is illustrated in Figure 3.

The main flow of the proposed algorithm is as follows: (1) feature extraction: interest points are detected in the input image represented by a feature descriptor, after which accurate interest point matches are obtained via a matching process; (2) affine transformation calculation: utilize a random verification algorithm to calculate the affine transformation matrix; (3) forgery region extraction: local search algorithm is applied to the image block level and the pixel level. The image block level realizes the location of the tampering region, and the pixel level is used to refine the tampering region boundary.

The image-level detection and pixel-level detection of the proposed model on the testing dataset show promising results. Our main contributions are as follows:

(i) A method combining image blocks and pixels is proposed. Based on the block, the forged region can be located, and the pixel points are used to make the area boundary more refined. This method can make up for the poor performance of only extracting tampered areas with points of interest, thereby improving detection performance.

(ii) Considering the balance between algorithm complexity and performance, design a two-level local search algorithm. In the first stage, the image is divided into small blocks by rectangular blocks. If the image block contains the point of interest, it is marked as a forgery unit and calculated by affine transformation. The search algorithm matches the result to get the forgery region. In the second stage, the boundary of the forged area is extracted at the pixel level, and a secondary search algorithm is used for improvement to further improve the accuracy of model detection.

(iii) Four different postprocessing operations were performed on the test dataset, and the experimental results show that our model still exhibits high robustness.

In the rest of this section, we present the process of this detection algorithm as illustrated in Figure 3. The details of our proposed algorithm are reflected in the following sections: Section 3.1 presents the feature extraction and description along with image segmentation using the adaptive oversegmentation algorithm to prepare for the next matching process. Section 3.2 outlines the feature-matching process using the two nearest neighbor (2NN) algorithm [19]. And then, the affine transformation is calculated. Section 3.3 introduces the local search algorithm. In Section 3.4, two-level local search algorithm using affine transformation matrix is utilized to locate the tampered region accurately. In the first stage, the image blocks are used as search units for feature matching, and the second stage is at the pixel level to refine the edge of the region.

3.1. Feature Extraction and Adaptive Oversegmentation. The first phase of the proposed algorithm involves interest point detection and feature extraction based on SIFT features, referring to local features of an image. SIFT remains invariant to rotation, scaling, and light intensity and maintains stable robustness to changes in the viewing angle, affine transformation, and noise. The interest points and their corresponding descriptors are obtained. Based on these results, the proposed algorithm performs a matching operation to identify similar local regions.

To obtain good performance in matching and calculation of the affine transformation matrix, the adaptive oversegmentation method is adopted [10]. Next, we find corresponding interest point pairs via the feature matching process. In our proposed method, the segmentation algorithm is simple linear iterative clustering (SLIC). SLIC algorithm can generate compact and nearly uniform superpixel, and has high comprehensive evaluation in terms of operation speed, object contour preservation, and superpixel shape, which is more in line with the expected segmentation effect. When the SLIC segmentation method is used, the balance between computational cost and detection precision must be guaranteed. Therefore, the adaptive over segmentation algorithm is adopted to adaptively define the size of superpixels according to the texture of the test images.

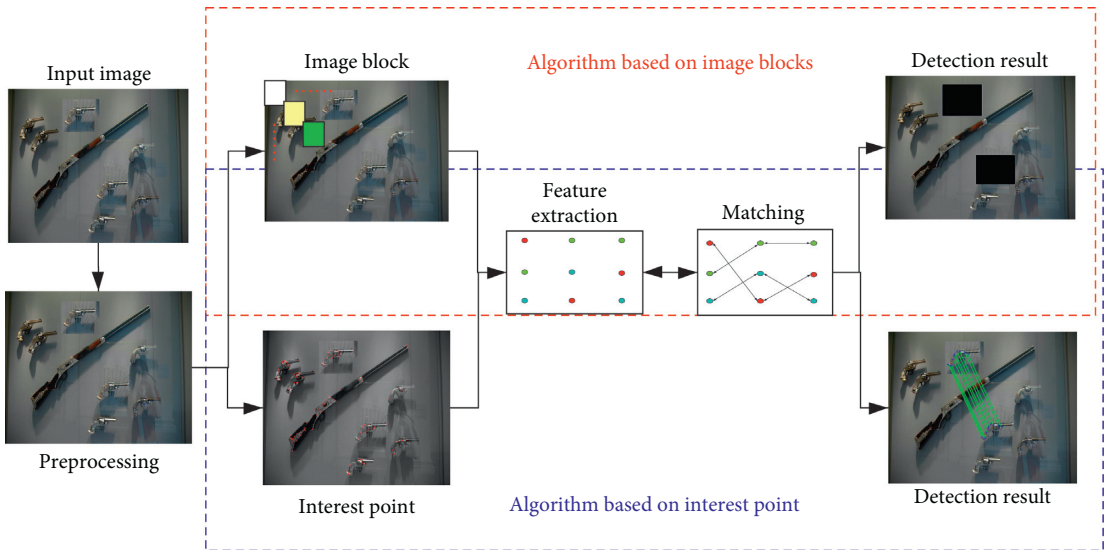


Figure 2: Common copy-move forgery detection framework-based CMFD.

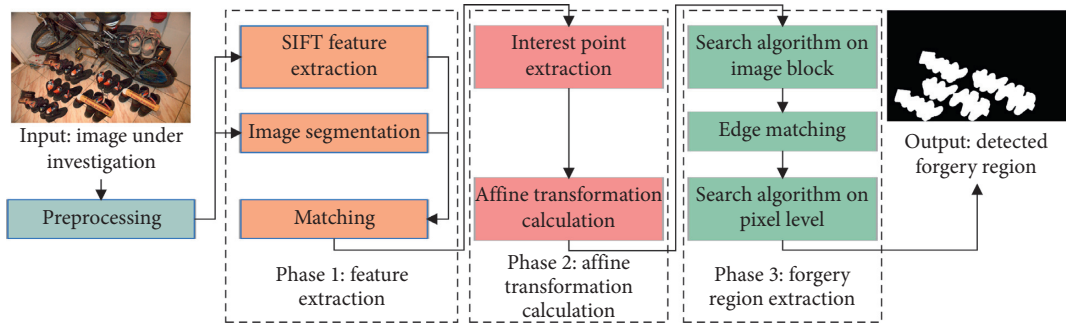


Figure 3: Framework of the proposed copy-move forgery detection method.

Next, a segmented image builds the image blocks set $B = (B_1, B_2, \ldots B_i, \ldots B_{NB})$, where NB is the total number of image blocks; the interest points and feature descriptors in the i^{th} image block are stored in B_i. Figure 4 depicts the relationship of the block set. Then, we find the corresponding interest point pairs via the feature-matching process.

3.2. Interest Point Matching and Affine Transformation Calculation. The 2NN algorithm utilizes the ratio of the distance between the nearest neighbor and the second nearest neighbor. If image blocks B_i and B_j must match, for any feature point, where is the k^{th} point in block B_i, the calculation is as follows:

$$T_b > \frac{T_1}{T_2}, \tag{1}$$

where T_b is the similarity threshold, d_1 is the closest neighbor, and d_2 is the second closest neighbor. The distance d_m is calculated as

$$d_m = \left\| f_i^k - f_j^m \right\|_2, \tag{2}$$

where d_m denotes the distance between point p_i^k and point p_i^m. p_j^m is the m^{th} point in P_j, and f_i^k and f_j^m are the corresponding feature descriptors.

In our experiment, T_b is set to 0.2. If constraint (1) is satisfied, then the inspected interest point p_i^k is matched with p_j^m (p_i^k and p_j^m denotes the interest pairs).

We iterate the 2NN process in different image blocks in our experiment until all blocks have been traversed, resulting in a dataset: $MP = \left\{M_1^2, M_1^3, M_1^4, \ldots M_i^j, \ldots M_{m-1}^m\right\}$, where the interest pairs between B_i and B_j are stored in M_i^j.

Matching operations between image blocks can avoid failed matching due to the proximity of points to coordinates. To further prevent match failure, assuming that M_i^j exists in MP, if the number of point pairs in M_i^j is too small, then the point pairs between the image blocks B_i and B_j are considered a failure and must be deleted. As such,

$$T_p \geq \text{size}(MP[x]), \tag{3}$$

where size () represents the number of point pairs in MP $[x]$ and the threshold T_p is set to 3 to filter the failed pairs. Thus, most missed matches are filtered.

To better display the tampered region, affine transformation matrix T is used to describe the relationship between

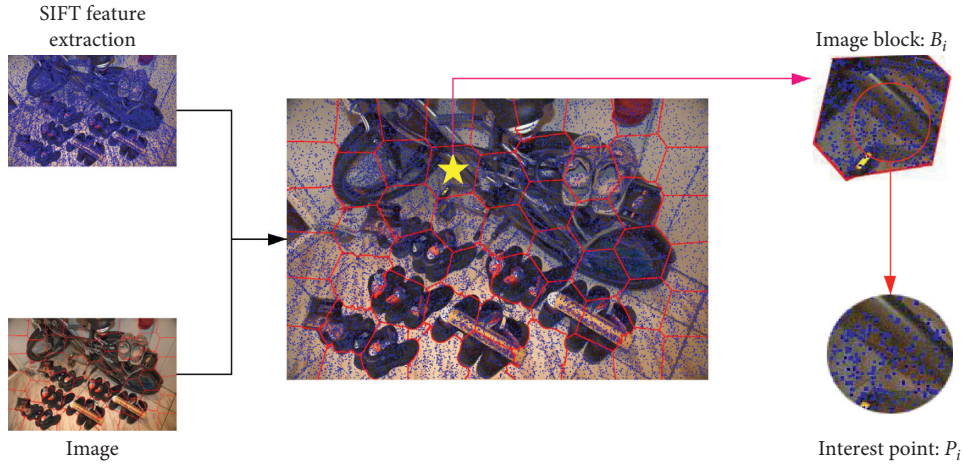


Figure 4: Framework of the proposed copy-move forgery detection method.

the source region and replication regions. The traditional method of estimating affine transformation is not suitable for the algorithm in this paper. In our proposed method, we propose a more efficient matrix estimation algorithm. If M_i^j exists in MP, then we randomly extract three point pairs M_i^j and store them in $C_{\text{matrix}} = \{(p_1, p_1'), (p_2, p_2'), (p_3, p_3')\}$. The affine transformation matrix T is described as follows:

$$\left|y'\right| = T|y|, \tag{4}$$

where the affine transformation matrix T is represented as

$$T = \left| a_3 \;\; a_4 \;\; t_v \right|, \tag{5}$$

where t_x and t_y denote translations and a_1, a_2, a_3, and a_4 are associated with scaling and rotation. C_{matrix} can obtain the affine transformation matrix T.

To verify the accuracy of matrix T, all point pairs in M_i^j must be tested using this matrix. For any interest point pairs (p, p') in M_i^j, point p can obtain the corresponding interest point p' using the following equation:

$$p' = T * p. \tag{6}$$

We verify the matrix accuracy based on the distance between $p_,$ and p'.

$$D_p = \left|x' - x_,\right| + \left|y' - y_,\right| < T_d, \tag{7}$$

where x', y', and x', y' are the coordinates of $p_,$ and p'. T_d is the similarity threshold of the matrix ($T_d = 1.5$ in our experiment). Then, we obtain the number of right point pairs count in M_i^j.

When rate is greater than 0.5, the matrix T is considered correct. In this case,

$$\text{rate} = \frac{\text{count}}{\text{size}}\left(M_i^j\right), \tag{8}$$

where size (M_i^j) is the amount of all point pairs in M_i^j.

In most cases, the source region and replication region may be covered by many image blocks. Many affine transformation matrices can be obtained through MP. We propose an algorithm to deal with this problem. Whenever any set M in MP must be calculated, we must examine the relationship between point pairs in M and existing matrix using formulas (6)–(8). If the label *rate* is more than 0.5, the set M is not to be calculated. Finally, the matrix set is described as follows:

$$T_{\text{end}} = \{T_1, T_2, T_3, \ldots\}. \tag{9}$$

Next, we will display the tampered region in the search algorithm.

3.3. Local Search Algorithm. Extracting the tampered region using only the interest point results in poor performance. By considering the balance between algorithm complexity and performance to more accurately extract the forgery region, we propose a local search algorithm that can be applied at the image block level and pixel level. The role of the local search algorithm is described in Figure 5, where the grid is used to replace the test image, the region outlined in red is the forged region, and the blue small block is the forged unit; when the first search algorithm is used, the forged unit is an image block, and the second forged unit is a pixel. Details of the search algorithm are provided in the following section.

The detection unit can find a corresponding unit via the affine transformation matrix, which is key to the local search algorithm. The detected unit can find corresponding unit through the matrix. Before executing the search algorithm, the forged units must be collated and added to the forgery region set (TR). Then, the local search algorithm is executed; steps are shown in Algorithm 1.

TR_{cnt} is the result of the current detection, D_{nei} is the set of neighborhood $p_{\text{nei}} = \{p_1, p_2, p_3, p_4\}$, and (1, 2, 3, 4) denotes four angles (0°, 90°, 180°, and 270°). Notably, the detection unit in p_{nei} may be the detected element; therefore, the detected elements in p_{nei} must be deleted. Then, the corresponding unit p_i is calculated by matrix T, and feature descriptors are used to measure the similarity. These descriptors are explained in detail in the following section.

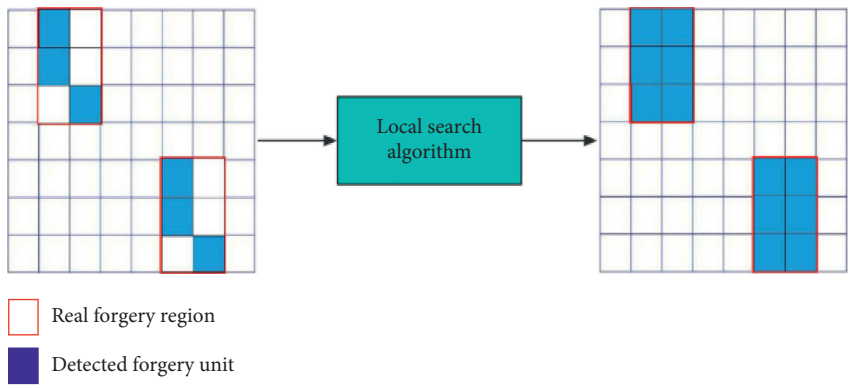


Figure 5: Role of local search algorithm. The region outlined in red is the forged region, and the blue small block is the forged unit.

The successfully matched unit pairs are added to $TR_{\text{cnt}+1}$. This operation is iterated until all elements in TR_{cnt} have been detected. Finally, the test result $TR_{\text{cnt}+1}$ is combined with the original result $TR_{\text{cnt}+1}$, and we obtain the final result $TR_{\text{cnt}+1} = TR_{\text{cnt}+1}$ U TR_{cnt}. To understand the algorithm flow and prove the validity of the local search algorithm, a flow chart is used for descriptive purposes (Figure 6).

Figure 6 presents the ordinary flow of the local search algorithm. There are only six forged units (a, b, c, a′, b′, and c′) at the beginning of the algorithm; the forged region is not completely covered. Implementation steps of the algorithm are described in Figure 6, where the blue blocks are forged units, green tags stand for detecting units, red blocks are nonforged units, and white blocks are units that have not been detected. Assume that there is only one affine transformation matrix T, and the final result was shown.

3.4. Tampered Region Localization. To balance the complexity and accuracy of the algorithm, the two-stage local search algorithm is proposed: the image block level. And, the second stage is at the pixel level to refine the edge of the tampered region. The framework of the algorithm is displayed in Figure 7.

3.4.1. The First Stage. In our method, interest points in the *MP* are extracted and stored in P_{right}. First, a small, nonoverlapping rectangular block is used to cover the host image, and all image blocks are scanned. If the image block contains interest points in P_{right}, the block is marked as a forged unit. Then, the image blocks as a detection unit are added to TR_0, and the search algorithm is employed on the image block level. Corresponding image blocks are calculated by the affine transformation T. Assume that image block B_i calculates corresponding image block B_i; in this case, image block B_i cannot reach the center of another block (B_i) and needs to extract the true matching image block B_i, so feature comparison must be executed between B_i and B_i. Then, the ZNCC (zero-based normalized cross-correlation) should be calculated between B_i and B_i as follows:

$$C(x) = \frac{\sum_{u\in B_i}(I(u) - \overline{I})(I''(u) - \overline{I''})}{\sqrt{\sum_{u\in B_i}(I(u) - \overline{I})^2}\sqrt{\sum_{u\in B_i''}(I''(u) - \overline{I''})^2}}, \quad (10)$$

where $I(u)$ and $I''(u)$ denote pixel intensities at location u, and I and I are the average pixel intensities of B_j and B. We apply a Gaussian filter of 7×7 pixels with a standard deviation of 0.5 to reduce noise; the threshold (T_{RD}) is set up to obtain similar image block pairs:

$$C(x) \geq T_{R\,D}. \quad (11)$$

In our work, T_{RD} is set to 0.55 once formula (11) has been calculated. The two image blocks $(B_i$ and $B_j)$ are similar, and the results of the search algorithm are stored in TR_1.

A filtering algorithm is used to render the test results more accurate. For each forged unit in TR_1, the neighbor of detection element D must be extracted, and the neighboring blocks are defined as $D_{\text{nei}} = \{d_0, d_1, d_2, d_3, d_4, d_5, d_6, d_7\}$. In our experiment, if the number of forged units in D_{nei} is less than 2, the detection element D is deleted.

3.4.2. The Second Stage. It is challenging to extract the forgery region at the image block level, and the algorithm does not have good performance at the edge of the tampered region. Thus, the edge of TR_1 is extracted, and we obtain an edge region ER_0 and a center region CR_1 on the image block level, where ER_0 is considered inaccurate and CR_1 is accurate. In matrix T, all pixels in ER_1 must be calculated. For the obtained pixel pairs, the ZNCC algorithm is used to measure similarities, and the threshold (TDR) is set to 0.55. The matching result is saved in ER_1, from which, forgery region TR_2 is obtained by combining the center region CR_1 and the matching result ER_1. To improve the edge of the forged region, the edge of ER_2 is extracted at pixel level in TR_2, and ER_2 is used to execute

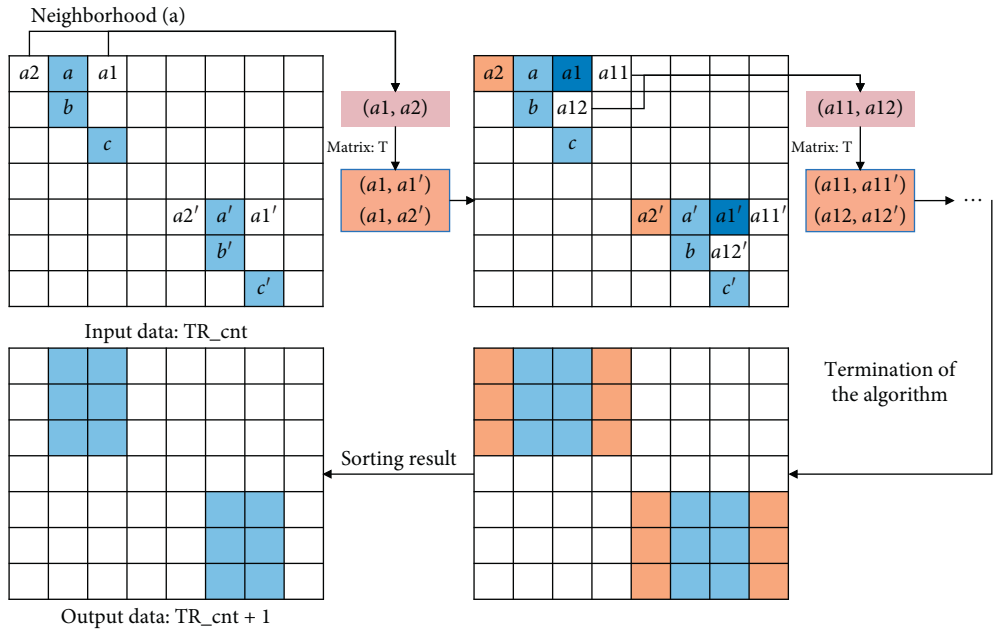


Figure 6: Framework of the local search algorithm. The blue blocks are forged units, green tags stand for detecting units, red blocks are nonforged units, and white blocks are nondetected units.

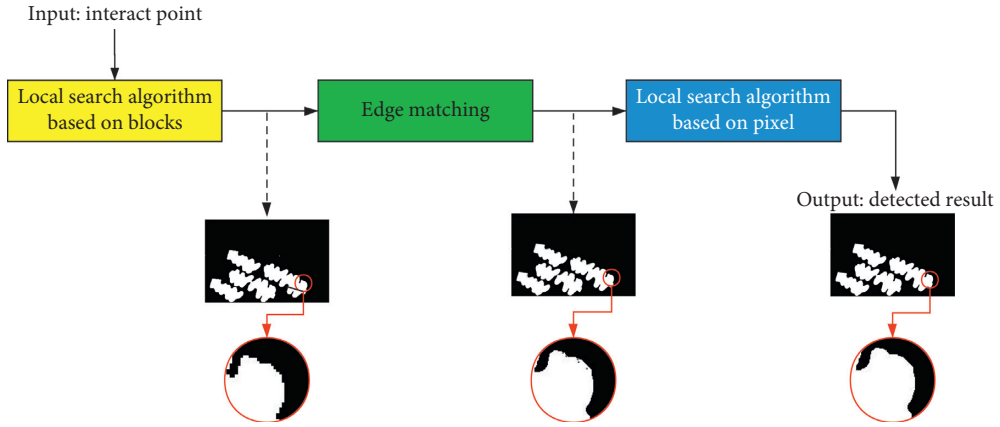


Figure 7: Framework of two-stage tampered region localization algorithm.

local search algorithm. Assume that we get (I, I') by matrix T; the color feature should be extracted, respectively, between I and I' as follows:

$$F_I = \frac{R(E_I) + G(E_I) + B(E_I)}{3},$$
$$F'_I = \frac{R(E_{I'}) + G(E_{I'}) + B(E_{I'})}{3}, \tag{12}$$

where R (), G (), and B () are three color channels of the detected image unit; F_I, $F_{I'}$ are the color features of I and I'; and if feature F_I and $F_{I'}$ conform to formula (11), matching is successful between unit I and I'.

$$\left|F_I - F_{I'}\right| \le T_{RD^2}, \tag{13}$$

where T_{RD^2} is the degree of similarity between I and I'. In our work, T_{RD^2} is 0.5. Results are stored in ER_3.

The tampered region TR_2 is obtained by combining ER_3 and center region CR_2. After the filtering step, the morphological close operation is applied to TR_3 to eliminate small gaps, after which the tampered region TR_{end} is generated. The algorithm is evaluated in the following section to demonstrate its effectiveness.

4. Experimental Results

In this section, a series of experiments are conducted to evaluate the performance of the proposed CMFD method. Section 4.1 introduces the image dataset used in our experiments and the evaluation criteria used to evaluate the performance of the proposed method. Section 4.2 shows the experimental results of the proposed algorithm. In section 4.3, the experimental results of the proposed CMFD method were finally compared with existing state-of-the-art CMFD

methods under different transforms, and the results of comparative analysis were outlined.

4.1. Datasets and Evaluation Criteria. In the following experiments, a benchmark database [20] that includes realistic copy-move forgeries was used to test the proposed scheme. This image dataset included 48 source images along with manually prepared per image, semantically meaningful regions to be copied. Each image measured 3000×2300 pixels. Forgery regions comprised approximately 10% of each image. The copied regions belonged to the categories of living, natural, artificial, and mixed textures ranging from smooth to complex. Transformed images, such as those that underwent rotation, scaling, JPEG artifacts, and added noise, were also included in the image dataset.

To quantitatively evaluate the detection performance, we adopted two metrics: precision and recall. Precision is the fraction of pixels identified as forgery that are truly forgery, defined as the ratio of the number of correctly detected forged pixels to the total number of detected forged pixels. Recall refers to the fraction of forged pixels that are correctly classified, defined as the ratio of the number of correctly detected forged pixels to the number of forged pixels in the ground truth forgery image. Precision and Recall are calculated using (14) and (16), where Ω denotes the set of the detected forged regions in forged images with the CMFD method at the pixel level and Ω' denotes the forged regions of the ground-truth of forged images. We provide the F_i score as a measure that combines precision and recall in a single value.

Using these metrics, we show how precisely the CMFD algorithms identified tampered regions. To reduce the effects of random samples, the average precision and recall were computed for all images in the dataset.

4.2. Experimental Results of the Proposed Algorithm

4.2.1. Experimental Results on Plain Copy-Move Forgery. Plain copy-move forgery is a kind of one-to-one copy-move method that does not involve other transformation operations. It is to cut the local area of the target image and then paste it into the target image again through rotation, scaling, and other operations to generate a new tampered image. We experimented on 48 plain copy-move forgery images in total. Figure 8 displays eight copy-move forgery detection results for the plain copy-move forgery, and the forgery content is either smooth (e.g., sky), rough (e.g., rocks), or structured (typically man-made buildings). From top to bottom are test images and corresponding ground-truth forged regions, and the final row is forged region detected by the CFModel. As can be seen from the figure, the proposed model obtains fine prediction masks and even in small forgery region. These groups can be used as categories for CMFD images.

$$\text{precision} = \frac{\left|\Omega \cap \Omega'\right|}{\Omega'}, \tag{14}$$

$$\text{recall} = \frac{\left|\Omega \cap \Omega'\right|}{\Omega'}, \tag{15}$$

4.2.2. Experimental Results under Various Attacks. In addition to one-to-one copy-move forgery, we also experimented on the various attacks to verify the effectiveness of the proposed algorithm.

(i) Scale: the tampered region is rescaled to between 91% and 109% of their original size with 2% step length.

In total, $48 \times 10 = 480$ images are experimented. Figure 9 displays eight copy-move forgery detection results for the scaling, and some scale resizing parameters are included: 91%, 93%, 95%, 97%, 103%, 105%, 107%, and 109%.

(ii) Rotation: the tampered region is rotated at a rotation angle varying from 2° to 10° with a step length of 2°. In total, $48 \times 5 = 240$ images are experimented. Figure 10 shows eight copy-move forgery detection results for the rotation, and some rotation angles, i.e., 2°, 4°, 6°, 8°, and 10°, are considered.

(iii) Gaussian noise: the image intensities of the tampered region is normalized between 0 and 1 with added zero-mean Gaussian noise with standard deviations of 0.02 to 0.10 and a step length of 0.02. In total, $48 \times 5 = 240$ images are experimented. Figure 11 illustrates eight copy-move forgery detection results for noise, and noise standard deviations are included: 0.02, 0.04, 0.06, 0.08, and 0.1.

(iv) JPEG compression: the forged image is JPEG compressed with quality factors varying between 100 and 20 and a step length of 10. In total, $48 \times 9 = 432$ images are experimented. Figure 12 shows eight copy-move forgery detection results for the JPEG, quality factor (QF) which included: 20, 30, 40, 50, 60, 70, 80, and 90.

$$F1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \tag{16}$$

4.3. Comparative Analysis of Algorithms. This section presents the comparison results between CFModel and the existing methods, and experiments on the dataset proposed in [20] including 1488 tampered images. Three recent methods based on SIFT [20] and SURF [20] along with iterative CMFD [17] were selected for comparison.

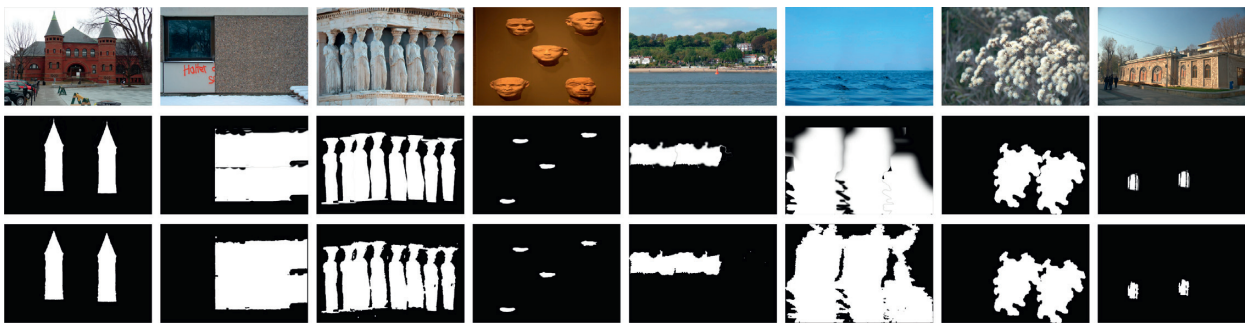


Figure 8: Copy-move forgery detection results under plain copy-move forgery. Top to bottom: test images from dataset, ground-truth forged regions, and forged regions detected by CFModel.

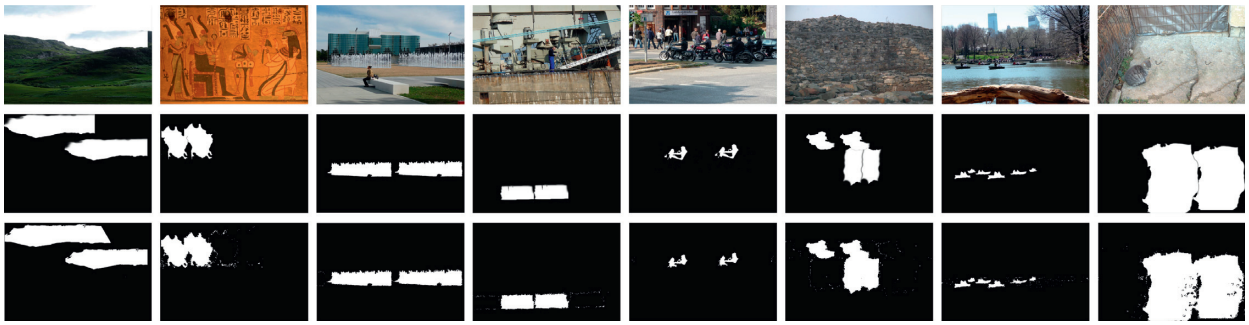


Figure 9: Copy-move forgery detection results for the scale. Top to bottom: test images from dataset, ground-truth forged regions, and forged regions detected by the proposed algorithm. Left to right represents the scale of 91%, 93%, 95%, 97%, 103%, 105%, 107%, and 109%.

Figure 10: Copy-move forgery detection results for the rotation. Top to bottom: test images from dataset, ground-truth forged regions, and forged regions detected by the proposed algorithm. Left to right, the angle of rotation is 2°, 4°, 4°, 6°, 6°, 8°, 8°, and 10°.

Figure 11: Copy-move forgery detection results for noise. Top to bottom: test images from dataset, ground-truth forged regions, and forged regions detected by the proposed algorithm. Left to right: the standard deviation is 0.02, 0.04, 0.04, 0.06, 0.06, 0.08, 0.08, and 0.1.

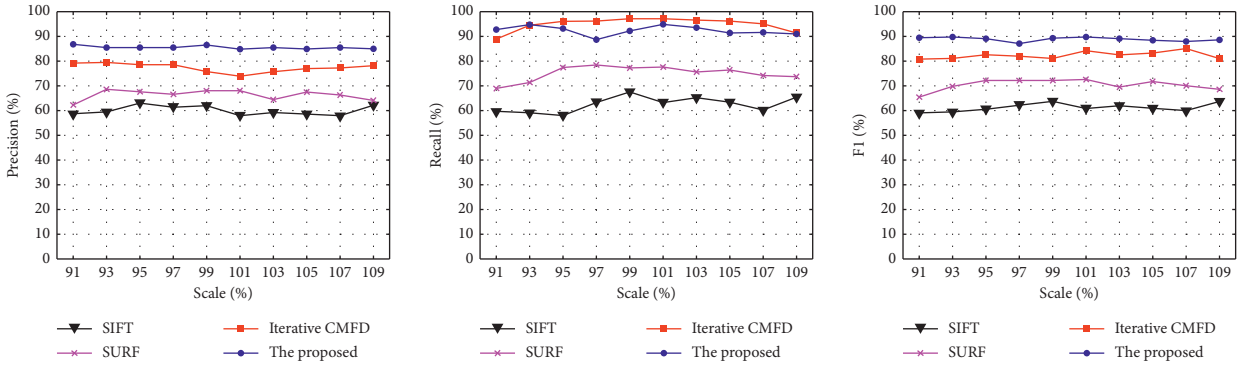
Figure 12: Copy-move forgery detection results for the JPEG. Top to bottom: test images from dataset, ground-truth forged regions, and forged regions detected by the proposed algorithm. Left to right, the quality factor is 20, 30, 40, 50, 60, 70, 80, and 90.

Input: forgery region set (TR_{cnt}) (block or pixel), affine transformation matrix (T_{end})
Output: forgery region set ($TR_{\text{cnt}+1}$)
(1) Detection unit p selected from TR_{cnt} and obtain the neighborhood p_{nei}; elements that have been detected in p_{nei} are deleted. p_{nei} is added to the set D_{nei}.
(2) Nondetection unit p_i is removed from D_{nei} and obtain detection unit p_i' by T. Calculate the similarity between p_i and p_i'; if successful, p_i and p_i' are added to $TR_{\text{cnt}+1}$, and the neighborhood of p_i is added to D_{nei}. Continue to execute step 2 until D_{nei} is empty.
(3) Iterate steps 1 and 2 until all elements in TR_{cnt} have been detected.

Algorithm 1: Local search algorithm.

Table 1: Detection results of plain copy-move forgery at image level.

Image level	Precision (%)	Recall (%)	F_1 (%)
SIFT [20]	88.37	79.17	83.52
SURF [20]	91.47	89.58	90.52
Iterative [17]	67.14	97.91	79.66
Hybrid [21]	78.33	97.92	87.04
CFModel	**97.82**	**93.75**	**95.74**



(a) (b) (c)

Figure 13: Continued.

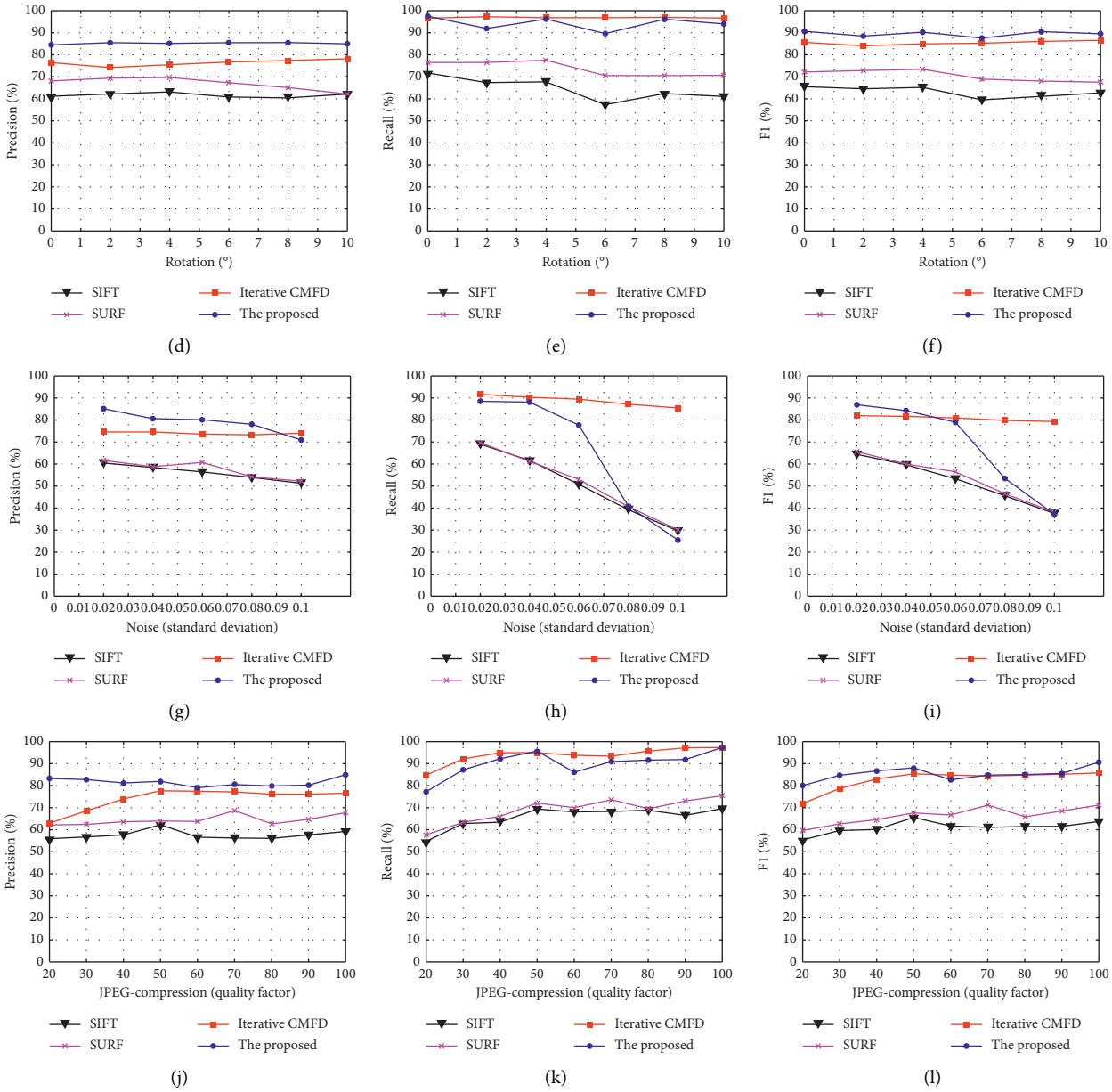


Figure 13: Detection results under various attacks. Top to bottom: scale, rotation, Gaussian noise addition, and JPEG compression. Left to right: precision, recall, and F_1.

4.3.1. Detection Results under Plain Copy-Move Forgery. We first evaluated our algorithm under plain copy-move forgery attack. We experimented on 48 original images and 48 forged images, which are tampered by one-to-one copy-move forgery. Tables 1 and 2 present the results of the evaluation at the image level and pixel level.

As noted in Table 1, the CFModel achieved 97.82% precision and 93.75% recall, better than the most state-of-the-art methods at image level. Our scheme also achieved better performance at the pixel level. As indicated in Table 2, the CFModel achieved up to 84.58% precision and up to 97.41% recall, surpassing most state-of-the-art methods. Compared to Bi [22] and Chen [23], *F1* score is slightly lower than them. The possible reason is that the proposed model is based on block and interest point, which focuses more on recall rate (whether the forged pixel is checked completely and correctly). These results show that the proposed method is more effective than others. Figure 8 also provides the representative results of eight examples. As is shown in the figure, we can see that our proposed algorithm can accurately locate the tampered region even in those small or smooth copy-move regions.

4.3.2. Detection Results under Various Attacks. In order to obtain a more detailed assessment of the discriminative properties of the method, the detailed data of copy-move forgery detection results, experimented on 1392 tampered images under various attacks in total, are shown in Figure 13.

Table 2: Detection results of plain copy-move forgery at pixel level.

Pixel level	Precision (%)	Recall (%)	Fi (%)
SIFT [20]	60.80	71.48	65.71
SURF [20]	68.13	76.43	72.04
Iterative [17]	72.23	96.46	82.61
Hybrid [21]	90.27	78.61	84.04
Bi [22]	—	—	92.87
Chen [23]	—	—	93.92
CFModel	**84.58**	**97.41**	**90.54**

"/" denotes that corresponding results are not provided in the literature.

We use 1392 images in total under different attacks. Figure 13 provides all qualitative results: top to bottom—scale attack, rotation attack, Gaussian noise addition, and JPEG compression; left to right—precision rate, recall rate, and $F1$ score.

As shown in the figure, the precision rate and recall rate of our scheme reached a higher level than other methods, the $F1$ score was particularly prominent under scale indicating that our method provides a good balance of precision and recall. The main reason is that our method proposes a two-stage local search algorithm, which can not only locate the tampered region at the image block level but also locate the edge at the pixel level. In other words, our scheme performed better than most state-of-the-art methods in most cases; however, our method has a very low score when the standard deviation exceeds 0.6, and we will address this deficiency in subsequent work.

5. Conclusion

With the development of digital technology, digital images can be easily forged using image processing software. Forged images must be identified given the potential legal and other implications. In this paper, we propose a copy-move forgery detection algorithm using SIFT as the interest point and feature extraction method. The affine transformation matrix was then calculated, followed by a local search algorithm to locate the forged region. Experimental results show that the proposed scheme performs much better than state-of-the-art copy-move forgery detection algorithms and demonstrates good performance under various attacks. However, performance was poor when images contained noise; we will focus on this image type in later work.

Future research is mainly as follows:

(1) To address the problem that the method cannot adapt to noisy operations, future plans are to incorporate richer texture feature information to achieve better robustness

(2) In future work, we will focus on detection tasks with multiple copy-move tampered regions at the same image to realize practical applications of the detection algorithm

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the National Key Research and Development Program of China (2018YFB080402 and 2018YFB0804203), Regional Joint Fund of NSFC (U19A2057), the National Natural Science Foundation of China (61672259 and 61876070), Jilin Province Science and Technology Development Plan Project (20190303134SF and 20180201064SF), CERNET Innovation Project (NGII20190802), and Undergraduate Innovation and Entrepreneurship Training Program of Jilin University (202010183389).

References

[1] Y. Yin, Z. Cao, Y. Xu, H. Gao, R. Li, and Z. Mai, "QoS prediction for service recommendation with features learning in mobile edge computing environment," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 4, pp. 1136–1145, 2020.

[2] Z. Shi, X. Shen, H. Chen, and Y. Lyu, "Global semantic consistency network for image manipulation detection," *IEEE Signal Processing Letters*, vol. 27, pp. 1755–1759, 2020.

[3] A. J. Fridrich, B. D. Soukal, and A. J. Lukáš, "Detection of copy-move forgery in digital images," in *Proceedings of the 2003 Digital Forensic Research Workshop*, Cleveland, OH, USA, August 2003.

[4] X.-Y. Wang, Y.-N. Liu, H. Xu, P. Wang, and H.-Y. Yang, "Robust copy-move forgery detection using quaternion exponent moments," *Pattern Analysis and Applications*, vol. 21, no. 2, pp. 451–467, 2018.

[5] C.-C. Chen, H. Wang, and C.-S. Lin, "An efficiency enhanced cluster expanding block algorithm for copy-move forgery detection," *Multimedia Tools and Applications*, vol. 76, no. 24, pp. 26503–26522, 2017.

[6] T. Mahmood, Z. Mehmood, M. Shah, and Z. Khan, "An efficient forensic technique for exposing region duplication forgery in digital images," *Applied Intelligence*, vol. 48, no. 7, pp. 1791–1801, 2018.

[7] H. Gao, C. Liu, Y. Li, and X. Yang, "V2VR: reliable hybrid-network-oriented V2V data transmission and routing considering RSUs and connectivity probability," *IEEE Transactions on Intelligent Transportation Systems*, vol. 99, pp. 1–14, 2020.

[8] H. He, X. Huang, and J. Kuang, "Exposing copy move forgeries based on a dimension reduced SIFT method,"

Information Technology Journal, vol. 12, no. 14, pp. 2957–2979, 2013.

[9] Z. Mohamadian and A. A. Pouyan, "Detection of duplication forgery in digital images in uniform and nonuniform regions," in *Proceedings of the 2013 UKSim 15th International Conference on Computer Modelling and Simulation*, pp. 455–460, Cambridge, UK, April 2013.

[10] C. M. Pun, X. C. Yuan, and X. L. Bi, "Image forgery detection using adaptive over-segmentation and feature point matching," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 8, pp. 1705–1716, 2015.

[11] R. C. Pandey, S. K. Singh, K. K. Shukla, and R. Agrawal, "Fast and robust passive copy-move forgery detection using SURF and SIFT image features," in *Proceedings of the 2014 2014 9th International Conference on Industrial and Information Systems (ICIIS)*, pp. 1–6, Gwalior, India, December 2014.

[12] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: speeded up robust features," in *Computer Vision-ECCV 2006*, pp. 404–417, Springer, New York, NY, USA, 2006.

[13] X. Bo, W. Junwen, L. Guangjie, and D. Yuewei, "Image copy-move forgery detection based on SURF," in *Proceedings of the 2010 International Conference on Multimedia Information Networking and Security*, pp. 889–892, Nanjing, Jiangsu, China, November 2010.

[14] H. Gao, W. Huang, Y. Duan et al., "The cloud-edge-based dynamic reconfiguration to service workflow for mobile ecommerce environments: a QoS prediction perspective," *ACM Transactions on Internet Technology*, vol. 21, no. 1, pp. 1–23, 2020.

[15] X. Yin, M. Cao, and S. Zhou, "An approach to alleviate the sparsity problem of hybrid collaborative filtering based recommendations: the product-attribute perspective from user reviews," *Mobile Networks Applications*, vol. 25, no. 2, pp. 376–390, 2020.

[16] P. Mishra, N. Mishra, S. Sharma, and R. Patel, "Region duplication forgery detection technique based on SURF and HAC," *The Scientific World Journal*, vol. 2013, no. 17, pp. 1–8, 2013.

[17] M. Zandi, A. Mahmoudi-Aznaveh, and A. Talebpour, "Iterative copy-move forgery detection based on a new interest point detector," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 11, pp. 2499–2512, 2016.

[18] M. Zandi, A. Mahmoudi-Aznaveh, and A. Mansouri, "Adaptive matching for copy-move forgery detection," in *Proceedings of the 2014 IEEE International Workshop on Information Forensics and Security. (WIFS)*, pp. 119–124, Atlanta, GA, USA, December 2014.

[19] X. Pan and S. Lyu, "Region duplication detection using image feature matching," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 6, pp. 857–867, 2010.

[20] V. Christlein, C. Riess, J. Jordan, C. Riess, and E. Angelopoulou, "An evaluation of popular copy-move forgery detection approaches," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 6, pp. 1841–1854, 2012.

[21] F. Yang, J. Li, W. Lu, and J. Weng, "Copy-move forgery detection based on hybrid features," *Engineering Applications of Artificial Intelligence*, vol. 59, pp. 73–83, 2017.

[22] X. Bi and C.-M. Pun, "Fast copy-move forgery detection using local bidirectional coherency error refinement," *Pattern Recognition*, vol. 81, pp. 161–175, 2018.

[23] B. Chen, M. Yu, Q. Su, H. J. Shim, and Y.-Q. Shi, "Fractional quaternion Zernike moments for robust color image copy-move forgery detection," *IEEE Access*, vol. 6, pp. 56637–56646, 2018.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 5563312, 18 pages
https://doi.org/10.1155/2021/5563312

Research Article

Blockchain-Enhanced Fair Task Scheduling for Cloud-Fog-Edge Coordination Environments: Model and Algorithm

Wenjuan Li,[1] **Shihua Cao**,[1] **Keyong Hu**,[1] **Jian Cao**,[2] **and Rajkumar Buyya**[3]

[1]*Qianjiang College, Hangzhou Normal University, Hangzhou, China*
[2]*Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China*
[3]*The Cloud Computing and Distributed Systems (CLOUDS) Laboratory, The University of Melbourne, Melbourne, Australia*

Correspondence should be addressed to Wenjuan Li; liellie@163.com

Received 6 January 2021; Revised 3 March 2021; Accepted 22 March 2021; Published 5 April 2021

Academic Editor: Honghao Gao

The cloud-fog-edge hybrid system is the evolution of the traditional centralized cloud computing model. Through the combination of different levels of resources, it is able to handle service requests from terminal users with a lower latency. However, it is accompanied by greater uncertainty, unreliability, and instability due to the decentralization and regionalization of service processing, as well as the unreasonable and unfairness in resource allocation, task scheduling, and coordination, caused by the autonomy of node distribution. Therefore, this paper introduces blockchain technology to construct a trust-enabled interaction framework in a cloud-fog-edge environment, and through a double-chain structure, it improves the reliability and verifiability of task processing without a big management overhead. Furthermore, in order to fully consider the reasonability and load balance in service coordination and task scheduling, Berger's model and the conception of service justice are introduced to perform reasonable matching of tasks and resources. We have developed a trust-based cloud-fog-edge service simulation system based on iFogsim, and through a large number of experiments, the performance of the proposed model is verified in terms of makespan, scheduling success rate, latency, and user satisfaction with some classical scheduling models.

1. Introduction

The traditional centralized cloud service mode, with all tasks being handled in the center, faces big challenges in practical applications, including high latency, network dependency, single point of failure, and failure scale effect and cannot adapt to the instant transaction scenarios. Therefore, researchers put forward the idea of distributing some time-sensitive and low-resource demand services to be processed at the edge of the network, which is the prototype of fog or edge computing.

Based on the idea, this paper designs a cloud-fog-edge hybrid computing architecture, in which the fog layer is actually a management middleware between edge and cloud, helping the scheduler decide where to deploy and implement a service, thus to better achieve the resource balancing and low service latency [1].

The cloud-fog-edge computing model provides a superior framework for the distributed coordination of resources and the timely process of tasks. However, due to the different level of heterogeneity and resource asymmetry, the possible cross-layer task offloading, and the dynamics and mobility of edge nodes, the three-tier hybrid architecture has been facing with a more severe crisis in terms of service credibility and rational resource allocation and scheduling [2–5]. On the other hand, the hybrid service provision environment, which contains the edge or IoT layer, has more abundant, random, and diversified application scenarios than traditional cloud computing and requires the task scheduling strategy to be more adaptive and robust.

At present, researchers have proposed many valuable solutions for the efficient task scheduling in the distributed systems (see [6–11]). However, the existing strategies cannot achieve full functions in a cloud-fog-edge environment due to the following reasons: (1) the centralized trust management model cannot handle the identity authentication and behavior management under the heterogeneous and

decentralized architecture, (2) the existing distributed or decentralized trust models cannot provide enough trust evidence with sufficient credibility to convince entities in the same domain or across domains, and (3) it is difficult to guarantee the relative fairness and global rationality of resource distribution in scheduling.

In response to these problems, this paper proposes a trust-enhanced fair scheduling model for the cloud-fog-edge environments. It firstly deploys the blockchain technology to build a decentralized trust framework to solve the problem of service credibility. And then, it introduces Berger's theory of wealth distribution to propose the concept and evaluation method of service fairness. Considering the mobile application scenarios, we divide scheduling into two levels. The first level is user scheduling, which deals with the matching of mobile terminals and access points, and the second one is task scheduling, which realizes the matching of tasks and resources in the local pool. Task scheduling in a cloud-fog-edge environment is treated as a multiobjective optimization problem, which comprehensively considers terminal mobility, QoS service requirements, and workload.

The main contributions of this paper are (1) it proposes a novel distributed decentralized trust management model based on blockchain technology; (2) it introduces Berger's fairness theory to design a preference-based fair task scheduling model for the cloud-fog-edge environments; and (3) it designs a new task scheduling algorithm that comprehensively considers user mobility, load balancing, and trust.

The rest of the paper is organized as follows. Section 2 briefly introduces the related work. Section 3 introduces the system architecture along with the design details of the double-blockchain structure-based service transaction framework. The trust-enhanced location-aware fair scheduling algorithm is proposed in Section 4. And performance evaluation is presented in Section 5. The last section concludes the paper.

2. Related Work

2.1. Task Scheduling Model. Task scheduling refers to the process of unified resource allocation among all resources and users based on a certain resource usage rules and user requirements in a specific service provision environment. Task scheduling is the core of cloud computing. The existing task scheduling models can generally be divided into three categories: performance-centric, user-centric, and hybrid.

The goal of the performance-centric models is to improve the performance of the systems, such as system throughput, makespan, and energy consumption. The classical task algorithms like min-min, max-min, greedy algorithm, swarm intelligence, and genetic algorithm all belong to this group. Aiming at solving the energy efficiency (EE) problem in fog computing, Y. Yang et al. proposed a collaborative task offloading algorithm named MEETS [12, 13]. In [14], they developed a delay and energy consumption balanced scheduling algorithm called DEBTS. H. Sun et al. designed a contract and cluster-based resource allocation model for fog-cloud hybrid platform [15]. N. Auluck et al. proposed a security aware task scheduling model for fog computing, through the classification of users and providers, achieving a better system performance [16].

Improving service QoS and enhancing user service experience are the primary tasks of the user-centric models [17–24]. In order to minimize the service delay, Z. Liu et al. proposed a dispersive stable task scheduling model named DATS [25]. M. Mukherjee et al. introduced a deadline-aware task scheduling algorithm for fog computing to complete as many tasks as possible before their deadlines [26]. G. Zhang et al. proposed a task offloading algorithm for fog computing systems to reduce the service delay [27]. G. Zhang et al. put forward a fog task offloading algorithm named DOTS to decrease latency with the help of the voluntary nodes (VNs) [28]. H. Apat et al. proposed a three-layered priority-based fog task scheduling model to meet the different deadline requirement of tasks [29].

In recent years, researchers have proposed many hybrid scheduling models considering multifactors, of which energy consumption and delay were their focus. For example, C. Tang et al. used genetic algorithms to design a hybrid task scheduling algorithm for mobile cloud computing taking into consideration multifactors such as energy consumption task requirement like deadline and cost [30]. J. Xu et al. proposed a task scheduling model combining laxity and ant colony algorithm to satisfy both energy consumption and latency [31]. In order to improve the makespan and execution cost, M. Yang et al. put forward an evolutionary heuristic-based multiobjective task scheduling model for fog environment [32]. Caching is an effective way to reduce the execution delay of tasks in edge computing, and thus some novel caching strategies have been proposed by researchers [33–35].

However, indicators considered by most current scheduling models are simply makespan, service QoS, load balancing, or economic principles, and the fairness of scheduling is usually ignored.

2.2. Trust Management in Distributed Systems. Trust is an efficient mechanism in dealing with the reputation and reliability issues in the distributed open network environment. A large number of highlight resulting in trust and trust-enabled interactions are achieved [36, 37].

W. Tian et al. made a survey on trust evaluation methods in sensor cloud systems [38]. J. Wang et al. proposed a recommendation trust evaluation method based on the cloud model and attribute weighted clustering [39]. P. Zhang et al. introduced a domain-based trust management model for the public cloud [40]. For the trust-base service management, Y. Li put forward an online learning aided service offloading model for mobile edge computing [41]. H. Yang et al. designed a context-aware trust prediction model for vehicle edge computing [42]. S. Jian et al. proposed a trust-based multiobjective task allocation model for cloud service transactions [43]. C. Hu et al. designed an e-commerce recommendation algorithm combining trust and distrust factors [44]. X. Meng et al. proposed a two-tier service selection method that matched

credibility and behavior patterns [45]. Z. Ma et al. introduced a trust-enabled edge data management model based on blockchain technology [46]. L. Cui et al. put forward an edge service configuration method combined with the decentralized trust [47]. We have also done some interesting work in this area, such as the service trust architecture for open cloud environment [48], trust decision strategy based on fuzzy clustering [49], and trust and preference learning-based service matching and combination models [50, 51].

However, the existing solutions cannot achieve full functions in mobile fog computing systems due to the following limitations: (1) the centralized trust framework cannot be accurately integrated with the cloud-fog-edge hybrid systems characterized by node dynamic distributed autonomy and topology loosely coupled, (2) trust crisis of the center node, which easily leads to single point of failure, (3) huge trust management overhead prevents it from being used in the instant trading scenario, and (4) trust evidence lacks transparency. Therefore, the decentralized trust management model and trust-enabled transaction mechanism for the cloud-fog-edge hybrid environments require further exploration.

3. System Overview

3.1. Trust-Enhanced Cloud-Fog-Edge Hybrid Framework. In a cloud-fog-edge hybrid framework, fog computing architects a management middleware between cloud servers and edge devices to coordinate resource allocation, thus improving system performance and fulfilling the customized demand of different devices and users. Figure 1 shows the hybrid scheduling framework of the proposed model. Blockchain technology is introduced to construct a decentralized trust model in the IoT device layer, thereby enhancing the interactive credibility of the entire cloud-fog-edge hybrid architecture.

The framework consists of three layers: (1) trust-enhanced edge/IoT layer, (2) fog layer, and (3) cloud layer. The trust-enhanced edge/IoT layer implements the peer-to-peer interconnections through the ubiquitous sensors and communication protocols over the traditional IoT infrastructure layer and constructs the distributed and decentralized trust management with a blockchain architecture. The fog layer is introduced to decide the allocation and management of resources and handle complex transactions such as cross-group/cross-cloud tasks. The fog layer consists of a large number of fog servers deployed at the network edge. Fog servers are much closer to users or terminals than the centralized cloud server and are more capable than the terminals. Therefore, they can ensure lower latency and meet the needs of cross-group or cross-domain interactions. Trust management becomes one part of the responsibility of edge/fog servers. The cloud layer is located at the top of the framework. It is mainly used to deal with some high-level and highly complex tasks, such as data mining and behavior or preference analysis, which impose high requirements on computing and storage capacity and relatively loose requirement on the response time.

3.2. Service Transaction Model Based on a Double-Blockchain Structure. Generally speaking, a complete trust authentication contains two parts: identity authentication and behavior evaluation. And the first part is easy to obtain or evaluate because the identity information of a node is relatively statically stable even in a P2P network topology. In contrast, trading behavior is dynamic, requiring a lot of computing power to record and assess. Therefore, to improve the integrity and efficiency of the trust management in a real-time trading environment, a cloud service transaction model based on a double-blockchain structure is proposed, as shown in Figure 2.

The double-blockchain structure contains two blockchains, including a trust authentication blockchain and a trading behavior blockchain.

Definition 1. Trust Authentication Blockchain (TAB). TAB is a blockchain structure that stores the trust data of the cloud-fog-edge hybrid transaction system and assists trust transaction decisions. TAB adopts an alliance chain structure, in which blocks store the trust data of nodes in the transaction system.

TAB is responsible for managing trust data in the cloud service markets and provides trust evaluation results to other nodes. Each block in TAB contains two parts: identity trust data and behavior trust data. When a node initially joins, only the identity part is written; however, as time goes by with the transactions progressing, the behavior part is continuously written. Authentication is completed by a small number of supervisors, who can be the normal miners or some special nodes elected by the market authority. Miners are responsible for storing and authenticating trust data and ensuring the consistency of the data through some specifically designed consensus mechanisms. When nodes apply to enter the trading network, they must pay a fee to run a smart contract for the initial identity authentication. In addition, when they want to obtain the trust data of the other nodes, they also have to pay a fee. The funding provides the incentive fee for the miners. Figure 3 shows the basic content of a trust block of TAB.

Definition 2. Trading Behavior Blockchain (TBB). TBB is a blockchain structure that stores the transaction data in the cloud-fog-edge transaction system. A block in TBB contains both the transaction data and the evaluation data, which will assist in generating the behavioral trust data.

TBB is responsible for generating and storing the trading data. In TBB, the miners have two tasks, one is to generate the new transaction blocks based on the latest transaction results and the other is to evaluate the behavior trust, generate a trust block, and then forward it to TAB. The corresponding trust block will be confirmed and stored by the miners in TAB. Figure 4 shows the basic content of a trading block in TBB. The block structure of TBB is very similar to that of TAB, and the only difference is the storage content.

TAB and TBB jointly ensure the credibility of cloud-fog-edge interaction. When an entity registers for the first time, its identity trust will be written into TAB. As the transaction

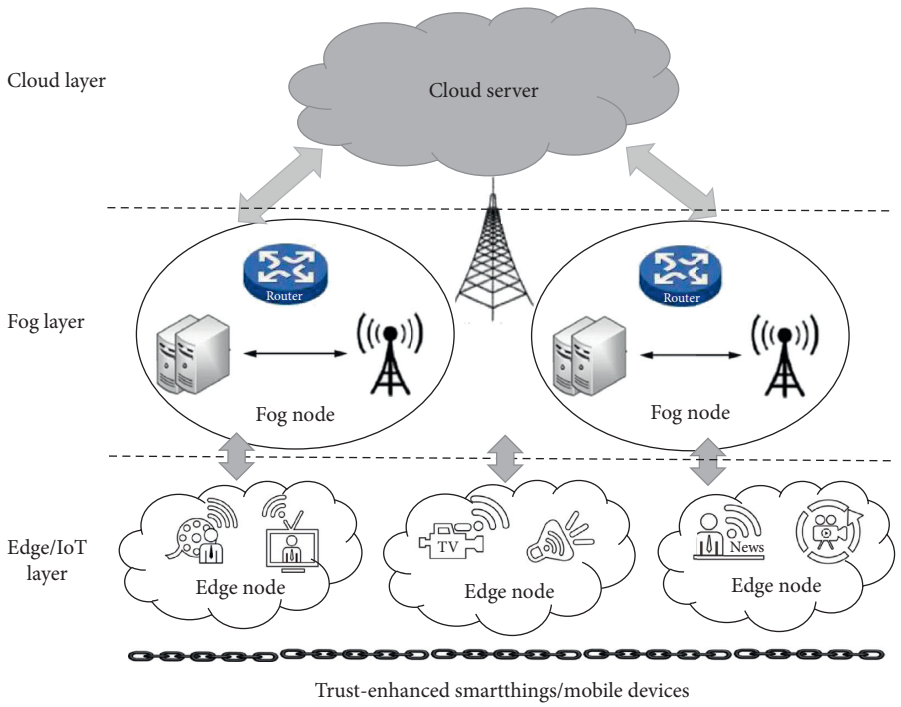


Figure 1: Trust-enhanced cloud-fog-edge hybrid framework.

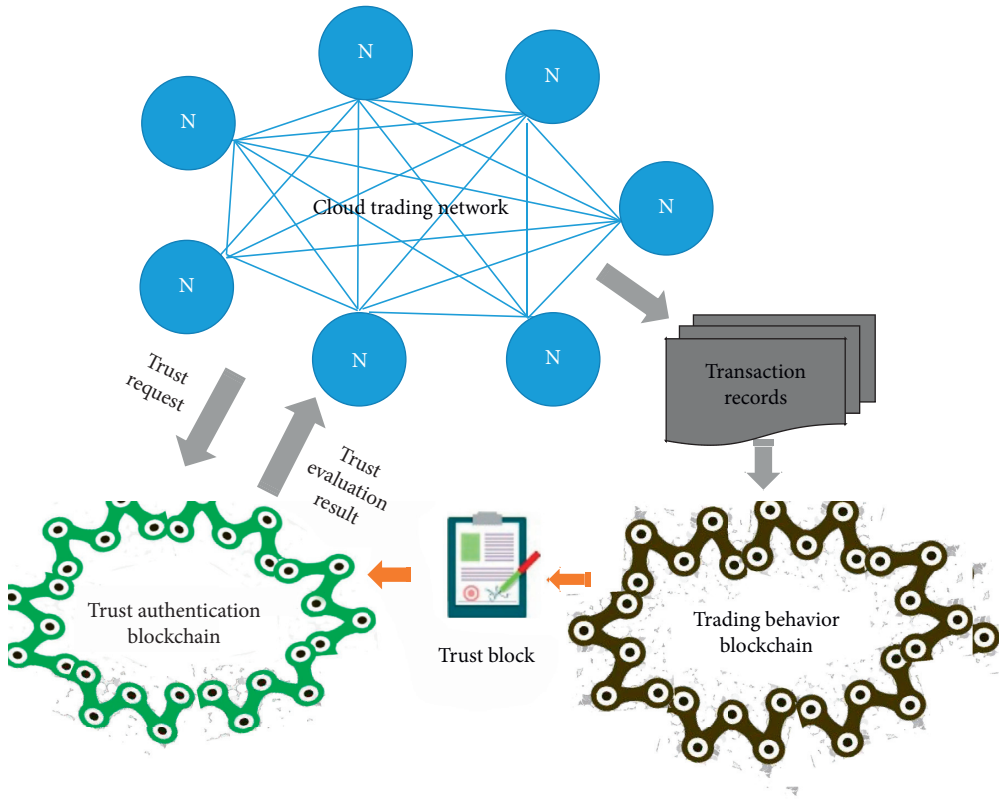


Figure 2: Service transaction model based on a double-blockchain structure.

progresses, the trading record between entities will be written into TBB one by one and then the behavior trust will be evaluated in TBB and be forwarded into TAB. When TAB gradually grasps enough trust data of the entities in the transaction chain, it can help entities make trust decisions more accurately, which improves the reliability and success

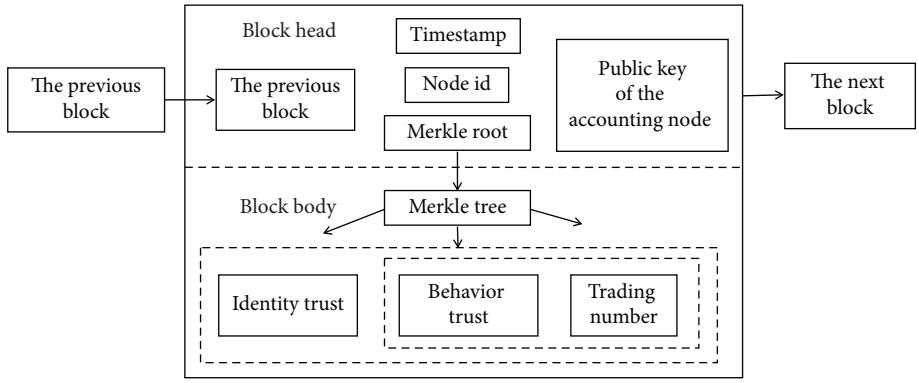


Figure 3: The basic content in a trust block of TAB.

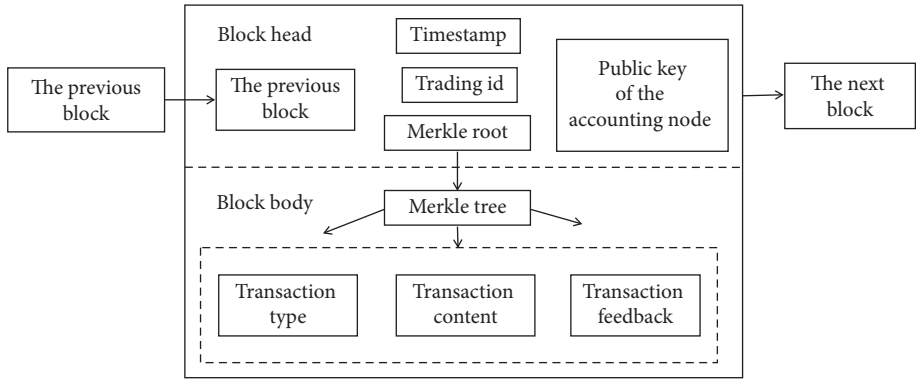


Figure 4: The basic content in a trading block of TBB.

rate of interactions. Furthermore, with the benefits of the double-blockchain structure, an efficient parallel computing is realized. Because trust value is provided by TAB, while the large-scale calculation or evaluation is done on the TBB side, this effectively reduces the latency caused by trust management and makes it possible for the application of blockchain in a real-time and high-reliability scenarios.

4. Trust-Enhanced Location-Aware Fair Scheduling Model in a Cloud-Fog-Edge Computing System

4.1. Service Justice. The theory of distribution justice (Berger's theory) proposed by Berger et al., is a theory of the distribution of social wealth. Berger et al. believed that justice is a subjective behavior, of which the conclusion is drawn from the comparison with the comparable objects. In the real world, people usually refer to the surrounding social information, such as the status and compensation of others, to generate their own expectations, which are used to judge whether they are fairly treated or not. Furthermore, the satisfaction of these expectations affects their future behaviors. Figure 5 is the diagram of Berger's model.

Here, c or C is the attribute set and go or GO is the expectation set; go represents the expectation value, while GO means the actually obtained value. In Berger's theory, the distribution justice holds only when the local self-structure and the reference structure are significantly related.

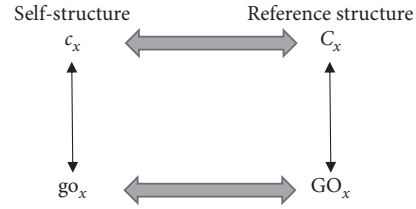


Figure 5: Berger's model.

In a cloud-fog-edge hybrid scheduling framework, fog provides services to users by coordinating resources from edge or cloud and users pay a service fee according to the usage of resources or time. Therefore, service fairness of users is reflected in the following: the scheduling system is able to provide services on user demand and preferences, ensuring a high user satisfaction and experience.

Definition 3. Service Justice (SJ). It represents the degree of agreement between the QoS of user actually obtained and the expected. How to evaluate SJ_i is given in the following formula:

$$SJ_i = \theta \ln \frac{OS_i}{ES_i}, \tag{1}$$

where θ represents the equilibrium coefficient, OS_i means the actual obtained service quality vector of $user_i$, and ES_i is the demand vector of $user_i$.

Obviously, the overall service justice of the scheduling system is the average of all system users' SJ, which is illustrated in the following formula:

$$SJ = \sum_{i=1}^{n} \varphi_i SJ_i. \tag{2}$$

The overall system balance is achieved by constraining the overall system service justice, namely, min SJ.

Suppose the capability vector of a resource can be described by $CR = (C_{cpu}, C_{ram}, C_{bd})$, where C_{cpu}, C_{ram}, and C_{bd} represent the CPU, memory, and bandwidth capability of the resource/vm. The resource expectation of a user to the resource is described by $ER = (E_{cpu}, E_{ram}, E_{bd})$, where E_{cpu}, E_{ram}, and E_{bd} represent the expected CPU, memory, and bandwidth capability of the resource. In order to eliminate the influence of dimensions of different capability, it uses the following formula for normalization:

$$\frac{C_i - C_{min}}{C_{max} - C_{min}} \text{ or } \frac{E_i - E_{min}}{E_{max} - E_{min}}. \tag{3}$$

After normalization, performance or expectation parameters of the virtual machine are mapped to the [0, 1]. However, different users or tasks have different service preferences, for example, some are CPU or computing power enthusiasts and some are bandwidth pursuers. Let $P = (P_{cpu}, P_{ram}, P_{bd})$ describe the service preference or weight of the attribute. It uses modified cosine function to compare the similarity and distance between expectation and obtained. Let SJ_{ij} denote the service justice when $task_i$ is scheduled to $resource_j$, and SJ_{ij} can be calculated by the following formula:

$$SJ_{ij} = \frac{CR_iP_i \cdot ER_iP_i}{\|CR_iP_i\| \|ER_iP_i\|}. \tag{4}$$

Accordingly, the SJ of the task set can be obtained by the following formula:

$$SJ = \sum_{i=1}^{n} \frac{CR_iP_i \cdot ER_iP_i}{CR_iP_iER_iP_i}. \tag{5}$$

Here, n denotes the number of tasks in the scheduling system.

4.2. Trust-Based Location-Aware Fair Scheduling Model

4.2.1. User/Smart Thing Model. A user or a smart thing is an intelligent entity who requests for mobile services. Let $ST = \{st_0, st_1, \ldots, st_{k-1}\}$ represents a user or a smart thing set. The i^{th} user st_i ($i \in [0, k-1]$) can be further described as $st_i = \{stI_D, st_{Name}, st_{Location}, st_{TaskSet}, st_{Trust}\}$. The meaning of each attribute is as follows:

(i) st_{ID} represents the unique system identity of a user/smart thing

(ii) st_{Name} represents the name of the user/smart thing

(iii) $st_{Location}$ represents the location of the user/smart thing

(iv) $st_{TaskSet}$ represents the task set submitted by the user/smart thing

(v) st_{Trust} represents the trust requirement of the user/smart thing

4.2.2. Task Model. $T = \{t_0, t_1, \ldots, t_{n-1}\}$ represents a task set, where the i^{th} task ti ($i \in [0, n-1]$) can be further described as $ti = \{t_{ID}, t_{State}, t_{RRes}, t_{ORSet}, t_{DeadLine}\}$. The meaning of each attribute is as follows:

(i) t_{ID} represents the identity of a task

(ii) t_{State} represents the state of the task

(iii) t_{RRes} represents the resource requirements of the task, t_{RRes} can be further described as $t_{RRes} = \{t_{Comp}, t_{BW}, t_{Stor}\}$, where t_{Comp}, t_{BW}, and t_{Stor} represent the computation, network, and storage requirement of the task

(iv) t_{ORSet} represents the obtained resource set of the task

(v) $t_{DeadLine}$ represents the latest completion time of the task

4.2.3. Provider Model. $AP = \{ap_0, ap_1, \ldots, ap_{k-1}\}$ represents the provider set. In this paper, service provider refers to the access point or gateway server. In a cloud-fog-edge hybrid platform, a service provider can be a cloud server, a fog server, or even an edge server.

The i^{th} provider ap_i ($i \in [0, k-1]$) can be further described as $api = \{ap_{ID}, ap_{Name}, ap_{ResouceSet}, ap_{Load}, ap_{Location}\}$. The meaning of each attribute is as follows:

(i) ap_{ID} represents the unique system identity of a provider

(ii) ap_{Name} represents the name of the provider

(iii) $ap_{ResourceSet}$ represents the resource set managed by the provider

(iv) ap_{Load} represents the load of the provider

(v) $ap_{Location}$ represents the location of the provider

4.2.4. Resource Model. $R = \{r_0, r_1, \ldots, r_{m-1}\}$ represents the resource set. The j^{th} resource rj ($j \in [0, m-1]$) can be further described as $rj = \{r_{ID}, r_{Name}, r_{Provider}, r_{Cap}\}$. The meaning of each attribute is as follows:

(i) r_{ID} represents the unique system identity of a resource

(ii) r_{Name} represents the name of the resource

(iii) $r_{Provider}$ represents the manage provider of the resource

(iv) r_{Cap} represents the capability of the resource, $rCap = \{rComp, rBW, rStor\}$, where r_{Comp}, r_{BW}, and r_{Stor} represent the computation, network, and storage capability of the resource.

4.2.5. Two-Level Service Scheduling Model. This paper divides scheduling in a cloud-fog-edge hybrid environment into two levels, user scheduling and task scheduling. User scheduling realizes the binding of the smart thing (mobile user or terminal) to the credible nearest access point, while task scheduling realizes the matching of tasks issued by the user with resources/vms managed by the access point.

The user scheduling model is defined as USM = (ST, AP, ACS), which selects the most suitable gateway or access point for the user from the list of providers:

(i) ST means the smart thing set, as defined in 4.2.1.

(ii) AP represents the gateway or access point set, as defined in 4.2.3.

(iii) ACS: AP->AP*i* represents the selection of the access point *ap* for smart thing *st* using a certain strategy. In a cloud-fog-edge environment, since the location of a user or terminal is constantly changing and a fog or edge server also has a limited service capability, distance between *st* and *ap* must be considered. Besides, for a trustworthy trading, credibility is another important factor. Therefore, the primary principles of ACS include distance and trust.

The task scheduling model is defined as TSM = (*T*, *R*, RCS), which matches the most suitable resources for the task set of the user.

(i) *T* means the task set, as defined in 4.2.2.

(ii) *R* means the resource set managed by the service provider, as defined in 4.2.4.

(iii) RCS: *R*- > *Ri* represents the selection of the resource *r* for task *t* using a certain strategy. The main selection principle of RCS in this paper is service justice defined in 4.1 and work load.

Figure 6 shows the general steps of the proposed model in this paper.

(1) The user scheduler searches appropriate access points for a specific user/smart thing according to the current location and makes recommendation according to distance and trust;

(2) When the recommendation arrives, the user/smart thing requests a trust service from the TAB and makes decision after obtaining the trust data. If it agrees to trade, then proceed to step (3); otherwise, it asks the scheduler to resume recommendation and returns to step (1);

(3) The access point (gateway) starts the task scheduler to select specific resources for the user;

(4) The task scheduler selects the appropriate virtual machines to perform tasks according to the principle of service justice and load balance;

(5) The user obtains the service and makes service evaluation. The transaction data enter the TBB;

(6) TBB regularly performs trust evaluation and forwards trust blocks into TAB.

4.3. User Scheduling Algorithm. The user scheduler chooses the most suitable access point for user or smart thing as shown in Algorithm 1. The basic principle is distance and trust.

In Algorithm 1, coverage means the service coverage of an *ap*, and *trust_threshold* is the minimum required trust value of the smart thing.

The user scheduler chooses the nearest and reliable access point for a user. The general steps of user scheduling are as follows: (1) the user scheduler selects the candidate subset $AP_{candidate}$ from AP that can cover st_i and meet its minimum trust requirement, (2) it calculates the distance between ap and st_i and chooses the nearest ap, namely, ap_{chosen} from $AP_{candidate}$, and (3) it connects st_i with ap_{chosen}.

4.4. Task Scheduling Algorithm. The task scheduling algorithm references Berger's model to ensure the overall system justice.

Algorithm 2 shows how task scheduler (running on an access point) chooses the suitable resources/servers for mobile tasks triggered by a user or a smart thing.

In Algorithm 2, *load_safe_range* represents the safe load limit of a server or a resource. The general steps of task scheduling are as follows: (1) it puts the load-safe virtual resources into the candidate resource set $R_{\text{candidate}}$, (2) it calculates the service fairness of each resource r_k in $R_{\text{candidate}}$ according to the demand of *ti* and the capability of r_k, and selects the most suitable resource r_{chosen}, which achieves the maximum *SJ*, and (3) it schedules *ti* to r_{chosen} and updates the workload of r_{chosen}.

4.5. Blockchain-Architected Lightweight Trust Algorithm. Trust runs through the entire process of scheduling. The node initiates the transaction requests trust service from the trust blockchain TAB. After paying a certain fee, the trustworthiness of the candidate provider (the access server or the application server) is obtained, thereby helping it to make trustworthy decisions. After each transaction, transaction-related data (transaction type, content, and evaluation data) will be used to generate transaction blocks and be added to the transaction blockchain TBB after consensus. Thereafter, TBB initiates a trust evaluation transaction at regular intervals and pushes the new trust block to TAB.

This paper uses a lightweight method to compute the trust of a specific trading node as shown in the following formula:

$$T_i = \sum_{k=1}^{n} \theta V_{ki}, \tag{6}$$

where T_i is the trust value of node i, θ is the evaluation weight of trustor node k, and V_{ki} represents the trust evaluation value from k to i.

In the proposed mode, in order to deploy blockchain-based framework, we defined some transaction-related classes and trust-related classes. The transaction-related classes are used to maintain the transaction data and keep it traceable and tamper-proof. The trust-related classes are

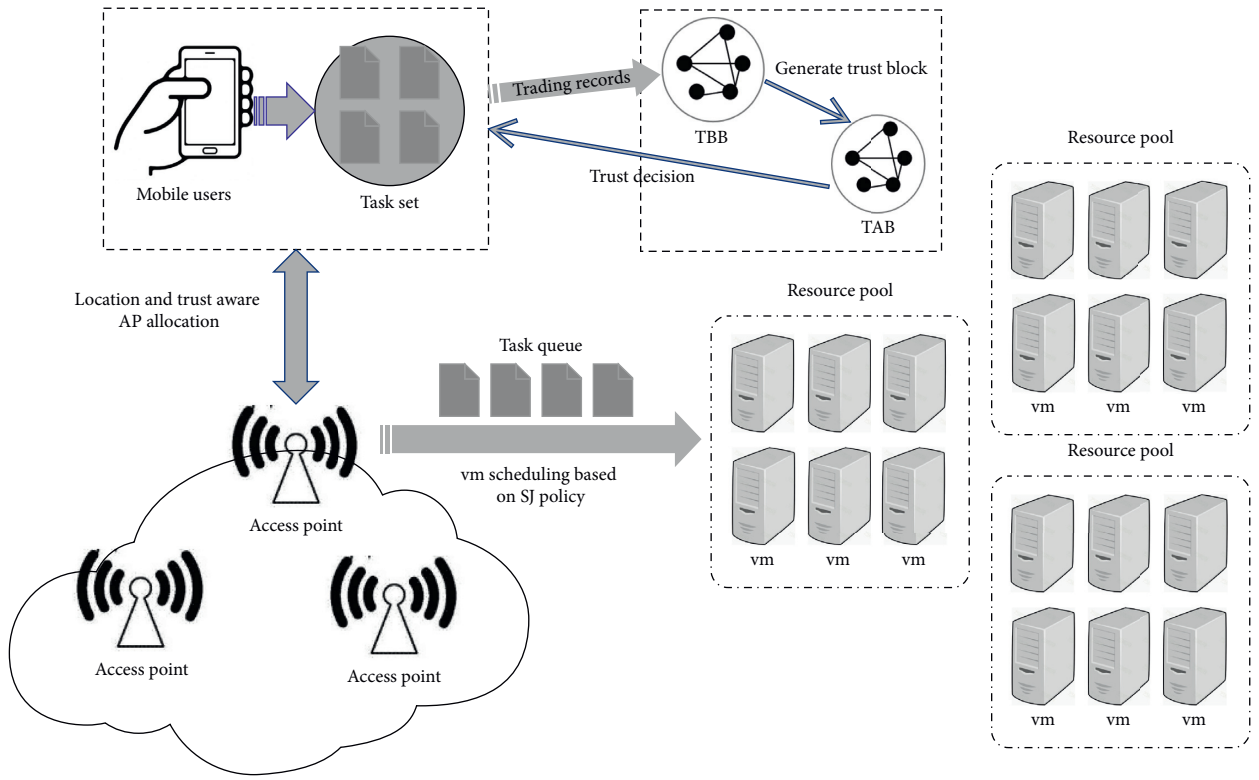


Figure 6: Trust-based location-aware fair scheduling model.

Input: AP, st_i, coverage, trust_threshold
Output: the matching of st_i to an appropriate ap

(1) **for** each ap_j in AP **do**
(2) Calculate the distance from st_i to ap_j;
(3) Calculate trust from st_i to ap_j;
(4) **if** ($distance_{sti\_apj} \leq coverage$) && ($trust_{sti\_apj} \geq trust\_threshold$) **then**
(5) $ap_j -> AP_{candidate}$
(6) **end if**
(7) **done**
(8) **for** each ap_k in $AP_{candidate}$ **do**
(9) **if** ($distance_{sti\_apk} < min\_distance$) **then**
(10) $min\_distance = distance_{sti\_apk}$;
(11) $ap_{chosen} = ap_k$;
(12) **end if**
(13) **done**
(14) **if** $ap_{chosen} != NULL$ **then**
(15) connect st_i to ap_{chosen};
(16) **end if**

Algorithm 1: Connect st_i with an appropriate access point ap.

used to maintain trust data. They are mainly generated offline and can support trust decisions in the real-time transactions. Following is the definition of the critical data structures. The class "transaction" defined in Algorithm 3 stores transaction data of the trading system, through which trust value from trustor to trustee can be obtained. And the class "TrustBlock" defined in Algorithm 4 is used to store the trust value of a certain node. Each block in TAB or TBB has the normal content of a blockchain block (timestamp, hash value, and the pointer to the previous block) and some special functions (mineBlock, calculateHash, and addTransaction). (Algorithms 3 and 4)

5. Performance Evaluation

5.1. Experimental Design. This paper designed a task scheduling prototype for performance test in a cloud-fog-edge environment based on iFogsim [52]. In iFogsim, FogDevice is

(i) **Input:** R, ti, $load\_safe\_range$
(ii) **Output:** the matching of ti to an appropriate r
(1) **for** each r_j in R **do**
(2) observe the workload of r_j;
(3) **if** ($workload_{rj} \leq load\_safe\_range$) **then**
(4) r_j - > $R_{candidate}$
(5) **end if**
(6) **done**
(7) **for** each r_k in $R_{candidate}$ **do**
(8) Calculate service justice $SJ_{ti\ \_rk}$ of ti in r_j according to formula (4);
(9) **If** ($SJ_{ti\ \_rk} < min\_sj$) **then**
(10) $min\_sk = SJ_{ti\ \_rk}$;
(11) $r_{chosen} = r_k$;
(12) **end if**
(13) **done**
(14) **if** r_{chosen}! = $NULL$ **then**
(15) schedule t_i to r_{chosen};
(16) update the workload of r_{chosen};
(17) **end if**

Algorithm 2:. Matching mobile task ti with the suitable resource r.

used to represent a common device from terminal to fog server. In order to reflect the mobility and distinguish the functions of different devices, new entities are added: MobileDevice, AccessPointDevice, MobileSensor, and MobileActuator. Inheriting from parent class FogDevice, MobileDevice handles mobile tasks, making trust decision and service selection. A mobile device randomly appears in a specific location and has its own specific resource preferences. MobileSensor inherits from sensor, acting as a mobile sensor, and is able to generate and transmit tasks. While MobileActuator inherits from Actuator to deal with the execution and output of tasks. AccessPoint also inherits from FogDevice, which is an intermediary for resource allocation and service scheduling according to the location of the mobile devices. It helps a mobile device to choose the most cost-effective service resources and handles real-time task migration. The resource pool that actually executes tasks is a collection of traditional fog devices, and each fog device is configured with certain capabilities. Figure 7 shows the key classes and their inherent relationships in the scheduling model.

This paper draws on the idea of blockchain to construct a trust-enhanced task scheduling model. The proposed model contains a dual-blockchain structure, including a transaction blockchain and a trust blockchain. Thus, the new classes including the class of TransactionBlock and TrustBlock class are derived. The transaction block records the actual transactions of the system in a traceable and nontamperable manner. And the trust block records the trust value of the nodes, where the identity trust depends on the authenticity of the node, while the behavior trust is continuously updated by the transaction evaluation results. The generation of the trust block mainly adopts the offline mode, and a trust block is generated according to the feedback data obtained from the transaction blockchain. Trust management contains three parts: trust initialization, trust decision, and trust maintenance. Trust decision helps entities to choose the credible trading partners. Figure 8 is the UML diagram of the trust-related classes and their relationships.

In order to serve mobile devices, we designed the mapping and migration rules for tasks in virtual machines, which were implemented in the class MigrationPolicy. The main strategies include load balance, shortest distance between smartThing and AccessPoint or between smartThing and foglets (vms), lowest latency, and trust-based. Figure 9 shows the migration-related classes and their relationships.

5.2. Simulation Indicator and Benchmarks. In the experiments, we use various evaluation indicators for performance test of the different scheduling models, such as makespan, tuple loss rate, distribution justice, latency, and transaction success rate. Here, makespan represents the overall execution time of the task set. Tuple loss rate refers to the percentage of tasks dropped due to the limited processing capacity of the edge server or task migration. And the system justice is defined in formula (5).

The mentioned strategies were implemented and tested in the EEG Tractor Beam Game [53]. It is a latency-critical game requiring each player to wear an EEG headset to process the EEG signals and obtain his brain state. In EEG Tractor Beam Game, there are seven types of tuples carried between the different modules of the application, as shown in Table 1. The capabilities of cloud, fog, and edge devices are shown in Table 2, and the task sending interval of two different EEG headsets is 10 ms and 5 ms, respectively.

The speed of MobileDevice is set to be 20 m/s, including nine different moving directions: NONE, EAST, NORTH, NORTH EAST, NORTHWEST, WEST, SOUTHWEST,

```
    class Transaction {
(1) String transactionId;
(2) PublicKey trustor;//the public key of trustor
(3) PublicKey trustee;//the public key of trustee
(4) double trust;//trust value from trustor to trustee
(5) byte[] signature;
(6) ArrayList < Transaction > transactions;
(7) boolean processTransaction() {
(8)             verifySignature();
(9)             transactionId = calulateHash();
(10)            generateTrust();//generate trust according to transaction details
(11)            TransactionOutput(this.recipient, trust, transactionId));
(12)            return true;
(13) } //add transactions into transactionChain TBB
(14) boolean addTransaction(Transaction transaction) {
(15)            verifyTransaction();
(16)            processTransaction();
(17)            transactions.add(transaction);
(18)            return true;
(19) }
    }
```

Algorithm 3: Class of transaction.

```
    TrustBlock {
(1) String hash;
(2) String previousHash;
(3) String merkleRoot;
(4) long timeStamp;
(5) int nonce;//the proof of miners
(6) double trust;//the integrated trust of a certain node
(7) TrustBlock(String previousHash) {//the basic structure of a trust block
(8)         this.previousHash = previousHash;
(9)         this.timeStamp = new Date().getTime();
(10)        this.hash = calculateHash(); }
(11) void mineBlock() {
(12)        merkleRoot = StringUtil.getMerkleRoot(transactions);
(13)        while(!hash.substring(0, difficulty).equals(target)) {
(14)            nonce ++;
(15)            hash = calculateHash();
(16) }
(17) }
    }
```

Algorithm 4: Class of TrustBlock.

SOUTH, and SOUTHEAST. The maximum switch radius of vm to a certain access point is set to be 40 meters. Various fog devices (resources and access points) are randomly and evenly distributed on the map.

This paper aims at improving the performance of resource scheduling in a cloud-fog-edge hybrid computing architecture, including the total execution time, service delay, the fairness, and reliability. Therefore, we chose the min-min and Berger's model as the benchmark comparison models. At the same time, in order to measure the influence of the different factors in the proposed model, we also compared it with the subtraction models, namely, locate-ware and load-balance. The locate-ware is the subtraction model which only remains the location switching strategy of

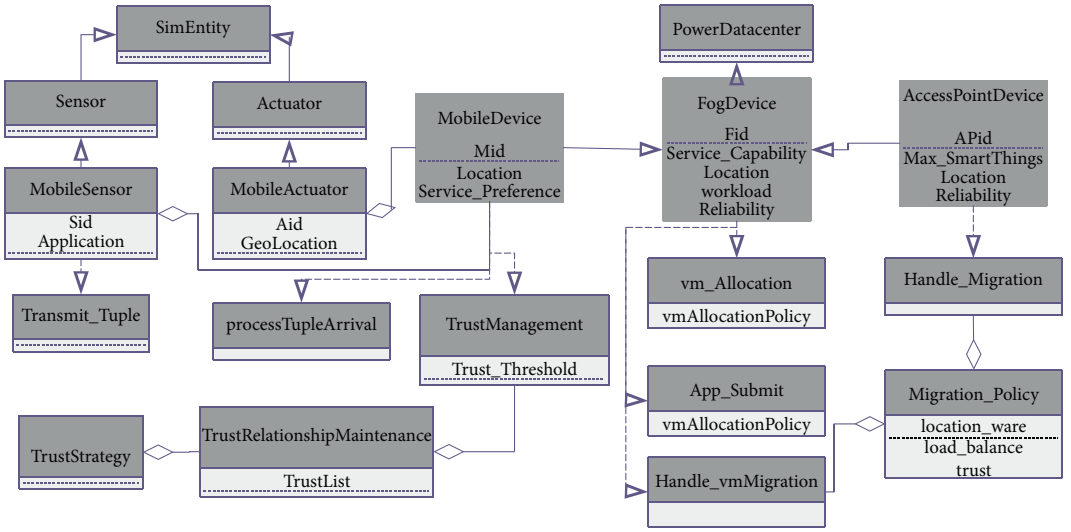


Figure 7: The key classes in scheduling procedure.

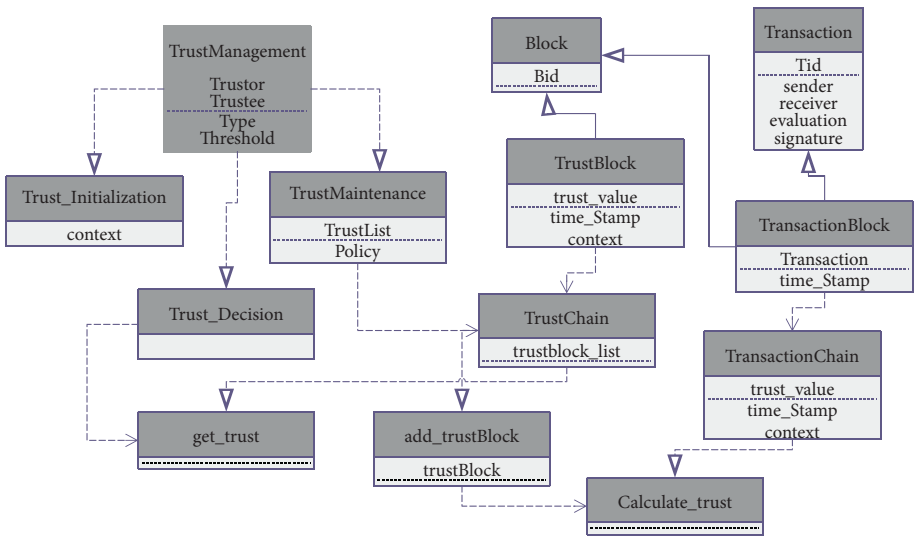


Figure 8: Blockchain-enhanced trust architecture.

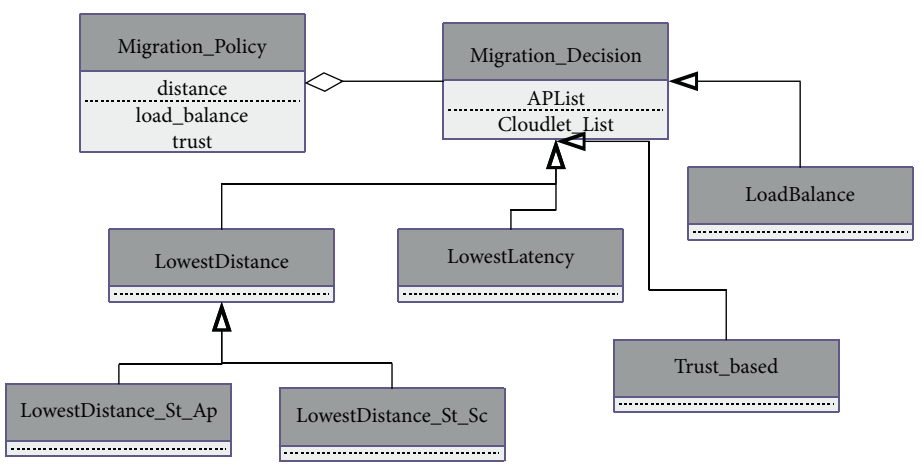


Figure 9: Types of migration rules.

Table 1: The parameters of the intermodule tuples in the EEG tractor game.

Tuple type	CPU length (MIPS)	N/W length
EEG	2000(A)/2500(B)	500
_SENSOR	3500	500
PLAY_GAME_STATE	1000	1000
CONCENTRATION	14	500
GLOBAL_GAME_STATE	1000	1000
GLOBAL_STATE_UPDATE	1000	500
SELF_STATE_UPDATE	1000	500

Table 2: The configuration of the device.

Tuple type	CPU (GHz)	RAM (GB)	Power (W)
EEG	2000(A)/2500(B)	500	107.339(M) 83.433(I)
_SENSOR	3500	500	107.339(M) 83.433(I)
PLAY_GAME_STATE	1000	1000	83.53(M) 82.44(I)
CONCENTRATION	14	500	107.339(M) 83.433(I)

the proposed model in scheduling, and the load-balance is the subtraction model that only maintains the load-sensitive strategy.

The task scheduling algorithms including min-min, Berger's Model, proposed model, location-aware, and load-balance model are compared from various perspectives.

5.3. Simulation Result. The experimental results are shown in Figures 10–17. From the results, we can see that the proposed model has achieved better effect than the other models on almost all the experimental metrics. And two subtraction models also gained better results than the other benchmark models.

5.3.1. Comparison on System Performance. For the test of system performance, we compared the mentioned models in terms of makespan, service latency, and tuple loss rate.

The proposed model and the location-aware model work well in the total completion time and service delay. Because the built-in location-sensitive scheduling strategy takes into account the mobility of nodes to match it with the closer access point, which is able to effectively reduce the transmission overhead, the service delay, and the total execution time. The load-balance model gains a good performance in the index of the tuple loss rate because it considers the load status of resources in the task scheduling stage, which can effectively reduce the waiting time in queue and avoid tuple loss caused by overload.

The overall system performance of the min-min model is not bad. However, since it does not consider the resource preferences, mobility, and load, it may cause tuple loss and redistribution, which ultimately affects the overall execution time. Berger's model does not work well because it focuses only on the fairness of scheduling, which tends to satisfy the needs of users rather than the overall performance of the system. In a cloud-fog-edge hybrid environment with nodes moving frequently, although the global fairness algorithm can find the most suitable resource for nodes, the scheduling result may not be implemented due to the distance, which increases the possibility of tuple loss and redistribution, thus eventually increasing in the total execution time. In particular, the overhead of Berger's model is relatively large. When the total number of tasks increases, its execution time is very long, which may affect its deployment in the actual applications.

5.3.2. Comparison on User Satisfaction. User satisfaction, measured by service justice (SJ) in this paper, is an important index to evaluate the quality of scheduling algorithms. Berger's model performed the best in this respect because scheduling fairness was its core concern. However, using the traditional Berger's model to perform global matching and fairness calculations during task scheduling often leads to high computational overhead, which may cause the paralysis of the resource allocation system when the task scale is large. The experimental results indicate that when the number of tuples is greater than 125, Berger's model cannot give a resource allocation decision within the tolerable time.

The proposed model still performs well in terms of user satisfaction because it adopts a two-level scheduling mode, and through the fairness factors, it takes into consideration the specific requirements of tasks in the process of task scheduling.

5.3.3. Influence of the Decentralized Trust Mechanism. This section tested the performance of the double-blockchain-based trust mechanism proposed in this paper.

Figure 17 shows the result of the transaction success rate. It can be seen that the decentralized trust can maintain a high degree of the transaction success rate despite the increase in the malicious nodes, indicating that it can effectively assist a reliable transaction decisions. In sharp contrast, in the random transaction scenario, the transaction success rate drops sharply as the number of malicious nodes increases.

This paper also used NetLogo [54] to test the efficiency of the proposed trust mechanism. Table 3 shows the parameters of the simulations.

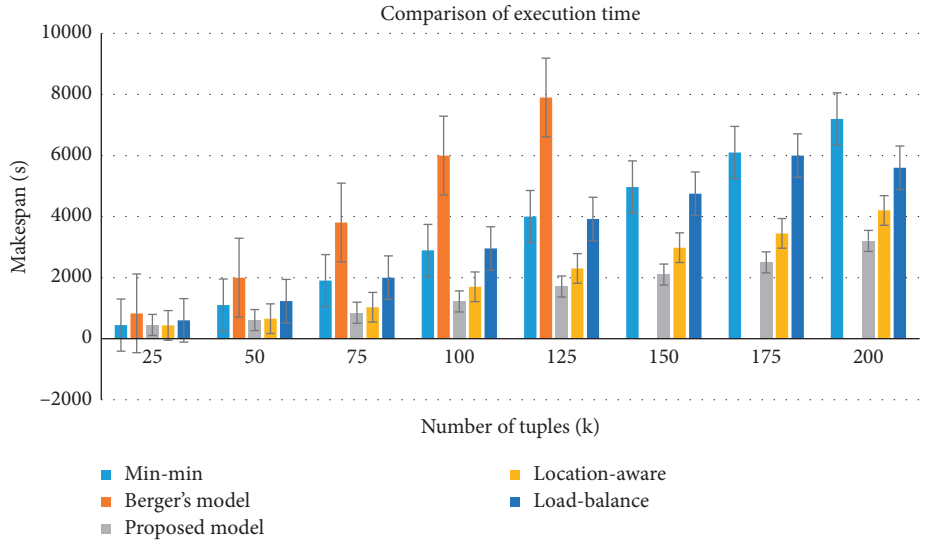


Figure 10: Comparison of makespan.

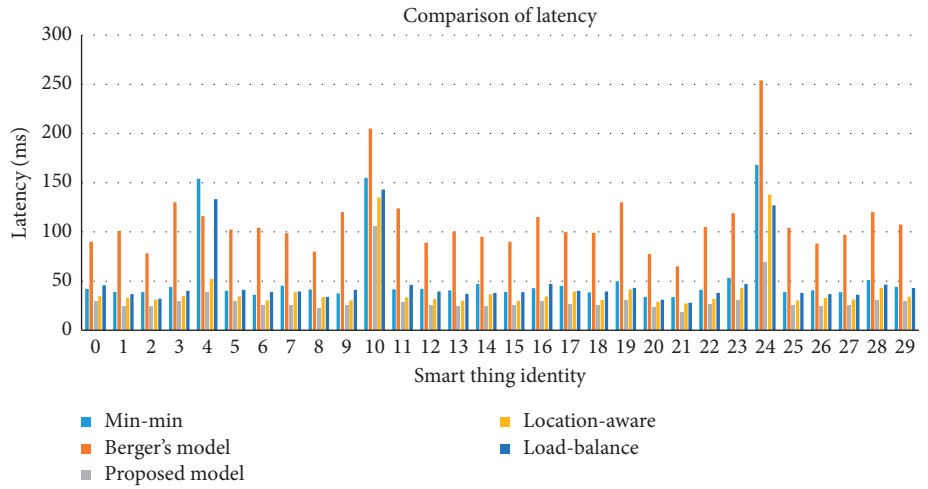


Figure 11: Comparison of service latency.

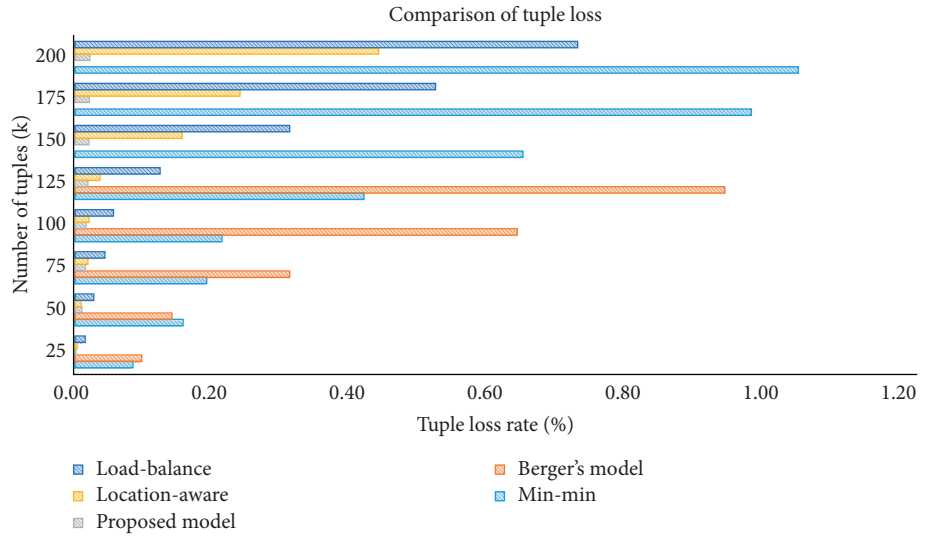


Figure 12: Comparison of tuple loss rate.

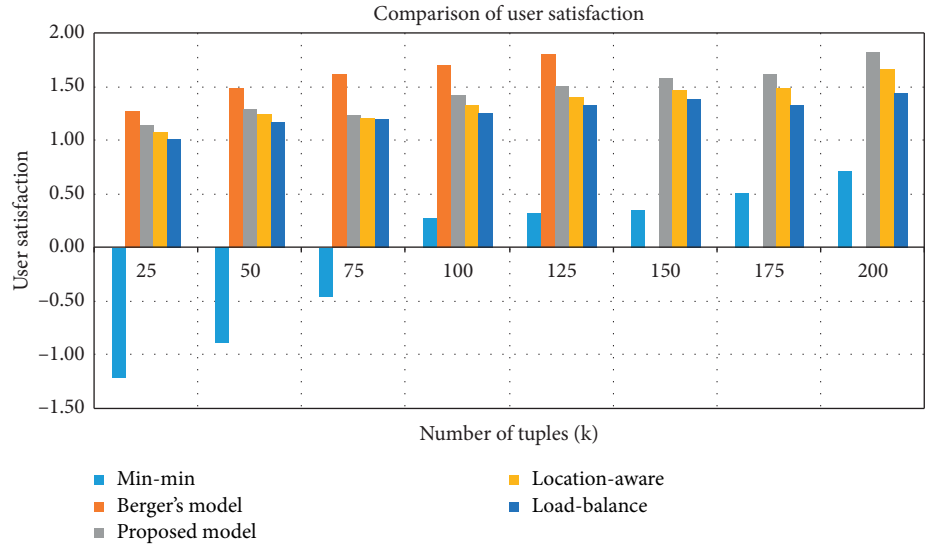


Figure 13: Comparison of user satisfaction.

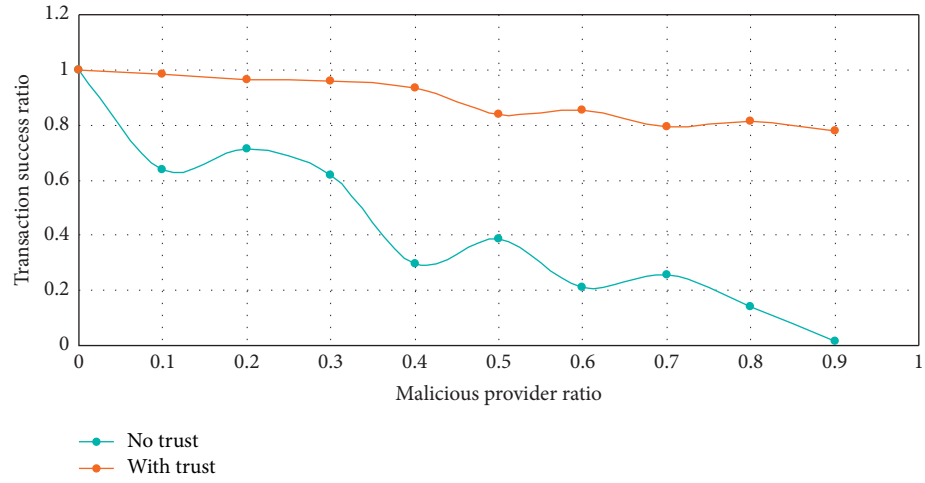


Figure 14: Comparison of transaction success ratio.

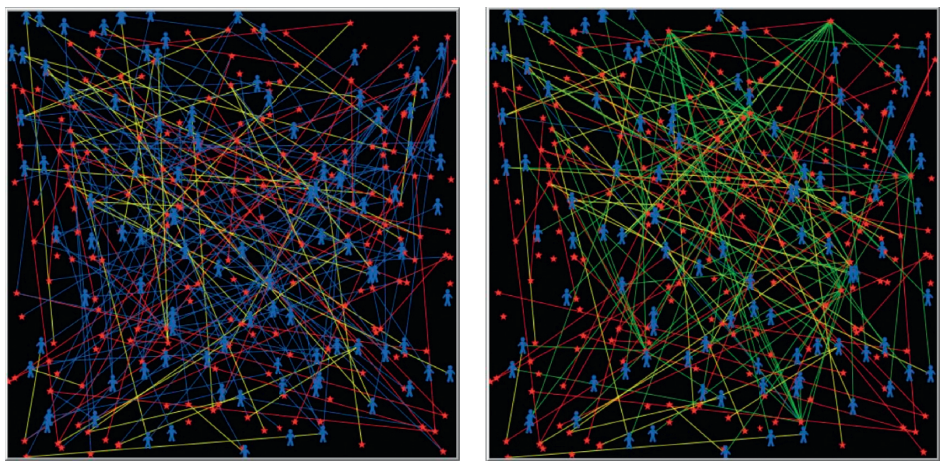
Several experiments were done to observe the impact of trust on the success rate of transactions when the proportion of malicious providers was fixed to 30%. Figure 15 is the experimental result when trust mechanism was loaded, and Figure 16 is the experimental data without any trust mechanisms (in the case of random transactions).

In the simulation chart, users are represented by the blue color person-shaped turtles, and the fog providers (access point or service providers) are the red color star-shaped turtles. The links in the charts represent the relationship between the different entities, including the trading relationship (blue edges), the recommendation relationship (yellow edges), and the cooperation between service providers (red edges).

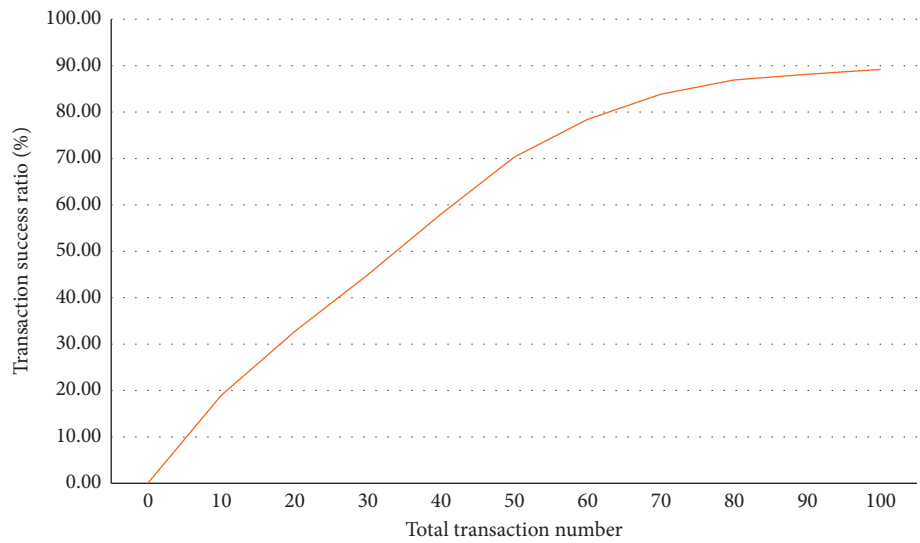
Figures 15(a) and 16(a) are the initial state of the simulation, Figures 15(b) and 16(b) are the final state of the trading cycle, and Figures 15(c) and 16(c) are the records of transaction success ratio output from NetLogo reporter. It can be seen from the figures that, when the trust mechanism is loaded, the transaction success rate gradually increases with the increasing of the number of transactions, which remains almost the same when no trust is loaded.

In order to measure the overhead of trust, we also evaluated the influence of trust on the total execution time. Figure 17 shows the additional time overhead brought by the loading of trust, from which we can see that trust does not have much influence on the total execution time, with even a few nodes showing the shorter execution time. The reason is that after trust has been running for a period of time, when entities in the system are able to correctly select the credible trading partner, the reselection or transaction failure in the random transactions can be effectively avoided, thus, to a certain extent, reducing the total execution time.

In summary, in the cloud-fog-edge hybrid environment, in order to improve the effective matching of terminals, access points, and their expected resources, it is necessary to fully consider the location, the workload, and also the

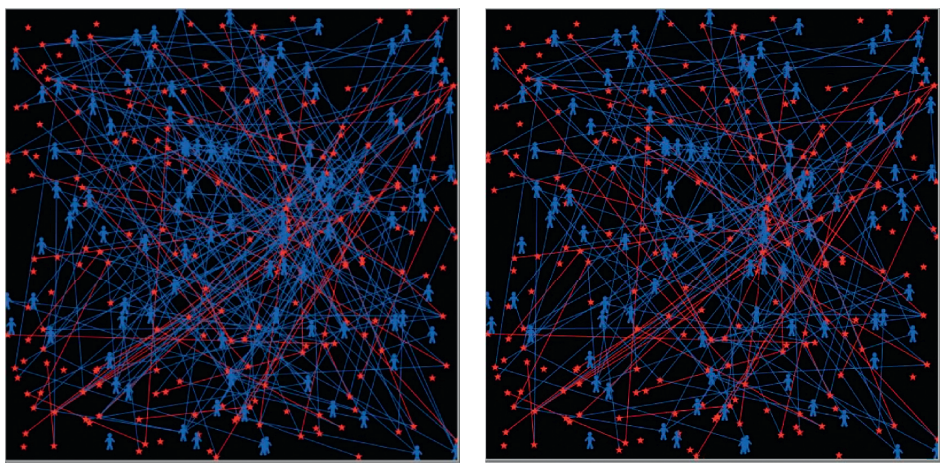


(a) (b)



(c)

Figure 15: (a) The initial state (with trust). (b) The final state (with trust). (c) The transaction success rate (with trust).



(a) (b)

Figure 16: Continued.

Figure 16: (a) The initial state (without trust). (b) The final state (without trust). (c) The transaction success rate (without trust).

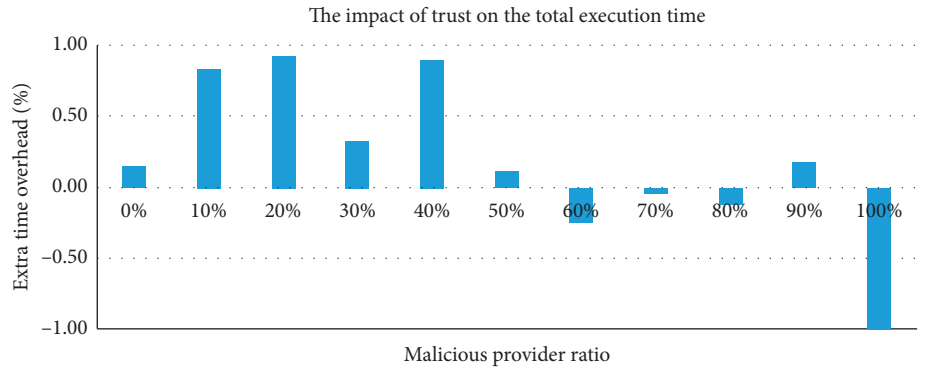


Figure 17: Percentage of increased time overhead by trust.

Table 3: The parameters in the trust performance test.

Number of providers	Number of users	Link probability	Ratio of malicious providers (%)
200	100	0.20	30

credibility of nodes. The proposed decentralized trust strategy is able to guarantee the safety and the reliability of interaction.

6. Conclusion and Future Work

This paper proposes a trust-enhanced location-aware fair task scheduling model for the cloud-fog-edge hybrid environment. The new model contains a three-layer architecture of IoT, fog, and cloud. The fog layer is utilized to achieve the cloud-fog or fog-edge resource coordination and unified scheduling. The proposed task scheduling algorithm comprehensively considers user mobility, system justice, load balance, and trust requirement. Berger's theory is introduced to solve the fairness problem in task scheduling. In resource allocation, it comprehensively considers the location, task's QoS requirements, the capability, and the load of the resources. In addition, to improve the credibility of service interaction, it draws on the idea of blockchain to build a decentralized trust framework. The performance of the new model and the related strategies was evaluated by a series of related experiments.

However, the proposed model still has some imperfections. For example, it adopts a two-level scheduling mode to solve the problem of mobile access. However, the location of the mobile terminals is constantly changing, requiring the timely replacement of the access point. Therefore, it is necessary to determine where and when to switch and also the resource reservation algorithms. Things like the moving direction, speed, and preferences can be used to predict the

next access point and make resource reservations for reducing the handover delay. In addition, the dual-blockchain-based trust management, with all the transactions recorded in the trading behavior blockchain, needs to find the most appropriate time to generate the trust blocks, whether in a fix time period or a certain number of transactions. All these problems are our future work.

Data Availability

Data is available upon request to the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under grant 61702151, 61702320, 61772334, the National Key Research and Development Plan under grant 2018YFB1003800, and the Joint Funds of the Zhejiang Provincial Natural Science Foundation of China under grant LHY21E090004.

References

[1] C. Chang, S. Srirama, R. Buyya, and I. Fog, "An efficient fog-computing infrastructure for the internet of things," *Computer*, vol. 50, no. 9, pp. 92–98, 2017.

[2] C. Huang, X. Mei, G. Zhao, J. Wu et al., "Transaction modelling and execution analysis of uncertainty composition service in mobility computing environments," *Science China: Information Science*, vol. 45, no. 1, pp. 70–96, 2015.

[3] X. Deng, P. Guan, E. Liu et al., "Integrated trust based resource cooperation in edge computing," *Journal of Computer Research and Development*, vol. 55, no. 3, pp. 449–477, 2018.

[4] J. Zhang, Y. Zhao, B. Chen et al., "Survey on data security and privacy-preserving for the research of edge computing," *Journal on Communications*, vol. 39, no. 3, pp. 1–21, 2018.

[5] M. Mukherjee, R. Matam, L. Shu et al., "Security and privacy in fog computing:challenges," *IEEE Access*, vol. 5, pp. 19293–19304, 2017.

[6] L. Bittencourt, J. Diaz-Montes, R. Buyya et al., "Mobility-aware application scheduling in fog computing," *IEEE Cloud Computing*, vol. 4, no. 2, pp. 26–35, 2017.

[7] M. Lopes, W. Higashino, M. Capretz et al., "MyiFogSim: a simulator for virtual machine migration in fog computing," in *Proceedings of UCC Companion 17, the 10th International Conference. ACM*, pp. 47–52, Zurich, Switzerland, December 2017.

[8] Y. Yin, F. Yu, Y. Xu, L. Yu, and J. Mu, "Network location-aware service recommendation with random walk in cyber-physical systems," *Sensors*, vol. 17, no. 9, p. 2059, 2017.

[9] Y. Yin, J. Xia, Y. Li, Y. Xu, W. Xu, and L. Yu, "Group-wise itinerary planning in temporary mobile social network," *IEEE Access*, vol. 7, pp. 83682–83693, 2019.

[10] B. Xu, C. Zhao, E. Hu et al., "Job scheduling algorithm based on Berger model in cloud environment," *Advances in Engineering Software*, vol. 42, no. 7, pp. 419–425, 2011.

[11] C. Zhao, "Research and realization of job scheduling algorithm in cloud environment," Master's thesis, Beijing Jiaotong University, Beijing, China, 2009.

[12] Y. Yang, K. Wang, G. Zhang, X. Chen, X. Luo, and M. Zhou, "Maximal energy efficient task scheduling for homogeneous fog networks," in *Proceedings of the IEEE INFOCOM 2018—IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 274–279, Honolulu, HI, USA, April 2018.

[13] Y. Yang, K. Wang, G. Zhang, X. Chen, X. Luo, and M. Zhou, "MEETS: maximal energy efficient task scheduling in homogeneous fog networks," *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 4076–4087, 2018.

[14] Y. Yang, S. Zhao, W. Zhang, Y. Chen, X. Luo, and J. Wang, "DEBTS: delay energy balanced task scheduling in homogeneous fog networks," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 2094–2106, 2018.

[15] H. Sun, H. Yu, and G. Fan, "Contract-based resource sharing for time effective task scheduling in fog-cloud environment," *IEEE Transactions on Network and Service Management*, vol. 17, no. 2, 2020.

[16] N. Auluck, O. Rana, S. Nepal, A. Jones, and A. Singh, "Scheduling real time security aware tasks in fog networks," *IEEE Transactions on Services Computing*, 2019.

[17] H. Gao, Y. Xu, Y. Yin, W. Zhang, R. Li, and X. Wang, "Context-aware QoS prediction with neural collaborative filtering for internet-of-things services," *IEEE Internet of Things Journal*, vol. 7, 2019.

[18] H. Gao, Y. Duan, L. Shao, and X. Sun, "Transformation-based processing of typed resources for multimedia sources in the IoT environment," *Wireless Networks*, 2019.

[19] J. Yu, J. Li, Z. Yu, and Q. Huang, "Multimodal transformer with multi-view visual representation for image captioning," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 8, 2019.

[20] J. Yu, M. Tan, H. Zhang, D. Tao, and Y. Rui, "Hierarchical deep click feature prediction for fine-grained image recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[21] H. Gao, C. Liu, Y. Li, and X. Yang, "V2VR: reliable hybrid-network-oriented V2V data transmission and routing considering RSUs and connectivity probability," *IEEE Transactions on Intelligent Transportation Systems(T-ITS)*, 2020.

[22] H. Gao, L. Kuang, Y. Yin, B. Guo, and K. Dou, "Mining consuming behaviors with temporal evolution for personalized recommendation in mobile marketing apps," *ACM/Springer Mobile Networks and Applications (MONET)*, vol. 25, no. 4, pp. 1233–1248, 2020.

[23] Y. Yin, Z. Cao, Y. Xu et al., "QoS prediction for service recommendation with features learning in mobile edge computing environment," *IEEE Transactions on Cognitive Communications and Networking*, 2020.

[24] X. Yang, S. Zhou, and C. Min, "An approach to alleviate the sparsity problem of hybrid collaborative filtering based recommendations: the product-attribute perspective from user reviews," *Mobile Networks & Applications*, vol. 25, no. 2, pp. 376–390, 2020.

[25] Z. Liu, X. Yang, Y. Yang, K. Wang, and G. Mao, "DATS: dispersive stable task scheduling in heterogeneous fog networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 3423–3436, 2019.

[26] M. Mukherjee, M. Guo, J. Lloret, R. Iqbal, and Q. Zhang, "Deadline-aware fair scheduling for offloaded tasks in fog computing with inter-fog dependency," *IEEE Communications Letters*, vol. 24, no. 2, pp. 307–311, 2020.

[27] G. Zhang, F. Shen, Y. Zhang, R. Yang, Y. Yang, and E. A. Jorswieck, "Delay minimized task scheduling in fog-enabled IoT networks," in *Proceedings of the 2018 10th International Conference on Wireless Communications and Signal Processing (WCSP)*, pp. 1–6, Hangzhou, China, October 2018.

[28] G. Zhang, F. Shen, N. Chen, P. Zhu, X. Dai, and Y. Yang, "DOTS: delay-optimal task scheduling among voluntary nodes in fog networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 3533–3544, 2019.

[29] H. K. Apat, "An optimal task scheduling towards minimized cost and response time in fog computing infrastructure," in *Proceedings of the 2019 International Conference on Information Technology (ICIT)*, pp. 160–165, Bhubaneswar, India, 2019.

[30] C. Tang, S. Xiao, X. Wei, M. Hao, and W. Chen, "Energy efficient and deadline satisfied task scheduling in mobile cloud computing," in *Proceedings of the 2018 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pp. 198–205, Shanghai, China, 2018.

[31] J. Xu, Z. Hao, R. Zhang, and X. Sun, "A method based on the combination of laxity and ant colony system for cloud-fog task scheduling," *IEEE Access*, vol. 7, pp. 116218–116226, 2019.

[32] M. Yang, H. Ma, S. Wei, Y. Zeng, Y. Chen, and Y. Hu, "A multi-objective task scheduling method for fog computing in cyber-physical-social services," *IEEE Access*, vol. 8, 2020.

[33] X. Wei, J. Liu, Y. Wang, C. Tang, and Y. Hu, "Wireless edge caching based on content similarity in dynamic environments," *Journal of Systems Architecture*, vol. 115, pp. 1–8, 2021.

[34] J. Zhang, X. Hu, Z. Ning et al., "Joint resource allocation for latency-sensitive services over mobile edge computing networks with caching," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4283–4294, 2019.

[35] W. Wen, Y. Cui, T. Q. S. Quek, F.-C. Zheng, and S. Jin, "Joint optimal software caching, computation offloading and communications resource allocation for mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 7, pp. 7879–7894, 2019.

[36] J. Jiang, G. Han, L. Shu, S. Chan, and K. Wang, "A trust model based on cloud theory in underwater acoustic sensor networks," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 1, pp. 342–350, 2017.

[37] P. Zhang, Y. Kong, and M. Zhou, "A domain partition-based trust model for unreliable clouds," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 9, pp. 2167–2178, 2018.

[38] T. Wang, G. Zhang, S. Cai et al., "Survey on trust evaluation mechanism in sensor-cloud," *Journal on Communications*, vol. 39, no. 6, pp. 37–51, 2018.

[39] J. Wang, Z. Yu, H. Zhang et al., "Service recommended trust algorithm based on cloud model attributes weighted clustering," *Journal of System Simulation*, vol. 30, no. 11, pp. 275–289, 2018.

[40] P. Zhang, Y. Kong, and M. Zhou, "A novel trust model for unreliable public clouds based on domain partition," in *Proceedings of the of IEEE 14th International Conference on Networking, Sensing and Control (ICNSC)*, pp. 275–280, IEEE, Calabria, Italy, May 2017.

[41] Y. Li, X. Wang, X. Gan, H. Jin, L. Fu, and X. Wang, "Learning-aided computation offloading for trusted collaborative mobile edge computing," *IEEE Transactions on Mobile Computing*, vol. 8, 2019.

[42] H. Yang, J. Cho, H. Son, and D. Lee, "Context-aware trust estimation for realtime crowdsensing services in vehicular edge networks," in *Proceedings of the 17th Annual Consumer Communications & Networking Conference (CCNC)*, pp. 1–6, IEEE, Las Vegas, NV, USA, January 2020.

[43] J. Shu, C. Liang, and J. Xu, "Trust-based multi-objectives task assignment model in cloud service system," *Journal of Computer Research and Development*, vol. 55, no. 6, pp. 1167–1179, 2018.

[44] C. Hu, X. Tong, and W. Liang, "The real-value restricted Boltzmann machine recommendation algorithm based on trust-distrust relationship," *Systems Engineering-Theory & Practice*, vol. 39, no. 7, pp. 1817–1830, 2019.

[45] X. Meng, J. Ma, D. Lu, and Y. Wang, "Trust and behavioral modeling based two layer service selection," *Journal of Xidian University (Natural Science)*, vol. 41, no. 4, pp. 198–204, 2014.

[46] Z. Ma, X. Wang, D. Jain, H. Khan, H. Gao, and Z. Wang, "A blockchain-based trusted data management scheme in edge computing," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 2013–2021, 2020.

[47] L. Cui, S. Yang, Z. Chen, Y. Pan, Z. Ming, and M. Xu, "A decentralized and trusted edge computing platform for Internet of Things," *IEEE Internet of Things Journal*, vol. 11, 2019.

[48] W. Li, L. Ping, and X. Pan, "Trust model to enhance security and interoperability of cloud environment," in *Proceedings of CloudCom'09*, pp. 69–79, Springer, Bangalore, India, September 2009.

[49] W. Li, L. Ping, Q. Qiu, and Q. Zhang, "Research on trust management strategies in cloud computing environment," *Journal of Computational Information Systems*, vol. 8, no. 4, pp. 1757–1763, 2012.

[50] W. Li, J. Cao, S. Qian, and R. Buyya, "TSLAM: a trust-enabled self-learning agent model for service matching in the cloud market," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 13, no. 4, pp. 1–41, 2019.

[51] W. Li, J. Cao, K. Hu, J. Xu, and R. Buyya, "A trust-based agent learning model for service composition in mobile cloud computing environments," *IEEE Access*, vol. 7, pp. 34207–34226, 2019.

[52] H. Gupta, A. Dastjerdi, S. Ghosh, and R. Buyya, "iFogSim: a toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments," *Software Practice & Experience*, vol. 47, pp. 1275–1296, 2017.

[53] J. K. Zao, T. T. Gan, C. K. You et al., "Augmented brain computer interaction based on fog computing and linked data," in *Proceedings of the International Conference on Intelligent Environments (IE)*, pp. 374–377, IEEE, Shanghai, China, June-July 2014.

[54] NetLogo, "NetLogo user manual version 6.0.3," 2020, https://ccl.northwestern.edu/netlogo/docs/.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 8864552, 11 pages
https://doi.org/10.1155/2021/8864552

Research Article

Research on Coordinated Scheduling Strategy of Heat Storage Thermoelectric Units Based on Wind Power Data Acquisition System Using Edge Computing

D. J. Guan,[1] X. Y. Qian,[1] S. X. Lu,[1] N. An,[1] P. Ye,[1] M. L. Zhang,[2] and N. Zhang[2]

[1]*Shenyang Institute of Engineering, Shenyang 110136, China*
[2]*Economic and Technological Research Institute of Liaoning Electric Power Co., Ltd., Shenyang 110016, China*

Correspondence should be addressed to P. Ye; yepeng_126@sina.com

Received 29 September 2020; Revised 4 October 2020; Accepted 15 March 2021; Published 26 March 2021

Academic Editor: Honghao Gao

Cogeneration units use heat and electricity as the main operation mode in the heating season. It suffers the lack of peak shaving ability, resulting in intense wind curtailment. This study presents a coordinated scheduling strategy of heat storage thermoelectric units considering the uncertainty of wind power. The developed method performs a wind power data measurement and process system based on edge computing and deep learning to acquire operation data quickly and independently and adopts a wind power prediction method based on interval scenario. Furthermore, the coordinated control model of the thermal storage unit is established after the analysis of the characteristics of coordinated scheduling strategy of heat storage thermoelectric units based on wind heat complementation, and then the improved particle swarm optimization (PSO) is adopted to the solving process. The proposed method is verified by applying it to two typical thermal power units.

1. Introduction

In recent years, the installed capacity and scale of wind power installed in China have been increasing, and there has been a serious phenomenon of wind abandonment. According to the latest information released by the national energy administration, in 2017, China's wind abandonment mainly concentrated in the northeast, northwest, and North China, among which Gansu, Xinjiang, Jilin, and Inner Mongolia were the most serious provinces, with wind abandonment rates of 43%, 38%, 30%, and 21%. One of the main reasons for this phenomenon is that the heating demand of the thermal power unit in winter leads to a sharp decline in the peak shaving ability of the system [1, 2]. Due to the climate of the heating season in the north of China, the main power supply is the cogeneration unit, which lacks the capacity of peak load regulation. In order to respond to the demand of the heating load in the heating period, the thermal power unit adopts the mechanism of "power by heat," and generating output and heating output have a coupling relationship. With the increase of the minimum generating capacity, the contradiction of the lack of peak shaving ability of the system is intensified. This problem is also one of the important directions in the related research process of energy management system, by the establishment of layered distributed management system to collect and manage the data of power production, energy storage, and energy consumption and with the means of energy monitoring, energy statistics, and conversion efficiency analysis used in the system to distribute properly the energy facilities allocation and control functions which can significantly increase the utilization efficiency of the facilities and energy and reduce costs.

In order to improve the scale of wind power input network, energy storage technology has attracted extensive attention in recent years due to its excellent wind power complementarity. However, except for the mature pumped storage power stations, almost all other energy storage methods are faced with problems such as scale difficult to meet the requirements of modern power system and life,

cost, and efficiency. In order to expand the feasible area of wind power operation and solve the contradiction between wind power and heating units, a joint scheduling model of cogeneration unit with heat storage and electric heating system is established in [3], which shows that some heat storage devices with large capacity can effectively improve the flexibility of system scheduling and increase the wind power consumption. Literature [4] puts forward the concept of electricity-heat combined system, which goes beyond the scope of traditional power system, makes full use of the complementarity between power system and thermal system, and improves the ability of optimal allocation of resources in a broader space and time scope. Cogeneration units are of great importance for solving the energy crisis [5], but their thermoelectric coupling relationship is in contradiction with the new energy grid connection. Literature [6] uses heat storage devices to break the thermoelectric coupling relationship of "fixing electricity by heat" and analyzes the effects generated by different installation locations of heat storage devices. Literature [4] summarizes a scheduling model of comprehensive utilization of wind power consumption of heat storage device and electric boiler based on the thermoelectric decoupling characteristics of heat storage device and electric heating characteristics of electric boiler and compares the economic situation of different scheduling methods. In [7], under the influence of uncertain factors of wind power, the scheduling model of joint operation of the wind farm and cogeneration unit with heat storage is established. Considering the interaction between the online revenue and penalty cost under uncertain factors, the dual operation scheduling strategy is formulated with the goal of the highest final revenue. Under the condition of the German spot market, the efficiency of thermal power unit greatly improved by increasing cogeneration and heat storage device with the appropriate capacity. Scholars first analyzed the principle of peak shaving and valley filling of heat storage device to improve the peak shaving capacity, established a joint scheduling model of wind power, thermal power, and heat storage device, and tested the economy of wind power and heat storage device with the coal consumption rate as the index. At the same time, considering the uncertainty of wind power and the heat release characteristics of a heat storage device, a bilinear model scheduling method proposed solving the combined scheduling problem of electric heating [8–11].

Through the above research, it can be found that the thermal storage device can decouple the thermal electric characteristics of the thermal power unit. Through the reasonable distribution of the electrical power and thermal power of the unit, the peak shaving ability of the system can improve, the generation of new energy such as wind power can connect to the grid, the consumption space can improve, and the operation cost can reduce. For the seasonal uncertain factors of wind power, a wind turbine data measurement and process system based on edge computing and deep learning is adopted to control each WT in real time, quickly, and independently, and a scheme to improve the capacity of peak load regulation of thermal storage unit based on the coordinated scheduling of wind and heat is proposed. This paper mainly introduces the basic principle of the heat storage scheme and then analyzes the external characteristics of the thermal operation of the thermal storage unit. The coordinated scheduling strategy of heat storage thermoelectric units based on wind/heat complementary power system is proposed, and the coordinated control model of thermal storage unit is set up. Finally, the coordinated scheduling strategy is applied to typical thermal power units, and the improved particle swarm optimization (PSO) is used to solve and verify the effectiveness of the strategy.

2. Wind Turbine Measurement and Control System Based on Edge Computing and Deep Learning

2.1. Edge Computing. Along with the appearance of the ubiquitous power Internet of things, the Internet of everything era is coming [12–14]. The increasing amount of data generated by distribution network edge devices (such as large-scale DG and FL), at the same time, puts forward higher requirements for network communication bandwidth, data processing timeliness, data privacy, and so on, which cannot be effectively addressed by the centralized cloud computing model of traditional distribution network EMS. In this context, edge computing has received extensive attention from scholars at home and abroad. The basic idea is to carry out computing tasks at the data source side near the edge of the network, so as to reduce communication delay, reduce communication bandwidth requirements, relieve the computing pressure of the cloud master station, and realize data privacy protection of the edge nodes [15, 16].

2.2. Wind Turbine Measurement and Control System. A wind turbine measurement and control system based on edge computing and deep learning is adopted. The system includes the master server, the terminal sensor, the terminal controller, the terminal server, and WT master controller installed on the wind turbine. The terminal sensor is placed at the location of the parts to be monitored on the wind turbine. The structure diagram of the method is illustrated in Figure 1.

The terminal controller is connected to the terminal sensor and the terminal server. The terminal controller is stored with a deep learning model. The terminal controller is used to obtain the collected data of the terminal sensor, conduct edge computing and processing of the collected data through the deep learning model, store the processing results in the terminal server, and generate WT control instructions [17]. The terminal server is connected to the master server, which is used to obtain processing results through the terminal server and issue the deep learning model to the terminal controller. WT master connection terminal controller or the terminal server gets the WT control instructions and controls the WT according to the instructions. The data acquisition and control system can control each WT in real time, quickly, and independently, and wind turbine control is more efficient and intelligent. The implementation process is shown in Algorithm 1.

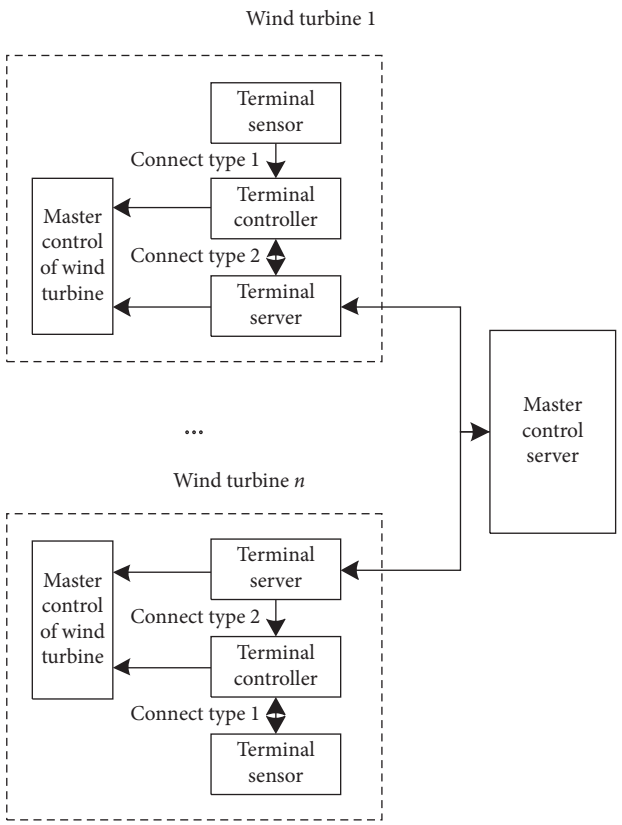


Figure 1: The structure diagram of wind turbine measurement and control system.

3. Coordinated Scheduling Model of Heat Storage Thermoelectric Units Based on Wind/Heat Complementary Power System

3.1. Electrothermal Characteristics of Heat Storage Thermoelectric Units. Poor regulation ability of the power system in the northern region during the heating period is largely due to the operation of the heating unit following the "heat to power" mode. Although the heat and electric load of the extraction unit change in a specified range, the power generation is greatly affected by the heat supply. With the increase of the heat supply, the adjustable range of the power generation becomes increasingly smaller. In the heating period, the heating task of a thermal power plant is very large, so the actual power adjustment range is very small [18,19]. The characteristic curve of thermoelectric unit is shown in Figure 2.

According to the above working condition diagram, there is a linear relationship between the generating power range of the heating unit and the change of the thermal power, which is expressed by the following formula [20]:

$$P_{ti}^{e\max} = P_{ti0}^{e\max} - K_1 P_{ti}^{h\max},$$
$$P_{ti}^{e\min} = \begin{cases} P_{ti0}^{e\min} - K_2 P_{ti}^{\text{heat}l}, \\ P_{ti0}^{e\min} - K_2 P_{ti}^{\text{heat}l} + K_3\left(P_j^{h\max} - P_{ti}^{\text{heat}l}\right), \end{cases} \quad (1)$$

where $P_{ti}^{e,\max}$ and $P_{ti}^{e,\min}$ are the minimum and maximum values of generating power when the heating power of the unit is P_{ti}^{heat} at time t. $P_{ti0}^{e,\min}$ and $P_{ti0}^{e,\max}$ are the minimum and maximum values of generating power when the unit is in condensate mode. P_{ti}^{heat} is a certain heating power of the unit. K_1, K_2, and K_3 are coefficients.

The following formula is the energy relationship between the input and output of the extraction heating unit:

$$Q_{\text{coal}} = \frac{P_E\left(1+\varepsilon_1\right)}{\lambda_{\text{coal}}}, \quad (2)$$

where Q_{coal} is the energy of coal, P_E is the generating power, ε_1 is the thermal power ratio, and λ_{coal} is the unit efficiency.

Through the analysis of the extraction type heating unit, it can be seen that the thermal power unit can automatically respond to the change of heating load, so the controllable thermal power unit can be used to make up for the uncontrollable wind power, form a complementary operation mode, and realize the stable heating of the user end. The necessity of decoupling the mode of "power by heat" to improve the peak load regulation ability of the system is explained.

3.2. Wind Power Prediction considering Uncertainty. The uncertainty of wind power is directly reflected in the large prediction error of wind power. Due to the characteristics of wind energy, such as randomness, volatility, intermittence, and poor controllability and the limitations of existing wind power prediction methods, wind power prediction error is inevitable. Taking the day ahead forecast as an example, the average absolute error of the wind power forecasting software actually put into commercial operation in the global scope is about 14%–20%, which will have a significant impact on the optimization results of the power flow analysis, unit combination, economic scheduling, and other issues considering wind power access.

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \cdot e^{\left((x-\mu)^2/2\sigma^2\right)}. \quad (3)$$

3.3. Objective Function. At present, China has started to implement energy-saving power generation scheduling. The most common scheduling mode is the lowest operating cost and the lowest coal consumption of the system. Therefore, the objective function is the minimum cost, that is, the smallest coal consumption of the system. The short-term operational cost of the heat storage device is very small, so it is unnecessary to consider when modeling. For conventional thermal power units, the coal consumption C_c can be expressed as the secondary form of power generation:

$$C_c(i,t) = a_i \cdot \left(P_{el,i,j}^s\right)^2 + b_1 \cdot P_{el,i,j}^s + c_i, \quad (4)$$

where a_i, b_i, and c_i are the coal consumption coefficients of unit I and the generating power of unit i at time 1.

The relationship between the coal consumption C_e of the extraction unit and the thermal power and electric power of the unit is as follows:

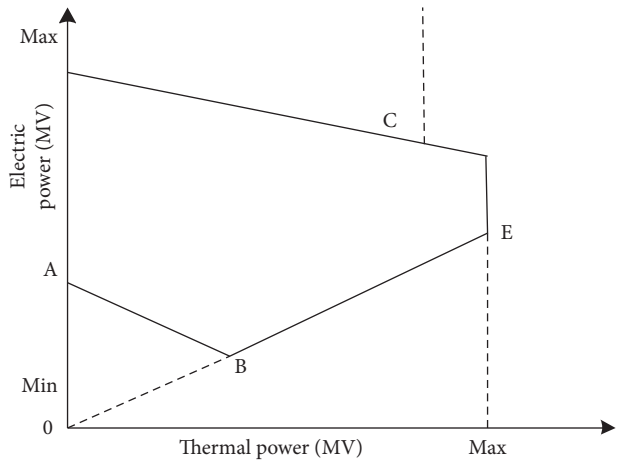


Figure 2: Characteristic curve of thermoelectric units.

$$C_e(i,t) = A_i \cdot \left(P_{el,i,j}\right)^2 + B_i \cdot P_{el,i,j} + C_i, \tag{5}$$

where A_i, B_i, and C_i are coal consumption coefficients of thermal power unit.

Based on the cost function forms of the above two types of units, the following objective functions can be established:

$$\min C(i,t) = \left\{ \sum_{t=1}^{T} \left[\sum_{i \in G\Updownarrow} C_c(i,t) + \sum_{j \in Ge} C_e(j,t) \right] \right\}, \tag{6}$$

where T is the total number of periods; G_c is the set of conventional thermal power units; G_e is the set of cogeneration units.

The coordinated control model considering environment includes the cost of conventional thermal power generating units, cogeneration unit, operation maintenance of wind power generation, coal desulfurization denitration processing, emissions of sulfur dioxide (SO_2), nitrogen oxides (NO_x), and the emission required for joining account [21, 22].

Assume that f is the environmental cost; then, the objective function is as follows:

$$\begin{aligned} f = {} & C_1(P_{it}) + C_2\left(P_{e,it}\right) + C_3\left(P_{w,it}\right) + C_4(A_S, A_N) \\ & + C_5(L_S, L_N), \end{aligned} \tag{7}$$

where C_1 is the generation cost of conventional thermal power units; C_2 represents the generation cost of cogeneration units; C_3 is the operation and maintenance cost of wind turbines; C_4 is the removal and treatment cost of SO_2 and NO_x; C_5 is the pollutant emission penalty cost for the escape of SO_2 and NO_x; P_{it} is the generation power of conventional thermal power unit at time; $P_{e,it}$ is the generation power of the cogeneration unit at any time; $P_{w,it}$ is the generating power of wind turbine at the moment; A_S and A_N, respectively, represent the mass of SO_2 and NO_x generated in the process of power generation; L_S and L_N, respectively, represent the mass of SO_2 and NO_x that have escaped [23].

3.4. Constraints

3.4.1. Power Balance Constraints.

$$\sum_{i \in N} P^s_{el,i,t} + P^s_{w,t} = P_{D,el,t} + \rho \cdot \sum_{k=1}^{M} \left| h_{k,t} \right|, \tag{8}$$

where N is the collection of thermal power units in the area, $N = G_c + G_b + G_e$. The coefficient $P_{sw,t}$ is the wind power of the wind farm connected to the grid at time t in scenario S. $P_{D,el,t}$ is the electrical load of the system at time t. ρ is the power consumption when the electric pump stores unit heat. $h_{k,t}$ is the heat storage and release power of the heat storage tank at time t.

3.4.2. Heating Constraint.

$$\sum_{i \in G^k_e \cup G^k_b} P_{h,j,t} + h_{k,t} \geq P_{D,h,j,t}, \tag{9}$$

where $k = 1, \ldots, M$, M is the total number of heating zones; $p_{D,h,c,t}$ is the total heat load of the k zone thermal power plant at time t; G_a and G_b are the collection of extraction type and backpressure type units of the k zone, respectively.

3.4.3. Start-Stop Output Constraint of Unit.

When a generator set changes from a stagnant state to an operating state or is about to change from an operating state to a stagnant state, the minimum output of the generator set shall be maintained stable.

$$\begin{aligned} & u_{i(t-1)} = 0, u_i = 1, \\ & u_i = 1, u_{i(t+1)} = 0, \\ & P_{it} = P_{i\min}. \end{aligned} \tag{10}$$

3.4.4. Constraints on the Upper and Lower Limits of Unit Climbing Speed.

$$D_i T_1 \leq P^t_i - P^{t-1}_i \leq U_i T_1, \tag{11}$$

where D_i and U_i are the upper and lower limits of unit climbing speed.

3.4.5. Wind Power Output Constraint.

$$P^s_{w,t} \leq P^s_{w,\text{fore},t}, \tag{12}$$

where $P^s_{w,\text{fore},t}$ is the predicted wind power of the wind farm at time t.

3.4.6. Operation Restriction of Heat Storage Device.

$$h_{k,c\max} \leq h_{k,t} \leq h_{k,f\max}, \tag{13}$$

$$h_{k,t} = (1 - \lambda) \cdot S_{h,k,t-1} - S_{h,k,t}, \tag{14}$$

in which formula (13) is the heat storage and release capacity of heat storage device and $h_{k,c\max}$ and $h_{k,f\max}$ are the maximum heat storage and release power of the heat storage device. $h_{k,t}$ is calculated by formula (14); $S_{h,k,t}$ is the heat storage capacity of k zone heat storage device at time t. λ is the heat loss rate of the heat storage tank in a single period.

3.4.7. Capacity Constraints for Heat Storage.

$$S_{h,k,0} = S_{h,k,T}, \tag{15}$$

where $S_{h,k,max}$ is the heat storage capacity of the heat storage device. In the model, it is assumed that the heat storage capacity $S_{h,k,t}$ of the heat storage device at the end of the cycle is equal to its initial heat storage capacity $S_{h,k,0}$.

3.4.8. System Rotation Standby Constraint.

$$\begin{aligned} U_{SR} &= \sum_{i=1}^{N} \min\left(P_{i\max} - P_i(t), P_{ui}\right)^3 P_{sr}, \\ U_{DR} &= \sum_{i=1}^{N} \min\left(p_i(t) - P_{i\min}, P_{di}\right)^3 P_{sr}, \end{aligned} \tag{16}$$

where U_{SR} and U_{DR} are the sum of the positive and negative rotating spare capacity provided by each thermal power and thermoelectric unit, respectively. P_{sr} is the amount of rotation reserve set.

3.5. Solution Method. In this paper, the improved PSO algorithm is used to improve the learning factors, combined with the prior method. For a certain time, if the output value of a unit exceeds the maximum value of the output range of the unit, the output value of the unit in that period is the maximum value of the unit output. If the output value of the unit is from zero to the minimum value of the output range of the unit, the output value of the unit in this period shall be zero or the minimum value of the unit output. If the output value of the unit is lower than zero at this time, it will be zero. After the output of each unit is set in this way, if all start-up units have reached the upper limit of output or the allowable value limit of climbing rate, all start-up units have reached the lower limit of output or the allowable value limit of downward climbing rate, unable to meet the requirements of rotating reserve and additional equality constraints. Then run or shut down the selected unit in turning. Then, the above operations are conducted at all times to achieve the correction of each particle position, to ensure that the solution obtained in each iteration of the particle swarm is feasible. The specific solution steps are as follows (Algorithm 2).

The flowchart of the overall scheme is shown in Figure 3.

4. Example Analysis

4.1. Experimental Description. The example study is divided into four parts, part A is the introduction to the experimental environment, part B is based on interval scene, and two contrast tests were conducted in parts C and D, respectively.

In part C, the specific data of two cogeneration units are shown in the literature [15], the basic parameters of four conventional thermal power units are shown in Table 1, and the predicted power curve of a wind farm is shown in Figure 4 to verify the effectiveness and feasibility of the proposed model. One day is divided into 24 scheduling periods, and a swarm optimization algorithm is used to solve the model to determine the joint output scheduling of cogeneration unit with heat storage.

To make a comparison, the configuration, parameters, and load of the units are consistent with those in the previous part D cogeneration units; 4 conventional thermal power units and 1 wind farm are also selected to verify the effectiveness and feasibility of the proposed model. In the simulation analysis, the operation and maintenance cost of the wind farm in the selection area in this paper is 110 yuan/MW·h, and the cost of desulfurization and denitrification treatment unit for SO_2 and NO_x per kilogram is 3 yuan/kg and 15 yuan/kg, respectively. The generator unit uses coal burning at a price of 520 yuan per ton, and the mass of SO_2 and NO_x released by coal burning per ton is 8 kg and 7 kg, respectively. According to the reference, the treatment efficiency of the desulfurization and denitrification unit is set as 80%, and a penalty fee of 0.65 yuan is imposed per kilogram of SO_2 and NO_x. One day is divided into 24 scheduling periods, and the swarm optimization algorithm is used to solve the model to determine the joint output scheduling of cogeneration unit with heat storage and wind power considering the environmental cost.

4.2. Wind Power Prediction Based on Interval Scene. The test results show that the expected value of the predicted relative error $\mu_f = 0.0203$, and the standard deviation of the predicted relative error $\sigma_f = 0.0667$. Through the calculation of probability distribution, the prediction error is mainly distributed between 16% and 22%, reaching 99.52%. It is discredited in steps of 0.02, the curve with the largest probability of occurrence is called 1.0 curve, and the curves on both sides are called 1.02, 1.04, . . ., 0.98, 0.96, ... For the statistical wind power data of a wind farm in 2018 during the heating period, see Figure 5 and Table 2.

According to the distribution characteristics of wind power prediction error and considering the obvious seasonal difference of wind power, the discretization interval of wind power prediction can be simplified as a curve with I number, corresponding to a scene with I number. Each curve with I number corresponds to an empirical probability value f_i, where $I = 1, 2, \ldots, I_{\max}$ (upper limit to the lower limit in order), indicating the probability of its possible occurrence, and the sum of all f_i is 1.

4.3. Coordinated Scheduling of Heat Storage Thermoelectric Units. For the coordinated scheduling of heat storage thermoelectric units, through the proposed solution, it is found that the lowest comprehensive cost is 495600 yuan/d,

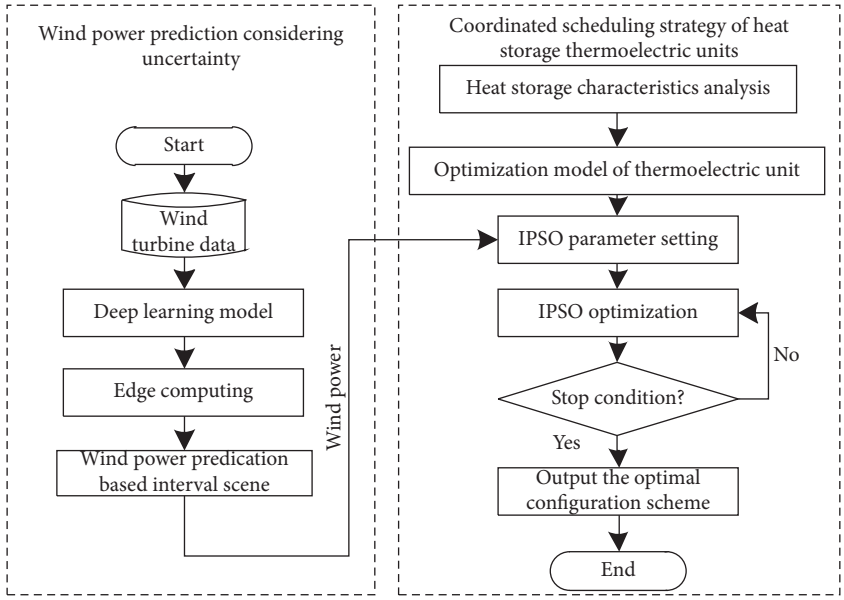


Figure 3: The flowchart of the overall scheme.

(1) Initialization
(2) **for** $WT_i = 1$
(3) Send model DL_i to terminal controller C_t.
(4) **end for** $WT_i = n$
(5) End of Initialization
(6) **repeat**
(7) **for** $S_t = 1$
(8) collects monitoring data
(9) Send data to C_t.
(10) edge calculation
(11) Results compiled and stored
(12) **end for** $S_t = m$
(13) Send control instructions
(14) **for** $WT_i = 1$
(15) obtains the control instructions
(16) transmits the processing results
(17) **end for** $WT_i = n$
(18) **until** achieve control effect

Algorithm 1: Wind turbine measurement and control system.

and the power output scheduling of each unit within the scheduling day is shown in Figure 6. It can be seen from Figure 6 that the sum of electric output and thermal output of the unit at each time under this dispatching mode is equal to the electric load and thermal load, to meet the balance of heat, power supply, and demand. In the peak period of the heat load (1–7 h; 21–24 h), the electric load is in the low period, and there is an obvious contradiction between power on and heat demand. Because the fuel cost of cogeneration unit is significantly lower than that of thermal power unit and its thermal output and electrical output have certain coupling characteristics, in the peak period of electrical load, in order to reduce the output of thermal power unit, it is necessary to increase the electrical output of cogeneration unit. The heat storage device releases heat during the peak period of the electric load, to reduce the thermal output of the cogeneration unit, then achieves the purpose of increasing its electricity output, and achieves the goal of reducing the comprehensive operation cost.

From the curve in Figure 7, it can be seen that the period of no downregulation capacity of thermal power generation mostly occurs in the period of low-level operation of the power grid and almost coincides with the period of heat storage input at night. When the period of no downregulation capacity coincides with the period of heat storage input, it is less than the period of no downregulation capacity or heat storage input. It shows that when the power grid is in a low period and the thermal power has no

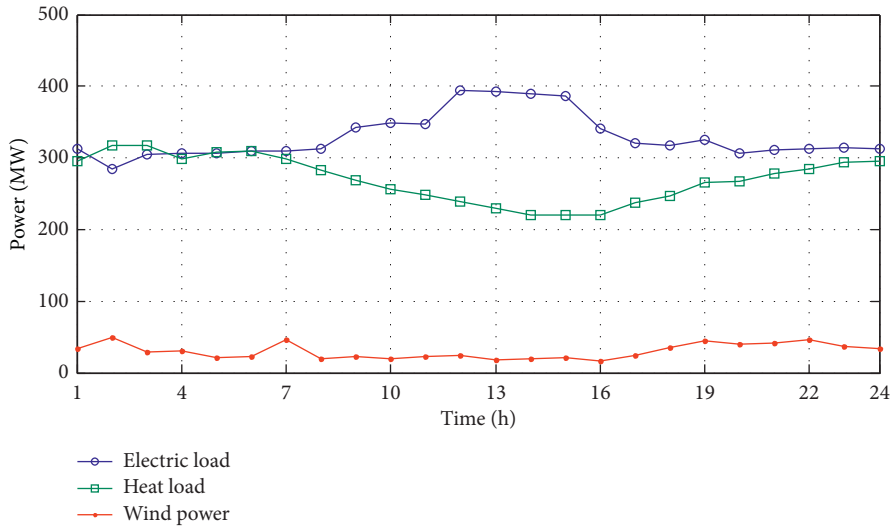


Figure 4: Prediction curve of thermal load, electric load, and wind power output.

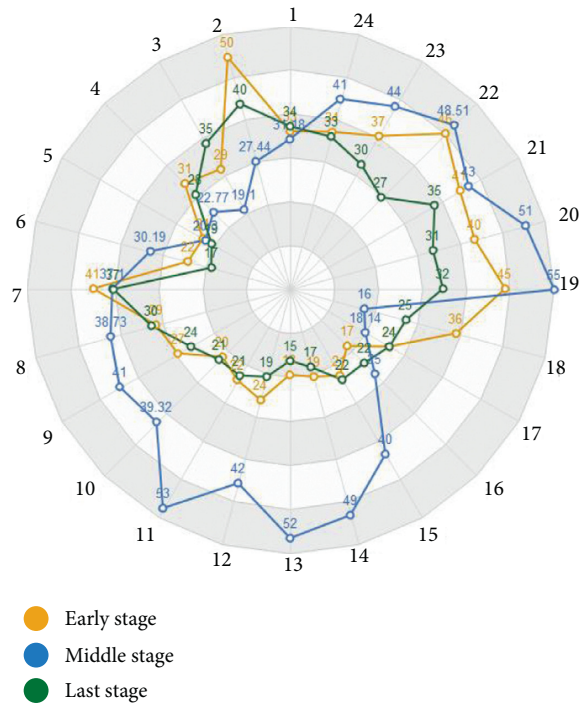


Figure 5: Statistical wind power in heating period.

downregulation capability, there is no need to put heat storage into operation. Although this rarely happens, one explanation is that the wind power output is in the process of decreasing.

4.4. Coordinated Scheduling of Heat Storage Thermoelectric Units considering Environmental Costs. After calculation, the total cost of the coordinated control model considering environmental costs is 576,000 yuan per day, including 104,000 yuan for the operation of the desulfurization and denitrification device and 0.14 million yuan for the penalty charged for the gas escaping SO_2 and NO_x. Figure 8 shows the electric load and output curves of the thermal storage unit, and Figure 9 shows the wind power consumption of different models. Table 3 is the comparison of wind power consumption between different models.

Figure 9 shows the main running time of wind power is 8 ~ 16 h; by the time, the internal heat load is in a low state, and the wind power is in its peak, wind curtailment still exists subject to the operational cost of wind power.

(1) Initialization
(2) Parameters setting: particle number n_p, iteration number $g_{\max}$, constraints: cons.
(3) Initialize the position p_i and speed s_i
(4) End of Initialization
(5) **repeat**
(6) optimal value calculation f_g
(7) **for** $g = 1$
(8) **for** each n_i
(9) update the p_i, s_i
(10) **end for** all particle updated
(11) new particles judgment
(12) **if** particles $\in$ cons
(13) next step
(14) **if** particles $\notin$ cons
(15) particles correction and judgment
(16) **if** particles $\notin$ cons
(17) replace the last particle value
(18) determine the optimal output value f_g
(19) $g = g + 1$
(20) **until** $g = g_{\max}$

Algorithm 2: Improved particle swarm optimization (PSO).

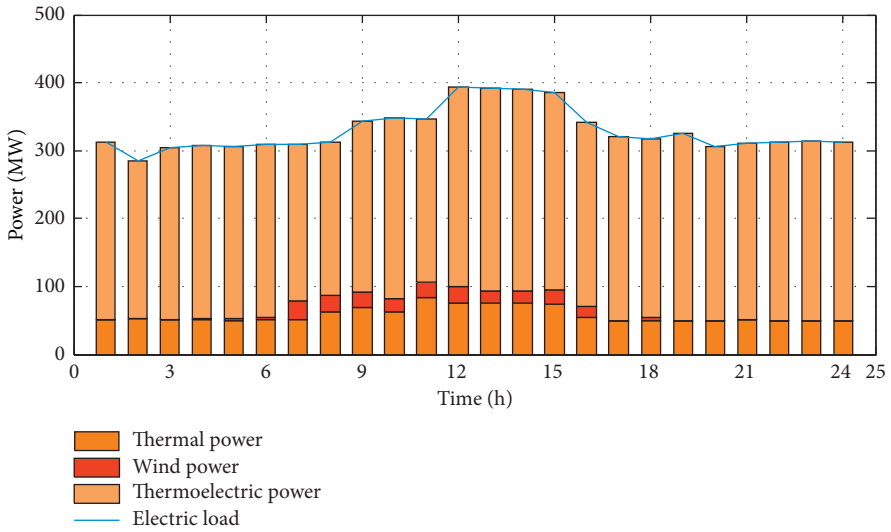


Figure 6: Generating power dispatching value of unit.

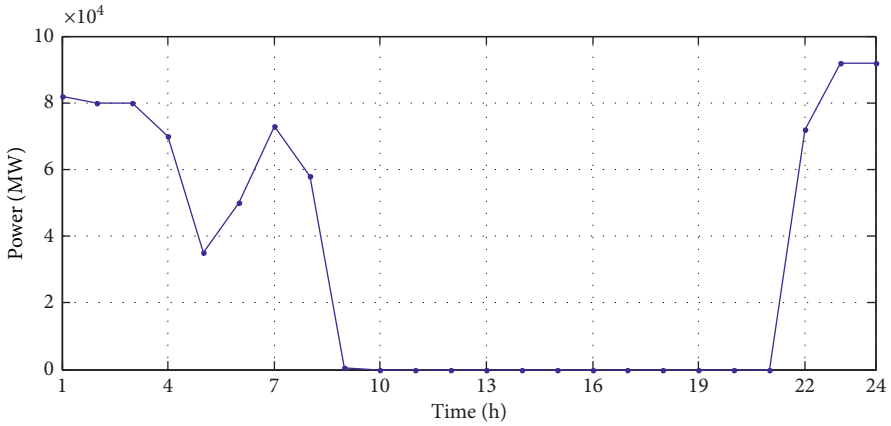


Figure 7: Equivalent heat-load curve of heat storage device of cogeneration unit.

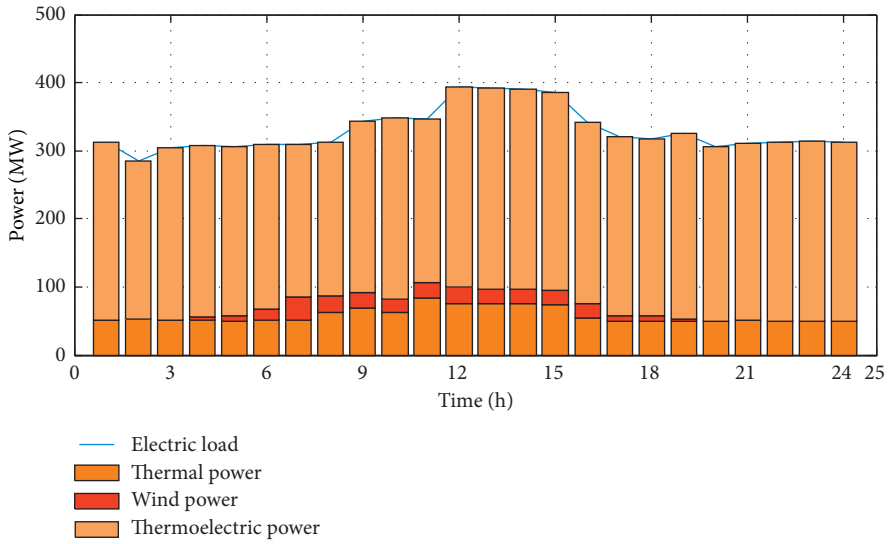


Figure 8: Electric load and output curve of thermal storage unit.

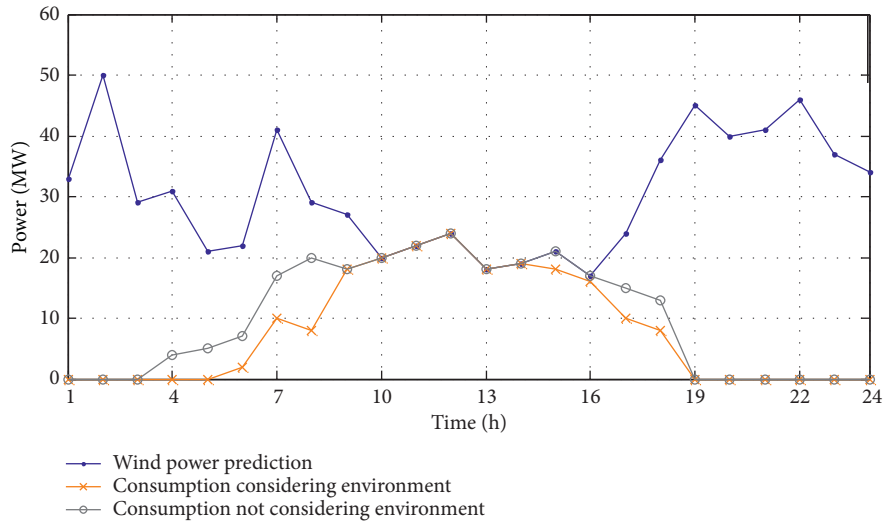


Figure 9: Wind power consumption of different models.

Table 1: Parameters of the conventional thermal power unit.

Thermal power unit	Upper limit of output P_{max} (MW)	Lower limit of output P_{min} (MW)	Unit climbing rate r_{ui} (MW/h)	Fuel cost factor a_l (yuan/MW^2)	b_l (yuan/MW)	c_l (yuan)
1	50	25	25	0.011	18.84	11.50
2	35	10	18	0.070	25.97	32.87
3	30	10	15	0.030	38.02	18.11
4	40	12	20	0.010	13.01	7.07

In the traditional coordinated control model without environmental costs, the daily cost of power generation is 471,000 yuan, and the wind power consumption is 21.9% of the predicted value, while in the coordinated control model considering the environment and wind power operation and maintenance, the daily cost of power generation is 576,000 yuan, and the wind power consumption is 30.7% of the predicted value. Obviously, in the coordinated control model that takes into account environment and wind power operation and maintenance, wind power operation and maintenance costs and pollution control costs will increase the power generation cost to a certain extent, but the emission of pollution gases will significantly change, among which SO_2 emission will be reduced by 5.8 t and NO_x emission will be reduced by 5.2 t.

Table 2: Distribution of wind power in heating period.

Scene I	Empirical probability value f_i
Early stage	0.21
Middle stage	0.47
Last stage	0.32

Table 3: Environmental cost on wind power consumption and pollution emission.

Environmental costs	Power generation cost/ten thousand yuan	Wind power consumption (%)	SO_2 pollution (t)	NO_x pollution (t)
Considering	57.6	30.7	1.4	1.1
Not considering	47.1	21.9	7.2	6.3

5. Conclusions

In view of the problem that a large number of winds are abandoned due to the lack of peak load regulation capacity caused by the operation of cogeneration unit with heat as power, this paper proposes a research on coordinated scheduling strategy of heat storage thermoelectric units based on wind power data acquisition system using edge computing. A wind power data measurement and process system based on edge computing and deep learning is introduced, and a wind power prediction method based on interval scenario is adopted to process the uncertainty of wind power. Finally, the coordinated scheduling strategy of thermal power units is applied to typical thermal power units, and the swarm optimization is used to verify the effectiveness of the strategy. By reasonably dispatching conventional thermal power units, cogeneration units, and wind power, the comprehensive operation cost can significantly reduce, the consumption of wind power can increase, and the output of thermal power units and cogeneration units can reduce, to achieve the goal of energy conservation and emission reduction.

Data Availability

The data used to support the findings of this study are currently under embargo while the research findings are commercialized. Requests for data, 12 months after publication of this article, will be considered by the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] T. Nuytten, B. Claessens, K. Paredis, J. Van Bael, and D. Six, "Flexibility of a combined heat and power system with thermal energy storage for district heating," *Applied Energy*, vol. 104, pp. 583–591, 2013.

[2] D. Xu, Q. Ding, G. D. Huang et al., "Cogeneration unit dynamic scheduling model considering peak-load regulation ability," *China Electric Power Research Institute*, vol. 45, no. 11, pp. 59–64, 2017.

[3] G. Yuan, L. Wang, B. Wang et al., "Optimal dispatch of heat-power load and economy benefit analysis based on decoupling of heat and power of virtual power plant," *Proceedings of the CSEE*, vol. 37, no. 17, pp. 4974–4985, 2017.

[4] H. R. Abdolmohammadi and A. Kazemi, "A benders decomposition approach for a combined heat and power economic dispatch," *Energy Conversion and Management*, vol. 71, pp. 21–31, 2013.

[5] L. Chen, F. Xu, X. Wang, and Y. Min, "Implementation and effect of thermal storage in improving wind power accommodation," *Proceedings of the CSE*, vol. 35, no. 17, pp. 4283–4290, 2015.

[6] F. Xu, M. Yong, L. Chen et al., "Combined electricity-heat operation system containing large capacity thermal energy storage," *Proceedings of the CSEE*, vol. 34, no. 29, pp. 5063–5072, 2014.

[7] Y. Cui, Z. Chen, G. Yan, and Y. Tan, "Coordinated wind power accommodating dispatch model based on electric boiler and CHP with thermal energy storage," *Proceedings of the CSEE*, vol. 36, no. 15, pp. 4072–4081, 2016.

[8] S. Mueller, R. Tuth, D. Fischer, B. Wille-Haussmann, and C. Wittwer, "Balancing fluctuating renewable energy generation using cogeneration and heat pump systems," *Energy Technology*, vol. 2, no. 1, pp. 83–89, 2014.

[9] B. V. Mathiesen and H. Lund, "Comparative analyses of seven technologies to facilitate the integration of fluctuating renewable energy sources," *IET Renewable Power Generation*, vol. 3, no. 2, pp. 190–204, 2009.

[10] G. Streckiene, V. Martinaitis, A. N. Andersen, and J. Katz, "Feasibility of CHP-plants with thermal stores in the German spot market," *Applied Energy*, vol. 86, no. 11, pp. 2308–2316, 2009.

[11] S. Rinne and S. Syri, "The possibilities of combined heat and power production balancing large amounts of wind power in Finland," *Energy*, vol. 82, pp. 1034–1046, 2015.

[12] C. Chen, G. Li, G. Zheng et al., "Applied research of artificial intelligence in power grid enterprises under the context of ubiquitous power internet of things," *Journal of Jiangsu University of Technology*, vol. 6, no. 1, pp. 205–220, 2019.

[13] S. Li, L. D. Xu, and S. Zhao, "The internet of things: a survey," *Information Systems Frontiers*, vol. 17, no. 2, pp. 243–259, 2015.

[14] H. Gao, C. Liu, Y. Li, and X. Yang, "V2VR: reliable hybrid-network-oriented V2V data transmission and routing considering RSUs and connectivity probability," *IEEE Transactions on Intelligent Transportation Systems (T-ITS)*, vol. 99, 2020.

[15] H. Li, K. Ota, and M. Dong, "Learning IoT in edge: deep learning for the internet of things with edge computing," *IEEE Network*, vol. 32, no. 1, pp. 96–101, 2018.

[16] H. Gao, W. Huang, and Y. Duan, "The cloud-edge-based dynamic reconfiguration to service workflow for mobile ecommerce environments: a QoS prediction perspective," *ACM Transactions on Internet Technology*, vol. 21, no. 1, pp. 1–23, 2020.

[17] M. D. Deassuncao, A. D. S. Veith, and R. Buyya, "Distributed data stream processing and edge computing: a survey on resource elasticity and future directions," *Journal of Network & Computer Applications*, vol. 103, no. 2, pp. 1–17, 2018.

[18] Y. Li, H. Zhang, X. Liang, and B. Huang, "Event-triggered-based distributed cooperative energy management for multienergy systems," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 4, pp. 2008–2022, 2019.

[19] Y. Li, D. W. Gao, W. Gao, H. Zhang, and J. Zhou, "Double-mode energy management for multi-energy system via distributed dynamic event-triggered Newton-raphson algorithm," *IEEE Transactions on Smart Grid*, vol. 11, 2020.

[20] T. Li, R. Huang, L. Chen, C. S. Jensen, and T. B. Pedersen, "Compression of uncertain trajectories in road networks," *Proceedings of the VLDB Endowment*, vol. 13, no. 7, pp. 1050–1063, 2020.

[21] X. Ma, H. Gao, H. Xu, and M. Bian, "An IoT-based task scheduling optimization scheme considering the deadline and cost-aware scientific workflow for cloud computing," *EURASIP Journal on Wireless Communications and Networking*, vol. 249, 2019.

[22] X. Yang, S. Zhou, and M. Cao, "An approach to alleviate the sparsity problem of hybrid collaborative filtering based recommendations: the product-attribute perspective from user reviews," *Mobile Networks and Applications*, vol. 25, no. 2, pp. 376–390, 2020.

[23] H. Gao, L. Kuang, Y. Yin, B. Guo, and K. Dou, "Mining consuming behaviors with temporal evolution for personalized recommendation in mobile marketing apps," *Mobile Networks and Applications (MONET)*, vol. 25, no. 4, pp. 1233–1248, 2020.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 8830879, 13 pages
https://doi.org/10.1155/2021/8830879

Research Article

An Intelligent Offloading System Based on Multiagent Reinforcement Learning

Yu Weng, Haozhen Chu, and Zhaoyi Shi

College of Information Engineering, Minzu University of China, Beijing 100081, China

Correspondence should be addressed to Haozhen Chu; 595158846@qq.com

Received 18 August 2020; Revised 9 November 2020; Accepted 11 March 2021; Published 25 March 2021

Academic Editor: Luigi Coppolino

Intelligent vehicles have provided a variety of services; there is still a great challenge to execute some computing-intensive applications. Edge computing can provide plenty of computing resources for intelligent vehicles, because it offloads complex services from the base station (BS) to the edge computing nodes. Before the selection of the computing node for services, it is necessary to clarify the resource requirement of vehicles, the user mobility, and the situation of the mobile core network; they will affect the users' quality of experience (QoE). To maximize the QoE, we use multiagent reinforcement learning to build an intelligent offloading system; we divide this goal into two suboptimization problems; they include global node scheduling and independent exploration of agents. We apply the improved Kuhn–Munkres (KM) algorithm to node scheduling and make full use of existing edge computing nodes; meanwhile, we guide intelligent vehicles to the potential areas of idle computing nodes; it can encourage their autonomous exploration. Finally, we make some performance evaluations to illustrate the effectiveness of our constructed system on the simulated dataset.

1. Introduction

With the rapid development of intelligent vehicles, the vehicular network based on artificial intelligence has attracted extensive attention; its wide application has encouraged researchers all over the world to develop more applications, but there is still a problem to compute-intensive services on vehicles; as a promising solution, mobile edge computing (MEC) lets users upload services to edge computing servers (e.g., offloading), which can reduce the computing load of the terminal, just like roadside unit (RSU), building cloud, and other entities with computing [1]. Caching and network communication capabilities can become the MEC platform; they can not only reduce communication delay but also ease the workload of central BS.

However, the performance of traditional methods [2, 3] will decline sharply in the vehicular network; it is urgent to develop an effective MEC intelligent offloading solution. In recent years, machine learning is mainly used in intensive computation tasks, such as navigation and automatic driving, except in MEC; it is very difficult to build a suitable model because many vehicles are participating in the offloading system. Deep reinforcement learning (DRL) uses agents as interactive entities to learn strategies from the environment. DRL has been applied to the MEC offloading system [4], mainly for task scheduling and resource allocation and for studying DRL-based networking and caching [5, 6].

We focus on the tradeoff between the QoE of users and the profit of servers [7]; we need to schedule the corresponding edge computing nodes for intelligent vehicles; this process is similar to the order dispatch in modern taxi networks [8–10]; they provide information about passenger demand and taxi movement for finding the most appropriate pairs; some car-hailing services provide significant improvements over traditional taxi systems in terms of reducing taxi cruising time and waiting time [10, 11]; therefore, the online car-hailing is a vivid scene [8, 12–14] which can be migrated to the edge computing. Xu et al. [15] introduced reinforcement learning algorithms to have foresight, which can achieve better profits for servers and better services for users, but there is a limitation that each server pair with the user in a finite distance, it will put user in an idle state; if this state can be avoided, we can achieve

better. For this reason, we regard our goal as a decision problem for multiagent systems, which is implemented using the classical deep deterministic policy gradient (DDPG) [16] in reinforcement learning.

Furthermore, we introduce a central system with our optimization KM [17]; the algorithm performs better matching between users and servers. The positions of users are determined separately, and users cannot get any information from others at the beginning; we design a communication module referring to [18]; then, they can share information beneficial to the next decision.

The remainder of the paper is organized as follows. We provide a brief overview of the background and related works in Section 2, we will show the system architecture in Section 3, Section 4 describes improvement and algorithm description, Section 5 explains the experimental details and results, and finally, in Section 6, we summarize all the work.

2. Related Works

This paper proposes a multiagent reinforcement learning algorithm with global scheduling and applies it to the scene of intelligent offloading about edge computing. The following content introduces some related researches and the application of a multiagent decision algorithm.

An approach is proposed in [19] to find the optimal auction for computation resources of edge computing in blockchain networks; it uses a monotone transform function to anonymize prices. To decrease the number of duplicated contents in networks, a DRL method for caching in smart cities is designed in [20]; the agent in the system collects the status from MEC servers and learns to choose the optimal action to get the best policy for resource arrangement. Qi et al. [21] constructed an intelligent offloading system for vehicular edge computing by leveraging deep reinforcement learning; its communication and computation states are modeled by finite Markov chains, task scheduling, and resource allocation. The strategy is formulated as a joint optimization problem to maximize users' QoE.

Classical scheduling algorithms, such as greedy methods, are widely used in large companies, such as finding the nearest driver to serve customers [22], or using a first-in, first-out queue strategy [23]; although they are easy to dispatch, it only obtained nice profits in the short term; the spatiotemporal sequence does not match the supply-demand relationship in the long-term operation, which will lead to some suboptimal results [15]. Later, this dispatching process improved by using the central system through the taxi GPS trajectory and brute force method for the best path recommendation [24, 25], considering whether the driver took the initiative to find hot spots to provide the scheduling strategy [11] and focusing on minimizing total customer waiting time by simultaneously scheduling multiple taxis and allowing taxis to exchange their booking tasks [9], taking into account the overall benefits of a more global and far-sighted approach [26]. These methods have been put into practice and have shown valuable effects.

The vehicular network is often divided into several zones, each zone has one BS with abundant computation resources, and the BS can play the role of the central system. We refer to the work of [15] in global scheduling, which constructs a set of preference functions and calculates the corresponding different agents in each time slice with a global view. The average preference function in the region is conducive to the formulation of the reward and punishment process in the subsequent multiagent decision algorithm. We have adopted KM algorithm in order matching, which has a wide range of applications in the dispatch, network communication, system architecture, etc. [27], which is usually used for minimum weight matching [28]; however, it can also be achieved by setting up negative samples to maximize the sum of weights [29]. In this model, we use the KM algorithm, taking advantage of preference function, as a metric to form a strategy for node scheduling.

The multiagent intensive learning aim includes the learning stability of a single agent and the adaptability to the behaviors from other agents [30, 31]. Adaptability ensures that performance can be improved when other agents change their policies [32]. Some aims can be extended to dynamic games by requiring a phased satisfaction condition for all states of the dynamic game. In this case, the aim is based on the stage strategy rather than the global strategy and the expected return rather than the reward [18]. Multiagent systems have been extensively studied in various fields, such as robotics teams, resource management, distributed control, games, e-commerce [33], and several MARL algorithms that have been discussed, indicating that these algorithms combine time difference RL with game theory solvers [34, 35]. For static games generated in the state of dynamic environments [36], Alibaba also uses its system in the environment of a multiscenario e-commerce system to give priority to recommending products of interest to customers and obtain the maximum benefit of the system [37, 38].

In general, the research on multiagent reinforcement learning problems is complicated and computationally intensive. However, this model adds many idealized constraints and excludes irrelevant factor variables, such as data transfer rate and propagation delay. Simplified model is beneficial to implement our reinforcement learning algorithm.

3. General Architecture

The vehicle network in the city can be divided into several zones according to streets or other criteria; each zone has one central BS with abundant computation resources, as shown in Figure 1. We mainly analyze one of these areas; RSUs are equipped with MEC servers and have their signal coverage; therefore, intelligent vehicles can only upload services within a certain range of RSUs. In the case of no RSU nearby, the vehicle can upload its services to the BS directly, but in this model, we only take into account the matching relationship between vehicles and RSUs, because our goal is to schedule the edge computing nodes.

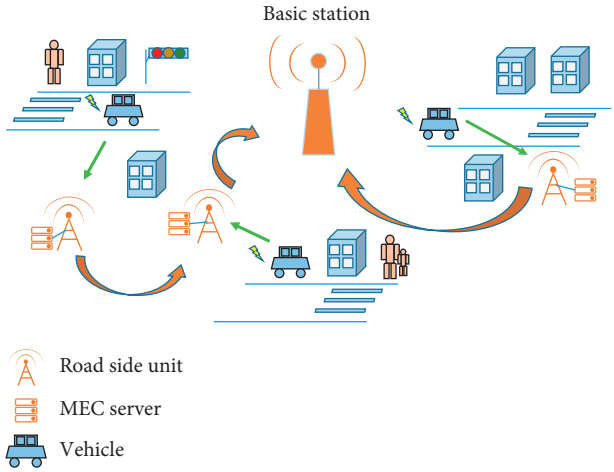


Figure 1: The architecture of MEC-based vehicular network.

Here, we introduce the model design and matching strategy of advantage algorithm at first; they are the basis of our work, but the advantage algorithm cannot cover some special situations. Inspired by [6], we have proposed our architecture to improve it.

3.1. Model Design and Matching Strategy. If we simplify the model for reinforcement learning, an intelligent vehicle can be regarded as an independent agent; its mobile strategy can be abstracted as a Markov decision process (MDP).

The model here refers to the toy example used in [18]; we did not set a more realistic vehicle movement and increase the grid size; because the actual model will consider too many details, it is difficult to reproduce a reasonable model and apply our algorithm; the algorithm compared in [18] is trained on the toy example; then, it was applied to a more complex simulation environment; we also train through a simple model and verify the advantages of our proposed algorithm. We build a simulation model (Figure 2) corresponding to each spatiotemporal state; it can show the movement strategy of the agent in each state.

Matching strategy uses a preference value as the weight between the vehicle i and the RSU j, which is calculated as the value function of the vehicle in the future state (when its service offloaded to the RSU j has finished) minus the value function of the vehicle in the current state; we call this formula of weight as the calculation of preference [17]; call it as an advantage function:

$$A_\pi(i,j) = \gamma^{\Delta t_j} V\left(s_{ij}'\right) - V\left(s_i\right) + R_\gamma(j). \tag{1}$$

Our second goal is to maximize the QoE of vehicles while guaranteeing the profits of network operators (i. e., RSUs). The network operators are in charge of the wireless access network and the mobile core network; the overall profit from finished services will be affected by the amount of transferred data and energy consumption of calculating in reality, which is not involved in our model; for the sake of simplicity, we use some random distributions to set the certain profit at each RSU instead of that; here, we use $R_\gamma(j)$ to represent.

The preference function can generate a corresponding value through (1) to determine the best matching between each vehicle and the RSU; the matching rules are

$$\begin{aligned} &\arg\max_{a_{ij}} \sum_{i=1}^{m}\sum_{j=1}^{n} A_\pi(i,j)a_{ij}, \\ &s.t. \quad \sum_{i=1}^{m} a_{ij} \le 1, \quad j = 1,2,3,\ldots,n, \\ &\qquad \sum_{j=1}^{n} a_{ij} \le 1, \quad i = 1,2,3,\ldots,m, \end{aligned} \tag{2}$$

where a_{ij} depends on the pairing relationship. If RSU j is matched with vehicle i, then a will be 1, i presents all the unmatched vehicles of this time slot, and j means that these RSUs can be allocated. During the RSU allocation phase, vehicles that are not matched to RSUs will be treated as if they were waiting to enter the next time slot. Its final effect is shown in Figure 3.

Moreover, RSU can only be dispatched to one vehicle within 2 grids (as shown in Figure 4), which can reduce the loss of profits caused by the partial allocation and also facilitate the timely response to more vehicles.

However, there is a special situation in this model. If there is no valid RSU in the area when the distribution of RSUs is too sparse, the central system will not allocate a RSU for any vehicle. This leads to the idle state, but actual vehicle can explore around to seek for some RSUs which have stronger signal source during this time.

3.2. Modules and Intermodule Connection. We design an overall architecture based on [15] to avoid the idle state; it analyzes areas having more RSUs in the future and guides a route for vehicles which are in the idle state; they can rush to potential high preference areas in advance; this way will lead to high QoE of users and the well profit ultimately. The analysis of the architecture will refer to historical memory and strategies from different agents.

To ensure that the parameters of the network can finally converge stably, our target network modules are constructed according to the DDPG algorithm. The loss function of the value network narrows the gap between the two value network modules and soft updates the parameters of the target value network. The same update method also applies to the parameter update of the target policy network.

He et al. [6] applied the reinforcement learning to the multiscene recommendation system of e-commerce [39], each scene is regarded as an agent; the different scenes are arranged by the respective strategies of the agents; its center system works as a critic to evaluate the profits from overall scenes. Our model requires adjustments to the driving path of different agents to get better overall profit; this point is very similar to our model.

Inspired by [6], we introduce actor-critic to our architecture with DDPG; an algorithm named MARDDPG (multiagent recurrent deep deterministic policy gradient) is proposed, refer to Figure 5.

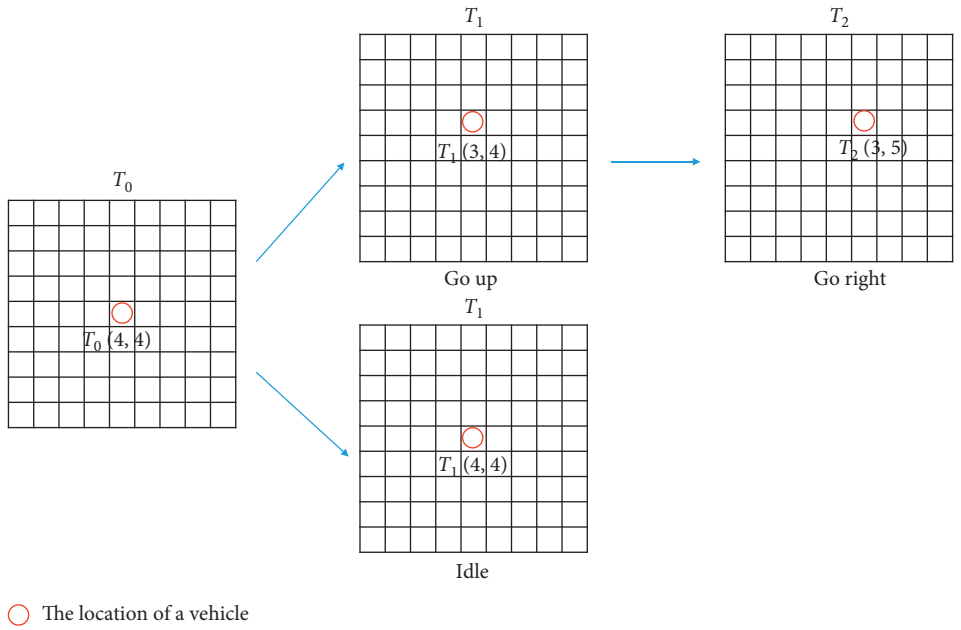


Figure 2: The 9×9 grid of the simulation model; the vehicle can choose to go in one direction or stay in place.

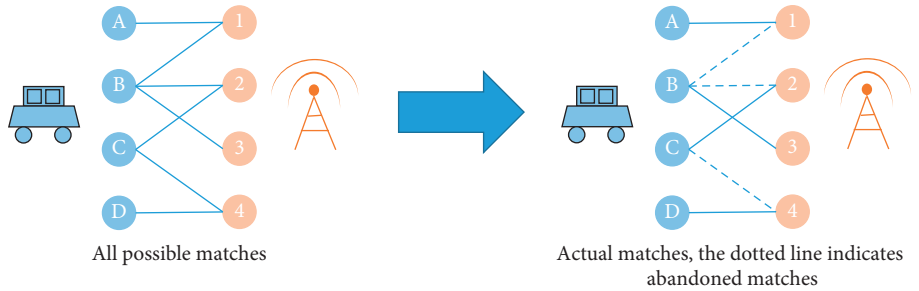


Figure 3: Matching the vehicle to the RSU by KM algorithm.

There are three important modules designed to the collaboration of multiple agents, respectively, a global critic module, independent agents, and a communication mechanism.

3.2.1. Actor Module. Each agent actor$_i$ in the group module called Actor is a separate actor module that accepts the local observation o_t^i such as the location and the shared message m_{t-1} from the previous moment and chooses one action. The behaviors of vehicles are a finite set of discrete actions: up, down, left, and right; we define the behavior variable as a preference probability of different behaviors at that moment:

$$p_i = \left(w_1^i, \ldots, w_{n_i}^i\right), \; p_i \in R_i^n,$$
$$\text{s.t.} \quad \sum_{j=1}^{n_i} w_j^i = 1. \tag{3}$$

Therefore, each behavior is a n_i dimension vector; this vector will eventually select the corresponding behavior with the highest probability of preference as the next behavior of the agent.

Inspired by DDPG-related work, a method of determining the strategy is used instead of a random strategy. The actor$_i$ of each agent corresponds to the function $\mu_i(s_t; \theta_i)$, where the parameter is θ_i, and the function maps a state to a behavior. At time t, the agent decides its behavior according to the actor$_i$ network:

$$a_t^{i_t} \approx \mu^{i_t}\left(m_{t-1}, o_t^{i_t}; \theta_{i_t}\right), \tag{4}$$

where $s_t \approx \{m_{t-1}, o_t\}$ represents the approximate global state; the behavior of actor$_i$ depends on both the message m_{t-1} and its current observation o_t.

To be exact, each agent here corresponds to the vehicle in the idle state of the forecast period, and the period during which the vehicle whose service is still being computed will not be taken into account; each agent i will be according to its strategy $\mu_i(s_t)$ each time the behavior a_t^i is chosen, and then an immediate reward $r_t^i = r(s_t, a_t^i)$ was obtained from the environment. After all the observations and rewards of the agent are accumulated, the state changes from $s_t$to update to s_{t+1}.

3.2.2. Critic Module. The purpose of multiple agents is to achieve the global maximum profits. We pass a global critic

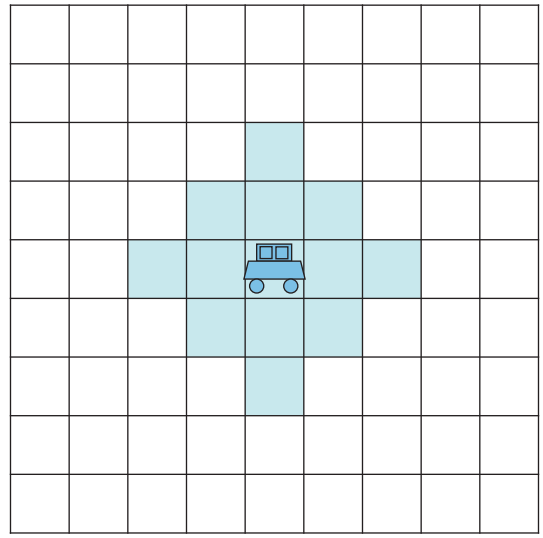


Figure 4: The request range (grid covered by blue) from one smart vehicle.

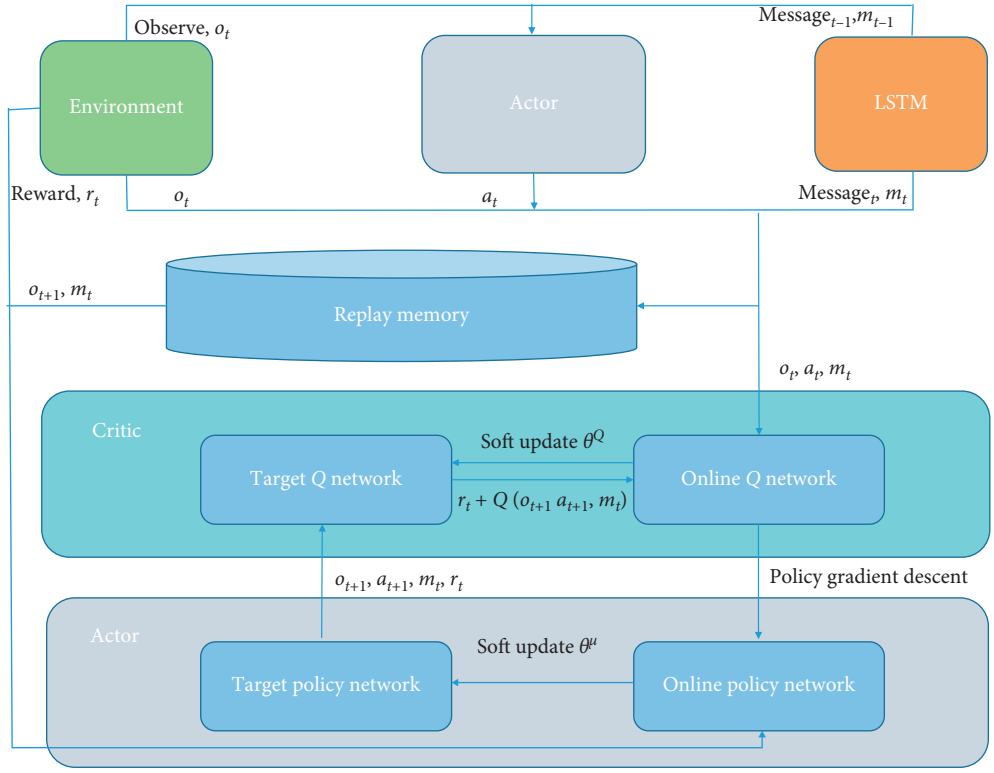


Figure 5: An illustration of the proposed architecture.

behavior value function $Q(s_t, a_t^1, a_t^2, \ldots, a_t^n)$ to evaluate the overall profit; each agent performs a local behavior after obtaining local observations.

A critic network is designed to fit the behavioral value function, which is used to assess the impact of overall behavior on future expectations when taking actions. Because all agents share message, we use a global evaluation function for

$$Q = \sum_{j=1}^{n} q_j, q_j = f\left(s_t, a_t^j; \varphi\right). \tag{5}$$

Each q_j represents the assessment of the behavior of each agent in the global state by the critic network, which ultimately needs to be summed to form a total assessment of agents. The details of the actor-critic network inside an agent are shown in Figure 6.

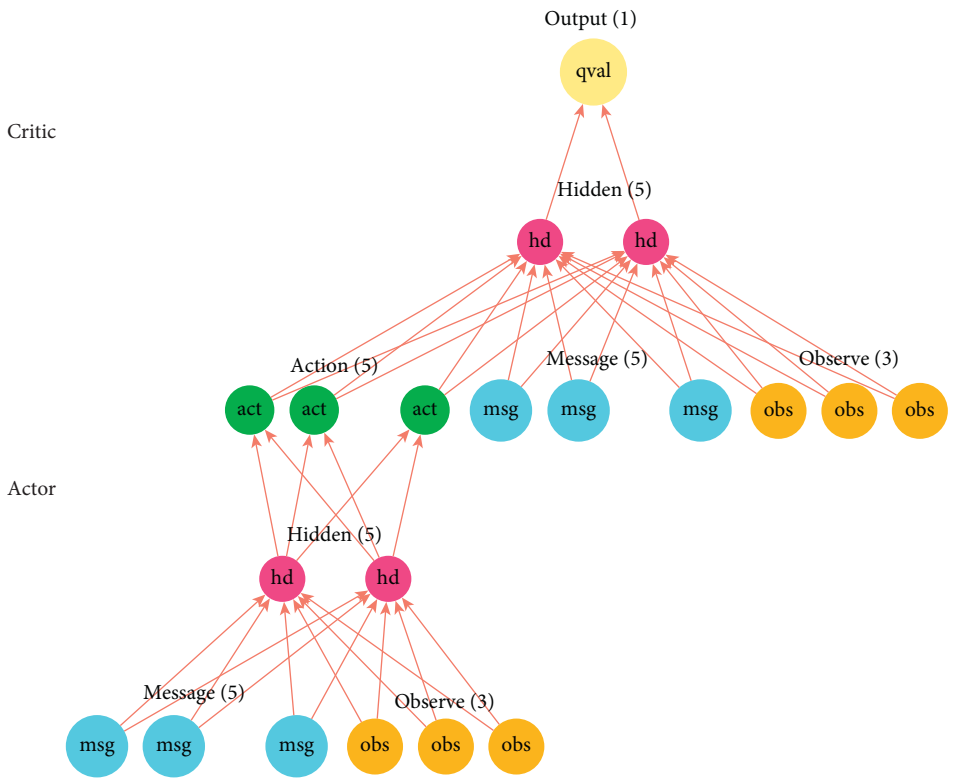


Figure 6: An actor-critic network inside an agent.

The local observation corresponds to a three-dimensional vector, the five-dimensional memory information vector of the previous moment is used as input, and the five-dimensional motion vector is output and combined with local observation as input, finally the critic network outputs one-dimensional evaluation vector, the middle hidden layers are all consists of 5 neurons.

3.2.3. Communication Mechanism. In our communication mechanism based on long-short-time memory network (LSTM), we refer to the setting of LSTM in [18]; the author applies it to the multiscenario ranking task, different scenarios are regarded as an agent, the LSTM encodes all local observations and actions of all agents into a message vector, and the message will be sent between agents. We input all observations and behaviors of all agents into the same communication module simultaneously; it will generate a message vector that memorizes the global information of the current moment. The vector will be sent to the actor module of the different agents at the next moment and form a new input together with their observation information, which has an effect of cooperating. The message m_{t-1} is updated in the communication module, which memorizes the observation o_t and behavior a_t of different agents at the same time.

Because of this mechanism, each agent's decision is based not only on its state and previous behavior but also on the state and behavior of other agents. This communication mechanism gives agents the ability to approximate the state of the global environment, allowing them to make more global and long-term decisions. Then, the o_t, a_t, and m_{t-1} of all the agents at each moment are stored in the playback memory area and transmitted to the global critic module Critic. The online behavior value function $Q(m_{t-1}, o_t, a_t)$ evaluates the effect of the behavior a_t when each agent accepts the message m_{t-1} and the local state o_t.

4. Improvements and Algorithm Description

4.1. Optimized Preference Function. The preference function (1) is only used as a weight for matching. We will calculate the average preference value from one region as improved preference function; it reflects the appearance probability of valid RSUs. A larger value means that this region will have more services which can produce higher overall profits; the system will recommend vehicles rush to this region in advance. This behavior recommendation will reduce the amount of vehicles which are in the idle state; even if there is no valid RSUs, the matching strategy cannot be performed; the agent will actively seek for RSUs. The formula of improved preference is as follows:

$$F_{x,y} = \sum_{j=1}^{n_i} \frac{A_\pi\left(\text{order}_j, \text{car}_i\right)}{n_i}, \tag{6}$$

where n_i is the number of RSUs that had finished the services within a certain distance of the agent i; A_π is based on a preference function (1). Later, we call $F_{x,y}$ as the preference density.

Replay buffer stores data about (m_t, o_{t+1}, r_t) from every episode at each time step; the policy network randomly extracts the sample data from the buffer in the train phase,

and thereby the agent can choose far-sighted action by the past message from other agents, observation, and reward; this way can remove correlations in the observation sequence and smooth over changes in the data distribution. Value function in the target network is calculated by the sum of the immediate return r_t. r_t represents the reward and punishment value; they are obtained by the interaction between the agents and the environment; we define it as the future potential comprehensive income of an agent; whether it is high or not is totally judged by the average value of the preference function; r_t^i is defined as follows:

$$r_t^i = \begin{cases} -1, & F_{x_t,y_t} > F_{x_{t+1},y_{t+1}}, \\ 0, & F_{x_t,y_t} = F_{x_{t+1},y_{t+1}}, \\ 1, & F_{x_t,y_t} < F_{x_{t+1},y_{t+1}}. \end{cases} \tag{7}$$

The preference density of this new area is lower than the original, and a penalty value of −1 is assigned to discourage the exploration of the region in the state s_t; similarly, if the preference density does not change, then to reduce the unnecessary exploration of the agent, which is better than the case of exploring the low preference density, we assign a value of 0, namely, neither encourage nor suppress it; Ultimately, exploring high-density areas is our desired aim with a reward value of 1.

4.2. Optimized Matching for Vehicle Dispatching

4.2.1. Problem Description. The traditional-KM algorithm is suitable for the exact matching of bipartite graphs. Because our model only allocates within a certain range, there is likely a mismatch to the existing bipartite graphs (exact matching: bipartite graph in the case where the left node can match the right node one by one), for example, there is a single area overlap phenomenon (as shown in Figure 7). If only this RSU exceeds the request range of vehicles, it cannot be matched; the traditional-KM algorithm has been modified, because we do not know the lower limit of the worst case of $A(i, j)$ in reinforcement learning; we cannot simply set the negative weight to 0; we need a considerable negative weight to identify the pairing situation in which the RSU is unable to match the vehicle; in the extreme case, the RSU with negative weight cannot be discarded; otherwise, it will affect the satisfaction of users.

A new problem arises with negative weight, when the number of matching on both sides is inconsistent; according to the classical KM algorithm, the nodes with a small number are preferentially matched to ensure that they can be matched, but in the actual matching process, if a considerable negative weight is taken into account in the matching, a lot of time about updating will be wasted; we need to avoid it as much as possible, but it will leave some cases (Figure 8): the edges of the negative weight (non-responsive) are omitted in the figure. When there is a situation as shown in Figure 8, the matching between vehicle c and 2 is better, but in the traditional-KM algorithm, since vehicle b cannot find a new matching in a short time, a loop is stuck in a long calculation.

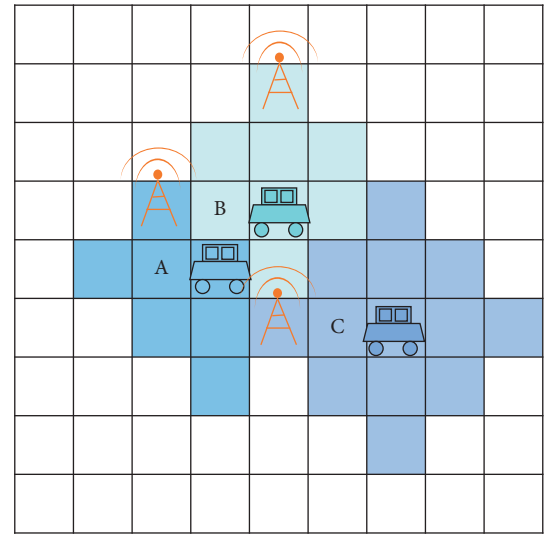


Figure 7: The overlapped answer range of three vehicles.

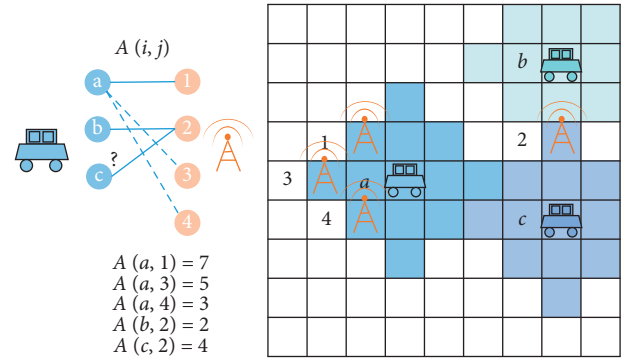


Figure 8: Potential matching situation; it may imply a better match as a to 3 or a to 4.

There will be a more suitable matching; nodes with a higher sum of overall pairing weights will be excluded from the existing matching situation. This requires us to further optimize the existing matching strategy.

We introduce the concept of the loser; we call it the loser-KM algorithm. Once the situation in Figure 8 appears, the vehicle will be eliminated immediately, and then all the vehicle nodes that are eliminated will be collected to Loser$\{\text{loser}_0, \ldots, \text{loser}_n\}$; loser_i will be given one more chance to challenge the vehicle who has already matched the RSU j. If it is greater, the corresponding RSUs will be collected in the master node set Challenger$\{\text{challenger}_0, \ldots, \text{challenger}_n\}$, and finally, the process of attacking will use the classical KM algorithm. The loser-KM algorithm can be used to deal with bipartite graphs that need to consider negative weight matching, and there are some unmatchable conditions. See Algorithm 1 for details.

4.3. Multiagent Recurrent Deep Deterministic Policy Algorithm.

By calculating the regional preference function, the agent i chooses an action by its behavior decision based on its approximate global state s_t; it moves to a new area;

Input: calculate all vehicle-RSU pair matches using KM algorithm and the weight of vehicles pairing with RSUs' weight; record the vehicle nodes that are not matched as the loser set.
Output: updated pairs match
(1) Collect matched orders $j \in [1, \ldots n]$
(2) Initialize a directory challenge, arrays attacker, and challenger
(3) Travel across all the weights between Loser i and RSU j:
(4) If weight [Loser [i]] [j] > weight [match [j]] [j], then
(5) Challenge [Loser [i]] [j] = weight [Loser [i]] [j]]−weight [match [j]] [j]
(6) If Loser [i] is not in attacker, then
(7) Attacker push (Loser [i])
(8) End if
(9) If j is not in attacker, then
(10) Challenger push (Loser [i])
(11) End if
(12) End if
(13) Initialize a two-dimension array chaWeight
(14) Check if challenger j is in the challenge list of attacker i:
(15) If challenger [j] is not in Challenge [attacker [i]], then
(16) Challenge [attacker [i]] [challenger [j]] = 0
(17) End if
(18) chaWeight [i][j] = Challenge [attacker [i]] [challenger [j]]
(19) Calculate attacker-challenger pairs chaMatch with chaWeight by using KM algorithm
(20) For i = 1:challenger, do
(21) Match [challenger [i]] = attacker [chaMatch [i]]
(22) End for

Algorithm 1: Loser Kuhn–Munkres Algorithm.

central system uses the loser-KM matching algorithm to determine a set of scheduling strategies. Since the regional preference takes historical information into account, it encourages vehicles to go to the area where the RSU likely takes services in advance; both the response rate and the total profits have improved in the end.

Then, we define the network update rule; we try to minimize the difference between the calculated behavior value and old value in online Q-network; correspondingly, due to the characteristics of the DDPG network, the value of weight in target Q-network is the value in the last training. Because of the existence of the target network, the update of weight is delayed; it makes the training more convergent and the network more stable. We refer to the settings of the loss function in [6].

The detailed process is shown in Algorithm 2.

5. Experiments

We designed our experiments to investigate the following questions:

(1) How to reflect the role of the improved km algorithm and improve the performance of matching?

(2) MARDDPG is related to prior methods but makes several changes; how does it compare with others when applied to the same simulated environments, with our experimental metric?

To answer (1), we compare the performance on the average profits in a round of matching between the traditional and improved km algorithm. About (2), we show that both the results of traditional and MARDDPG under the same RSU and different numbers of vehicles, it is used to compare different algorithms and highlight the performance under different sparsity levels of the agent. The use of different metrics is to show the performance of each algorithm, it is difficult to finish this matching problem in limited time, it is obvious that each algorithm has its limitations and merits.

5.1. Implementation Details. To verify the reliability of the algorithm, we designed the entire dispatch process to be executed in a 9×9 grid, and for 20-time steps, uniform finite time and small space can effectively reduce the interference of external noise, such as cross-regional and cross-day information transmission; we simplify the behavior of vehicles, each vehicle can only stay in one-time spot, or do a horizontal/vertical move.

While setting the dispatch distance to 2, the vehicle can only upload services to the RSU not exceeding this distance range. If a RSU has not been requested to any vehicle for a long time, it will be canceled, and the cancellation time is set to a range from 0 to 5 of the truncated Gaussian function; its average is 2.5, and the standard deviation is 2.

The RSU generation model also simulates the morning and peak traffic patterns of commuting and the situation with the residential area; the RSU position uses a two-component mixed Gaussian to generate x- and y-axis coordinates and truncates them into integers in the grid. The initial positions of vehicles and RSUs are generated by a discrete uniform distribution function.

Input: the environment
Output: $\theta = \{\theta^1, \ldots, \theta^N\}$
(1) Initialize the parameters $\theta = \{\theta^1, \ldots, \theta^N\}$ for the N actor network and ϕ for the critic network
(2) Initialized the replay memory M
(3) For training step = 1:all steps, do
(4) m_0^0=initialized message, $t = 0$
(5) While $t < T$, do
(6) For $i = 1{:}N$, do
(7) Select the action $a_t^i = \mu_t^i(m_{t-1}, o_t^i)$ for agent i_t
(8) Receive reward r_t^i which is calculated by regional average preference function
(9) Receive observation o_{t+1}^i
(10) Update message by $m_{t-1}^i = \mathrm{LSTM}(m_{t-1}^{i-1}, [o_t^i, a_t^i])$
(11) End for
(12) $m_t^0 = m_{t-1}^N$
(13) $t = t + 1$
(14) End while
(15) Store episode $\{m_0, a_1, r_1, m_1, o_2, a_2, \ldots\}$ in M
(16) Sample a random minibatch of episodes from replay memory M
(17) Each episode and each time, we do
(18) Update the critic, actor, and LSTM network by minimizing the loss
(19) Soft-update the target critic and target actor network
(20) End for

Algorithm 2: Multiagent recurrent deep deterministic policy algorithm (MARDDPG).

5.2. Improved Performance on Matching. Our algorithm is compared with the distance priority algorithm, the profit priority greedy algorithm, and the advantage algorithm. Each algorithm uses the original KM version and the loser-KM version to match these RSUs so that we can observe the impact of existing algorithm factors; in the bipartite graph, the weight of the distance-first algorithm is based on the distance between the RSUs and the vehicles; the weight of the profit priority is based on the calculation of services with the highest profit, and the algorithms combine with the two preference matching strategies; we call them as advantage algorithm whose name comes from advantage function and MARDDPG.

We use average profits at first, namely, the profit obtained by the RSU on each vehicle as a metric for the improvement of the strategy algorithm, because traditional KM in the advantage algorithm uses the preference function as the weight, compared with our improved KM matching algorithm, which has become an adaptation of MARDDPG; it takes the regional preference as the weight, which means that once the vehicle reaches the area with a higher preference value, the profit obtained will be higher so that average profits will be higher because it focuses on long-term returns compared to advantage algorithm.

As shown in Figure 9, each bar represents the average content of 20 independent experiments. Here, we uniformly take the ratio of RSUs to vehicles, which is 100 : 25, as the experiment environment. We are not listing the distance algorithm because it is unrelated to the matching strategy. With using the loser-KM algorithm, the average profits are higher than the traditional-KM algorithm, which is also due to the loser-KM which is based on the principle of the traditional algorithm; its improvement of performance is made of the change according to the actual problem of

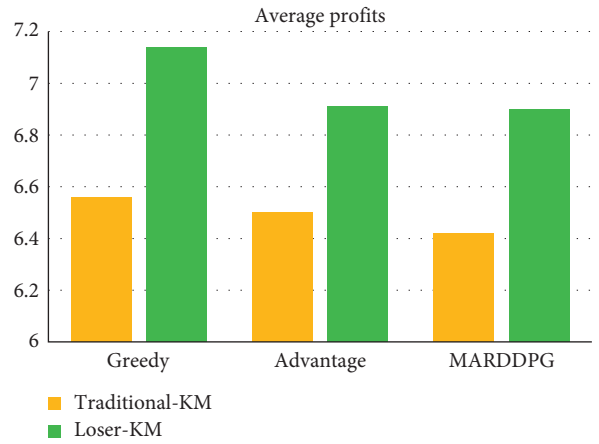


Figure 9: The comparison between traditional-KM and loser-KM in average profits.

negative weight with an unknown lower limit; therefore, the loser-KM algorithm used in the RSU allocation process contributes to certain performance improvement.

Following that, we use four strategies compared with four metrics (average profits, overall profits, pickup distance, and response rate) under the condition of 100 RSUs and the number of vehicles is 25, 50, or 75, respectively, with ten independent experiments.

We compare the performance of average profit in Figure 10; when the ratio of RSUs to vehicles is lowest, the average profits tend to be high; because the overall number of finished services is relatively small, average profits will decrease first and then slowly rise with the ratio increasing. The performance of distance algorithm in the average profit is not outstanding, when the vehicles density increases; its

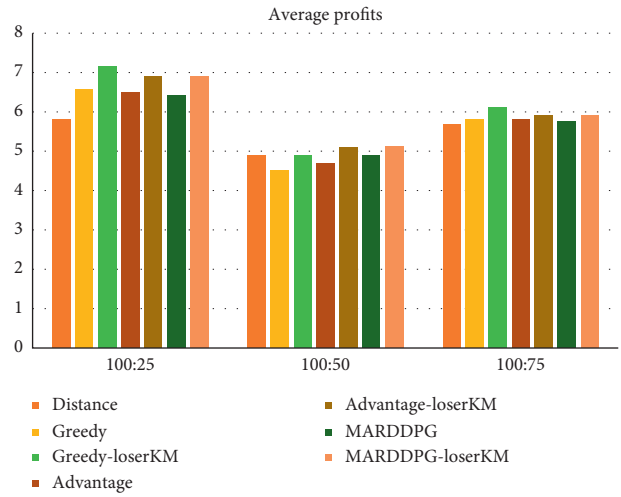


Figure 10: The comparison in average profits under three magnitudes of vehicles.

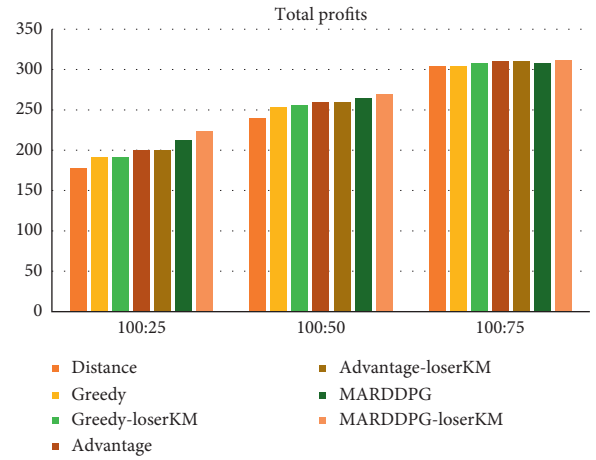


Figure 11: The comparison in total profits under three magnitudes of vehicles.

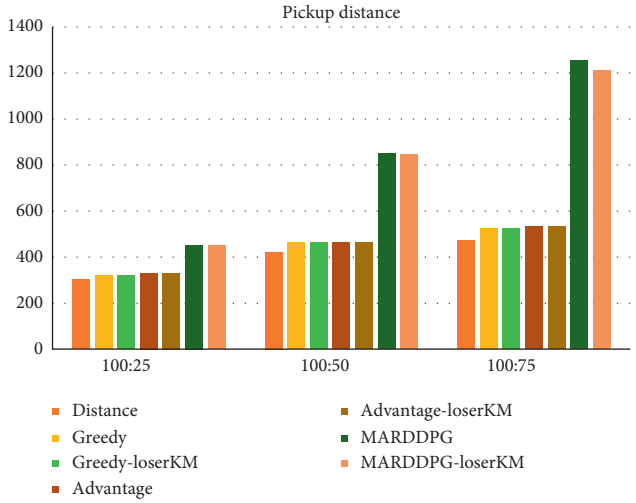


Figure 12: The comparison in pickup distance under three magnitudes of vehicles.

performance also rallies, because the distance between vehicles and RSUs is not too long at this situation; RSU can quickly finish computing services; then, go to next. The greedy algorithm has the best performance in the average profit due to being profit-driven, but it has decreased at a ratio of 100 : 50; we think this is caused by an extreme situation in which the driving distance is too long. The average profit of advantage algorithm will lag behind MARDDPG in some cases, but the experiment shows that the performances of advantage and MARDDPG are almost same in average profit; although we think MARDDPG takes regional preference into account in a period, vehicles will rush to the RSU with high information transmission. Our time step which is only 20 is relatively short. When RSU receives the service from one vehicle, it will finish it without interference; they cannot respond to a new service; therefore, this improvement is weakened.

Figure 11 shows the comparison of overall profits; it measures the degree of different matches. We can find that it does not have a regular relationship with the average profit. The overall performance of the distance algorithm is not outstanding because its weights are independent of the preference function. The greedy algorithm performs generally in terms of overall profits; although it prefers a higher average profit, it does not consider the performance of time spent, which makes it unable to achieve enough profits in a limited time slice, resulting in performance degradation. The advantage algorithm is more inclined to match RSUs with better overall profits; with the ratio rising, its overall profits are almost equal to our algorithm. MARDDPG will actively seek for RSUs because it takes into account the idle state; this will have a considerable improvement when the ratio is low. As the ratio increases, a large number of RSUs can enter the effective matching distance of vehicles, which reduces the probability of idle state.

We show the result of pickup distance in Figure 12; pickup distance represents the total distance that a vehicle moves; it can reflect the activity for exploring the area. The distance algorithm shines here because its weight is based on shorter distances. The performances of the greedy algorithm and advantage are also taken into account; as ratio increases, there will not be a large change in performance, because this will not produce extra mobile behavior. MARDDPG has the "worst" pickup distance because it will actively break the idle state and rush to the potential area. It can be truly proved with the ratio increasing. You can see that the distance has greatly increased in the real world, and it can be worked as a route recommendation because the vehicle can also rely on experience to rush to areas with more timely RSUs.

Until the end of one cycle, we evaluate the response rate from the RSUs (the number of the response/the total number of responses), as shown in Figure 13; it serves as a feeling of QoE and can also indirectly reflect the number of finished services. The distance algorithm has a good response rate, it prioritizes distance, and vehicles can quickly rush to the nearly RSUs; then, RSUs begin to compute the services from vehicles. The greedy algorithm is not good at responses, because the higher the profit is, the longer the calculation time is, and the more times RSU cannot respond,

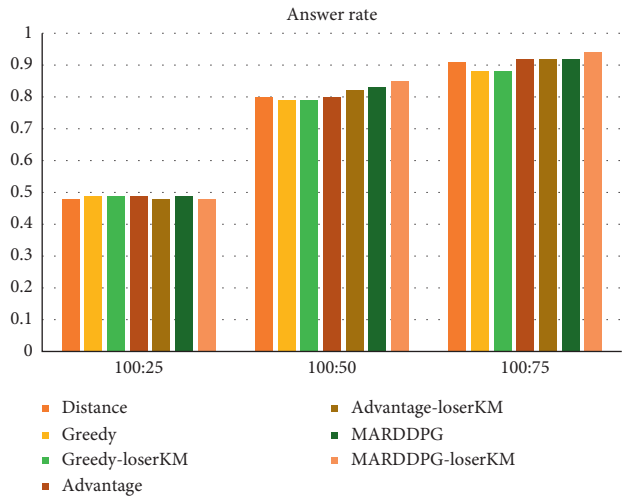


Figure 13: The comparison in answer rate under three magnitudes of vehicles.

leading to the lower response rate. Advantage algorithm is based on the weight of overall profits; it balances the tradeoff between time and distance; it also performs exceptionally in terms of response rate. MARDDPG reduces the time that vehicle stays in idle state and encourages them to actively seek more potential free RSUs, so that it can respond immediately. As ratio increases, this advantage is increasingly manifested; if one area is lacking free RSUs, it will guide vehicles to go to other areas.

5.3. Results Comparison and Analysis. The greedy algorithm is the most profitable because it is profit-driven; however, it also results in a lower response rate. The distance-driven algorithm has the smallest distance and its response rate is improved relative to the greedy algorithm because RSU can respond in time when the previous services are completed. The advantage algorithm has excellent performance, but there may be a situation where the agent may be vacant, so it is at a disadvantage in response rate and average profits.

MARDDPG gives the vehicles the ability to actively explore based on the advantages of retaining the preference matching algorithm, under the definition of the reward function; the vehicle is encouraged to go to the high preference area by central scheduling system, where it is more likely to encounter the free RSUs; its response rate and profits achieve better performance; correspondingly, this exploration also leads to more distances per vehicle, so it can be used as the path recommendation but not a signal source wizard when the MEC servers are sparse in the actual application for the reference to vehicle. As the ratio of RSUs to vehicles decrease, it seems that the performance gain is not large.

When the RSU density is high, the RSU coverage is close to the entire area, and the exploration ability will no longer affect. There are also some cases where the response rate and the total profits are not high, because the high-profit transmission needs to be transported for a long time, and the RSU cannot answer other vehicles during this transportation phase. When the running of the simulation model reaches the time T, the result statistics will be summarized. Upload services that are not completed because of a limited period will affect the relevant data.

These figures unify the performance of the various algorithms mentioned above under different metrics; it helps us to further analyze the experiment from a general point of view. We can find that the distance cost of MARDDPG is relatively large. The overall profit becomes flat with an increase of the ratio. We can find using the loser-KM and the preference function benefit total profits before the ratio of 100 : 75 because the system considers the future profit. The ratio of 100 : 50 is a kind of ideal environment; it has improved the total profits by 60% and approximately 100% response rate compared to the ratio of 100 : 25. We can also see that comparing the region preference with algorithms other than being profit-driven, it takes into account the factors of average return and response rate. The total profits from all servers and the total response rate which is indirectly related to the individual profits are used as the tradeoff metrics; they can embody the robustness and foresight of an algorithm when the RSUs exist quite far from the vehicles.

6. Conclusions and Future Work

In this paper, we introduce the multiagent reinforcement learning algorithm into the agent exploration and combine it with using the loser-KM algorithm in matching. According to the comparison of the traditional algorithms, we can find that it can perform better and satisfy the requirements of high response rate and total profits optimization. As shown in the figures, this algorithm is more suitable for the medium density, and it may lead a long way based on the prediction, so it had better be applied to the recommendation system with sparse RSUs; in large density areas, its merits are not obvious. The related algorithms of these comparisons are all achieved by ourselves, and the details mentioned in the referenced papers are restored as much as possible. In every experiment, each series of comparison algorithms use the same dataset generated by the RSUs generation models, but there are some extreme cases of experiment because a gap between theory and reality exists.

We will try to regard this reinforcement learning algorithm as a decision model in more fields and overcome the defect in the distance; it is related to the nature of the algorithm, whose prediction is a trial and error that leads to more distances. These may become the focus of our future work.

Data Availability

Performance evaluations illustrate the effectiveness of our constructed system on the simulated dataset. The data used to support the findings of this study are available within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant no. 61772575, in part by the National Key R&D Program of China under Grant no. 2017YFB1402101, and in part by the Independent Research Projects of Minzu University of China.

References

[1] H. Gao, C. Liu, Y. Li, and X. Yang, "V2VR: reliable hybrid-network-oriented V2V data transmission and routing considering RSUs and connectivity probability," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–14, 2020.

[2] R. Deng, R. Lu, C. Lai, and T. H. Luan, "Towards power consumption-delay tradeoff by workload allocation in cloud-fog computing," in *Proceedings of the 2015 IEEE International Conference on Communications (ICC'15)*, pp. 3909–3914, London, UK, June 2015.

[3] R. Deng, Z. Yang, J. Chen, N. R. Asr, and M.-Y. Chow, "Residential energy consumption scheduling: a coupled-constraint game approach," *IEEE Transactions on Smart Grid*, vol. 5, no. 3, pp. 1340–1350, 2014.

[4] P. Dong, X. Wang, J. J. P. C. Rodrigues, F. Xia, and Z. Ning, "Deep reinforcement learning for vehicular edge computing: an intelligent offloading system," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 6, 2019.

[5] L. T. Tan and R. Q. Hu, "Mobility-aware edge caching and computing in vehicle networks: a deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 11, pp. 10190–10203, 2018.

[6] Y. He, N. Zhao, and H. Yin, "Integrated networking, caching, and computing for connected vehicles: a deep reinforcement learning approach," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 1, pp. 44–55, 2018.

[7] X. Ma, H. Gao, H. Xu, and M. Bian, "An IoT-based task scheduling optimization scheme considering the deadline and cost-aware scientific workflow for cloud computing," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, p. 249, 2019.

[8] B. Chen and H. H. Cheng, "A review of the applications of agent technology in traffic and transportation systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 2, pp. 485–497, 2010.

[9] K.-T. Seow, N. H. Dang, and D.-H. Lee, "A collaborative multiagent taxi-dispatch system," *IEEE Transactions on Automation Science and Engineering*, vol. 7, no. 3, pp. 607–616, 2010.

[10] L. Zhang, T. Hu, M. Yue et al., "A taxi order dispatch model based on combinatorial optimization," in *Proceedings of the 2017 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2151–2159, New York, NY, USA, August 2017.

[11] B. Li, D. Zhang, L. Sun et al., "Hunting or waiting? Discovering passenger-finding strategies from a large-scale real-world taxi dataset," in *Proceedings of the 2011 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, pp. 63–68, IEEE, Seattle, WA, USA, March 2011.

[12] Y. Tong, Y. Chen, Z. Zhou et al., "The simpler the better: a unified approach to predicting original taxi demands based on large-scale online platforms," in *Proceedings of the 2017 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1653–1662, Halifax, Nova Scotia, Canada, August 2017.

[13] C. Yang, L. Bai, C. Zhang, Y. Quan, and J. Han, "Bridging collaborative filtering and semi-supervised learning: a neural approach for POI recommendation," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1245–1254, New York, NY, USA, August 2017.

[14] J. Liu, L. Sun, Li Qiao, J. Ming, Y. Liu, and H. Xiong, "Functional zone based hierarchical demand prediction for bike system expansion," in *Proceedings of the 2017 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 957–966, Halifax Nova Scotia Canada, August 2017.

[15] Z. Xu, Z. Li, Q. Guan et al., "Large-scale order dispatch in on-demand ride-hailing platforms: a learning and planning approach," in *Proceedings of the 2018 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, London, UK, July 2018.

[16] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proceedings of the 2014 31st International Conference on Machine Learning*, Beijing, China, June 2014.

[17] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the Society for Industrial and Applied Mathematics*, vol. 5, no. 1, pp. 32–38, 1957.

[18] J. Feng, H. Li, M. Huang et al., "Learning to collaborate: multi-scenario ranking via multi-agent reinforcement learning," in *Proceedings of the 2018 International World Wide Web Conference*, Lyon, France, April 2018.

[19] N. C. Luong, Z. Xiong, P. Wang, and D. Niyato, "Optimal auction for edge computing resource management in mobile blockchain networks: a deep learning approach," in *Proceedings of the IEEE International Conference on Communications (ICC'18)*, Kansas City, MO, USA, May 2018.

[20] Y. He, F. Richard Yu, N. Zhao, C. Victor, M. Leung, and H. Yin, "Software-defined networks with mobile edge computing and caching for smart cities: a big data deep reinforcement learning approach," *IEEE Communications Magazine*, vol. 55, no. 12, pp. 31–37, 2017.

[21] Q. Qi, J. Wang, Z. Ma et al., "Knowledge-driven service offloading decision for vehicular edge computing: a deep reinforcement learning approach," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 5, pp. 4192–4203, 2019.

[22] Z. Liao, "Real-time taxi dispatching using global positioning systems," *Communications of the ACM*, vol. 46, no. 5, pp. 81–83, 2003.

[23] R. Zhang and M. Pavone, "Control of robotic mobility-on-demand systems: a queueing-theoretical perspective," *The International Journal of Robotics Research*, vol. 35, no. 1, pp. 186–203, 2016.

[24] M. Qu, H. Zhu, J. Liu, G. Liu, and H. Xiong, "A cost-effective recommender system for taxi drivers," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 45–54, New York, NY, USA, August 2014.

[25] S. Rahili, B. Riviere, S. Olivier, and S. Chung, "Optimal routing for autonomous taxis using distributed reinforcement

learning," in *Proceedings of the 2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 556–563, Singapore, November 2018.

[26] H. T. Wai, Z. Yang, Z. Wang et al., "Multi-agent reinforcement learning via double averaging primal-dual optimization," in *Proceedings of the Proceedings of the 32nd International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, December 2018.

[27] Y. Han, X. Zhou, L. Yang, and S. Li, "A bipartite matching based user pairing scheme for hybrid vlc-rf noma systems," in *Proceedings of the 2018 International Conference on Computing, Networking and Communications (ICNC)*, pp. 480–485, Maui, HI, USA, March 2018.

[28] T. Yuan, X. Huang, M. Ma, and J. Yuan, "Balance-based SDN controller placement and assignment with minimum weight matching," in *Proceedings of the 2018 IEEE International Conference on Communications (ICC)*, pp. 1–6, Kansas City, MO, USA, May 2018.

[29] H. Sun, Z. Chen, S. Yan, and L. Xu, "MVP matching: a maximum-value perfect matching for mining hard samples, with application to person Re-identification," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 6737–6747, Seoul, South Korea, October 2019.

[30] C. A. S. De Witt, J. N. Foerster, G. Farquhar et al., "Multi-agent common knowledge reinforcement learning," 2018, http://arxiv.org/abs/1810.11702.

[31] R. Raileanu, E. Denton, A. Szlam et al., "Modeling others using oneself in multi-agent reinforcement learning," in *Proceedings of the 2018 Machine Learning Research*, Stockholm, Sweden, July 2018.

[32] P. Mannion, K. Mason, S. Devlin, and J. Duggan, "Multi-objective dynamic dispatch optimisation using multi-agent reinforcement learning," in *Proceedings of the 15th International Conference On Autonomous Agents And Multiagent Systems*, Singapore, March 2016.

[33] H. Gao, W. Huang, and Y. Duan, "The cloud-edge-based dynamic reconfiguration to service workflow for mobile ecommerce environments," *ACM Transactions on Internet Technology*, vol. 21, no. 1, pp. 1–23, 2021.

[34] R. Raileanu, E. Denton, A. Szlam et al., "Modeling others using oneself in multi-agent reinforcement learning," in *Proceedings of the 35th International Conference on Machine Learning*, Stockholmsmässan, Stockholm, Sweden, July 2018.

[35] J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*, Sao Paulo, Brazil, May 2017.

[36] A. Nowé, P. Vrancx, and Y. M. D. Hauwere, "Game theory and multi-agent reinforcement learning," *Reinforcement Learning: State-of-the-Art*, Springer, New York, NY, USA, 2012.

[37] D. Qing and A.-X. Zeng, *Reinforcement Learning beyond Games: To Make a Difference in Alibaba*, Electronic Industry Press, Beijing, China, 2018.

[38] H. Gao, L. Kuang, Y. Yin, B. Guo, and K. Dou, "Mining consuming behaviors with temporal evolution for personalized recommendation in mobile marketing apps," *Mobile Networks and Applications*, vol. 25, no. 4, pp. 1233–1248, 2020.

[39] X. Yang, S. Zhou, and M. Cao, "An approach to alleviate the sparsity problem of hybrid collaborative filtering based recommendations: the product-attribute perspective from user reviews," *Mobile Networks and Applications*, vol. 25, no. 2, pp. 376–390, 2020.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6666628, 10 pages
https://doi.org/10.1155/2021/6666628

WILEY | Hindawi

Research Article

Reverse Auction-Based Services Optimization in Cloud Computing Environments

Hongkun Zhang[1,2] **and Xinmin Liu**[3]

[1]*College of Economics and Management, Shandong University of Science and Technology, Qingdao, China*
[2]*College of Economics and Management, Qingdao University of Science and Technology, Qingdao, China*
[3]*College of Management, Qingdao Agricultural University, Qingdao, China*

Correspondence should be addressed to Hongkun Zhang; zixin-zhang@163.com

Received 10 December 2020; Revised 31 January 2021; Accepted 24 February 2021; Published 18 March 2021

Academic Editor: Honghao Gao

Cloud-based services have been increasingly used to provide on-demand access to a large amount of computing requests, such as data, computing, resources, and so on, in which it is vitally important to correctly select and assign the right resources to a workload or application. This paper presents a novel online reverse auction scheme based on online algorithm for allocating the cloud computing services, which can help the cloud users and providers to build workflow applications in a cloud computing environment. The online reverse auction scheme consists of three parts: online algorithm design, competitive ratio calculation, and performance valuation. The online reverse auction-based algorithm is proposed for the cloud user agent to choose the final winners based on Vickrey–Clarke–Groves (VCG) mechanism and online algorithm (OA). The competitive analysis is applied to calculate the competitive ratio of the proposed algorithm compared with the offline algorithm. This analysis method is significant to measure the performance of proposed algorithm, without the assumption of the distribution of cloud providers' bids. The results prove that the proposed online reverse auction-based algorithm is the appropriate mechanism because it allows the cloud user agent to make purchase decisions without knowing the future bids. The difference of auction rounds and transaction cost can impressively influence and improve the performance of the proposed reverse auction algorithm.

1. Introduction

Cloud computing, as a new computing paradigm, has rapidly emerged in recent years [1, 2], and it is able to solve the large-scale problems arising in industry, science, commerce, and engineering [3]. In recent years, there are some famous commercial cloud platforms, e.g., Amazon elastic compute cloud (EC2), Microsoft Azure, and Linode, which organize a shared resource pool for cloud users and providers to trade services [4, 5]. The basic idea of cloud computing is to deliver computational resources, such as central processing unit (CPU), random-access memory (RAM), and storage, as services across the Internet. The cloud users can purchase the cloud computing services on their dynamic and fluctuating demands. On the other hand, the cloud providers pack their resources into different types of virtual machines and design pricing mechanism. However, the highly dynamic, uncontrollable, and distributed features of grid environment cause the hindering of cloud providers to price the cloud computing services. Thus, how to better manage cloud resources becomes the important problem of cloud computing between the cloud providers and users.

At present, the current literature often uses a fixed price method and auction-based method to allocate cloud computing services. For these two methods, the fixed price method is criticized for inherently lacking market efficiency and failing to rapidly adapt to real-time demand-supply relation changes [6]. The auction-based algorithm has been proved to have more advantages [7, 8]. Firstly, fixed price algorithm cannot reflect the change of supply-demand relationship in cloud resource market. Since the auction is a

kind of flexible market mechanism, it is the preferred design for the cloud resource business. Secondly, auction-based method requires little global information and enables trade-off more easily to be implemented with decentralized structure [9]. There are a series of literature about auction-based algorithm design in cloud resource markets from different aspects, achieving some new results [10–12]. Although the researchers have achieved many good results using auction-based methods to allocate cloud computing services, there are the following problems. (1) The one-round auction-based method is often conducted, but the multiple round auctions are not considered. (2) The cloud providers should be in a complete information situation to use the above auction-based method, but not in the incomplete information cases. For example, a cloud provider will leave the auction platforms without the prompt reply from the cloud user. Then, the cloud user has to make decision before the coming of the next cloud provider.

Therefore, it is quite essential to discover the novel pricing algorithm to satisfy the properties of cloud computing services' trading. Because of the limitations of current literature, this paper proposes an online reverse auction-based algorithm to allocate cloud computing services using the online algorithm (OA).

(1) Construct an online reverse auction-based algorithm. Based on the reverse auction theory, after the cloud user has service demand requirement, the cloud providers take price to bid this service demand. In reality, the transactions for allocating cloud resources are in an online setting, where the cloud user has to make purchase decisions without knowing the future bids taken by other cloud providers. Thus, this paper constructs an online reverse auction-based algorithm by using the online algorithm proposed by Nisan and Ronen [13].

(2) Calculate the competitive ratio. Competitive analysis proposed by Goldberg et al. [14] is used to calculate the competitive ratio of the performance of online reverse auction-based algorithm. The advantage of competitive analysis is that it does not need any assumption of the distribution from the future events.

(3) valuate the performance. The competitive ratio calculated by the competitive analysis is used for measuring the performance of the online reverse auction-based algorithm by comparison with optimal offline auction algorithm.

The main contributions of this paper are as follows. First, this paper introduces the multi-round time dimension into the traditional auction-based method to propose a novel online reverse auction-based algorithm to allocate cloud computing services. Second, the online reverse auction-based algorithm can help the cloud user make purchase decisions while he has no information about the future bid sequences of the cloud providers. Third, the transaction cost of auction service is introduced into the online reverse auction-based algorithm, in which the cloud user has to charge a fee to an agent for auction service. Numerous simulating experiments show that the proposed online reverse auction-based algorithm is effective in cloud service utilization and has better performance in user utility. The auction rounds and the transaction cost play an important role for the cloud user agent's decision. The auction efficiency can be improved from these two factors.

This paper is organized as follows. Section 2 introduces the related work. Section 3 describes the reverse auction market. Section 4 presents the novel online reverse auction-based algorithm. Section 5 explains the trading strategies and competitive analysis. Section 6 introduces the simulation and summarizes the experimental results. Conclusions are given in Section 7.

2. Related Work

The auction-based algorithm has been applied to various fields, such as grid computing, wireless networks, and cloud computing. The purpose is to analyze how the users and providers behave in a competitive environment. The studies show that different classes of auctions have been considered in the design of the algorithms. An important achievement of auction design is the Vickrey–Clarke–Groves (VCG) mechanism, which is a general method for the construction of truthful mechanisms in an auction market. All reasonable approximations or heuristics for combinatorial auctions are VCG-based mechanisms and have truthful features, which are suitable for a wide class of cost minimization problems [13]. The results essentially indicate that the only truthful auction algorithm is efficient. Thus, for the cloud computing services market, the auction-based algorithm is also truthful if a cloud user or a cloud provider has no incentives to lie about their private information. The current studies have designed some auction-based algorithms to allocate the cloud resources. The most popular auction forms are the English auction, the combinatorial auction, and the double auction. For example, Fujiwara et al. [15] proposed a combinatorial auction to design marketplace mechanism for cloud computing services. The presented auction algorithm helps the cloud users build workflow applications in a cloud computing environment, where the cloud users need to compose multiple types of services at different timeslots. Wang et al. [16] studied an English auction-based algorithm for cloud computing service. They presented an auction framework for cloud computing. Different designs of auction-based algorithms were also discussed. Zaman and Grosu [17] presented two auction schemes to allocate virtual machines for cloud users. They argued that combinatorial auction-based algorithms were more efficient than the fixed price algorithms. The reason is that the virtual machines having the highest valuation are assigned to cloud users. Kang et al. [11] proposed the multi-unit Vickery auctions and one-sided VCG-combinatorial auctions to address cloud service allocation. They proved that these two auction-based algorithms were efficient methods. Lee et al. [18] proposed a new instantiation of the negotiation protocol between the cloud user and cloud provider by using a

continuous double auction model. They analyzed and showed different scheduling strategies, which can be applied into the real cloud resource trading. Some studies introduced the double auction forms into combinatorial auction design. For example, Tafsiri and Yousefi [12] studied a combinatorial double auction-based algorithm where a broker performed the allocation of the providers' virtual machines according to the users' requests for the cloud computing. In fact, in the above auction mechanism the cloud providers sell the cloud resources and the cloud users take bids to make purchase decisions, which is called forward auction.

In recent years, a new auction pattern, that is, the reverse auction, has attracted more attention. The reverse auction model has a high potential for cloud computing. It is different from the traditional auction forms, where the cloud user is an auctioneer and cloud provider is a bidder. Namely, in a reverse auction model, the cloud providers submit bids. The cloud users evaluate the bids by auctioning rules and determine the final winners. Roovers et al. [19] investigated the design of a reverse auction market. They pointed out that the reverse auction had the flexibility and the ability to model and integrate currently deployed pricing schemes of the real world. Prodan et al. [20] extended the continuous double auction problem and introduced a continuous reverse auction that was paired with a novel bidding language based on tag and constraint sets. Prasad and Rao [21] used reverse auction to design a cloud resource procurement approach and showed three possible reverse auction algorithms for cloud resource procurement.

However, the above auction-based algorithm designs are considered in offline situations where the final winners are announced after collecting all bids. In fact, both the cloud service providers and users often make decisions under incomplete information [22]. For example, different cloud providers arrive at different times and the cloud users are required to decide whether to accept each bid as it is received without knowing the future bids. In 1999, an online auction algorithm was originally proposed to solve this kind of dynamic grid or cloud resource allocation problem, and the authors used the competitive analysis to calculate the competitive ratio to measure the performance of online auction algorithm [14]. The competitive analysis has gained much recognition in the areas of finance, economics, and operation researches. It is different from the traditional average-case analysis, which focuses on the assumption of the distribution from the future events according to some known probabilities and tries to achieve the good average-case performance. Many researchers extend Goldberg's work from the following aspects. Hajiaghayi et al. [23] considered an online truth telling mechanism based on the offline Vickrey model. For the limited supply goods, Lavi and Nisan [24] presented an incentive compatible online auction and proved this auction had an optimal competitive ratio with respect to the revenue and the total social efficiency. Zhu et al. [25] studied a reverse online auction problem and designed online reverse auction algorithm based on multi-attribute bids, which achieved a better performance.

In recent years, some studies began to design online auction algorithm to allocate the cloud computing service. For example, Zhang et al. [26] conducted a framework for truthful online auctions where the cloud users with heterogeneous demands could come and leave on the fly. They designed a novel bidding language, in which the cloud users' heterogeneous demands were generalized to be regulated and kept in consistent forms. Based on this bidding language, they proposed an incentive compatible online cloud auction algorithm and got some new results. Shi et al. [6] gave the first online combinatorial auction algorithm for the cloud computing problem. The same results were shown where the auction systems were expressive enough to optimize system efficiency across the temporal domain instead of at an isolated time point. Ding et al. [7] introduced the online auction into the resource scheduling in grid computing networks and designed the online auction-based algorithm. They presented a new multi-attribute multi-round reverse auction, where the gird resource user's satisfaction degree was introduced into the traditional grid resource allocation problem to help the grid resource broker make multi-attribute decisions under incomplete information.

3. The Reverse Auction Market Description

There is a reverse auction market which maintains the requests and bids collected from cloud users and providers, respectively. After the cloud user agent receives instructions from the cloud user, it computes when and how to allocate funds to purchase the cloud computing services from which cloud providers. Figure 1 is a flowchart of the reverse auction scheme and shows how the participants work in a market. The cloud user submits his request and private information to the cloud user agent (Label 1). In the reverse auction market, the cloud user agent searches the cloud providers that meet the user's request and invites them to participate in the auction (Label 2). After the cloud providers take bids in an online manner (Label 5), the cloud user agent reports and announces to the users who are the winners/losers of the auction (Label 3). Once the charging and payment are completed (Label 4), the cloud users and providers establish the connection and complete the deal (Label 6).

In the reverse auction market, the cloud users and providers have different objectives, strategies, and supply-demand patterns. These participants are presented in the following with a brief explanation.

3.1. Cloud Users. In the reverse auction market, the cloud user's task is to submit his demand constrained by budget, price preferences, and memory size to his agent. It supposes that the cloud user has a budget of d_0, which means the cloud user has total funds of d_0 to purchase cloud services. Each service is denoted by a three-tuple $J_i = (\mathrm{T}_i, \mathrm{RP}_i, E_i)$, where T_i is the deadline of the service, RP_i represents the secret reservation price, and E_i is the minimum memory size. The goal of the cloud user is to spend his total funds to maximize the utility within his corresponding deadlines.

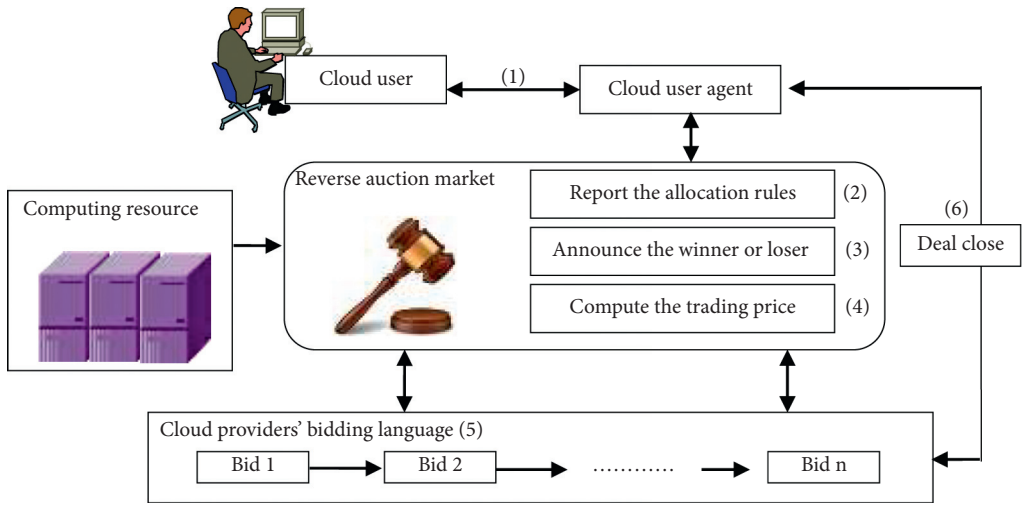


Figure 1: A flowchart of the proposed reverse auction market.

3.2. Cloud User's Agent. Each cloud user has a cloud user agent or a broker. In the auction process, a cloud user agent represents the cloud user to announce the cloud resource requests. His job depends on the user's request to search cloud providers which can meet user's demand, and then invite the cloud provider to bid. At the end of game, he chooses the winner of the auction on behalf of the cloud user. In fact, the cloud user agent can be seen as the auctioneer of the reverse auction market. Different from the above literature, we assume that there is transaction cost when trading the cloud resource. Instead of one round, this paper assumes that there are n rounds with the expired time $T > 0$. In each round i, the allocation quantity of the funds of cloud user is denoted by $s_i (s_i \geq 0)$. And, there is a fixed transaction cost denoted by a for the cloud user to pay the agent.

3.3. Cloud Providers. In the reverse auction market, the cloud provider's job is to decide whether to participate in the auction when receiving the invitation from the user agent according to his own capability. If he accepts the invitation, then he takes bid. It supposes that a cloud provider i submits a bid defined by $B_i = (e_i, p_i, \overrightarrow{t_i})$, where e_i is the memory size of cloud and p_i is the provider's valuation as a bidding price, which indicates the maximum price that is acceptable for the provider to supply the requesting instances. $\overrightarrow{t_i} = (t_i^s, t_i^e)$ is a length of time during which the cloud provider i wants to reserve a bid between starting time t_i^s and ending time t_i^e. Here, the cloud providers arrive at different times in sequences, which is referred to as the online manner.

4. A Novel Online Reverse Auction-Based Algorithm

This paper studies the online reserve auction problem in the online setting. In each round, facing the historical bids and limited current information, the cloud user agent has to make decisions without knowing the future bids [16], or the distribution functions of bids [17]. This paper aims to design a novel online reverse auction-based algorithm to be carried out by the cloud user agent, which guides cloud resource allocation in the cloud computing service market. We use the competitive analysis to evaluate the performance of the online reverse auction-based algorithm.

Definition 1. The online reserve auction is defined as follows:

(1) In each round, the cloud provider i arrives and announces his bid $B_i = (e_i, p_i, \overrightarrow{t_i})$, where each p_i is a real number in the interval of $[m, M]$.

(2) The cloud user agent determines whether to buy the cloud computing service from the cloud provider i and if so, at what price and quantity before opening the next bid.

(3) The game ends when the last cloud provider announces his bid during the time of $[0, T]$.

Definition 2. The competitive ratio of online reverse auction-based algorithm is defined as follows:

(1) Let $\mathrm{OPT}(B)$ be the optimal benefit by an offline algorithm denoted by OPT for any bid sequences B.

(2) Let $\mathrm{ALG}(B)$ be the benefit generated by the online reverse auction-based algorithm denoted by ALG for any bid sequences B.

(3) The competitive ratio r of the online reverse auction-based algorithm is calculated by $r = \mathrm{OPT}(B)/\mathrm{ALG}(B)$.

Definition 3. The competitive analysis of online reverse auction-based algorithm is defined as follows. It says an online reverse auction-based algorithm is r-competitive if the benefit of ALG satisfies $\mathrm{OPT}(B) \leq r \cdot \mathrm{ALG}(B)$.

In this definition, r is the competitive ratio of the online reverse auction-based algorithm *ALG*. That is, the infimum over all r is called the competitive ratio of the online reverse

auction-based algorithm *ALG*. On the other hand, an online reverse auction-based algorithm is said to be best possible if there does not exist another online algorithm with a strictly smaller competitive ratio. The closer to 1 the competitive ratio, the more benefit the online reverse auction-based algorithm *ALG* can obtain.

Next, we design a novel online reverse auction-based algorithm, which extends the threat-based algorithm proposed by Lavi and Nisan [16] through taking the transaction cost into consideration. Although this paper considers the online reverse auction-based algorithm as VCG mechanism, it is different from the study of Lavi and Nisan [16], which studies the forward auction, relative to the reverse online auction. In the online reserve auction market for the cloud computing services, the cloud user agent has to consider a risk by assuming that bid sequences will increase to the highest price. Hence, the proposed online reverse auction-based algorithm helps the cloud user agent reserve enough funds to ensure a competitive ratio of r, even though the bid sequences stay at a higher price. Specially, even if the cloud user agent meets the worst case and has to buy services at the highest price at the end of the auction game, the competitive ratio is no more than r.

A novel online reverse auction-based algorithm is proposed as follows.

Let *SP* and *NSP* denote the sets of preferred and not-preferred bids, respectively. We design a novel online reverse auction-based algorithm for the cloud user agent to update his decisions. Given r, m, M, and a new bid p_i, the cloud user agent makes a decision according to the following rules:

Step 1. Set $i = 1$ and SP = NSP = Φ.

Step 2. In round i, one cloud provider comes and presents his bid, i.e.,$B_i = (e_i, p_i, \overrightarrow{t_i})$.

Step 3. Before the end of round t, the cloud user agent decides whether to accept bids. If the bid value reaches a new low, that is, $p_i < \min_{i>j} p_j$, and the memory size of cloud satisfies $e_i > E_i$, the cloud user agent places the accepted bid in set *SP*, and go to Step 4. Otherwise, place the refused bid in set *NSP* and go to Step 7.

Step 4. The cloud user agent uses such a rule to allocate the cloud user's original funds that is spend less money to buy service to keep the competitive ratio as a constant of r. That is, compute the trading funds of s_i based on the following rules:

$$\begin{cases} s_1 = \dfrac{1}{r} \cdot \dfrac{(d_0 - a)(1 - rmp_1)}{1 - mp_1} + \dfrac{a}{1 - p_1 m} & i = 1, \\ s_i = \dfrac{1}{r} \cdot \dfrac{(d_0 - a)(p_{i-1} - p_i)}{p_{i-1} - p_i p_{i-1} m} + \dfrac{a}{1 - p_i m} & 1 < i. \end{cases} \tag{1}$$

Step 5. Inform the winning cloud provider in set *SP* of the trading price and quantities between the time interval of $[t_i^s, t_i^e]$.

Step 6. Set $i = i + 1$. Go to Step 2. If $i = n$, go to Step 7.

Step 7. Stop. The winning cloud providers are bids in *SP*. If there are remaining funds, then the cloud user agent has to purchase the job or service at the highest price of M. Even in this worst case, the competitive ratio is no more than r.

5. Competitive Analysis of the Online Reverse Auction-Based Algorithm

In this section, the competitive analysis is used to evaluate the performance of the online reverse auction-based algorithm. We divide the online reverse auction-based algorithm into two cases as follows. One is that the cloud user agent charges a fee for his auction service. The other is that the cloud user agent is free for auction service.

5.1. Case 1: The Cloud User Agent Charges a Fee. If the cloud user agent knows all the bid sequences, the optimal offline benefit can be achieved by the optimal offline algorithm. According to Definition 2, in round i, the optimal offline benefit is $\mathrm{OPT}(B) = d_0 - a/p_i$. The benefit of online reverse auction-based algorithm is $\mathrm{ALG}(B) = \sum_{i=1}^{n} s_i/p_i$, where s_i is the trading funds. Hence, the competitive ratio denoted by r of the online reverse auction-based algorithm can be computed as follows:

$$\begin{aligned} r &= \min \max \frac{d_0 - a}{p_i \sum_{j=1}^{i} s_j/p_j}, \\ \text{s.t.} \quad & \forall i, s_i \geq 0, \sum_{i=1}^{n} s_i = d_0. \end{aligned} \tag{2}$$

In order to solve the optimal competitive ratio of r, we assume that D_i is the remaining funds owned by the cloud user agent after the round i. The accumulative quantities of cloud computing services bought by the cloud user agent before round i are $Y_i = \sum_{j=1}^{i} s_j/p_j$. If $i = n$, then the total quantities of services are $Y_n = \sum_{j=1}^{n} s_j/p_j$. At the same time, it is shown that $s_i = D_{i-1} - D_i$ with the range of $s_i \in [0, d_0 - \sum_{j=1}^{i-1} s_j]$. In Section 4, the online reverse auction-based algorithm implies us that the trading only occurs when the new bid is lower than the previous one. Hence, it assumes that some bid sequences keep declining to the lower bound of $p_{\min}$ until round k, and then the remaining bid sequences keep the upper bound of $p_{\max}$. Namely, the bid sequences are $p_{\min} < p_k < \cdots < p_2 < p_1$ and $p_{k+1} = p_{k+2} = \cdots = p_n = p_{\max}$. For simplicity, suppose that $b_i = 1/p_i$, $b_{\min} = 1/p_{\max} = m$, and $b_{\max} = 1/p_{\min} = M$. Therefore, the cloud user agent has to encounter such a worse case that bid sequences are $m < b_1 < b_2 < \cdots < b_k < M$ and $b_{k+1} = b_{k+2} = \cdots = b_n = m$.

Based on (2), the competitive ratio also can be translated into the other form as follows:

$$r = \frac{d_0 - a}{p_i \sum_{j=1}^{i} s_j/p_j} = \frac{(d_0 - a) b_i}{Y_i + (D_i - a)m}. \tag{3}$$

From equation (3), we get

$$Y_i + (D_i - a)m = \frac{(d_0 - a)b_i}{r}. \tag{4}$$

It means that in round i the accumulative quantities of cloud computing services Y_i and the remaining funds D_i have a linear relationship. Hence, we can decompose the formula about the accumulative quantities of services into the following forms:

$$Y_i = Y_{i-1} + \frac{s_i - a}{p_i} = Y_{i-1} + (s_i - a)b_i. \tag{5}$$

The function about remaining funds also can be divided into the new form:

$$D_i = D_{i-1} - s_i. \tag{6}$$

According to equation (3), the competitive ratio also can be described as follows:

$$r = \frac{(d_0 - a)b_i}{Y_i + (D_i - a)m} = \frac{(d_0 - a)b_{i-1}}{Y_{i-1} + (D_{i-1} - a)m}. \tag{7}$$

Substituting equations (5) and (6) into equation (7), we solve equation (7) to achieve the trading funds in the round of i:

$$s_i = \frac{1}{r} \cdot \frac{(d_0 - a)(b_i - b_{i-1})}{b_i - m} + \frac{ab_i}{b_i - m}. \tag{8}$$

Taking $b_i = 1/p_i$ into equation (8), we achieve that

$$s_i = \frac{1}{r} \cdot \frac{(d_0 - a)(p_{i-1} - p_i)}{p_{i-1} - p_i p_{i-1} m} + \frac{a}{1 - p_i m}. \tag{9}$$

When $i = 1$, the cloud user agent makes a decision based on the first bid of p_1 by using equation (9); we can get that

$$s_1 = \frac{1}{r} \cdot \frac{(d_0 - a)(1 - rmp_1)}{1 - mp_1} + \frac{a}{1 - p_1 m}. \tag{10}$$

Competitive ratio of the online reverse auction-based algorithm in Case 1 is presented.

When the cloud user agent chooses the online reverse auction-based algorithm to allocate funds, the adversary would present the bad bid sequences to make him in a worst case. For example, for any $k(0 < k \le n)$, if the accumulative trading quantities are $\sum_{i=1}^{k} s_i \le d_0$, then the best for the cloud user agent is to spend all funds to buy services before the end of n. Hence, substituting (8) and (10) into $\sum_{i=1}^{k} s_i = d_0$, we obtain that

$$\frac{1}{r} \cdot \frac{(d_0 - a)(b_1 - rm)}{b_1 - m} + \frac{ab_1}{b_1 - m} + \sum_{i=2}^{k} \frac{(d_0 - a)(b_i - b_{i-1})}{r(b_i - m)} + \sum_{i=2}^{k} \frac{ab_i}{b_i - m} = d_0, \tag{11}$$

where $b_i = 1/p_i$.

Solving equation (11), we have that

$$r = r^{(k)}(b_1, b_2, \ldots, b_k) = \frac{(d_0 - a)b_1 + \left(\sum_{i=2}^{k} (d_0 - a)(b_i - b_{i-1})/b_i - m\right)(b_1 - m)}{\left(d_0 - ab_1/b_1 - m - \sum_{i=2}^{k} ab_i/b_i - m\right)(b_1 - m) + (d_0 - a)m} = \frac{b_1 + \left(\sum_{i=2}^{k} b_i - b_{i-1}/b_i - m\right)(b_1 - m)}{\left(d_0 - \sum_{i=1}^{k} ab_i/b_i - m\right) \cdot (b_1 - m/d_0 - a) + m}. \tag{12}$$

Since $k < n$ and $m \le b_1 < b_2 < \cdots < b_k \le M$, we optimize $r^{(k)}(b_1, b_2, ..., b_k)$ to get the optimal competitive ratio of the online reverse auction-based algorithm. For equation (12), the difficulty is to solve the cumulative sum. Next, we solve and simplify these two cumulative sums. Hence, we have

$$\sum_{i=2}^{k} \frac{b_i - b_{i-1}}{b_i - m} = \sum_{i=2}^{k} \frac{(b_i - m) - (b_{i-1} - m)}{b_i - m}. \tag{13}$$

Because the cumulative summation formula is larger than the cumulative multiplication formula for the same sequences of x_{i-1}/x_i, that is, $\sum_{i=2}^{k} x_{i-1}/x_i \ge (k-1)(x_1/x_2 \cdot x_2/x_3 \cdot \cdots \cdot x_{k-1}/x_k)^{1/(k-1)} = (k-1)(x_1/x_k)^{1/(k-1)}$, we have the following relation between them:

$$\sum_{i=2}^{k} \frac{b_i - b_{i-1}}{b_i - m} = (k-1) - \sum_{i=2}^{k} \frac{b_{i-1} - m}{b_i - m}. \tag{14}$$

We take first-order derivation of (14) with respect to $(b_i - m)$ and find that

$$\frac{\partial \Sigma}{\partial (b_i - m)} = 0 \Rightarrow \frac{b_i - m}{b_{i-1} - m} = \frac{b_{i+1} - m}{b_i - m}. \tag{15}$$

Hence, the cumulative summation formula can be described as follows:

$$\sum_{i=2}^{k} \frac{b_i - b_{i-1}}{b_i - m} \ge (k-1)\left(1 - \left(\frac{b_1 - m}{b_k - m}\right)^{1/(k-1)}\right). \tag{16}$$

For the other cumulative summation formula of $\sum_{i=2}^{k} ab_i/b_i - m$, we have that

$$\sum_{i=1}^{k} \frac{ab_i}{b_i - m} = ak + am \sum_{i=1}^{k} \frac{1}{b_i - m}. \tag{17}$$

Since $b_i - m/b_{i+1} - m = (b_1 - m/b_k - m)^{1/k-1}$, we set $(b_1 - m/b_k - m)^{1/k-1} = q$ and obtain that

$$\frac{1}{b_k - m} = \frac{q^{k-1}}{b_1 - m}. \tag{18}$$

Hence, we get

$$\sum_{i=1}^{k} \frac{1}{b_i - m} = \sum_{i=1}^{k} \frac{q^{i-1}}{b_1 - m} = \frac{1 - q^k}{(b_1 - m)(1 - q)}. \tag{19}$$

We bring equation (19) into formula (17) and get that

$$\sum_{i=1}^{k} \frac{ab_i}{b_i - m} = ak + \frac{am\left(1 - (b_1 - m/b_k - m)^{k/(k-1)}\right)}{(b_1 - m)\left(1 - (b_1 - m/b_k - m)^{1/(k-1)}\right)}. \tag{20}$$

Substituting (16) and (20) into (12), the new competitive ratio formula is to maximize the function of $r^{(k)}(b_1, b_2, \ldots, b_k)$ as follows:

$$\max r^{(k)}(b_1, b_2, \ldots, b_k) = \frac{b_1 + (k-1)\left(1 - (b_1 - m/b_k - m)^{1/(k-1)}\right)(b_1 - m)}{(d_0 - ak)(b_1 - m/d_0 - a) - am\left(1 - (b_1 - m/b_k - m)^{k/(k-1)}\right)/1 - (b_1 - m/b_k - m)^{1/(k-1)} + m}. \tag{21}$$

Taking the derivation of $r^{(k)}(b_1, b_2, \ldots, b_k)$ with respect to b_k, we get $\partial r/\partial b_k > 0$. Therefore, it is shown that $r^{(k)}(b_1, b_2, \ldots, b_k)$ would go up with the increasing of b_k. There is a positive relationship between them. Hence, while $b_k = M$ and $k = n$, the competitive ratio of $r^{(k)}(b_1, b_2, \ldots, b_k)$ is maximized. Because of the complexity of equation (21), we do not replace $1/p_1$ with b_1. The simple function of competitive ratio can be described as the function of M,m,b_1,n, and a.

$$r = r(b_1) = \frac{\left[b_1 + (n-1)\left(1 - (b_1 - m/M - m)^{1/(n-1)}\right)(b_1 - m)\right](d_0 - a)\left((b_1 - m/M - m)^{1/(n-1)} - 1\right)}{\left[(d_0 - an)(b_1 - m) + (d_0 - a)m\right](d_0 - a)\left((b_1 - m/M - m)^{1/(n-1)} - 1\right) - am\left((b_1 - m/M - m)^{n/(n-1)} - 1\right)}. \tag{22}$$

From equation (22), it is shown that the first bid b_1 is very important. If it is lower than rm, then the cloud user agent would not do any trading by the online reverse auction-based algorithm. Hence, the bid b_1 should be more than rm. If we want to get the optimal competitive ratio, the first bid b_1 should be given based on equation (22). The online reverse auction-based algorithm presents an idea that the adversary would like to take the worst bid sequences to make the cloud user agent be worst case. Then, we introduce the average idea into the online reverse auction-based algorithm to help the cloud user agent reduce risk in the worst case. Suppose that the cloud user agent divides the total funds into n; that is, $s_1 = d_0/n$. Therefore, we have that

$$\frac{1}{r} \cdot \frac{(d_0 - a)(b_1 - r_1 m)}{b_1 - m} + \frac{ab_1}{b_1 - m} = \frac{d_0}{n}. \tag{23}$$

Solving equation (23), we obtain that

$$b_1 = \frac{(d_0 - d_0 n + an)rm}{r + (a - ra - d_0)n}. \tag{24}$$

5.2. Case 2: The Cloud User Agent Is Free. For this case, the cloud user agent is free for the auction and helps the cloud users to choose suitable cloud service price. Based on an adversary argument, we show that the online reverse auction-based algorithm without transaction cost can achieve a smaller competitive ratio.

In this case, it means that $a = 0$. Hence, we can achieve the trading funds s_i' by similar derivation from Case 1 when the competitive ratio is r'. That is,

$$s_i' = \frac{1}{r'} \cdot \frac{d_0(p_{i-1} - p_i)}{p_{i-1} - p_i p_{i-1} m}. \tag{25}$$

When $i = 1$, the cloud user agent can decide the trading funds based on the first bid of p_1, the same as equation (10); we have that

$$s_1' = \frac{1}{r'} \cdot \frac{d_0(1 - r' m p_1)}{1 - m p_1}. \tag{26}$$

Competitive ratio of the online reverse auction-based algorithm in Case 2 is presented.

In this case, the cloud user agent uses the same rules of the online reverse auction-based algorithm to price the cloud service. Substituting (25) and (26) into $\sum_{i=1}^{k} s_i = d_0$, for any $k(0 < k \leq n)$ we obtain that

$$\frac{1}{r'} \cdot \frac{d_0 (1 - r' m p_1)}{1 - m p_1} + \sum_{i=2}^{k} \frac{1}{r'} \cdot \frac{d_0 (p_{i-1} - p_i)}{p_{i-1} - p_i p_{i-1} m} = d_0. \quad (27)$$

Set $b_i = 1/p_i$. Then, we get

$$\frac{1}{r'} \cdot \frac{d_0 (b_1 - r' m)}{b_1 - m} + \sum_{i=2}^{k} \frac{d_0 (b_i - b_{i-1})}{r' (b_i - m)} = d_0. \quad (28)$$

Comparing equation (11) with equation (28), it shows that equation (28) is the special case of equation (11) when $a = 0$. Hence, we simplify the solving process of competitive ratio of the online reverse auction-based algorithm without transaction cost. Based on equation (22), the competitive ratio of the online reverse auction-based algorithm without transaction cost is achieved by the following equation:

$$r' = r(b_1) = \frac{b_1 + (n-1)\left(1 - (b_1 - m/Mt - nm)^{1/(n-1)}\right)(b_1 - m)}{d_0 b_1}. \quad (29)$$

In the first round, the adversary presents p_1 to the cloud user agent. Maybe if the cloud user agent spends less than d_0/n funds for this bid, then the adversary ends the game. Then, the cloud user agent may encounter the loss. Namely, the cloud user agent has to use his remaining funds to buy cloud computing services at the maximum price. Hence, the cloud user agent also can use the average strategy to guarantee the competitive ratio of r'. Substituting $s_1' = d_0/n$ into equation (26), we have that

$$b_1 = \frac{d_0 (n-1) r' m}{d_0 n - r'}. \quad (30)$$

For the competitive analysis of these two cases, the emphasis is to discuss how the transaction cost affects the performance of online reverse auction-based algorithm. Based on equations (22) and (29), we can compare them to discover the smaller competitive ratio. Also, the sensitivity analysis about the competitive ratios in these two cases is presented in Section 6.

6. Simulation and Experimental Results

In this section, we provide some experimental results of competitive ratios attained by the online reverse auction-based algorithm discussed in two cases. For simplicity, we assume that competitive ratio is r_1 and r_2, respectively, for free case and chargeable case. Consider Figure 2. Clearly, when we set $M = 290$, $d_0 = 200$, $n = 50$, and $a = 0.1$, the competitive ratios of r_1 and r_2 are all decreasing with the increasing of m, which means that these online reverse auction-based algorithms are always significantly better than all other algorithms. Notice that the competitive ratio of online reverse auction-based algorithm for chargeable case is worse than the one for free case. The reason is that the transaction cost increases the cloud user burden and reduces his utility. In general, it is not hard to show that the limits of the competitive ratios of r_1 and r_2 for free case and chargeable case are all 1.55.

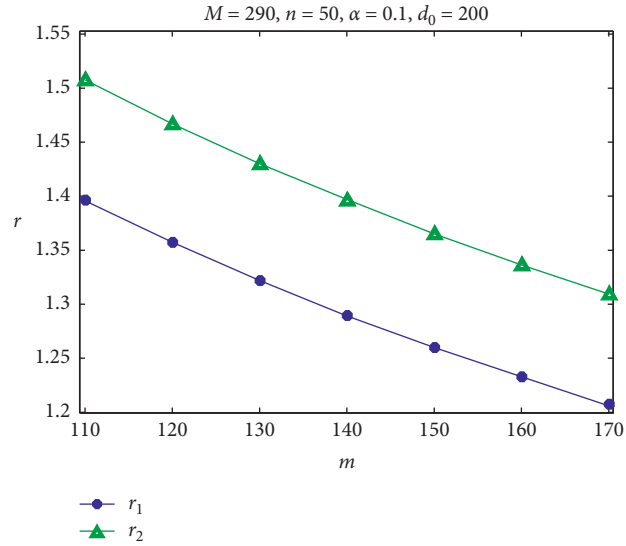


Figure 2: The comparison between r_1 and r_2.

Figure 3 shows that the online reverse auction-based algorithm enables the cloud user to manage his funds and utilize the transaction cost to achieve better performance. In Figure 3, the curve indicates the competitive ratio of r_2 is strictly less than 1.52. That is, the performance curve of online reverse auction-based algorithm expresses the behavior of competitive ratio, where lower bound m serves as the independent, and the competitive ratios serve as the dependent. Across these axes, the performance curves slope downward, which represents a negative relationship between the lower bound and the competitive ratios.

From Figure 3, we find that increasing the auction rounds will result in worse performance of the online reverse auction-based algorithm. Specifically, we compare the performance of online reverse auction-based algorithm for different n. It can be seen that when $M = 290$, $d_0 = 200$, and $a = 0.1$, and the auction round is 15, 50, and 100, the competitive ratio of online reverse auction-based algorithm decreases from 1.52 to 1.48. In nature, the auction rounds increase the competition between the cloud providers. But, it is lower efficiency for the cloud user to make a decision when he is in an incomplete environment.

Figure 4 shows the results about the performance of online reverse auction-based algorithm with a. It finds that the competitive ratio function is represented by transaction cost. That is, there is a negative relationship between the competitive ratio and the transaction cost. When increasing the transaction cost from 0.1 to 0.3, the competitive ratio of r_2 gradually decreases from 1.75 to 1.5. Many studies take no account of the transaction cost. This is usually contrary to the facts. However, this kind of cost would offset the benefit with a corresponding auction trading. Hence, it is necessary to introduce transaction cost into the design of auction

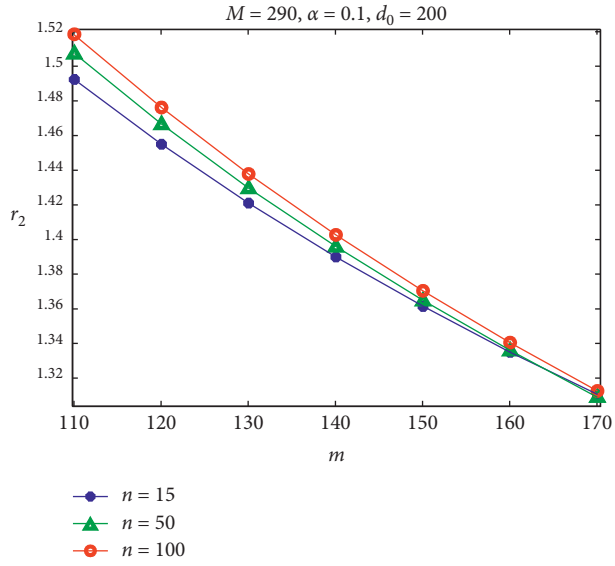


Figure 3: The performance of online reverse auction-based algorithm with n.

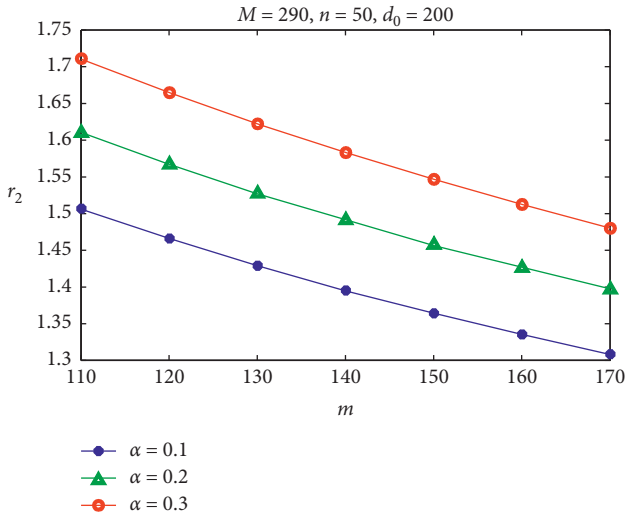


Figure 4: The performance of online reverse auction-based algorithm with a.

system for cloud computing services, which is suitable for all participants in the auction market.

We simulate the process of online reverse auction-based algorithm (see Figure 5) and then show the difference of cloud user agent's decision with different transaction cost. If the cloud user agent has known all bid sequences, he could make an optimal offline strategy and get the maximum benefit. However, in the online manner, the cloud user agent has no idea about the future bid of cloud providers. We present the online reverse auction-based algorithm to help the cloud user agent to purchase service. That is, the cloud user agent chooses the winning cloud providers according to the online reverse auction-based algorithm, while a bidder's bid meets the request of online reverse auction-based algorithm. In Figure 5, the blue bar, green bar, and brown bar represent the trading

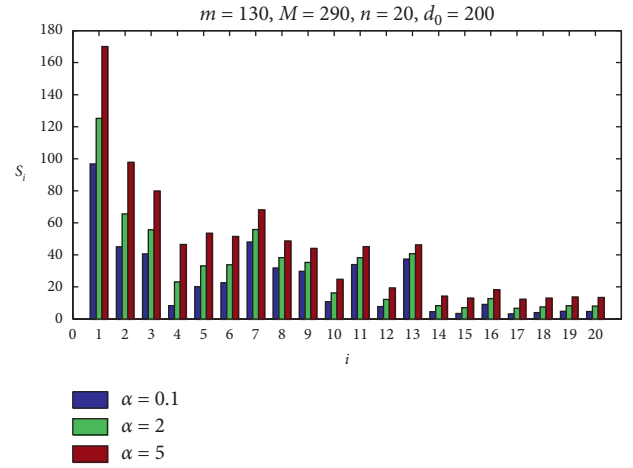


Figure 5: The efficiency of online reverse auction-based algorithm with a.

funds, respectively, while $a = 0.1$, $a = 2$, and $a = 5$. It is shown that when $b_1 = 133$, the trading funds are the highest among all s_i. This accords with the risk ideas. Since the adversary would make the cloud user agent in worst case, the cloud user agent would purchase more services when the first bid satisfies the trading rules based on a-threat based algorithm. Furthermore, compared with Ding et al. [7], the results are more instructive and meaningful for taking into account transaction cost and transaction price in this paper.

7. Conclusions

This paper proposes a novel online reverse auction method for allocating the cloud computing services, which can help the cloud users and providers to build workflow applications in a cloud computing environment. The proposed online reverse auction-based algorithm is evaluated by the competitive analysis from the free case and chargeable case. The results prove that the proposed reverse auction algorithm is an appropriate mechanism, because it allows the cloud user agent to make purchase decisions without knowing the future bids. The difference of auction rounds and transaction cost can impressively influence and improve the performance of the proposed reverse auction algorithm. In the future, except for the price of cloud services, some other factors, e.g., cloud service efficiency and cloud service quality, can be introduced into the averse auction market.

Data Availability

All data, models, and codes generated or used during the study appear in the submitted article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was supported by the Qingdao Social Science Project (no. QDSKL1801169).

References

[1] X. J. Ma, H. H. Gao, H. H. Xu, and M. J. Bian, "An IoT-based task scheduling optimization scheme considering the deadline and cost-aware scientific workflow for cloud computing," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, no. 1, pp. 1–19, 2019.

[2] Y. Yin, L. Chen, Y. Xu, J. Wan, H. Zhang, and Z. Mai, "QoS prediction for service recommendation with deep feature learning in edge computing environment," *Mobile Networks and Applications*, vol. 25, no. 2, pp. 391–401, 2020.

[3] H. Gao, L. Kuang, Y. Yin, B. Guo, and K. Dou, "Mining consuming behaviors with temporal evolution for personalized recommendation in mobile marketing apps," *Mobile Networks and Applications*, vol. 25, no. 4, pp. 1233–1248, 2020.

[4] X. Yang, S. Zhou, and M. Cao, "An approach to alleviate the sparsity problem of hybrid collaborative filtering based recommendations: the product-attribute perspective from user reviews," *Mobile Networks and Applications*, vol. 25, no. 2, pp. 376–390, 2020.

[5] H. Gao, C. Liu, Y. Li, and X. Yang, "V2VR: reliable hybrid-network-oriented V2V data transmission and routing considering RSUs and connectivity probability," *Institute of Electrical and Electronics Engineers Transactions on Intelligent Transportation Systems*, vol. 2020, Article ID 2983835, 14 pages, 2020.

[6] W. Shi, L. Zhang, C. Wu, Z. Li, and F. C. M. Lau, "An online auction framework for dynamic resource provisioning in cloud computing," *ACM Sigmetrics Performance Evaluation Review*, vol. 42, no. 1, pp. 71–83, 2014.

[7] L. L. Ding, L. Chang, and L. Wang, "Online auction-based resource scheduling in grid computing networks," *International Journal of Distributed Sensor Networks*, vol. 12, no. 10, pp. 1–12, 2016.

[8] L. Ding, M. Liu, W. Kang, and X. Zhao, "Prior-free auction mechanism for online supplier with risk taking," *Computers & Industrial Engineering*, vol. 133, pp. 1–8, 2019.

[9] K. Chard, K. Bubendorfer, S. Caton, and O. F. Rana, "Social cloud computing: a vision for socially motivated resource sharing," *Institute of Electrical and Electronics Engineers Transactions on Services Computing*, vol. 5, no. 4, pp. 551–563, 2012.

[10] P. Samimi, Y. Teimouri, and M. Mukhtar, "A combinatorial double auction resource allocation model in cloud computing," *Information Sciences*, vol. 357, pp. 201–216, 2016.

[11] K. Kang, S. X. Xu, R. Y. Zhong, B. Q. Tan, G. Q. Huang, and T. Peng, "Double auction-based manufacturing cloud service allocation in an industrial park," *Institute of Electrical and Electronics Engineers Transactions on Automation Science and Engineering*, vol. 278, pp. 1–13, 2020.

[12] S. A. Tafsiri and S. Yousefi, "Combinatorial double auction-based resource allocation mechanism in cloud computing market," *Journal of Systems and Software*, vol. 137, pp. 322–334, 2018.

[13] N. Nisan and A. Ronen, "Computationally feasible VCG mechanisms," *Journal of Artificial Intelligence Research*, vol. 29, pp. 19–47, 2007.

[14] A. Goldberg, J. Hartline, and A. Wright, "Competitive auctions and digital goods," Inter Trust Technical Report STAR-TR-99-01, Elsevier Science Publishers B. V, Amsterdam, Netherlands, 1999.

[15] I. Fujiwara, K. Aida, and I. Ono, "Combinatorial auction-based marketplace mechanism for cloud service reservation," *IEICE Transactions on Information and Systems*, vol. E95-D, no. 1, pp. 192–204, 2012.

[16] H. Wang, H. Tianfield, and Q. Mair, "Auction based resource allocation in cloud computing," *Multiagent and Grid Systems*, vol. 10, no. 1, pp. 51–66, 2014.

[17] S. Zaman and D. Grosu, "Combinatorial auction-based allocation of virtual machine instances in clouds," *Journal of Parallel and Distributed Computing*, vol. 73, no. 4, pp. 495–508, 2013.

[18] C. Lee, P. Wang, and D. Niyato, "A real-time group Auction system for efficient allocation of cloud internet applications," *Institute of Electrical and Electronics Engineers Transactions on Services Computing*, vol. 8, no. 2, pp. 251–268, 2015.

[19] J. Roovers, K. Vanmechelen, and J. Broeckhove, *A Reverse Auction Market for Cloud Resources. Economics of Grids, Clouds, Systems, and Services*, Springer, Berlin Heidelberg, Germany, 2011.

[20] R. Prodan, M. Wieczorek, and H. M. Fard, "Double auction-based scheduling of scientific applications in distributed grid and cloud environments," *Journal of Grid Computing*, vol. 9, no. 4, pp. 531–548, 2011.

[21] A. S. Prasad and S. Rao, "A mechanism design approach to resource procurement in cloud computing," *Institute of Electrical and Electronics Engineers Transactions on Computers*, vol. 63, no. 1, pp. 17–30, 2014.

[22] W. L. Kang, L. Wang, and W. C. Yu, "Multi-attribute double auction models for resource allocation in computational grids," *Open Cybernetics and Systemics Journal*, vol. 8, no. 1, pp. 357–362, 2014.

[23] M. T. Hajiaghayi, R. Kleinberg, and D. C. Parkes, "Adaptive limited-supply online auctions," in *Proceedings of the 5th ACM Conference on Electronic Commerce*, pp. 71–80, New York, NY, USA, January 2004.

[24] R. Lavi and N. Nisan, "Competitive analysis of incentive compatible on-line auctions," *Theoretical Computer Science*, vol. 310, no. 1-3, pp. 159–180, 2004.

[25] G. Zhu, S. Sangwan, and T. Lu, "Mechanism design of online multi-attribute reverse auction," in *Proceedings of the 42nd Hawaii International Conference on System Sciences*, pp. 1–7, Waikoloa, HI, USA, January 2009.

[26] H. Zhang, B. Li, H. Jiang, F. Liu, and J. Liu, "A framework for truthful online auctions in cloud computing with heterogeneous user demands," in *Proceedings of the 2013 Institute of Electrical and Electronics Engineers INFOCOM*, pp. 1510–1518, April 2013, Turin, Italy.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 6642121, 18 pages
https://doi.org/10.1155/2021/6642121

Research Article

PB: A Product-Bitmatrix Construction to Reduce the Complexity of XOR Operations of PM-MSR and PM-MBR Codes over GF(2^w)

Chuqiao Xiao, Xueqing Gong, Yefeng Xia, and Qian Zhang

East China Normal University, Software Engineering Institute, Shanghai 200062, China

Correspondence should be addressed to Xueqing Gong; xqgong@sei.ecnu.edu.cn

Received 21 October 2020; Revised 18 November 2020; Accepted 18 December 2020; Published 30 January 2021

Academic Editor: Honghao Gao

Edge computing, as an emerging computing paradigm, aims to reduce network bandwidth transmission overhead while storing and processing data on edge nodes. However, the storage strategies required for edge nodes are different from those for existing data centers. Erasure code (EC) strategies have been applied in some decentralized storage systems to ensure the privacy and security of data storage. Product-matrix (PM) regenerating codes (RGCs) as a state-of-the-art EC family are designed to minimize the repair bandwidth overhead or minimize the storage overhead. Nevertheless, the high complexity of the PM framework contains more finite-domain multiplication operations than classical ECs, which heavily consumes computational resources at the edge nodes. In this paper, a theoretical derivation of each step of the PM minimum storage regeneration (PM-MSR) and PM minimum bandwidth regeneration (PM-MBR) codes is performed and the XOR complexity over finite fields is analyzed. On this basis, a new construct called product bitmatrix (PB) is designed to reduce the complexity of XOR operations in the PM framework, and two heuristics are used to further reduce the XOR numbers of the PB-MSR and PB-MBR codes, respectively. The evaluation results show that the PB construction significantly reduces the XOR number compared to the PM-MSR, PM-MBR, Reed–Solomon (RS), and Cauchy RS codes while retaining optimal performance and reliability.

1. Introduction

Edge computing has emerged as a new paradigm for addressing local computing needs and moving data computation or storage to an edge node near the end user [1–3]. In contrast to cloud storage built on a network, edge computing-based storage is distributed among the edges of the network structure [4, 5]. Consequently, these edge nodes are particularly in need of fault-tolerant techniques to ensure system reliability and availability and even more importantly to ensure data privacy and security [6]. Nowadays, whether it is a distributed storage system [7] or a decentralized system [8, 9], it needs to guarantee the storage reliability of the source data, and these systems have used an erasure code (EC) strategy as opposed to a replication strategy which has a higher storage overhead. [10]. The well-known Reed–Solomon (RS) codes or more generally, maximum distance separable codes (MDS), have been applied to many storage systems [11]. But in a word, an EC is essentially a particular linear combination of data symbols that involves a large number of finite field matrix product operations, and the calculation complexity over finite field is too high. There have been many studies on this issue [12–14]. In addition to the problem of high computational complexity over GF(2^w), another problem of EC is that their repair bandwidth is equal to the size of the entire source data [15]. It has been reported that, based on RS coding, an average of 95,500 coded blocks per day needs to be recovered, and more than 180 TB of data per day must be transmitted through the TOR switch for data recovery [16]. Recently, a new class of ECs called regenerating codes (RGCs) has emerged that trades off both storage overhead and repair bandwidth [17]. RGC maintains the same reliability as ECs, but both of them are calculated over a finite field. There are two principal RGC classes,

namely, minimum storage regenerating (MSR) and minimum bandwidth regenerating (MBR) codes, which imply two extreme points in a trade-off known as the *storage-repair bandwidth trade-off*. There have been many frameworks designed for MSR and MBR codes, respectively [7, 18–20], but as far as we know, the product-matrix (PM) framework proposed by Rashmi et al. is the only framework that constructs two codes in a unified way [21], and the PM framework provides exact repair of PM-MSR $(2k-2 \le d)$ and PM-MBR $(k \le d \le n-1)$ codes. While there has been some research on the optimization of extensions based on this framework, such as reducing the disk I/O overhead or dynamically changing the number of helper nodes according to system requests and network changes, these studies have not analyzed and optimized the framework itself [22–24]. In this paper, we focus on the PM framework, analyzing the computational complexity of the encoding, decoding, and repair processes over GF(2^w). Since the PM framework has a large number of XORs at each step, especially in the decoding process of the PM-MSR code, the computational complexity of the inversion of the Vandermonde matrix is $O(n^3)$. Therefore, we propose a new construction called product bitmatrix (PB). Our contributions include the following:

(i) The computational complexity of the PM-MSR and PM-MBR codes is strictly theoretically derived in combination with the finite field arithmetic operations.

(ii) A new construction called product bitmatrix for MSR and MBR codes is designed, and we elaborate on the encoding, decoding, and repair processes in detail.

(iii) Two new heuristic algorithms are presented that find a locally optimal PB which has low XORs, thereby reducing computational complexity. The experimental results show that the number of XORs dropped by 11.73% of the PB-MSR and 20.17% of the PB-MBR codes.

The remainder of this paper is organized as follows. The background of the two codes and arithmetic complexity analysis over the finite field are analyzed in Section 2. Section 3 is the analysis of the computational complexity of the encoding, decoding, and repair processes of the PM framework. In Section 4, a new PB construction is presented and implemented. In Section 5, two different heuristic algorithms are designed to find the locally optimal product bitmatrix that has the least number of XORs. Then, the results of many evaluation experiments conducted to verify the performance of optimization are presented in Section 6. Finally, Sections 7 and 8 are related works and conclusions.

2. Foundations

2.1. MSR and MBR Codes. RGCs are state-of-the-art ECs that aim to reduce the amount of data during the repair process, which means a new replacement node needs to connect to any d helper nodes and download β symbols from each node. As shown in Figure 1, a comparison between the reconstruction and repair processes is presented. Any $k\alpha$ symbols in an α-level stripe are transferred to reconstruct origin symbols, while $d\beta$ symbols downloaded for repair are much smaller than the total size of B. It is shown in the following formula:

$$B \le \sum_{i=0}^{k-1} \min(\alpha, (d-i)\beta). \tag{1}$$

Since both storage overhead α and bandwidth overhead β are costs, it is desirable to minimize both α as well as β. In [17], when (n, k, d, B) are fixed values, there is a trade-off between α and β; the two extreme points in this trade-off are termed the MSR and MBR. Clearly, (α, β) of the minimum storage point is given by $((B/k), (\alpha/d-k+1))$, and at the the minimum bandwidth point of the trade-off, (α, β) is given by $(d\beta, (2B/k(2d-k+1)))$. RGC over GF(2^w) is associated with a set of parameters $\{n, k, d, \alpha, \beta, B\}$. A comparison about metrics between Replication, EC, MSR, and MBR is shown in Table 1.

As shown in Table 1, the upper bound of storage efficient $\mathbf{R}_{\mathrm{MBR}}$ code is reached (1/2) when $k = d = n-1$, but there is no such limitation of R_{MSR}. For the sake of simplicity, if the size of a file is $B = 6$, $(n, k, d) = (6, 3, 5)$, the repair bandwidth γ of EC, MSR, and MBR is 6, 3.33, and 2.5,respectively, and the total storage overhead $n\alpha$ of EC, MSR, and MBR is 12, 12, and 15, respectively. Although the EC and MSR codes use the same storage, the repair bandwidth of EC is higher than MSR. MBR code is where storage space is allowed to expand, but repair bandwidth is at the lowest bound. The MSR codes minimize γ based on the optimal α, while the MBR codes minimize α under the optimal γ.

2.2. Finite Field Arithmetic Complexity. Since the finite field of prime size is not suitable for computers, the finite field of a size equal to a power of 2 is generally preferred. In GF(2^w), each element is represented by the *formal power series*, which is a way of writing infinite sequences of bits. For example, a and b are bit sequences over the finite field GF(2); $\mathbf{a} = (a_0, a_1, a_2, a_3, \ldots)$; $\mathbf{b} = (b_0, b_1, b_2, b_3, \ldots)$. Addition of a and b could be easily done by bitwise XOR, but multiplication is more complicated, $a \cdot b = (a_0 b_0, a_0 b_1 + a_1 b_0, \ldots, \sum_{i=0}^{l-1} a_i b_{l-1-i\ldots})$, where l is length of sequence. A notation $a(z)$ is used as a formal power series (polynomial) to represent an infinite sequence:

$$a(z) = a_0 + a_1 z + a_2 z^2 + \cdots + a_{l-1} z^{l-1}. \tag{2}$$

The multiplication over GF(2^w) is performed by multiplying two polynomials with coefficients in binary field GF(2) and then reducing the product by irreducible polynomials [25]. Hence, a polynomial represents a field element, and the set of polynomials is denoted by $\mathbb{F}_2[z]$. Due to the highest power series, *dega(z)* of $a(z)$ is equal to *w-1*, and the sum of nonzero terms of polynomials as the *weight* of $a(z)$ is denoted as $\|a(z)\|$ which equals w.

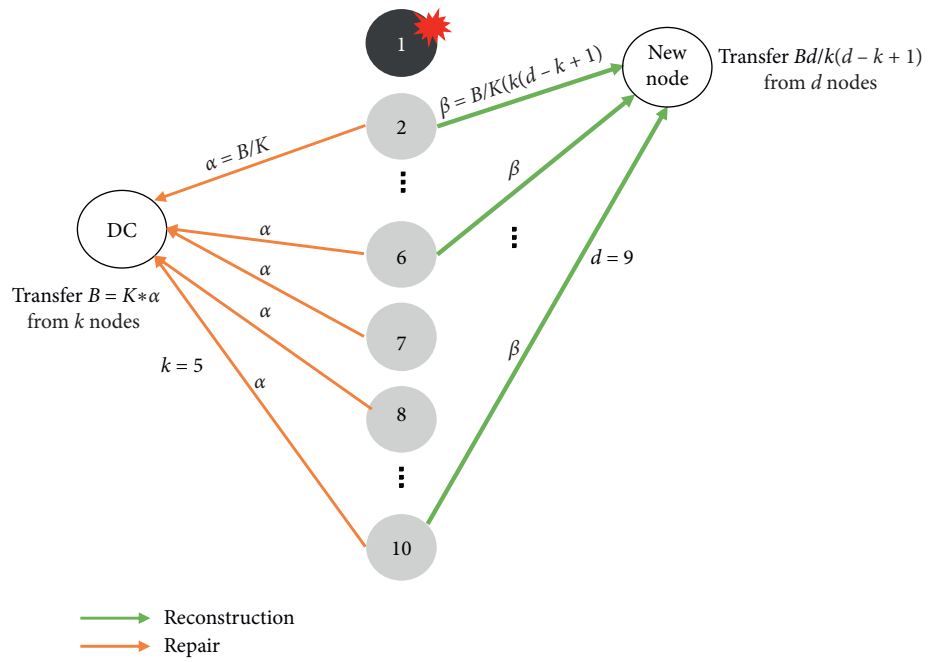


Figure 1: An example of parameters ($n = 10$, $k = 5$, $d = 9$): reconstruction by the data collector (DC) is shown on the left and repair by a new node when old node has failed is shown on the right.

Table 1: Comparison of replication, EC, MSR, and MBR.

	Replication	Erasure code	MSR	MBR
Parameters	–	(n, k)	$(n \geq 2k-1, k, d \geq 2k-2)$	(n, k, d)
Storage efficient **R**	(k/n)	(k/n)	(k/n)	$(2d-k+1/2d) \cdot (k/n)$
Storage overhead α	(B/k)	(B/k)	(B/k)	$(B/k) \cdot (2d/(2d-k+1))$
Repair bandwidth γ	(B/k)	B	$(B/k) \cdot (d/(d-k+1))$	$(B/k) \cdot (2d/(2d-k+1))$
Disk access	1	k	d	d

The addition is a bitwise XOR operation whose complexity is w.

For multiplication, where $c(z) := a(z)b(z)$, there are two steps to compute complexity: compute the *product* and *reduce* while $a(z), b(z) \in \in \mathbb{F}_2[z]$, $a(z) \neq b(z)$.

Step 1: in the \emph{product} stage, the complexity of this stage is the \emph{coefficients-XOR}, where

$$c(z) = a_0b_0 + (a_1b_0 \oplus a_0b_1)z + (a_2b_0 \oplus a_1b_1 \oplus a_0b_2)z^2 + \cdots + (a_{w-1}b_0 \oplus a_{w-2}b_1 \oplus a_{w-3}b_2 \oplus \cdots \oplus a_1b_{w-2} \oplus a_0b_{w-1})z^{w-1}. \quad (3)$$

For example, $a_0 = 1$, $a_1 = 1$, $a_2 = 1$, $b_0 = 0$, $b_1 = 1$, $b_2 = 1$. $c(z) = 0 + (0 \oplus 1)z + (0 \oplus 1 \oplus 1)z^2 + (1 \oplus 1)z^3 + z^4 = z^4 + z$. The sum of XORs is $S = 1 + \cdots + w - 2 + w - 1 + w - 2 + \cdots + 1 = (w-1)^2$, so the *product* stage is $(w-1)^2$.

Step 2: in the second stage, if $degc(z) \geq w$, $c(z)$ is reduced by an irreducible polynomial $g(z)$, until $degc(z)$ w. Let $l = degc(z)$, and convert the term $c_l z^l$ to $c_l(g(z) - z^w)z^{l-w}$. Because an irreducible polynomial $g(z)$ degree is w, $h(z) = g(z)$-z^w, $degh(z) = w'$, and maximum *weight* of $h(z)$ is $w' + 1$.

Divide $c(z)$ into three polynomials: *no-reduce* part $d(z)$, *once-reduce* part $e(z)$, and *twice-reduce* part $f(z)$ as shown in Table 2. While $w - 2 + w' \leq w - 1$, there is a nonexistent second reduction.

Because $c(z) = d(z) + e(z) + f(z)$, the reduction part is as follows:

Table 2: Dividing $c(z)$ into three polynomials.

	$d(z)$	$e(z)$	$f(z)$
Power range	$[0, w-1]$	$[w, 2w-w'-1]$	$[2w-w', 2w-2]$
Max*weight*	w	$w-w'$	$w'-1$
Product XORs	0	w'^2	$(w-2)^2 + w'^2$

$$\begin{aligned} e(z) + f(z) &= c_w z^w + c_{w+1} z^{w+1} + c_{w+2} z^{w+2} + \cdots + c_{2w-2} z^{2w-2} \\ &= z^w \left(c_w + c_{w+1} z + c_{w+2} z^2 + \cdots + c_{2w-w'-1} z^{w-w'-1} \right) \\ &\quad + z^w \left(c_{2w-w'} z^{w-w'} + \cdots + c_{2w-2} z^{w-2} \right) \\ &= h(z) \left(c_w + c_{w+1} z + c_{w+2} z^2 + \cdots + c_{2w-w'-1} z^{w-w'-1} \right) \\ &\quad + h(z) \left(c_{2w-w'} z^{w-w'} + \cdots + c_{2w-2} z^{w-2} \right). \end{aligned} \tag{4}$$

Therefore, the *once-reduce* part e(z) has w'^2 XORs. Then, we compute number of XORs of the *twice-reduce* part.

Let $h(z)(c_{2w-w'} z^{w-w'} + \cdots + c_{2w-2} z^{w-2}) = h(z) * m(z)$, and the new polynomial $m(z) = \sum_{i=2}^{w'} z^{w-i}$, $i \in (2, w')$, $i \in (2, w')$; $degm(z) = w-2$ and $\|m(z)\| = w'-1$. Then, $f(z) = f_0 + f_1 z + f_1 z^2 + \cdots + f_{w+w'-2} z^{w+w'-2}$ which needs $(w-2)^2$ XORs. And if $w' \geq 1$, f(z) also must be reduced.

$$\begin{aligned} f(z) &= \left(f_0 + f_1 z + \cdots + f_{w-1} z^{w-1} \right) + \left(f_w z^w + \cdots + f_{w+w'-2} z^{w+w'-2} \right) \\ &= \left(f_0 + f_1 z + \cdots + f_{w-1} z^{w-1} \right) + \underbrace{h(z) \left(f_w + \cdots + f_{w+w'-2} z^{w'-2} \right)}_{\text{product XORs is } w'^2} \\ &= \underbrace{\left(f_0 + f_1 z + \cdots + f_{w-1} z^{w-1} \right) + \left(n_0 + n_1 z + \cdots + n_{w-1} z^{2w'-2} \right)}_{\text{addition XORs is } 2w'-2}, \end{aligned} \tag{5}$$

where n(z) is a polynomial in which $\mathbf{degn}(\mathbf{z}) = 2w' - 2$. Thus, *twice-reduce* part f(z) needs $w'^2 + (w-2)^2 + 2w' - 1$ XORs.

Consequently, the sum of XORs μ of the field *multiplication* over GF(2^w) including two steps of the *addition* XOR among three parts is

$$\begin{aligned} \mu &= \text{XOR}_{\text{step1}} + \text{XOR}_{\text{step2}} \\ &= (w-1)^2 + \left[(w')^2 + w \right] + \left[w'^2 + (w-2)^2 + 2w' - 1 + w \right] \\ &= 2w(w-2) + 2w'(2w'+1) + 4. \end{aligned} \tag{6}$$

Generally, a multiplication operation over GF(2^w) takes $O(w^2)$ bit operations. For example, when $w = 3$, multiplication results are as shown in Table 3. Because the finite field size equals 2^8 that corresponds to a byte in the computer, w is usually set to be an integral times of 8 to make encoding operations convenient.

Matrix inversion is often used in EC decoding and repair processes. Suppose A is $(n \times n)$ matrix $(3 \leq n)$, and $A^{-1} = (A^*/|A|)$; the determinant of A is solved by trigonometric matrix. Elements in the solution process include addition and multiplication, which require $\sum_{i=2}^{n} i(i-1)\mu + \sum_{i=2}^{n} i(i-1)w$ XORs. Adjoint matrix $A^*$is an $(n \times n)$ matrix composed of algebraic cofactors in which each element $A_{i,j}$ equals the determinant value of A excluding row i and column j. Therefore, computing A^* requires $n^2 \sum_{i=2}^{n-1} i(i-1)\mu + n^2 \sum_{i=2}^{n-1} i(i-1)w$ XORs.

Consequently, the XORs of matrix inversion are

$$\begin{aligned} &\left(n^2 + 1 \right)(\mu + w) \sum_{i=2}^{n-1} i(i-1) + n(n-1)(\mu + w) \\ &= \frac{n^3 - 3n^2 + 2n}{3} \left(n^2 + 1 \right)(\mu + w) + n(n-1)(\mu + w) \\ &= \frac{n^5 - 3n^4 + 4n^3 - n}{3} (\mu + w). \end{aligned} \tag{7}$$

Table 3: Multiplication over GF(2^3) while $g(z) = z^3 + z + 1$ is an irreducible polynomial.

		0 000	1 001	x 010	$x+1$ 011	x^2 100	x^2+1 101	x^2+x 110	x^2+x+1 111
0	000	0	0	0	0	0	0	0	0
1	001	0	1	x	$x+1$	x^2	x^2+1	x^2+x	x^2+x+1
x	010	0	x	x^2	x^2+x	$x+1$	1	x^2+x+1	x^2+1
$x+1$	011	0	$x+1$	x^2+x	x^2+1	x^2+x+1	x^2	1	x
x^2	100	0	x^2	$x+1$	x^2+x+1	x^2+x	x	x^2+1	1
x^2+1	101	0	x^2+1	1	x^2	x	x^2+x+1	$x+1$	x^2+x
x^2+x	110	0	x^2+x	x^2+x+1	1	x^2+1	$x+1$	x	x^2
x^2+x+1	111	0	x^2+x+1	x^2+1	x	1	x^2+x	x^2	$x+1$

3. The Computational Complexity of PM-MSR and PM-MBR

The famous PM-RGC construction is introduced to provide MBR constructions for all feasible values of (n, k, d) and MSR constructions for all $(n, k, d \geq 2k-2)$[21]. A detailed description of the complexity of the PM-MSR and PM-MBR codes is presented in this section, where matrices are used in the framework listed in Table 4.

For an $(n, k, d = 2k-2)$ MSR code with $\beta = 1$, $\alpha = d - k + 1 = k - 1$, $d = 2\alpha$, $B = k\alpha = \alpha(\alpha+1)$ over GF(2^w). The original symbols $\{u_i\}_{i=1}^{B}$ are arranged in the form of $M = [S_1 \ S_2]^t$ where the $\binom{\alpha+1}{2}$ entries in the upper-triangular part of S_1 and S_2 are filled by $\binom{\alpha+1}{2}$ symbols.

Encoding is constructed using $\Psi = [\Phi \ \Lambda\Phi]$ where each element is chosen to satisfy the properties where (1) any d rows of Ψ are linearly independent; (2) any α rows of Φ are linearly independent; and (3) the n diagonal elements of Λ are distinct. Let the i-th row of Ψ be $(1 \times d)$ ψ_i^t, the i-th row of Φ be $(1 \times \alpha)$ ϕ_i^t, and the i-th row of Λ be $(1 \times n)$ λ_i. The α coded symbols stored by the i-th node are given by $\underline{c}_i^t = \psi_i^t M = \phi_i^t S_1 + \lambda_i \phi_i^t S_2$. The complexity of encoding equals the sum of addition and multiplication XORs, where the MSR encoding complexity is $n\alpha[d\mu + (d-1)w]$ and each data encoding needs $(n/k)(d\mu + (d-1)w)$ XORs.

Decoding is $\mathscr{DC}$ to reconstruct B. This process has three steps. Let $\Psi_{DC} = [\Phi_{DC}\Lambda_{DC}\Phi_{DC}]$ be the submatrix of Ψ corresponding to the k nodes. Then, get encoded data

$$\Psi_{DC} M = [\Phi_{DC}\Lambda_{DC}\Phi_{DC}][S_1 \ S_2]^t = \Phi_{DC}S_1 + \Lambda_{DC}\Phi_{DC}S_2. \tag{8}$$

(1) Let $\Psi_{DC} M \Phi_{DC}^t = \Phi_{DC}S_1\Phi_{DC}^t + \Lambda_{DC}\Phi_{DC}S_1\Phi_{DC}^t$; the result is $(k \times k)$ matrix. This step has $k^2[\alpha\mu + (\alpha - 1)w]$ XORs.

(2) Set $P = \Phi_{DC}S_1\Phi_{DC}^t$, $Q = \Phi_{DC}S_2\Phi_{DC}^t$, $\Phi_{DC} = [\phi_1^t \phi_2^t \ldots \phi_\alpha^t]^t$. Because S_1, S_2 is symmetric, P and Q are symmetric. $\Psi_{DC} M \Phi_{DC}^t = P + \Lambda_{DC} Q$. Since $i \neq j$ and $1 \leq (i, j) \leq k$, the $(i, j)^{\text{th}}$ element is $P_{ij} + \lambda i Q_{ij} = P_{ji} + \lambda i Q_{ji}$. Then, $(k(k-1)/2)$ sets of equations are derived and one set of equations is as follows:

$$\begin{cases} P_{ij} + \lambda_i Q_{ij} = \psi_i^t M \phi_i \\ P_{ij} + \lambda_j Q_{ij} = \psi_j^t M \phi_j \end{cases} \longrightarrow \begin{bmatrix} 1 & \lambda_i \\ 1 & \lambda_j \end{bmatrix} \begin{bmatrix} P_{ij} \\ Q_{ij} \end{bmatrix} = \begin{bmatrix} \psi_i^t M \phi_i \\ \psi_j^t M \phi_j \end{bmatrix}. \tag{9}$$

This step involves matrix inversion; the XORs to compute a pair (P_{ij}, Q_{ij}) is $16\mu + 8w$. Consequently, the DC solves the values of P_{ij}, Q_{ij} for all $i \neq j$. The sum of XORs in this stage is $(k(k-1)/2)(16\mu + 8w)$.

(3) After elements of P are known, the i-th row (excluding the diagonal element) is given by $\phi_i^t S_1 [\phi_1 \ldots \phi_{i-1}\phi_{i+1} \ldots \phi_{\alpha+1}]$ $(1 \leq i \leq \alpha + 1)$, and $\{\phi_i^t S_1 | 1 \leq i \leq \alpha + 1\}$ is recovered through right product $[\phi_1 \ldots \phi_{i-1}\phi_{i+1} \ldots \phi_{\alpha+1}]^{-1}$. Then, compute $[\phi_1 \ldots \phi_\alpha]^t S_1$, from which S_1 is recovered. S_2 is similarly recovered from Q. In this stage, XOR is

Table 4: The PM framework matrix list.

Notation	Size	Name
M	MSR: $d \times \alpha$, MBR: $d \times d$	Message matrix
S_1, S_2	$\alpha \times \alpha$	Symmetric matrix
S	$k \times k$	Submatrix of M
T, T^t	$k \times (d-k)$, $(d-k) \times k$	Submatrix of M
Ψ	$n \times d$	Encoding matrix
Φ	MSR: $n \times \alpha$, MBR: $n \times k$	Submatrix of Ψ
Λ	$n \times n$	Diagonal matrix
Δ	$n \times (d-k)$	Submatrix of Ψ
C	MSR: $n \times \alpha$, MBR: $n \times d$	Code matrix
Φ_{DC}	MSR: $k \times \alpha$, MBR: $k \times k$	Submatrix of Ψ_{DC}
Ψ_{DC}	$k \times d$	Decoding matrix
Ψ_{repair}	$d \times d$	Repair matrix

$$2\left[\frac{\alpha^5-3\alpha^4+4\alpha^3-\alpha}{3}(\mu+w)(\alpha+1)+2\alpha^3\mu+2\alpha^2(\alpha-1)w\right]. \tag{10}$$

For example, if $k=3$, $\alpha=2$ where

$$\Psi_{\mathrm{DC}}M\Phi_{\mathrm{DC}}^{t}=\begin{bmatrix}c_{11}&c_{12}&c_{13}\\c_{21}&c_{22}&c_{23}\\c_{31}&c_{32}&c_{33}\end{bmatrix}_{3\times3},$$

$$P=\begin{bmatrix}\backslash&\phi_1^tS_1\phi_2&\phi_1^tS_1\phi_3\\\phi_2^tS_1\phi_1&\backslash&\phi_2^tS_1\phi_3\\\phi_3^tS_1\phi_1&\phi_3^tS_1\phi_2&\backslash\end{bmatrix}_{3\times3}, \tag{11}$$

$$Q=\begin{bmatrix}\backslash&\phi_1^tS_2\phi_2&\phi_1^tS_2\phi_3\\\phi_2^tS_2\phi_1&\backslash&\phi_2^tS_2\phi_3\\\phi_3^tS_2\phi_1&\phi_3^tS_2\phi_2&N_3\end{bmatrix}_{3\times3}.$$

The sum of XORs during *decoding* is $126\mu+77w$.

Repair is to regenerate coded symbols on failed node f. Thus, the aim is to reconstruct $\underline{c}_f$ where

$$\underline{c}_f=\psi_f^tM=\left[\phi_f^t\lambda_f\phi_f^t\right]M=\phi_f^tS_1+\lambda_f\phi_f^tS_2. \tag{12}$$

There are two roles in this stage: helper(h) and replacement(r) nodes, respectively. At first, the i-th helper node $(h_i, i\in[d])$ computes the inner product $\psi_{h_j}^tM\phi_f$ and transmits this value to the r node. Then, then r node obtains the vector $\Psi_{\mathrm{repair}}M\phi_f=[\psi_{h_1}\psi_{h_2}\ldots\psi_{h_d}]^tM\phi_f$. Because Ψ_{repair} is invertible, the vector $M\phi_f$ is recovered. From $M\phi_f$: $S_1\phi_f, S_2\phi_f$, since S_1 and S_2 are symmetric matrices,

$$\left(M\phi_f\right)^t=\phi_f^tM^t=\left[\phi_f^tS_1\ \ \phi_f^tS_2\right]. \tag{13}$$

The content $\underline{c}_f$ is eventually recovered. The process of repair is not complicated. The number of XORs on each h node is $\alpha\mu+(\alpha-1)w$ and the total is $d[\alpha\mu+(\alpha-1)w]$. The computational complexity of r includes matrix inversion, matrix product, and coefficient product and is represented as follows:

$$\frac{d^5-3d^4+4d^3-d}{3}(\mu+w)+d^2\mu+d(d-1)w+\alpha(\mu+w). \tag{14}$$

All feasible values of (n, k, d) of MBR codes have $\beta=1$, $\alpha=d$, $B=\binom{k+1}{2}+k(d-k)$ over $\mathrm{GF}(2^w)$. The original symbols $\{u_i\}_{i=1}^{B}$ are arranged in the form of $M=\begin{bmatrix}S&T\\T^t&0\end{bmatrix}$ where the upper-triangular part of the symmetric matrix S is filled by $\binom{k+1}{2}$ symbols, and the remaining $k(d-k)$ symbols are used to construct matrix T.

Encoding is to obtain coding matrix C by matrix product with Ψ and M. Define the encoding matrix Ψ as the form $\Psi=\left[\Phi\ \ \Delta\right]$ where each element is chosen in such a way that (1)any d rows of Ψ are linearly independent and (2) any k rows of Φ are linearly independent. Then, C is given by $C=\Psi M$. During this process, the sum of XORs is $nd[d\mu+(d-1)w]$.

Decoding is *DC* to recover original symbols B by connecting to any k nodes. Because the i-th node stored the vector ψ_i^tM, the DC gets the $(k\times d)$ matrix $\Psi_{\mathrm{DC}}M=\left[\Phi_{\mathrm{DC}}S+\Delta_{\mathrm{DC}}T^t\ \ \Phi_{\mathrm{DC}}T\right]$. Thus, multiply the matrix Φ_{DC}^{-1} on the left of $\Phi_{\mathrm{DC}}T$ to recover first T, and the number of XOR in this step is $(k^5-3k^4+4k^3-k/3)(\mu+w)+k(d-k)[k\mu+(k-1)w]$. Subsequently, compute $\Phi_{\mathrm{DC}}S+\Delta_{\mathrm{DC}}T^t-\Delta_{\mathrm{DC}}T^t$ and then multiply Φ_{DC}^{-1} on the left to recover S. The XOR of this step is $k^2+k^2[k\mu+(k-1)w]$.

Repair is recover ψ_f^tM symbols stored in the failed node to a replacement node by connecting to an arbitrary set $\left\{h_j|j=1,\ldots,d\right\}$ of d helper nodes. First, each h_j computes the inner product $\psi_f^tM\psi_f$ and sends the result to the replacement node. The replacement node thus obtains the d symbols $\Psi_{\mathrm{repair}}M\psi_i$ and then multiplies on the left by $\Psi_{\mathrm{repair}}^{-1}$ to recover matrix $M\psi_f$. Since M is symmetric, $(M\psi_f)^t=\psi_f^tM$. The XOR of repair on the i-th node is $d\mu+(d-1)w$ and on the replacement node is $(d^5-3d^4+4d^3-d/3)(\mu+w)+d^2[d\mu+(d-1)w]$.

The computational complexity of the PM-MSR and PM-MBR codes is shown in Table 5. As seen from Table 5, the PM framework requires multiple matrix inversions, and the number of XORs in the decoding process is very high.

4. The New Product Bitmatrix Ψ Constructed by Cauchy Matrix

In this section, we introduce the process of converting finite field elements into bitmatrices first. Then, we construct a new Ψ with the Cauchy matrix and transform this matrix into a bitmatrix called PB. The encoding, decoding, and repair processes of the MSR and MBR codes based on PB are introduced in detail.

4.1. Transforming Finite Field Elements Using Bitmatrix. Through the analysis of the finite field arithmetic in the previous section, each element e in $\mathrm{GF}(2^w)$ is represented as a *formal power series* and the length of these polynomials is w. In [26], they have described a $1\times w$ row vector V or a $w\times w$ matrix M which is represented as an element over $\mathrm{GF}(2^w)$ in a new representation over $\mathrm{GF}(2)[z]$. For any $e\in\mathrm{GF}(2^w)$, use M(e) as the matrix whose i^{th} $(0\le i)$ column is $V(e*z^{i-1})$; M(1) is the identity matrix and M(0) is the all-zero matrix. For example, Figure 2 shows bitmatrices over $\mathrm{GF}(2^3)$. The bitmatrix of $e=3$ whose 1st column is 011, 2nd

Table 5: The number of XORs of the PM framework in encoding, decoding, and repair processes.

	PM-MSR	*PM-MBR*
Encoding: XORs	$n\alpha[d\mu + (d-1)w]$	$n\,d[d\mu + (d-1)w]$
Decoding: XORs	$(k^2[\alpha\mu + (\alpha-1)w] + /(k(k-1)/2)[16\mu + 8w]+)$ $2[(\alpha^5 - 3\alpha^4 + 4\alpha^3 - \alpha/3)(\mu + w)(\alpha + 1) + 2\alpha^3\mu + 2\alpha^2(\alpha-1)w]$	$(k^5 - 3k^4 + 4k^3 - k/3)(\mu + w) + k(d-k)[k\mu + (k-1)w] + k^2 + k^2[k\mu + (k-1)w]$
Repair: XORs of helper nodes	$d[\alpha\mu + (\alpha-1)w]$	$d[d\mu + (d-1)w]$
Repair: XORs of replacement nodes	$(d^5 - 3d^4 + 4d^3 - d/3)(\mu + w) + d^2\mu + d(d-1)w + \alpha(\mu + w)$	$(d^5 - 3d^4 + 4d^3 - d/3)(\mu + w) + d^2[d\mu + (d-1)w]$

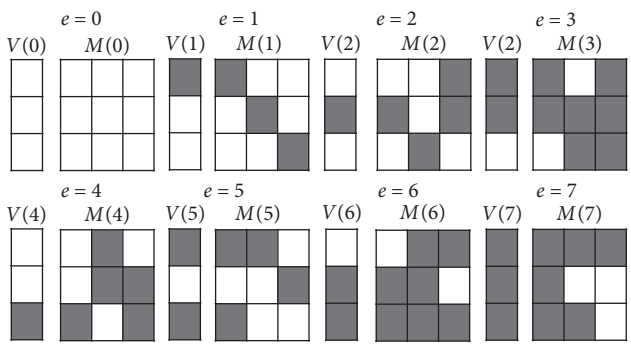


FIGURE 2: Bitmatrix of each element over GF(2^w) $(w = 3)$; M(0) is the all-zero matrix and M(1) is the identity matrix. Each columns except 1st column is calculated by the left-hand column$*2$.

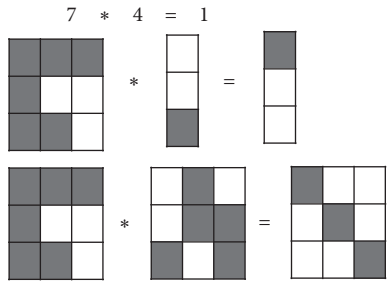


FIGURE 3: Two forms of multiplication based on bitmatrix over GF(8).

column is $3 * z = 3 * 2 = 110$, and 3rd column is $3 * z^2 = 3 * 4 = 111$. There are two forms of multiplication of bitmatrix as shown in Figure 3. Using the bitmatrix representation, the encoding and decoding of PB-MSR and PB-MBR are accomplished by XOR operations, together with some copying operations.

In paper [21], they have adopted classical "Vandermonde" construct Ψ, but the inverse of an $n*n$ Vandermonde-based matrix needs time of complexity $O(n^3)$. One well-known choice is to use the *Cauchy matrix* whose inverse has a time complexity $O(n^2)$. Let $X = \{x_1 \dots x_\alpha\}$ and $Y = \{y_1 \dots y_n\}$, where x_i and y_j are distinct elements over GF(2^w) and $X \cap Y = \varnothing$. Then, the Cauchy matrix defined by X and Y has $(1/x_i + y_j)$ in element (i, j). It is clear that any submatrix of a Cauchy matrix is still a Cauchy matrix. If using a Cauchy matrix as Ψ and converting it to bitmatrix, the number of ones in the bitmatrix means the number of XOR operations in encoding and o is the average number of ones per row in the matrix. Choosing different X and Y will get different o. Shown in Figure 4 is the instance over the finite field GF(8) of the PB-MSR code with $[n = 5, k = 3, d = 4, w = 3, B = 6, \alpha = 2, \beta = 1]$ and PB-MBR code with $[n = 4, k = 2, d = 43w = 3, B = 5, \alpha = 3, \beta = 1]$. When Ψ is constructed by Cauchy matrix, it uses bitmatrix transfer $\Psi \times M$.

4.2. New Product Bitmatrix Ψ of MSR Codes. For the PB-MSR code in the upper part of Figure 4, $X = \{1, 2\}$ and $Y = \{0, 3, 4, 5, 6\}$ are used to construct $(n \times \alpha)$ matrix Φ, and Λ is an $(n \times n)$ diagonal matrix. Because $\Psi = [\ \Phi \ \ \Lambda\Phi\ ]$, any d rows of Ψ are linearly independent, any α rows of Φ are linearly independent, and the n diagonal elements of Λ are distinct. According to these three conditions, encoded matrix $C = \Psi \times M$ is generated, and each row of coded symbols in C is stored on each corresponding node, which means the i-th symbols of a row are stored on the i-th node.

$$\mathbf{c}_1 = [\ u_1 + 5u_2 + u_4 + 5u_5u_2 + 5u_3 + u_5 + 5u_6\ ]. \tag{15}$$

In the reconstruction scene, DC links any three nodes and downloads $k\alpha = 6$ symbols.

$$\mathbf{\Psi}_{\mathrm{DC}} \times M = \begin{bmatrix} u_1 + 5u_2 + u_4 + 5u_5 & u_2 + 5u_3 + u_5 + 5u_6 \\ 5u_1 + u_2 + u_4 + 2u_5 & 5u_2 + u_3 + u_5 + 2u_6 \\ 2u_1 + 3u_2 + 6u_4 + 5u_5 & 2u_2 + 3u_3 + 6u_5 + 5u_6 \end{bmatrix}. \tag{16}$$

According to the decoding process introduced in Section 3, the content of the original symbols is decoded. When node 5 fails, helper nodes $d = 4$. Through connecting any 4 helpers $h_i\,(i \in [d])$ (nodes 1, 2, 3, 4 in the example), a replacement node r can exact regenerate symbols. Each h_i inner product bitmatrix ϕ_f as shown in Figure 5 sends results to replacement node r. Then r calculate equation (17) to obtain intermediate results:

$$\Psi_{\mathrm{repair}} \times M = \psi_{h_i}^t \times M \times \phi_f = \begin{bmatrix} 4u_1 + 5u_2 + 6u_3 + 4u_4 + 5u_5 + 6u_6 \\ 2u_1 + 2u_2 + 7u_3 + 4u_4 + 4u_5 + 5u_6 \\ 3u_1 + 2u_2 + 2u_3 + 5u_4 + 6u_5 + 6u_6 \\ u_1 + 5u_2 + u_3 + 4u_4 + 2u_5 + 4u_6 \end{bmatrix}. \tag{17}$$

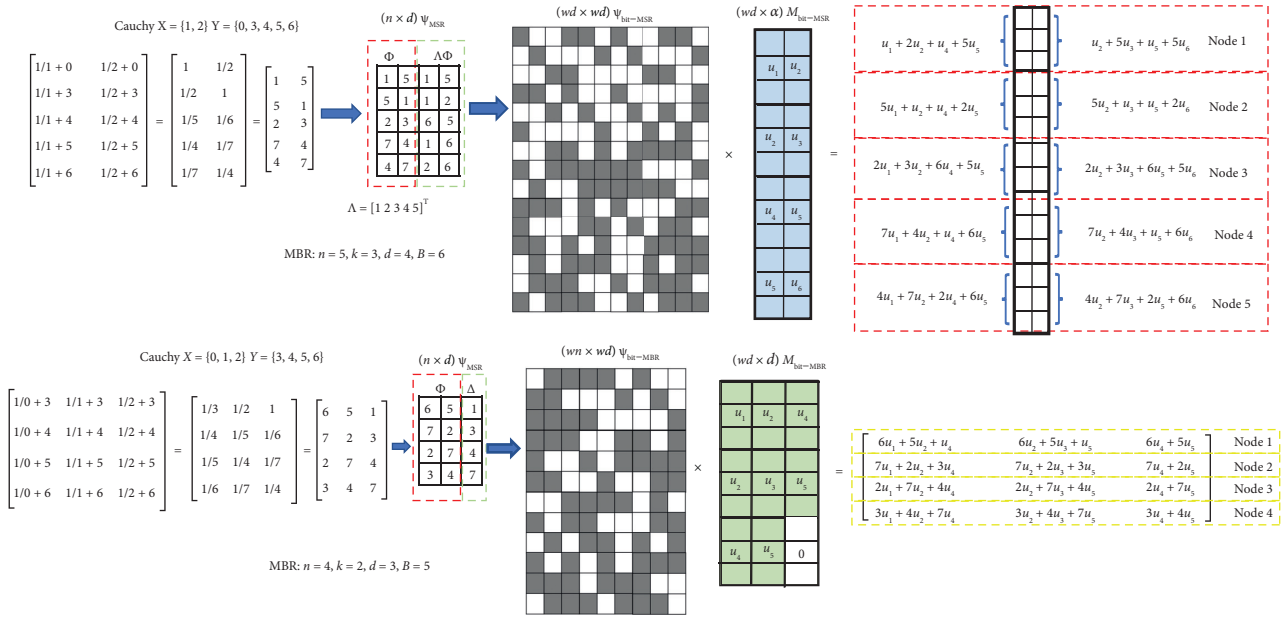


Figure 4: Using Bitmatrix transform $\Psi \times M$. Shown in the upper part of this figure is the transformation of PB-MSR codes for the parameters $n = 5, k = 3, d = 4$, and $B = 6$. The transformation of PB-MBR for the parameters $n = 4, k = 2, d = 3$, and $B = 5$ is shown in the bottom half of the illustration. All encoded blocks are stored on the corresponding nodes.

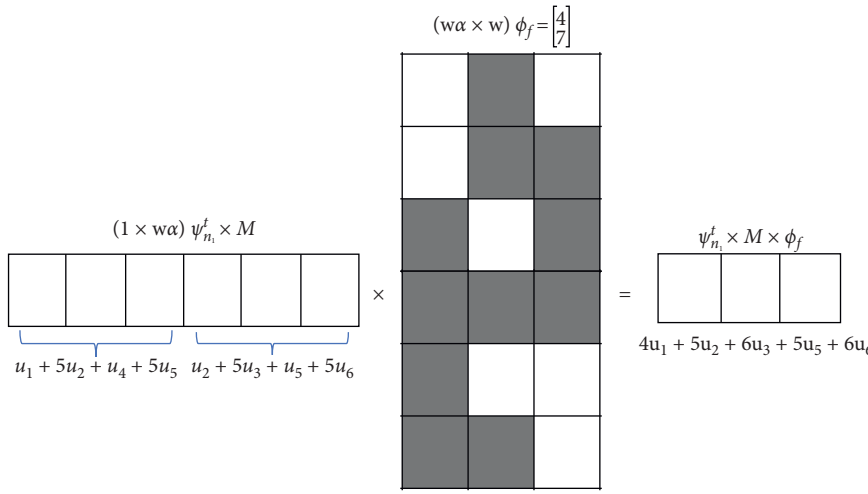


Figure 5: Product bitmatrix ϕ_f on node 1.

Then, r calculates $\Psi_{\text{repair}}^{-1}$ according to the number of helper node like this.

$$\Psi_{\text{repair}}^{-1} = \begin{bmatrix} 5 & 1 & 0 & 4 \\ 0 & 4 & 5 & 7 \\ 2 & 5 & 3 & 7 \\ 7 & 7 & 3 & 1 \end{bmatrix}. \tag{18}$$

Use $\Psi_{\text{repair}}^{-1} \times \Psi_{\text{repair}} \times M \times \phi_f$ to get vector $M\phi_f$.

$$M\phi_f = \begin{bmatrix} 4u_1 + 7u_2 \\ 4u_2 + 7u_3 \\ 4u_4 + 7u_5 \\ 4u_5 + 7u_6 \end{bmatrix}. \tag{19}$$

After the DC recalculates $\begin{bmatrix} S_1\phi_f & S_2\phi_f \end{bmatrix}$, the content of node 5 is recovered.

However, the method of random enumeration to find suitable Λ based on the third condition is not reasonable. Thus, we present a simple heuristic to create a better Λ.

Lemma 1. *In the finite field, any element a (except 0) has a unique multiplicative inverse element* a^{-1}*, so the formula* $aa^{-1} = (a^{-1})a = 1$ *holds.*

If $X = \{x_1, x_2\}$ and $Y = \{y_1, y_2, y_3, y_4\}$, where x_i and y_j are distinct elements over GF(2^w), then a Cauchy matrix is as Φ:

$$\Phi = \begin{bmatrix} \frac{1}{x_1 + y_1} & \frac{1}{x_2 + y_1} \\ \frac{1}{x_1 + y_2} & \frac{1}{x_2 + y_2} \\ \frac{1}{x_1 + y_3} & \frac{1}{x_2 + y_3} \\ \frac{1}{x_1 + y_4} & \frac{1}{x_2 + y_4} \end{bmatrix}. \tag{20}$$

Because every $(1/x_i + y_j)$ is distinct and their multiplicative inverse is also distinct according to Lemma 1, $(1/x_i + y_j)^{-1} = x_i + y_j$. Then, the square of the multiplicative inverse of each element in i-th row and j-th column of Φ is used to construct diagonal matrix Λ and Ψ as follows:

$$\Psi = \begin{bmatrix} \frac{1}{x_1 + y_1} & \frac{1}{x_2 + y_1} & x_1 + y_1 & \frac{(x_1 + y_1)^2}{x_2 + y_1} \\ \frac{1}{x_1 + y_2} & \frac{1}{x_2 + y_2} & x_1 + y_2 & \frac{(x_1 + y_2)^2}{x_2 + y_2} \\ \frac{1}{x_1 + y_3} & \frac{1}{x_2 + y_3} & x_1 + y_3 & \frac{(x_1 + y_3)^2}{x_2 + y_3} \\ \frac{1}{x_1 + y_4} & \frac{1}{x_2 + y_4} & x_1 + y_4 & \frac{(x_1 + y_4)^2}{x_2 + y_4} \end{bmatrix}. \tag{21}$$

Each column of Ψ is linearly independent. Nevertheless, for every operation based on the finite fields GF (2^w), when w and primitive polynomial change, the results of all operations change. Hence, matrix inversion is needed to determine whether the current structure is invertible. If the current Ψ has an irreversible submatrix, it changes each element of Λ by multiplying by the power to obtain a new completely different element. Since matrix inverses are encoded, decoded, and repaired only a few times and matrix products occur countless times, it is important to optimize the number of exclusive XORs for the matrix. The other details are covered in the next section.

4.3. New Product Bitmatrix Ψ *of MBR Codes.* For the lower part of Figure 4, combine $X = \{0, 1, 2\}$ and $Y = \{3, 4, 5, 6\}$ to construct $(wn \times w\,d)$ matrix Ψ of PB-MBR. Because $\Psi = [\,\Phi \;\; \Delta\,]$ where Φ and Δ are $(n \times k)$ and $(n \times (d - k))$, any d rows of Ψ are linearly independent and any k rows of Φ are linearly independent. As shown in the example, each node stores $(1 \times \alpha)$ vector, such as

$$\mathbf{c}_1 = [\,6u_1 + 5u_2 + u_4 6u_2 + 5u_3 + u_5 6u_4 + 5u_5\,]. \tag{22}$$

When decoding the original symbols, *DC* connects any two nodes and downloads $k\alpha = 6$ data, where

$$\mathbf{\Psi}_{\mathrm{DC}} \times M = \begin{bmatrix} 6u_1 + 5u_2 + u_4 & 6u_2 + 5u_3 + u_5 & 6u_4 + 5u_5 \\ 7u_1 + 2u_2 + 3u_4 & 7u_2 + 2u_3 + 3u_5 & 7u_4 + 2u_5 \end{bmatrix}. \tag{23}$$

When $\Psi_{\mathrm{DC}} M = [\,\Phi_{\mathrm{DC}} S + \Delta_{\mathrm{DC}} T^t \;\; \Phi_{\mathrm{DC}} T\,]$, T is recovered by multiplying Φ_{DC}^{-1} on the left.

$$\Phi_{\mathrm{DC}}^{-1} \times \mathbf{\Phi}_{\mathrm{DC}} \times T = \begin{bmatrix} 2 & 5 \\ 7 & 6 \end{bmatrix} \times \begin{bmatrix} 6u_4 + 5u_5 \\ 7u_4 + 2u_5 \end{bmatrix} = \begin{bmatrix} u_4 \\ u_5 \end{bmatrix}. \tag{24}$$

Subsequently, S is recovered.

For example, if node 4 fails, to accurately regenerate coded symbols on the r node, it requires connecting to any three helpers h_i(1,2,3 in the example). Each h_i inner product bitmatrix ψ_f, respectively, sends the result to the replacement node. r node obtains d vectors $\Psi_{\mathrm{repair}} M \psi_f$ and multiplies $\Psi_{\mathrm{repair}}^{-1}$ on the left where

$$\Psi_{\mathrm{repair}}^{-1} = \begin{bmatrix} 4 & 2 & 5 \\ 1 & 1 & 5 \\ 1 & 2 & 4 \end{bmatrix}. \tag{25}$$

Then, the DC recovers $(M\psi_f)^t = \psi_f^t M$ which represents the lost symbols on node 4.

5. Optimization of PB Framework Based on Minimizing the Number of Ones

An arbitrary Cauchy matrix with different X and Y brow the various number of ones, and the impact on performance is significantly different [27]. It is necessary to find a better bitmatrix Ψ which has fewer ones. Therefore, in this section, we optimize the two new Ψ by two new heuristic algorithms.

5.1. Optimizing the Number of Ones in Bitmatrix Ψ *of PB-MSR.* It is easy to know the number of ones of each element over GF (2^w). Figure 2 shows that 1–7 elements have several ones that are represented by an array as [3, 4, 7, 5, 4, 7, 6]. A bitmatrix with fewer ones can reduce the encoding computation, and element 1 has the least ones.

For j^{th} ($j \in [1, (n/2)]$) column of bitmatrix Ψ of PB-MSR,

(1) Count the number of ones.

(2) For each element divide by element $\Phi[i, j]$(i is row) and get a new column.

(3) Repeat the first two steps n times and obtain the best j^{th} column which has the minimum number of ones.

Then, repeat the first three steps for all columns and get a new matrix which has a minimal number of ones called Φ' as well as the invertibility property.

Next, it is necessary to find $\Lambda_{\min} \times \Phi'$ which has a minimal number of ones. Let $\lambda = \{\Lambda_1, \Lambda_2, \ldots, \Lambda_{d-k+1}\}$ denote the set of $(n \times n)$ diagonal matrices and let each Λ_j be n multiplicative inverses of j^{th} column of Φ'. For each Λ_j in λ, do the following:

(1) Construct $\Psi_j = \left[\Phi' \;\; \Lambda_j \times \Phi'\right]$.

(2) Determine whether the new Ψ is invertible. If it is, continue to the next step; if it is irreversible, $\Lambda_j = \Lambda_j \times \Lambda_j$.

(3) Count the number of ones of $\Lambda_j \times \Phi'$.

(4) Repeat the first three steps and whichever $\Lambda_{\min}$ gives the minimal number of ones, form Ψ by this $\Lambda_{\min} \times \Phi'$.

Algorithm 1 outlines the procedure to generate the set of λ and the process to find the best Λ_k matrix.

For example, as shown in Figure 6, Ψ_{MSR} has $22 + 22 + 22 + 24 = 90$ ones and XORs per row $o_{\text{before}} = 90/(4 * 3) - 1 = 6.5$, but optimized Ψ'_{MSR} has $18 + 18 + 18 + 18 = 72$ ones and XORs per row $o_{\text{after}} = 72/(4 * 3) - 1 = 5$, which is a 23% performance improvement.

5.2. Optimizing the Number of Ones in Bitmatrix Ψ of PB-MBR. For bitmatrix Ψ of PB-MBR, let $\lambda = \{\Lambda_1, \Lambda_2, \ldots, \Lambda_d\}$(d is the number of helper nodes) denote the set of $(n \times n)$ diagonal matrices and let each Λ_j be n multiplicative inverses of j^{th} column of Ψ. For each Λ_j, do the following:

(1) Use $\Lambda_j \times \Psi = \Psi'$.

(2) Count the number of ones of new Ψ'.

(3) Repeat the first two steps d times and obtain the best Ψ'.

After one column of Ψ' becomes element 1, Algorithm 2 does the same for the remaining d-1 columns, and each column needs to be optimized by dividing the value on this column so that the number of ones in the current column is minimized. The detailed process is written in Algorithm 2.

For example, as shown in Figure 7, original Ψ has $24 + 19 + 21 = 64$ ones and XORs per row $o_{\text{before}} = 64/(4 * 3) - 1 = 4.33$, but optimized Ψ'' only has $12 + 18 + 18 = 48$ ones and XORs per row $o_{\text{after}} = 48/12 - 1 = 3$, which is a 30.7% performance improvement.

6. Experiment

In this section, we have implemented the new PB-MSR and PB-MBR codes in C/C++ and employed the Jerasure 2.0 [28] with GF-complete [29] libraries for finite field arithmetic operations. All evaluations over GF (2^w) are about the reduction of XORs, encoding, decoding, and repair performances. All tests have been conducted on MacOS Catalina with 8 Intel Core i9 with 2.3 GHz clock speed and 16 GB 2667 MHz. In Section 6.1, we analyze the improved performances of the number of XORs based on new PB construction. In Sections 6.2–6.4, we report on experiments about encoding, decoding, and repair performances. The encoding performance = total encoded file size/encoding time and the decoding performance = total decoded file size/decoding time. Finally, the influence of finite field size about w is analyzed. All experimental results were average values after maximum and minimum values have been removed.

6.1. Analyzing New Product Bitmatrix. To better understand the effects of the new PB construct in speeding up the encoding computation, some evaluation experiments were designed on reducing the number of XORs, and the results are reported in Table 6. The baseline is based on the original Cauchy matrix without any additional optimization. Original PB-MSR and original PB-MBR have two subcolumns of data, which are the sum of XORs of Cauchy matrices formed by different combinations of X and Y. Since the original PB-MBR code is constructed by original Cauchy matrix, the two columns of before PB-MBR are the actual values of original Cauchy matrices. The latter two columns are reduction of PB-MSR and reduction of PB-MBR, respectively. Improvement is measured in terms of the decrease in XORs. In the last row of the table, the average improvement over all the tested parameters is included.

Since combinations of X and Y are different and the construction process of the PB-MSR code product matrix is also more complicated, each change will bring completely different results, so this experiment only observes the overall change trend. The size of the finite field is dependent on the size of computer bytes, which are 2^8, 2^{16}, and 2^{32}. And the choice of w needs to satisfy the range of $n + k \leq (w - 1)$, when $d = 2k$-2. It can be seen from Table 1 that PB-MSR provides a 11.73% reduction in the number of XORs and PB-MBR provides a 20.17% reduction in the number of XORs. Because the optimization of PB-MBR is the overall optimization of matrix Ψ, the degree of optimization is greater. Figure 8 shows the XOR change curve. The two line graphs are the original PB-MSR code and PB-MBR code, which are baselines. And the number of XORs optimized for the two codes is lower than the values of the line graphs.

6.2. Encoding Performance Evaluation. The RS and Cauchy codes use the Vandermonde and Cauchy matrices, which are the most classic experiment benchmarks for all erasure codes. Figure 9 shows the comparison results of the encoding performance of PB-MSR, PM-MSR, PB-MBR, PM-MBR, RS, and Cauchy codes when the total number of coded blocks n is fixed and r equals 2, 2.5, or 3. Cauchy-good is the optimized Cauchy-matrix encoding scheme provided by the Jerasure 2.0 library. Since encoding performance is directly related to the number of XORs, when the redundancy rate r is the maximum, the coding matrix of any coding scheme is the minimum. Therefore, when $r = 3$, all encoding performances are optimal. With a decrease of r, the multiplicative operations in finite fields increase gradually, and all coding rates decrease. It can be seen from the figure that as r becomes smaller, the PB Ψ becomes larger and more

(1) O_m is the number of ones of a matrix;
(2) O_{c_j} is the number of ones of the j^{th} column;
(3) Use Cauchy matrix construct Φ and count O_Φ;
(4) **for** each column $j = 1$ to $d - k + 1$ **do**
(5) **for** each row $i = 1$ to n **do**
(6) **for** each row $k = 1$ to n **do**
(7) Calculate $\Phi'[k, j] = \Phi[k, j]/\Phi[i, j]$;
(8) Count O_{c_j} of this new column;
(9) **end for**
(10) **end for**
(11) Choose the column with **min** O_{c_j} as new j^{th} column of Φ';
(12) **end for**
(13) Construct Φ' and each column has **min** O_{c_j};
(14) Count O'_Φ;
(15) **for** each column $j = 1$ to $d - k + 1$ **do**
(16) **for** each row $i = 1$ to n **do**
(17) $\Lambda[i, j] = 1/\Phi'[i, j]$;
(18) **for** each column $x = 1$ to $d - k + 1$ **do**
(19) $\Lambda[i, j] \times \Phi'[i, x]$;
(20) **end for**
(21) **end for**
(22) **if** $\Psi_j = [\ \Phi' \ \ \Lambda_j \times \Phi'\ ]$ is inverse **then**
(23) Count $O_{\Lambda_j \times \Phi'}$;
(24) **else**
(25) repeat 16–26 steps by multiplying the power of $\Lambda[i, j]$;
(26) **end if**
(27) **end for**
(28) Choose the best Ψ which has the **min** $O_{\Lambda_{\min} \times \Phi'}$;

Algorithm 1: Finding the optimized bitmatrix $\Psi = [\ \Phi' \ \ \Lambda\Phi'\ ]$ of **PB-MSR** (the algorithm first gets a new Φ' which has a minimal number of ones and then finds $\Lambda_{\min} \times \Phi'$ which has minimal number of ones).

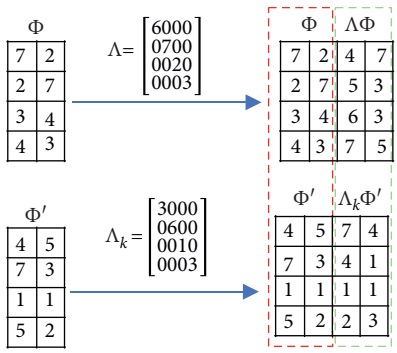


Figure 6: Comparison of original Ψ_{MSR} and optimized Ψ'_{MSR} that has a minimal number of ones.

optimization time is required. Therefore, the coding rate of the PB-MSR and PB-MBR codes decreases faster than RS or Cauchy. According to the analysis in the previous section, the different Cauchy matrices will generate different optimization results, so the rise of PB-MBR code in $r = 3$ or $r = 2.5$ does not have the same proportions. However, the PB-MSR encoding rate is not as fast as the PM-MSR using the Vandermonde matrix directly when the coded object is small. This is because it requires more time to generate the optimization bitmatrix. The encoding rate of the PB-MBR code is lower than that of PB-MSR because more redundancy is needed to be written to disk during the encoding process of the PB-MBR code. From Figure 9(b), it can be seen that with an increase of coding objects, the proportion of time to generate or optimize the matrix is relatively small, so the rate of all coding strategies increases.

6.3. Decoding Performance Evaluation. In the decoding/reconstruction experiments, the decoding performances for various values of r and fixed n for all coding schemes were tested. The decoding performance during reconstruction was measured in terms of the amount of data decoded per unit time. Figure 10 shows the comparison results. It can be observed that the decoding performances of all codes improve with increasing redundancy r. From Figure 10(a), we see that decoding performance of the PB-MSR code increased from 186.72 MB/s when $r = 2$ to 300.27 MB/s when $r = 3$. The PB-MBR code increased by 11.95% when $r = 3$ than when $r = 2$. From Figure 10(b), it can be seen that from $r = 2$ to $r = 3$, the decoding performance of the PB-MSR code increased by 48.26% and that of the PB-MBR code increased by 14.78%. Since the calculation complexity of PB-MSR and PB-MBR decoding is closely related to k, the smaller k, the lower the calculation complexity and the faster the decoding performances. Although the PB-MBR codes are similar to the PB-MSR codes, they are simpler and therefore have a higher decoding rate. As can be seen from Figure 10, PB-MSR

(1) O_m is the number of ones of a matrix
(2) O_{c_j} is the number of ones of the j^{th} column
(3) Use Cauchy matrix to construct Ψ and count O_Ψ
(4) $\lambda = \{\Lambda_1, \Lambda_2, \ldots, \Lambda_d\}$ as the set of $(n \times n)$ diagonal matrices and each matrix form from multiplcative inverses of Ψ
(5) **for** each $\Lambda_j \in \lambda$ **do**
(6) $\Psi\prime = \Lambda_j \times \Psi$
(7) Count $O_{\Psi'}$
(8) **end for**
(9) Choose the best Ψ' which has the **min** $O_{\Psi'}$
(10) The column of the matrix with all number 1 is x^{th}
(11) The set η includes all columns except x^{th} column
(12) **for** each column $j \in \eta$ **do**
(13) **for** each row $i = 1$ to n **do**
(14) **for** each row $k = 1$ to n **do**
(15) Calculate $\Psi''[k, j] = \Psi'[k, j]/\Psi'[i, j]$
(16) **end for**
(17) Count O_{c_j} of this new column
(18) **end for**
(19) Choose the column with **min** O_{c_j} as new j^{th} column of Ψ''
(20) **end for**
(21) Choose columns with the **min** O_{c_j} to construct Ψ''
(22) Count $O_{\Psi''}$

Algorithm 2: Finding the optimized bitmatrix Ψ of **PB-MBR** (the algorithm first denotes a set of λ and each $\Lambda_j \in \lambda$ and finds the best $\Psi' = \Lambda_j \Psi$ which has minimal number of ones).

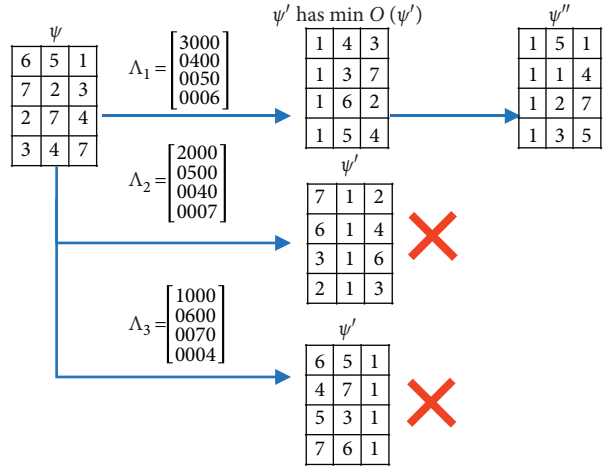


Figure 7: Comparison of original Ψ_{MBR} and optimized Ψ''_{MBR} that has a minimal number of ones.

Table 6: Performance improvement by PB construction.

(n,k,d,w)	Original PB-MSR		Original PB-MBR(Cauchy)		Reduction of PB-MSR		Reduction of PB-MBR	
(5, 3, 4, 8)	597	536	643	640	16.42%	5.78%	26.28%	33.44%
(5, 3, 4, 16)	1892	1948	2123	2296	19.24%	15.09%	23.60%	35.58%
(5, 3, 4, 32)	6551	7278	8043	8807	19.94%	10.51%	7.68%	29.24%
(10, 5, 8, 8)	2559	2543	2642	2620	11.92%	11.88%	16.28%	15.31%
(10, 5, 8, 16)	8487	9383	8916	9512	5.10%	14.78%	12.07%	14.27%
(10, 5, 8, 32)	35196	35816	40672	40166	14.36%	16.07%	22.84%	18.47%
(16, 8, 14, 8)	7292	7190	7342	7402	6.62%	6.43%	10.71%	9.70%
(16, 8, 14, 16)	26599	24591	26722	27324	5.34%	4.15%	14.68%	19.34%
(16, 8, 14, 32)	111559	103443	113806	116468	13.09%	14.44%	24.56%	28.96%
AVERAGE					11.73%		20.17%	

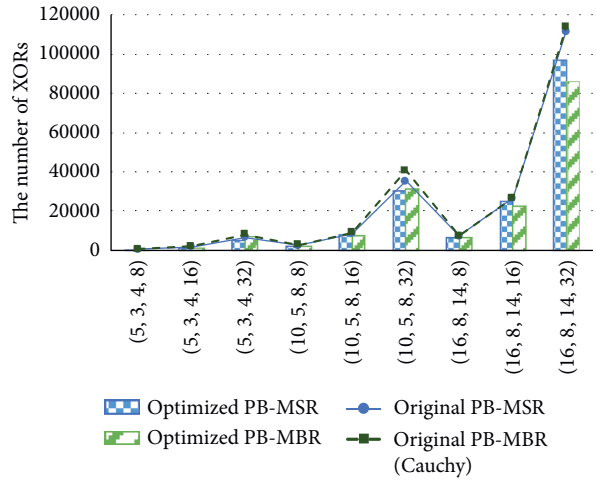


Figure 8: The number of XORs of PB-MSR and PB-MBR. The original PB-MSR and PB-MBR are the baselines represented by the line chart. The optimized PB-MSR and PB-MBR are represented by the bar graph.

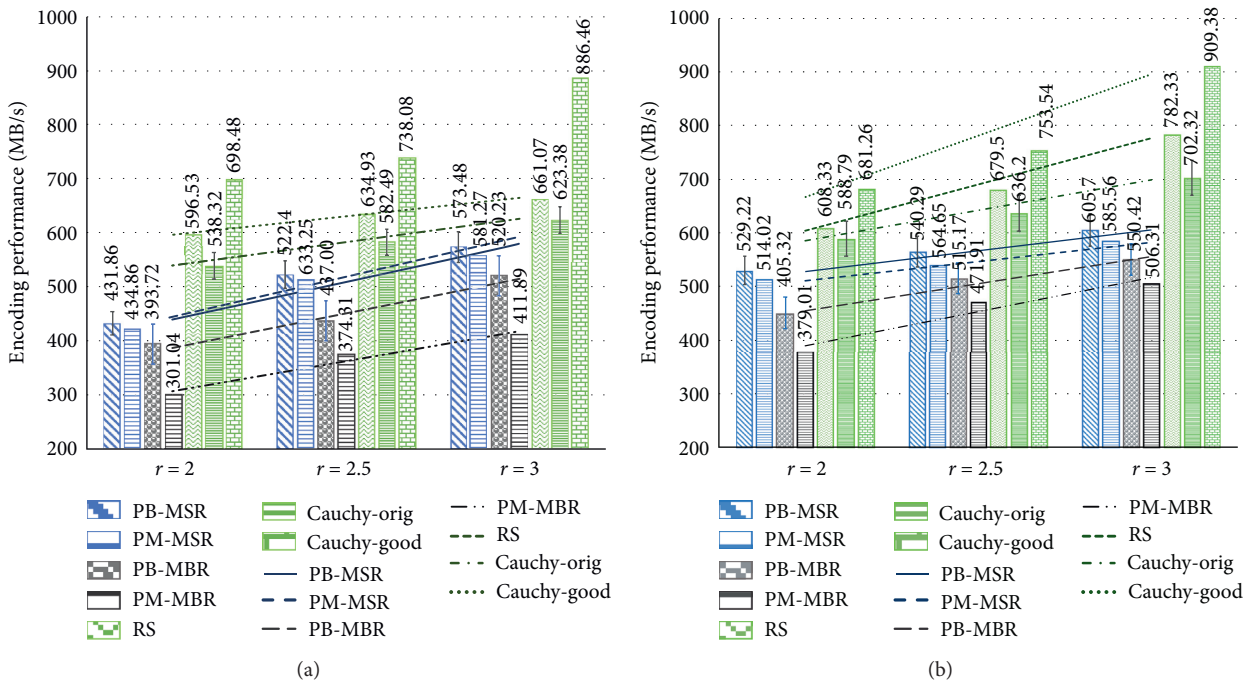


Figure 9: (a) The results of encoding performance under 5 MB. (b) The results of encoding performance under 200 MB.

decoding performance is about 17.35% higher than PM-MSR, and PB-MBR is about 15.29% higher than PM-MBR.

6.4. Repair Performance Evaluation. Figure 11 shows the data transfers across the network to repair one failed node. It can be seen that both the PB-MSR and PB-MBR codes have significantly lower data transfer compared to RS codes. As can be seen from the bar chart in Figure 11(a), when the encoding object is 5 MB, the classical RS code needs 1.5 times overhead of data transfer compared to the PB-MSR code and needs 2.25 times overhead of data transfer compared to the PB-MBR code. It also can be observed from Figure 11(a) that when the redundancy ratio r is fixed, changing the number of d will result in a smaller amount of data transferred across the network. When $d = n - 1$ in Figure 11(a), the data transferred overhead of PB-MSR code and PB-MBR code decreased by 35.74% and 16.22% compared to that of $d = 2 * k - 2$.

Additionally, the repair performances were tested as shown in Figure 12. From the overall trend, while the value of n is fixed, as the redundancy rate increases, k gradually decreases, and the unit storage capacity of each node increases; disk I/O overhead increases, and the overall repair rate decreases. In contrast to Figure 11, although PB-MSR has a larger network transmission overhead than PB-MBR, it

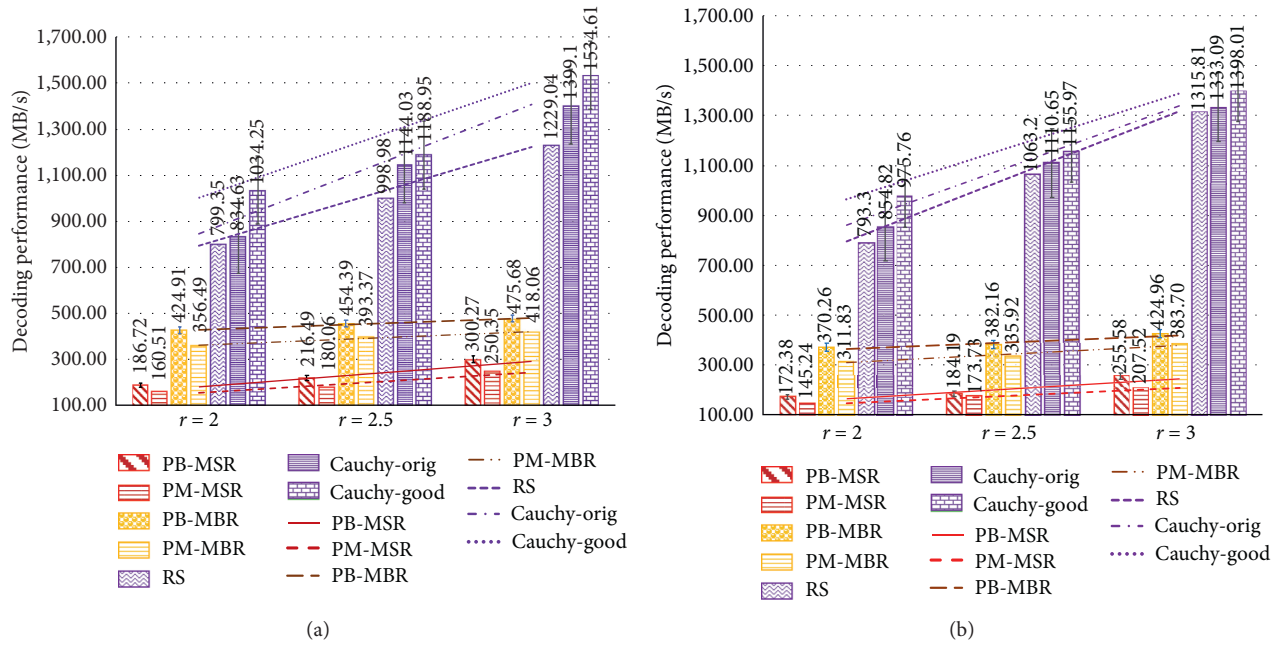


Figure 10: (a) The results of decoding performance under 5 MB. (b) The results of decoding performance under 200 MB.

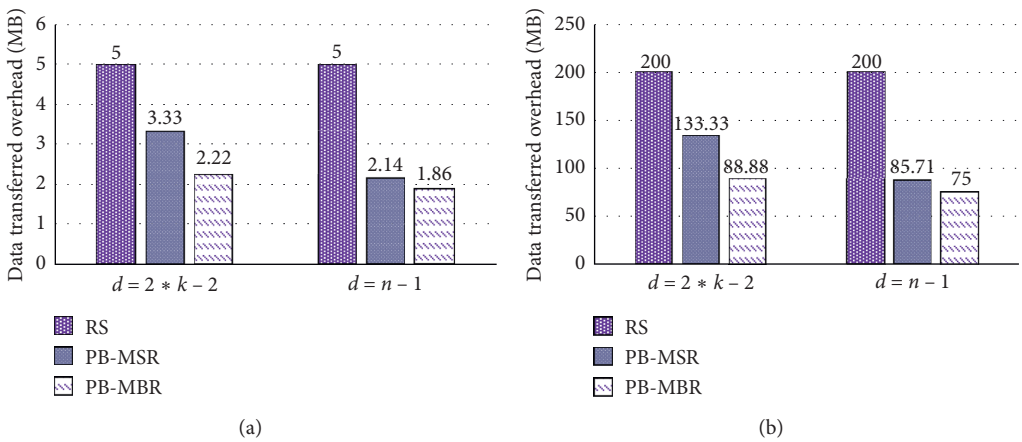


Figure 11: An example of total amount of data transferred across the network while repairing one failed node. (a) Data transferred overhead under 5 MB. (b) Data transferred overhead under 200 MB.

is always the smallest in-unit storage capacity α. Thus, the data that need to be written to the disk is also the smallest. Besides, besides, based on the calculation and analysis results about repair complexity in Table 5, it can be seen that PB-MSR has lower calculation complexity than PB-MBR, so its repair rate is slightly higher.

6.5. The Effect of Finite Field GF (2^w) on Coding Performance. Based on the analysis thus far, we know that the complexity of the encoding, decoding, or repair process is related to the finite field size w, which means every symbol $x \in [0, 2^w)$. Therefore, the experiment in this section analyzes the influence of the finite field size on the performance of ECs. The values of (n, k, d) were fixed, and the performances of $w = 8$, $w = 16$, and $w = 32$ were tested as shown in Figure 13. Figures 13(a), 13(b), and 13(c) are the comparison diagrams of encoding, decoding, and repair performances, respectively. From these figures, we see that as the finite field size w gradually increases, the calculation complexity increases and the rate declines in varying degrees.

7. Related Works

Regenerating codes are state-of-the-art ECs with optimal repair bandwidth that have been used by some storage systems. The NCCloud storage system was one of the earliest implementable designs for 2-parity functional MSR codes (F-MSR), which maintain the same data redundancy level and same storage requirement as traditional ECs but use less repair traffic [30]. In [31], Pamies-Juarez et al. have presented the evaluation of a novel MSR code known as the

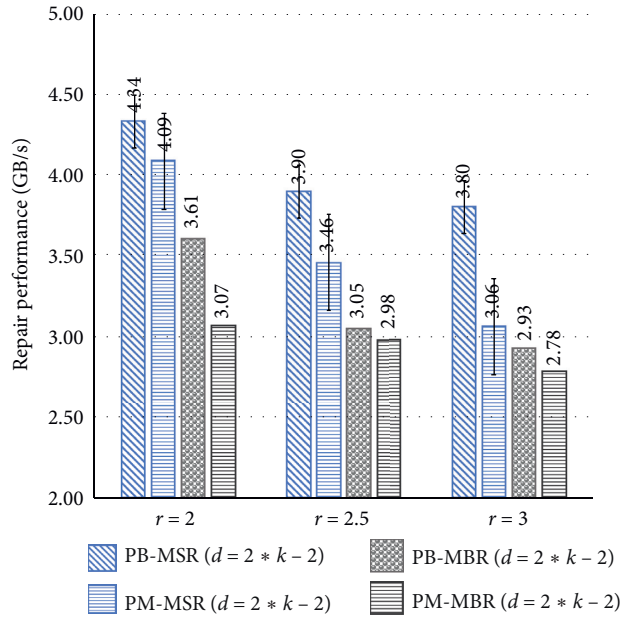


Figure 12: The result of repair performance.

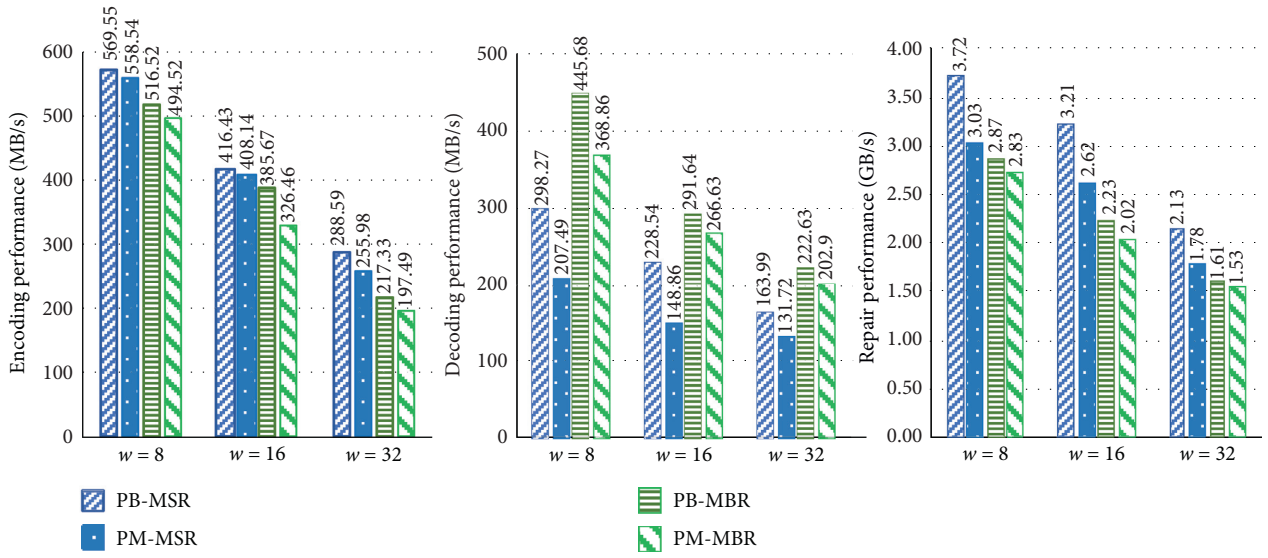


Figure 13: Comparison of performance under different w.

Butterfly codes in both Ceph and HDFS. Their analysis shows that Butterfly codes are capable of reducing network traffic and reading I/O access during repairs. In [20], the Coupled-Layer (Clay) code is an MSR code that offers a simplified construction for decoding/repair by using pairwise coupling across multiple stacked layers of any single MDS code and has been evaluated over an Amazon AWS cluster. The Clay code is simultaneously optimal in terms of storage overhead, repair bandwidth, optimal access, and sub-packetization level. In [22], Rashmi et al. wanted to minimize disk I/O consumed, while simultaneously retaining optimality in terms of both storage, reliability, and network bandwidth. They presented an algorithm to transform PM-MSR codes into I/O optimal codes (which they refer to as the PM-RBT codes). In addition to MSR codes, there was some new literature about MBR codes. In [32], Hu et al. presented NCFS which was a distributed file system under real network setting based on E-MBR codes [33]. In [34], Shah et al. presented the $(n = d + 1,\ k,\ d)$ exact-repair- (ER-) MBR codes without arithmetic operation in the repair process. In [35], the authors introduced a family of repair-by-transfer (RBT) codes which was a class of $(n,\ k,\ d = n\text{-}1)$ ER-MBR codes that were constructed based on congruent transformations applied to a skew-symmetric matrix of message symbols. In this construction, the encoding complexity decreases from

n^4 to n^3. Based on the new coding matrix for PM-MBR, the minimum of a finite field is reduced from $n\text{-}k+d$ to n. In [18], the authors first introduced a more natural extension to the classical PM-MBR framework, modified to provide flexibility in the choice of the number of helpers during node repair, permitting a certain number of error-prone nodes during repair. This was achieved by proving the nonsingularity of the family of matrices in large enough finite fields. To reduce the high coding and repair complexity that involves expensive multiplication operations in a finite field, another new class of RGC was proposed. In [36–38], the authors proposed a new framework of linear codes with binary parity-check code, named Binary Addition and Shift Implementable Cyclic-convolutional (BASIC) codes which enable coding and repair by XOR and bitwise cyclic shift.

Although some systems are currently using RGC, they are designed or optimized for coding structure, bandwidth overhead, storage overhead, and I/O overhead. However, there is still a lack of some classical coding framework complexity analysis from the perspective of the finite field.

ECs rely on finite field operations which can be performed using XOR operations. Many acceleration techniques have been proposed: optimizing bitmatrix design, optimizing computation schedule, common XOR operation reduction, cache management, and vectorization techniques [12]. In [13], two new heuristics have been derived called Uber-CHRS and X-Sets to schedule encoding and decoding bitmatrix by reducing the number of XORs. And several technologies were introduced in the same work by using different heuristic algorithms. In addition to smart scheduling and matching algorithms for reducing the number of XORs, other solutions were based on improving hardware performance. In [14], Plank et al. vectorized finite field operations directly based on single-instruction-multiple-data (SIMD).

8. Conclusion

In this paper, we have combined the finite field arithmetic operations to elaborate on the computational complexity of the famous PM framework in encoding, decoding, and repair processes and proposed a new construction called product bitmatrix (PB). Based on two heuristic algorithms, PB-MSR and PB-MBR codes find local minimum XORs that can improve encoding, decoding, and repair performances. Although PB has been optimized by the original framework in computation and has advantages in the decoding and repair processes, when the encoding object is small, the optimization time occupies a portion of the total encoding time. Compared with the PM adopted by Vandermonde, its advantage is insufficient.

In future research, in addition to theoretical research, we will conduct in-depth research on the fault-tolerant technology of edge computing nodes in combination with the real environment, for instance, improving network transmission performance [39], adapting the recommended data reliability storage algorithm of edge computing [40, 41], or to design new nonlinear coding methods with deep neural networks [42, 43].

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was supported by the National Key Research and Development Project (2019YFB2102600) and the National Natural Science Foundation of China (NSFC) (61572194 and 61672233).

References

[1] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, pp. 30–39, 2017.

[2] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[3] S. Deng, Z. Xiang, P. Zhao et al., "Dynamical resource allocation in edge for trustable internet-of-things systems: a reinforcement learning method," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 9, pp. 6103–6113, 2020.

[4] K. Dolui and S. Datta, "Comparison of edge computing implementations: fog computing, cloudlet and mobile edge computing," in *Proceedings of the 2017 Global Internet of Things Summit (GIoTS)*, IEEE, Geneva, Switzerland, 2017.

[5] H. Gao, W. Huang, and Y. Duan, "The cloud-edge based dynamic reconfiguration to service workflow for mobile ecommerce environments: a qos prediction perspective," in *Proceedings of the ACM Transactions on Internet Technology*, vol. 21, 23 pages, December 2020.

[6] W. Yu, L. Fan, X. He et al., "A survey on the edge computing for the internet of things," *IEEE Access*, vol. 6, pp. 6900–6919, 2017.

[7] S. B. Balaji, M. N. Krishnan, M. Vajha, V. Ramkumar, B. Sasidharan, and P. V. Kumar, "Erasure coding for distributed storage: an overview," *Science China Information Sciences*, vol. 61, no. 10, Article ID 100301, 2018.

[8] D. Vorick and L. Champine, *Sia: Simple Decentralized Storage*, Nebulous Inc, Boston, MA, USA, 2014.

[9] S. Wilkinson, T. Boshevski, J. Brandoff, and V. Buterin, *Storj a Peer-To-Peer Cloud Storage Network*, 2014.

[10] L. Liang, H. He, J. Zhao, C. Liu, Q. Luo, and X. Chu, *An Erasure-Coded Storage System for Edge Computing*, IEEE Access, New york, NY, USA, 2020.

[11] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of The Society for Industrial and Applied Mathematics*, vol. 8, no. 2, pp. 300–304, 1960.

[12] T. Zhou and C. Tian, "Fast erasure coding for data storage," *ACM Transactions on Storage*, vol. 16, no. 1, pp. 1–24, 2020.

[13] S. P. James, D. Catherine, and B. D. Robison, "Heuristics for optimizing matrix-based erasure codes for fault-tolerant storage systems," in *Proceedings of the IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2012)*, IEEE, Edinburgh, UK, 2012.

[14] S. P. James, K. M. Greenan, and E. L. Miller, "Screaming fast galois field arithmetic using intel simd instructions," in *Proceedings of the 11th Conference on File and Storage Technologies*, pp. 299–306, San Jose, CA, USA, 2013.

[15] V. A. Vins, S. Umamageswari, and P. Saranya, "A survey on regenerating codes," *International Journal of Scientific and Research Publication*, vol. 24, 2014.

[16] K. V. Rashmi, B. Nihar, D. Gu, H. Kuang, D. Borthakur, and R. Kannan, "A solution to the network challenges of data recovery in erasure-coded distributed storage systems: a study on the facebook warehouse cluster," in *Proceedings of the 5th Workshop on Hot Topics in Storage and File Systems*, vol. 13, San Jose, CA, USA, 2013.

[17] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Transactions on Information Theory*, vol. 56, no. 9, pp. 4539–4551, 2010.

[18] K. Mahdaviani, A. Khisti, and S. Mohajer, "Bandwidth adaptive & error resilient mbr exact repair regenerating codes," *IEEE Transactions on Information Theory*, vol. 65, no. 5, pp. 2736–2759, 2018.

[19] M. Ye and B. Alexander, "Explicit constructions of high-rate mds array codes with optimal repair bandwidth," *IEEE Transactions on Information Theory*, vol. 63, no. 4, pp. 2001–2014, 2017.

[20] M. Vajha, V. Ramkumar, B. Puranik et al., "Clay codes: moulding {*M DS*}codes to yield an c{*MSR*}ode," in *Proceedings of the 16th Conference on File and Storage Technologies*, pp. 139–154, Santa Clara, CA, USA, 2018.

[21] K. V. Rashmi, B. Nihar, and P. V. Kumar, "Optimal exact-regenerating codes for distributed storage at the msr and mbr points via a product-matrix construction," *IEEE Transactions on Information Theory*, vol. 57, no. 8, pp. 5227–5239, 2011.

[22] K. V. Rashmi, P. Nakkiran, J. Wang, B. Nihar, and R. Kannan, "Having your cake and eating it too: jointly optimal erasure codes for i/o, storage, and network-bandwidth," in *Proceedings of the 13th Conference on File and Storage Technologies*, vol. 15, pp. 81–94, Santa Clara, CA, USA, 2015.

[23] K. Mahdaviani, S. Mohajer, and A. Khisti, "Product matrix msr codes with bandwidth adaptive exact repair," *IEEE Transactions on Information Theory*, vol. 64, no. 4, pp. 3121–3135, 2018.

[24] K. Mahdaviani, S. Mohajer, and A. Khisti, "Product matrix minimum storage regenerating codes with flexible number of helpers," in *Proceedings of the 2017 IEEE Information Theory Workshop (ITW)*, IEEE, Kaohsiung, Taiwan, 2017.

[25] H. Cohen, G. Frey, R. Avanzi et al., *Handbook of Elliptic and Hyperelliptic Curve Cryptography*, CRC Press, Boca Raton, FL, USA, 2005.

[26] J. Bloemer, K. Malik, R. Karp, M. Karpinski, M. Luby, and D. Zuckerman, *An Xor-based erasure-resilient coding scheme*, ICSI Technical report, 1995.

[27] S. P. James and L. Xu, "Optimizing cauchy reed-solomon codes for fault-tolerant network storage applications," in *Proceedings of the Fifth IEEE International Symposium on Network Computing and Applications (NCA'06)*, pp. 173–180, IEEE, Cambridge, MA, USA, 2006.

[28] J. S. Plank and K. Jerasure, "A library in C facilitating erasure coding for storage applications–version 2.0," Technical Report UT-EECS-14-721, University of Tennessee, Knoxville, TN, USA, 2014.

[29] J. S. Plank, K. M. Greenan, E. L. Miller et al., "GF-Complete: A comprehensive open source library for Galois Field arithmetic," *Technical report UT-CS-13*, University of Tennessee, Knoxville, TN, USA, 2013.

[30] Y. Hu, H. C. H. Chen, P. P. C. Lee, and Y. Tang, "Nccloud: applying network coding for the storage repair in a cloud-of-clouds," *FAST*, vol. 21, 2012.

[31] L. Pamies-Juarez, R. Mateescu, C. Gyuot, E. Gad, and Z. Bandic, "Opening the chrysalis: on the real repair performance of codes," in *Proceedings of the 14th Conference on File and Storage Technologies*, vol. 16, pp. 81–94, Santa Clara, CA, USA, 2016.

[32] Y. Hu, Y. Chiu-Man, K. Yan, P. P. C. Lee, and J. C. S. Lui, "Ncfs: on the practicality and extensibility of a network-coding-based distributed file system," in *Proceedings of the 2011 International Symposium on Networking Coding*, pp. 1–6, IEEE, Beijing, China, 2011.

[33] K. V. Rashmi and B. Nihar Shah, P. V. Kumar and R. Kannan, Explicit construction of optimal exact regenerating codes for distributed storage," in *Proceedings of the 2009 47th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 1243–1249, IEEE, Monticello, IL, USA, 2009.

[34] B. Nihar, K. V. Shah, P. V. Kumar, and R. Kannan, "Distributed storage codes with repair-by-transfer and non-achievability of interior points on the storage-bandwidth tradeoff," *IEEE Transactions on Information Theory*, vol. 58, no. 3, pp. 1837–1852, 2011.

[35] S.-J. Lin and W.-H. Chung, "Novel repair-by-transfer codes and systematic exact-mbr codes with lower complexities and smaller field sizes," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 12, pp. 3232–3241, 2014.

[36] H. Hou, K. W. Shum, M. Chen, and H. Li, "Basic regenerating code: binary addition and shift for exact repair," in *Proceedings of the 2013 IEEE International Symposium on Information Theory*, pp. 1621–1625, IEEE, Istanbul, Turkey, 2013.

[37] H. Hou, K. W. Shum, and H. Li, "Construction of exact-basic codes for distributed storage systems at the msr point," in *Proceedings of the 2013 IEEE International Conference on Big Data*, pp. 33–38, IEEE, 2013.

[38] H. Hou, K. W. Shum, M. Chen, and H. Li, "Basic codes: low-complexity regenerating codes for distributed storage systems," *IEEE Transactions on Information Theory*, vol. 62, no. 6, pp. 3053–3069, 2016.

[39] H. Gao, C. Liu, Y. Li, and X. Yang, "V2VR:reliable hybrid-network-oriented v2v data transmission and routing considering rsus and connectivity probability," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[40] H. Gao, Li Kuang, Y. Yin, B. Guo, and K. Dou, "Mining consuming behaviors with temporal evolution for personalized recommendation in mobile marketing apps," in *Proceedings of the ACM/Springer Mobile Netw. Appl.(MONET)*, vol. 25, pp. 1233–1248, 2020.

[41] X. Yang, S. Zhou, and M. Cao, "An approach to alleviate the sparsity problem of hybrid collaborative filtering based recommendations: the product-attribute perspective from user reviews," *Mobile Networks and Applications*, vol. 2, pp. 1–15, 2019.

[42] Y. Yin, Z. Cao, Y. Xu, H. Gao, R. Li, and Z. Mai, "Qos prediction for service recommendation with features learning in mobile edge computing environment," *IEEE Transactions on Cognitive Communications and Networking*, 2020.

[43] K. Jack, K. V. Rashmi, and S. Venkataraman, "Learning a code: machine learning for approximate non-linear coded computation," 2018, http://arxiv.org/abs/1806.01259.

Hindawi
Security and Communication Networks
Volume 2021, Article ID 8873195, 13 pages
https://doi.org/10.1155/2021/8873195

Research Article

Deep Learning-Based Security Behaviour Analysis in IoT Environments: A Survey

Yawei Yue,[1,2] Shancang Li,[2] Phil Legg,[2] and Fuzhong Li[1]

[1]*School of Software, Shanxi Agricultural University, Jinzhong 030801, Shanxi, China*
[2]*Department of Computer Science and Creative Technologies, UWE Bristol, BS16 1QY, UK*

Correspondence should be addressed to Shancang Li; shancang.li@uwe.ac.uk

Received 30 August 2020; Revised 22 November 2020; Accepted 15 December 2020; Published 8 January 2021

Academic Editor: Honghao Gao

Internet of Things (IoT) applications have been used in a wide variety of domains ranging from smart home, healthcare, smart energy, and Industrial 4.0. While IoT brings a number of benefits including convenience and efficiency, it also introduces a number of emerging threats. The number of IoT devices that may be connected, along with the ad hoc nature of such systems, often exacerbates the situation. Security and privacy have emerged as significant challenges for managing IoT. Recent work has demonstrated that deep learning algorithms are very efficient for conducting security analysis of IoT systems and have many advantages compared with the other methods. This paper aims to provide a thorough survey related to deep learning applications in IoT for security and privacy concerns. Our primary focus is on deep learning enhanced IoT security. First, from the view of system architecture and the methodologies used, we investigate applications of deep learning in IoT security. Second, from the security perspective of IoT systems, we analyse the suitability of deep learning to improve security. Finally, we evaluate the performance of deep learning in IoT system security.

1. Introduction

The advancement of network theory and architecture in line with the development of sensors and microprocessors paved the way for the Internet of Things, and applications such as smart homes and smart cities are now becoming widely adopted. According to Gartner, 5.8 billion endpoints will be deployed in 2020, a 21% increase from 2019 [1]. The market for IoT was valued at $190 billion in 2018 and is projected to reach $1102.6 billion by 2026, exhibiting a compound annual growth rate (CAGR) of 24.7% in the forecast period [2]. Banking and financial services have the greatest market share, followed by information technology and telecommunications. Healthcare and government applications also account for a large proportion of the total IoT market. The explosive growth of IoT offers the potential for billions of devices to be connected and exchanging data for various applications. The unique characteristics that IoT offers have also brought a series of new security and privacy threats, of which are a major concern for sustainable growth of IoT adoption.

Often, IoT devices are reported to have vulnerabilities due to their limited resources which can make them an attractive target for attack. With billions of devices interconnected, many and other connected devices launched a targeted attack at the domain name provider Dyn [3], causing a denial of service (DoS) attack against many popular websites such as GitHub, Twitter, and others. Many of the devices used for this attack by the Mirai botnet were using default usernames and passwords. Connected autonomous vehicles (CAVs) are a unique form of IoT, yet attacks have been demonstrated to show how an Internet-enabled vehicle could be controlled remotely through a vulnerability in the media control system that could cause serious physical harm [4]. To be efficient and lightweight to deploy, many IoT applications run on embedded CPUs with limited memory and battery capacity. Many IoT system designs highlight the limitation in computing efficiency as a potential attack vector for security and privacy concerns. IoT devices are widely used as core controllers in critical infrastructures, and they convey valuable information. Stuxnet

[5] is a well-documented malicious computer worm that targeted a specific industrial control system (Uranium Enrichment Plant), which suspended the progress of nuclear weapons program of Iran. IoT technologies play a crucial role in enhancing real-life applications, such as healthcare, smart home, and surveillance.

Given the complexity of developing IoT systems integration, this can potentially provide a wide attack surface for an adversary. Like the Mirai botnet, devices that have weak authentication requirements can be easily compromised and controlled as part of an attack; as the number of connected devices increases, this attack surface continues to grow.

In this paper, we study how deep learning can be used to enhance security and privacy in the IoT era. Firstly, we review security and privacy concerns in IoT systems. We then survey deep learning-based IoT security and privacy applications and develop a taxonomy to consider these works from the viewpoint of deep learning algorithms used and the IoT security problems that they solve. Finally, we present the future research trends and challenges that we have identified. The main contributions of this paper are summarised as follows:

(1) We summarize and provide a taxonomy of recent work using deep learning to enhance the security and privacy property of IoT system and how deep learning can help to build a secure IoT system

(2) We identify the weaknesses that still exist in current research and the discrepancies between these weaknesses and the requirements of the IoT setting

(3) We investigate the possible future research directions toward deep learning enhanced IoT security

2. Background

Minerva et al. introduced the architecture of an IoT system and highlighted the set of features that a system must possess in order to be considered as an IoT system [6]. The main features include the following:

(1) *Interconnection of Things*. Here, the "Thing" means smart object that can collect, create, process, and store data from a user or application perspective.

(2) *Connectivity*. The IoT provides Internet connectivity for objects in the system, including devices, applications, and key IoT infrastructures.

(3) *Uniquely Identifiable Things*. IoT devices should be uniquely identifiable.

(4) *Ubiquity*. The IoT system can provide services that are available for users anywhere and at any time.

(5) *Sensing/Actuation Capability*. As the key component senses the environment, a smart sensor can collect data from environment and transmit this to the IoT systems. An actuator can conduct specific operations depending on the commands received from the IoT system.

(6) *Embedded Intelligence*. Advances in artificial intelligence are to be embedded into edge IoT systems.

(7) *Interoperable Communication Capability*. An IoT system should be able to communicate using standard and interoperable communication protocols.

(8) *Self-Configurability*. Due to the fact that there are a large number of heterogeneously connected devices in an IoT system, it is natural that IoT devices may need to manage and configure themselves, which could range from software and hardware management to resource allocation.

(9) *Programmability or Software Defined*. Physical devices in IoT systems can be easily customized with a user's command or software defined functions without physical changes.

In our previous works, we defined a service-oriented architecture (SoA) for the general IoT [7], as shown in Figure 1. This paper extends previous works by detailing the sensing layer, network layer, service layer, and interface layer. The sensing layer is integrated with available hardware objects to sense the statuses of things. The network layer is the infrastructure to support wireless or wired connections among things. The core of this architecture is the service layer, which consists of service discovery, service composition, service management, and service interfaces. The service layer allows developers to meet the request of end users with minimal workload. The interface layer consists of the interaction methods with users or applications. We adopt this architecture for the remainder of the paper.

3. Behaviour Modelling and Analysis of IoT Using Deep Learning

Deep learning (DL) is considered to be the founding pillar of modern artificial intelligence [8]. DL has been widely used in computer vision, speech recognition, robotics, and many other application areas. Compared with traditional machine learning techniques, deep learning has some key advantages. (1) The use of many hidden layers within a neural network structure means that deep learning can fit complex nonlinear relationships between attributes. (2) Popular architectures such as convolution neural networks (CNNs) and long short-term memory (LSTM) networks have the ability to extract and identify useful features directly from raw data (e.g., autoencoders) instead of relying on hand-crafted statistical features as performed in traditional machine learning. (3) Deep learning is particularly well suited for dealing with 'big data' challenges [9].

With billions of devices interconnected together to sense and share information worldwide, IoT systems naturally produce a huge volume of data. Deep learning has significant potential to help analyse user (events, apps) behaviours in complicated IoT systems. Furthermore, deep learning could enable IoT devices to learn complex behaviour patterns more effectively than traditional learning techniques.

The IoT is a complete ecosystem that contains a variety of devices and connections, a tremendous number of users, and a huge volume of data. To identify the potential vulnerabilities that exist within an IoT system, it is necessary to look at the whole IoT ecosystem and the behaviours exhibited

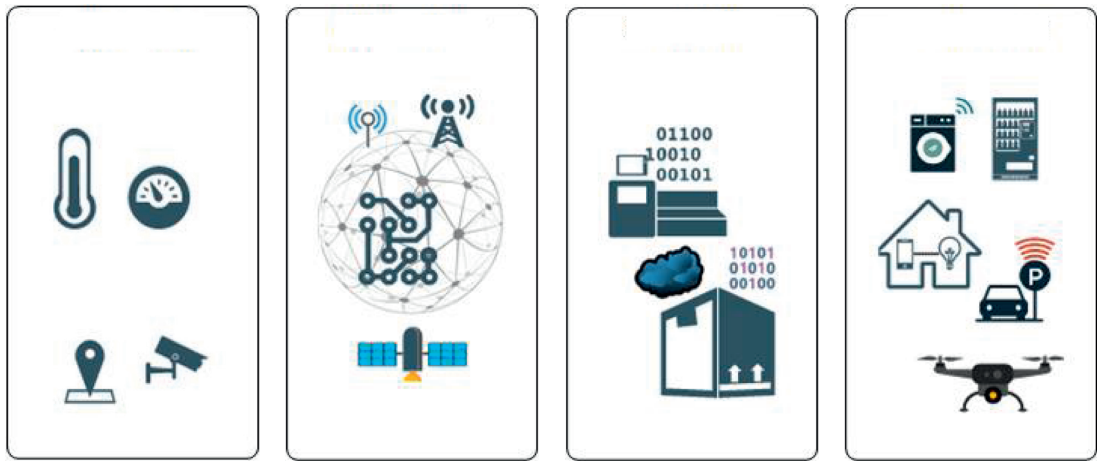


Figure 1: Service-oriented architecture (SOA) of the IoT system: (a) sensor layer; (b) network layer; (c) service layer; (d) interface layer.

rather than to focus on the individual device or layers. In this work, we focus on the following three problems: (1) to identify the uniqueness of each IoT device by classifying, training, and extracting device fingerprint of an IoT device; (2) to investigate the network behaviours in IoT; and (3) to model data abuse in IoT environment.

3.1. Identify the Uniqueness of IoT Devices Using DL. Each device in an IoT system will often have some fixed features, such as physical characteristics or services that it provides. Based on such features, we can profile a device to uniquely identify it from other IoT devices in the same system.

For example, an IoT digital camera could be used to take photographs and record audio/video and could even link with social networking data sources if permitted access. The CCD sensor in the digital camera has a unique sensor pattern noise (SPN) which could be used to create a unique fingerprint of the device. Such fingerprints for IoT devices could also be identified based on the device users, which can be further analysed using techniques such as deep learning. Having a means to fingerprint an IoT device and specifically the data generated by the IoT device (rather than merely relying on serial numbers, IMEI numbers, and so on) would be particularly beneficial should there be a need to identify malicious usage of devices in a complex interconnected IoT system. Likewise, this notion of fingerprinting can also be used as part of authentication and trust between connected devices:

(1) *Device Identification Using DL.* Traditional methods of device identification may use serial numbers, IMEI codes, or other static identifiers; however, these can potentially be spoofed or manipulated by an attacker. Deep learning has the potential to identify subtle differences between classes when considering a large feature set to characterise data and therefore could be effective for device identification as discussed previously. Deep learning methods can extract features from the signal or traffic produced by the device in order to recognise and identify the device.

An example of this is camera model identification, where the objective is to determine the device that captured the image provided. Work in [10] proposes the method of using deep CNNs to automatically extract features to identify the capture device. They calculate residual noise in the image by subtracting a denoised version of the image from the image provided. The residual noise is then used as input to the CNN model to extract and identify distinct features from various device types. Work in [11] uses a CNN to extract model-related features and then uses a support vector machine (SVM) to predict the camera model. In both of these cases, the role of deep learning is primary for the feature extractor. Similar examples have also been applied for audio device identification [12].

Radio fingerprinting has also been studied where devices are identified by their wireless radio device properties. In [13], Yu et al. propose a solution using partially stacking-based convolutional DAE to classify devices through reconstructing a high-SNR signal. Based on RF fingerprinting techniques, Bassey et al. proposed a framework to detect unverified smart devices with deep learning [14]. First, they use a convolutional neural network to automatically extract high-level features from RF traces; then, they perform dimensionality reduction and decorrelation on deep features. Finally, they use clustering techniques to classify IoT devices.

(2) *Service Fingerprint Extraction Using DL.* Due to the dynamic nature of IoT networks, it can be difficult to maintain static fingerprints for devices as they are connected or removed from the network. Therefore, establishing a dynamic behaviour baseline is essential. Fingerprinting IoT devices can also be a challenge due to the heterogeneous nature of IoT devices, protocols, and command interfaces. Service

fingerprints identify IoT devices based on the services that they provide, which then generates a profile that can be used to identify the type of device that it is likely to be. Typically, this would be achieved using system logs and web traffic as inputs to extract behavioural fingerprints.

Previously, researchers have employed machine learning to address challenges in IoT [15–17]. Meidan et al. propose an IoT device classification framework based on HTTP packet analysis [15]. They perform this as a two-pass classification to firstly distinguish between IoT devices and non-IoT devices and then perform a fine-grain classification model to differentiate between nine distinct IoT devices. In [16], the authors propose to approximately model IoT behaviour by the collection of communication protocols used, and the set of request and response traffic sequences observed, from which device features are then extracted from the network traffic. Finally, features are aggregated using a statistical model as a base profile for device identification. In [17], the proposed scheme extracts up to 23 features from each packet, from which they form a fingerprint matrix and use a random forest to develop a classification model.

More recently, deep learning has been adopted for IoT behaviour fingerprinting. Reference [18] proposes to use information from network packets to identify devices. They observed that packet interarrival time (IAT) is unique among devices. They extract and plot the IAT graph for packets where each graph contains 100 IATs. Then, they use the CNN to learn features from device graphs and distinguish different devices. Another study in [19] attempts to automatically identify the semantic type of a device by analysing its network traffic. First, they define a collection of discriminating features from raw traffic flows, and those features are used to characterise the attributes of devices.

Then, they use a LSTM-CNN model to infer the semantic type of a device. Due to the large variety of devices and manufacturers in IoT setting, other researchers [20] argue that traditional intrusion detection methods cannot suitably detect compromised IoT devices given the scale of devices being monitored. They propose DI¨OT, a self-learning distributed anomaly-based intrusion detection system, to identity compromised devices. DI¨OT can effectively build device-type-specific behaviour profile with minimal human efforts. Federated learning is utilized in DI¨OT to efficiently aggregate behaviour profiles across devices. Compared with traditional machine learning, in the works described using deep learning, features are often automatically extracted from raw device traffic.

(3) *Device Integrity Testing Based on DL*. Hardware Trojans are a major security concern where hardware can be accessed by untrusted third-party. Based on the availability of trusted (i.e., golden) chips, hardware Trojan detection methods can be split into methods that utilise golden chips and alternative approaches. Traditional methods include one-class anomaly detection, two-class classification, clustering, and outlier-based, utilising training data such as on-chip sensor data and on-chip traffic data.

Research on the topic of deep learning-based hardware Trojan detection methods is limited but increasing, with many currently based on simple neural networks as an anomaly detector. In works such as [21], they use power consumption data as the model input. To reduce the noise in data acquisition, wavelet transforms are used. A neural network is used to distinguish between normal chip power consumption and deviation in chip performance where a Trojan may be present. Wen et al. [22] use self-organizing maps (SOMs) to detect hardware Trojans. They employ Hotspot to catch the steady-state heat-map from running IC. Then, a 2-dimensional principal component analysis (PCA) is used to extract features from the heat-map. The SOM is used to automatically distinguish Trojan-infected chips. Both of these methods can efficiently detect hardware Trojan. Reshma et al. [23] argue that there exists a large intercluster distance between normal nodes and Trojan infected nodes, especially in the controllability and transition probability. They extract features from chips using autoencoders and use k-means to find Trojan nodes. Work by [24] proposes to extract features from netlists; for each netlist, they get 11 features. Then, the deep multilayer neural network is used to find out malicious netlist. However, the role they play is as an anomaly detector with predefined features. It is suggested that further research of deep learning in this application is still required.

3.2. Network Behaviours in IoT. Here, we focus on the modelling of network behaviours as a result of IoT devices, including device access control, connection-related activities, firmware upgrades, and remote access and control of devices. In particular, it would be beneficial to develop a model that can identify malicious behaviours across the network so as to block remote access. The following network activities will be considered: network nefarious activity/abuse, eavesdropping interception/hijacking, outage, damage/loss, and failures/malfunctions. Since IoT devices are typically constrained in terms of computational resource, detectors that are designed to operate on the devices will therefore need to be lightweight and maintain efficiency. Botnet and DDoS are two primary threats that have been observed on IoT networks in recent times, such as the Mirai botnet that managed to access and control millions of low-level devices. As the number of connected IoT devices increases, so will the nature of attacks that attempt to leverage these to conduct large-scale DDoS operations. Deep learning has recently been used to attempt to identify such attacks. Meidan et al. [25] use deep autoencoders to build normal behaviour profiles for each device. They extract statistical traffic features and train autoencoders with features from

benign traffic. When applied to new traffic observations for a new IoT device, there exists a bigger reconstruction error on the trained autoencoder which can be used to indicate that the device could be compromised. Similar approaches used in Kitsune [26] use ensembles of autoencoders to identify anomalies in IoT such as Mirai. Both of the above approaches assume that normal traffic activity can be approximately reconstructed, while an anomaly would cause large reconstruction error. While many detection methods borrow ideas from traditional intrusion detection or anomaly detection methods, the above two methods considered the heterogeneous and resource constraints in IoT environment.

Other methods use the CNN to automatically identify malicious traffic in IoT. In [27], they turn the payload in the traffic packet into a hexadecimal format and visualize it into a 2D image. Then, they employ a lightweight CNN framework called MobileNet to extract features from traffic images and malware classification. To deal with the volume of traffic needed to analyse this in a DDoS setting, in [28], they propose a deep learning lightweight DDoS detection system called LUCID. They exploit the weight sharing properties of the CNN to classify the traffic, which makes it efficient to be deployed in resource-constrained hardware. To efficiently extract features from network traffic, authors in [29] employ damped incremental statistics as basic features. They then use triangle area maps (TAMs)-based multivariate correlation analysis (MCA) to generate grayscale images as training data from normalized traffic features. They then use these as input to a CNN to learn a model for detecting anomalies.

3.3. Model Data Abuse in IoT Environment. Data gathered by IoT networks can be of great value, and abuse of this data can result in serious consequence, e.g., the case made against Cambridge Analytica. It is therefore crucial that IoT devices manage data responsibly. Data leakage can occur from the generation of data, the use of data, and the transmission/storage of data over the IoT network. For example, data collection by smart meters will reflect home usage patterns for electricity, gas, or water, which if leaked could expose attackers to information about when the house is occupied or not. Similarly, this information could be exposed by other smart devices such as kitchen and entertainment appliances. Intelligent IoT services will naturally aim to gather personal information to further inform the services being provided, where personalisation is deemed as enriching user experience. Five context parameters related to IoT data privacy are proposed by [30]: place ("where"), type of collected information ("what"), agent ("who"), purpose ("reason"), and frequency ("persistence"). In this section, we briefly review works related to data privacy and data integrity.

(1) *Data Privacy in IoT with Deep Learning.* In [31], they study visual privacy within an IoT setting. With low-end IoT cameras, they propose a method for constructing privacy protected and forgery-proof high frame-rate videos. They deployed their software prototype on three different IoT settings: on-site, vehicular, and aerial surveillance. In [32], the authors propose a deep and private-feature learning framework called deep private-feature extractor (DPFE). Based on information theoretic constraints, they are training a deep model which allows the user to prevent sharing sensitive information with a service provider and at the same time enables the service provider to extract approved information using the trained model. Similar work in [33] proposes a feature learning framework that leverages a double projection deep computation model (DPDCM). Different from the traditional deep learning framework, they use double projection layers to replace the hidden layers, which can learn interactive features from big data. Furthermore, they design a training algorithm to fit the DPDCM model. To improve the learning efficiency, cloud computation is used. They also propose privacy-preserving DPDCM based on BGV encryption to protect personal data.

(2) *Federated Learning.* Recently, there has been much interest in developing methods where a collective of devices can contribute towards a global shared model, whilst maintaining the privacy of data that is stored locally on each device. This is well suited in settings where there is a large population of devices that would benefit from collective knowledge but where there are not the rights or the ownership of the devices to control the data. Smart phone devices benefit from federated learning for the purpose of improving predictive services (e.g., predictive text and location recommendations) whilst not disclosing information about other mobile phone users. Smart meters and other IoT devices would benefit in a similar manner. Works by [34] demonstrate that decentralized federated learning can improve data privacy and security, while reducing economic cost. Works in [35] integrate deep reinforcement learning algorithms and the federated learning framework into an IoT edge computing system. The main focus of their work is to improve the efficiency of the mobile edge computing system. They design a framework called "In-Edge AI" to maximise the collaborative efficiency among devices and edge nodes. With this framework, learning parameters can be exchanged efficiently for better training and inference. Their framework can reduce unnecessary system communication while at the same time carry out dynamic system level optimization and application-level enhancement. Wang et al. studied a broad range of machine learning models optimized with gradient descent algorithms [36]. Their research first analyses the convergence bound of distributed gradient descent

algorithms. Then, they propose an algorithm to reach the best trade-off between local and global parameter learning while given limited resource budget.

(3) *Data Integrity in IoT with Deep Learning*. In an IoT setting, upholding integrity is vital to ensure that there is consistency between the actual, physical observation, and the transmitted data or signals that represent this activity. False data injection (FDI) is an attack against a cyber-physical system by modification of the sensor data, which could include SCADA (supervisory control and data acquisition) systems used widely in sectors supporting critical national infrastructure. For example, an FDI attack on an engine sensor could cause erroneous sensor outputs which would result in severe impact on physical maintenance algorithms. Likewise, the infamous Stuxnet attack [5] involved FDI to falsify the behaviour of the centrifuges that then caused physical destruction to the premises.

Recently, deep learning has been used in false data injection detection both in Internet and IoT. Works in [37] use deep learning algorithms to learn the behaviour feature model from historical sensor data and employ the learned model to infer the FDI behaviour in real time. Similar work was proposed by Wang et al. WangH2018 used a two-stage sparse scenario-based attack model to detect attack in smart grid given incomplete network information. To effectively detect established cyber-attacks, they develop a defense mechanism based on interval state model. In their model, they use a dual optimization method to model the lower and upper bounds of each state parameter, which will maximise variation intervals of the system variable. At last, they employ the deep learning model to properly learn nonlinear and nonstationary behaviour features from historical electric usage data.

3.4. Deep Learning Methods for IoT Security. In this section, we will summarize the methods using deep learning techniques to enhance IoT security. Building on this, we propose methodologies that can extend towards improved IoT security.

3.4.1. Feature Learning Process. Traditionally, feature extraction consists of data collection, data preprocessing, and feature extraction. For the purpose of our work, we will separate this out further to consider four steps: data collection, data encoding, feature definition, and feature extraction. Figure 2 shows the behaviour analysis using a multilayered hierarchical Bayesian networks, in which security features are categorised into static features, dynamic features, and causal features based on existing features extracted from IoT security behaviour databases.

In the data collection phase, raw data such as RF signals, device features, heat-map, and raw network packets are collected. Raw data can often be very large, of mixed data types, and can contain many unrelated records, and so there is a need to establish how to manage this information. Data encoding is the process of defining the basic element of interest that is contained within the input, such as individual pixels within a given image or individual packets within a network traffic stream. Here, we represent each element as x_i. In the feature definition stage, data are organised such that a coherent understanding of the data object can be analysed. Typically, input elements could be organized as a distribution, a sequence, a matrix, or more recently in deep learning, a tensor. Following data encoding, raw inputs can be transformed into a format that could be used as input for a deep learning model. Here, we define the feature definition process as D, and then the data after feature definition can be represented as

$$X = D(x_1, x_2, \ldots, x_k), \tag{1}$$

where D is the process to organize basic element into predefined orders. Based on feature definition, features are from inputs. Methods such as statistical methods, series analysis, frequency analysis, or machine learning are used to extract features from organized data elements. Here, we define the feature extraction as

$$V = F(X) = F(D(x_1, x_2, \ldots, x_k)), \tag{2}$$

where F is used to represent the feature extraction methods. Usually, the output in feature extraction is feature vectors with fixed length $V = (v_1, v_2, \ldots, v_m)$. In this work, a two-step data preprocessing phase is introduced: (1) a data encoding process that can be used to extract suitable features from mixed raw inputs and (2) a feature definition process that provides structure to our data.

3.4.2. Deep Learning for Device Feature Extraction. IoT networks can consist of a very large number of connected devices, which can mean that identifying a specific device within a network becomes challenging. Here, we focus on techniques to extract features that can identify a specific IoT device.

One of the most powerful aspects of deep learning is the ability to automatically learn useful features from raw inputs, e.g., autoencoders. For device identification, deep learning methods for device feature extraction can be classified based on the raw data that they use. Information such as sensor noise pattern, radio frequency features, or energy consumption can reflect the uniqueness of devices. Using deep learning, higher level features can be extracted and even very subtle differences between devices can be discovered. Such is the case in camera identification, where raw images captured by using the camera can be gathered. Here, we first define the raw image set as I and then extract the noise pattern from images, which are considered to be unique for a given device. Usually, noise patterns can be calculated as follows:

$$N = I - F(I), \tag{3}$$

where I is the original image containing the original noise and $F(i)$ is the denoised version of I. The residual noise N is called signal noise which is typically unique for a given

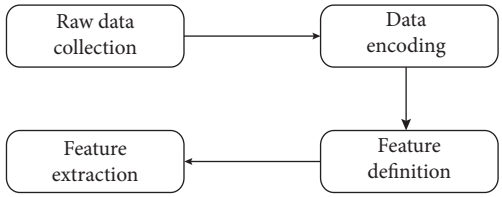


Figure 2: Feature learning processes.

device. To extract signal noise patterns from an image, statistical methods treat the residual noise as a two-dimensional distribution and extract features such as mean, max, skewness, and kurtosis. Using a frequency representation, noise signals can be treated as a two-dimensional signal, and then methods such as wavelet transform or Fourier transform can be used to identify the frequency of noise level. Different from methods above, deep learning methods such as in [10] feed the signal noise matrix directly into the CNN that aims to automatically extract features from noise with minimal human intervention.

Using deep learning, in [11], they learn the signal noise pattern with the signal noise extraction step. For each colour image I, with its camera model L, the authors extract K nonoverlapping patches P_k, $k \in [1, K]$, with each of size 64 ×,64 pixels. To avoid selecting uninformative regions of the image (e.g., dark or saturated pixels), they exclude all regions where the average pixel value is near to half of the image dynamic range. They use the CNN to extract noise feature representation from regions. After that a set of N $(N-1)/2$ linear binary SVMs are trained to identify the difference between different camera models.

Similar work has been performed for RF fingerprinting [13, 38]. In [38], RF signals (IQ) are collected from multiple devices. They consider the Zigbee device baseband as a complex time series represented as follows:

$$r(t) = s_1(t) + n_1(t), \tag{4}$$

where $n(t)$ represents the noise. The training data used here are historical in-phase and quadrature (I and Q) data from six ZigBee devices transmitting at 0, −1, −5, −10, and −15 dBm. They experiment using different window sizes of 16, 32, 64, 128, and 256 which represent the number of I and Q input sequences into the deep learning models. Finally, they utilise different deep learning architectures to assess how they perform for classifying Zigbee devices.

From the view of energy consumption, a heat-map of devices can formulate a normal device template. In this manner, malicious modification of hardware could be detected. In work [22], authors split chips into several equal sized grids. Then, they use a randomly generated "*excitation vector*" to feed the running chip. Finally, for each grid, they measure the steady-state temperature. A 2D PCA is used to identify the feature map from the original heat-map.

To train a device recognition model, data must first be collected from devices. Then, data must be transformed to provide features that can be used as input to a deep learning model. Typical input to the deep learning framework can be matrix-based, sequence-based or statistical-based. Then, with the help of deep learning, a normal template of devices can be formulated.

Traditional ML methods rely on human efforts to extract features which may not easily scale when considering IoT devices [39]. Also, manually curated features may be susceptible to attack or could be manipulated by an attack. Using deep learning techniques such as autoencoders, representative features could be identified automatically, which can be used for fingerprinting devices.

3.4.3. Network Behaviour Modelling with Deep Learning. The basic elements that are often considered for network behaviour modelling are packets, flows, and conversations between communication entities. Unlike other data, data in network traffic are heterogeneous. Basic input in network traffic can be divided into three parts: timestamp, connection identifier, and data description. A packet could therefore be represented as $p = <time, header, content>$. Network behaviour can be formally defined as sequence of packets running between communication nodes:

$$X = (p_1, p_2, \ldots, p_m), \tag{5}$$

where packets are sorted based on timestamps.

Given the heterogeneous nature of a network capture, it can be challenging to extract features directly from a packet sequence. Typically, statistical features are calculated over some short time interval to inform feature representation. Features such as interarrival time, packet length, packet count, bytes send, and bytes received can be extracted. This information can reflect network behaviour property such as communication frequency, traffic volume, and connectivity. Furthermore, these features can reflect buffer size and computational ability and also reflect the services that a device provides. This process can be defined as follows:

$$S = S(X) = S(w_1, w_2, \ldots, w_k), \tag{6}$$

where w_i can be represented as the sequence of packets that fall in the ith time window. Typically, researchers may extract uncorrelated statistical features from time, connection, and content.

Deep learning in network behaviour modelling plays two roles: (1) automatic extraction of high-level features from network traffic and (2) automatic identification of corresponding features across feature dimensions. Behaviour modelling based on deep learning can be defined as

$$V = H(S) = H(w_1, w_2, \ldots, w_k), \tag{7}$$

where H represents the black box, nonlinear function used in deep learning. After that a fixed length behaviour vector to represent network behaviour can be achieved.

As mentioned, features such as interarrival time and packet length could inform of device properties such as buffer size, computational ability, and the services that the device provides. With CNN, LSTM, and other deep learning methods, complex service pattern can be extracted. Such as work in [18], they use interarrival time (IAT) as features to

generate a graph of IAT for 100 packets. The graphs are then treated as images, and all images are converted to a size of 150×150, where they are then given as input to a neural network to identify device behaviour pattern. Research from [19] considers network traffic from devices as sequences of packets. They first split traffic into subflows with fixed time interval T. For each subflow, features related to the number of packets, packet length statistics, and protocol-related features are extracted. Then, a LSTM-CNN cascade model is used to extract high-level features from the whole flow. Both [25, 26] use autoencoders to get normal profiles of IoT devices. Both of them extract packet size, packet count, packet jitter, and packet size from the flow of packets, and then the autoencoder is used to reconstruct the original input to find devices that exhibit deviation in their behaviour.

3.4.4. Proposed Approach

(1) *Semantically Meaningful Device Modelling.* Although deep learning methods may achieve greater accuracy in device identification, semantically meaningful device modelling is still lacking. Since the range of possible IoT devices increases every day, signature-based methods would fail to recognise new device types and provide accurate device property records. Here, we propose a semantically meaningful device identification framework. The main concept is to decouple the feature learning process and devices identification process via a middle-layer process called service identification or functional identification. For example, a functional collection may consist of the following activities: {*capture image, record video, share photo, make phone call, play music*}. We could then represent a camera and a smart phone as follows:

(1) Camera (1, 1, 1, 0, 0)
(2) Smart phone (1, 1, 1, 1, 1)

From the example above, it can be inferred that semantic representation would be more robust for device modelling. Even though devices may be updated frequently, the basic functionality that such devices offer will change at a much slower rate. Using deep learning, a three-step device identification framework can be introduced: (1) using deep learning, basic features of devices can be extracted; (2) from the features extracted, the functional or behaviour model can be constructed, and the output of this step would be the functions or services that the device provides; and (3) using the inferred service type to identify the device. The training process of our framework can be described formally as follows:

Step 1. Extract data from devices including signal noise pattern, network traffic, device heat-map, and other relevant attributes. This data are then transformed to a tensor input for a deep learning model. We define the raw data of the ith device as d_i, the training data sets as $D = d_i, d_2, \cdots, d_n$, the device type of ith device as y_i, the training labels of devices as $Y = y_1, y_2, \cdots, y_n$, and the function set of devices as $A = a_1, a_2, \cdots, a_k$ where k is the size of function set.

Step 2. Use deep learning to extract features from raw data, where the feature extraction process can be defined as Φ, and the extract features are defined as $f_i = \Phi(d_i)$, where f_i is the vector of dimension m. The feature extraction process can be seen as a nonlinear mapping from n dimensional raw data space to m dimensional feature space.

Step 3. Find the mapping between features and attributes. Here, the mapping can be a linear or nonlinear decision or even based on a decision tree. The mapping here is defined as Ψ, which maps input of m dimension into a k dimension binary vector, e.g., (1, 0, 1, 0, 0).

Step 4. Based on attributes, use Bayesian theorem to calculate the most possible device type.

Compared against previous deep learning approach, our method clearly separates feature learning and device type recognition using an intermediate layer. Whilst features of a device type may change rapidly, the functions served may change much slower, and so by using a functional (or service) layer, we can develop a much more robust framework for device identification.

(2) *Behaviour Analysis via Multilayered Hierarchical Bayesian Network.* Compared with traditional approaches, deep learning can learn to extract features from a collection of basic statistics automatically. One advantage is that deep learning can find high-level complex features which may be hard to identify in a statistical setting. Similarly, as feature extraction can be used in device identification, network behaviour modelling using deep learning can be considered using the same approach, with the main difference being the product of the deep learning model. In IoT network behaviour analysis, the aim of deep learning is to identify patterns in network behaviour. Using this network behaviour model, malicious behaviours can then be identified at the network level.

However, it is important to note that deep learning models may fail to capture the causal relationship between traffic features and traffic behaviour. Consequently, the inner relationships between features and behaviour may be overlooked by the training process, which is clearly a vital aspect when examining the network characteristics. Here, we follow the approach in our previous work [40], as illustrated in Figure 3, where

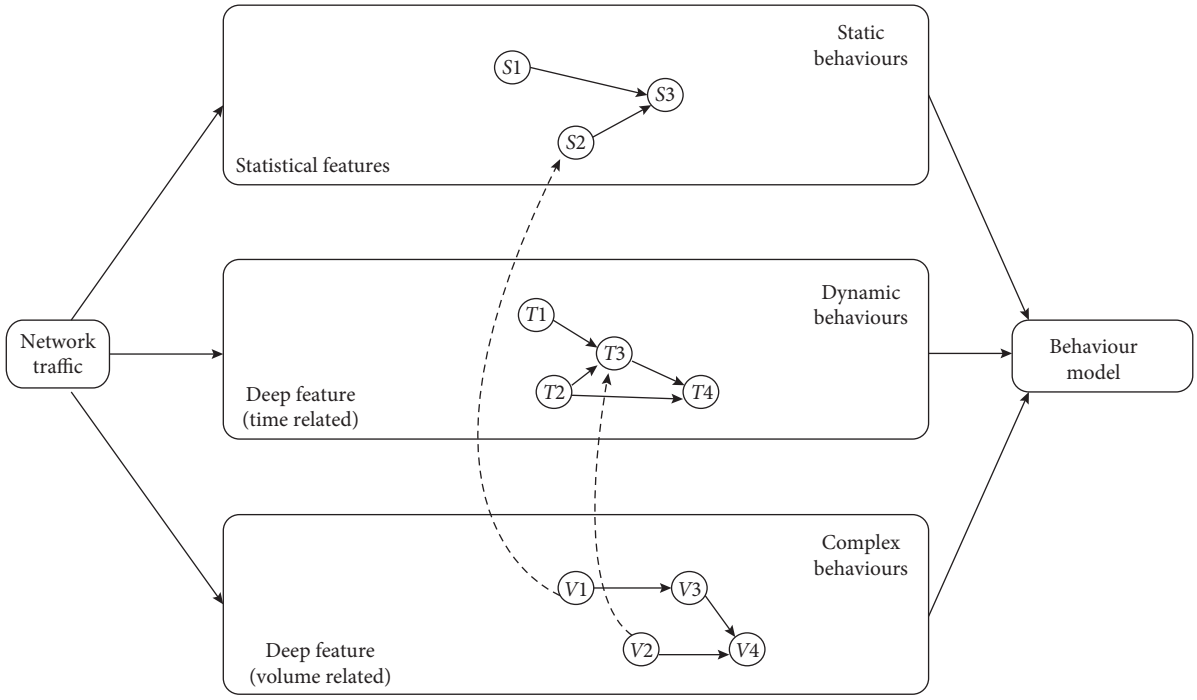


Figure 3: Behaviour analysis via multilayered hierarchical bayesian networks.

features may be extracted using three approaches simultaneously: statistical, time-related, and volume-related.

Then, using a three-layer Bayesian network, we can capture interactions between these different feature representations.

4. Evaluation

To evaluate the performance of DL-based methods, we need a set of performance measures and we need some benchmark dataset to verify the deep learning framework.

4.1. Evaluation Measure. Precision is one of the most common used measures in machine learning. Precision is defined as the number of true positive predictions over the total number of positive predictions made (where either true or false). In an intrusion detection system, f_p would be the number of false alerts.

$$\text{Precision} = \frac{t_p}{t_p + f_p}, \tag{8}$$

where t_p represents the number of correct cases that were classified as positive examples and f_p represents the number of incorrect cases that were classified as positive. Another common performance measure is recall (also known as sensitivity). Recall is defined as the number of true positive predictions over all positive instances. Recall can be defined as follows:

$$\text{recall} = \frac{t_p}{t_p + f_n}, \tag{9}$$

where f_n is the number of positive instances which are incorrectly classified as negative (false negatives). In an intrusion detection system, f_n would be the number of attacks that go undetected. Often, there is an inverse relationship between precision and recall, where it is possible to increase one at the cost of reducing the other.

In binary classification, usually the F1 score (also known as F-Measure or F-score) is used to measure accuracy. It combines the precision and recall at the same time to compute the final score. The F1 score is defined as the harmonic mean of precision and recall. F1 score is defined as follows:

$$F1 = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}. \tag{10}$$

The maximum score for the F1 measure is 1, where essentially both perfect precision and perfect recall are both achieved.

The receiver operating characteristic (ROC) curve is a graphical plot that can help us find the discrimination threshold of a binary classification system. It is a graphical plot created by plotting the true positive rate against the corresponding false positive rate with different threshold setting. The area under curve (AUC) is the performance measure used for classification systems with different threshold settings. The AUC represents the size of area under the ROC curve, which can be seen as a measurement of separability. It can give us information about how much the learning model is capable of distinguishing between classes. Higher the AUC, the better the classifier can predict a result that matches with the

provided labels. Essentially, the higher the AUC, the higher the model's classification power.

4.2. Evaluation Dataset

(1) *Device Identification.* Miettinen et al. [17] introduce the *captures _IoT Sentinel* dataset. They collect network traffic during the setup stage of 31 smart home IoT devices across 27 different device types (4 types are represented by 2 devices each). They repeat the collection process 20 times for each device type.

The IEEE Signal Processing Society [39] provides a camera model identification dataset, where images exhibit noise patterns that are specific to camera models. They use 10 different camera models to produce images for the training set. For each camera model (one device per camera model), they take 275 full images. They use the same camera models to create a test set but using different devices. Each instance of the test data consists of a 512×512 pixel image that is cropped from the centre of an image taken with the specific camera device.

(2) *Intrusion Detection and Anomaly Detection.* The Bot-IoT dataset [41] was created by researchers at UNSW Canberra Cyber. They simulated attacks including DDoS, DoS, OS and service scans, keylogging, and data exfiltration attacks in a designated realistic network environment. The dataset incorporates a combination of both normal and malicious traffic. The dataset is provided in various data format, such as original pcap files, the generated Argus files, and extracted features in csv format. To assist with the labelling process, they separate the data based on attack category and subcategories.

IoT-23 [42] is a dataset created by researchers at the Stratosphere Laboratory to model realistic network traffic behaviour from IoT devices. The dataset contains 23 captures, with 3 captures representing benign IoT traffic and 20 representing malware traffic captures. For each malicious packet capture, they execute a specific malware sample on a Raspberry Pi, with each malware using several protocols to perform various actions. Samples in their dataset include raw pcap files and Bro generated json files. They separate the dataset into different folders to provide labels.

AWID [43] is a dataset created for wireless network intrusion detection. It contains traffic collected from real-world WiFi traffic. Kolias et al. [43] apply several traditional supervised machine learning approaches to perform intrusion detection on the AWID dataset. They use mutual information to select the top 20 most informative features to then train 8 classifiers. Result shows that their framework can achieve an overall accuracy ranging from 89.43% to 96.2%. In the original AWID dataset, each data instance includes 155 features along with the corresponding training label.

(3) *Data Privacy.* Lee et al. investigated 14 IoT scenarios in [30] to randomly structure five contextual parameters to create IoT scenario descriptions. They request for users to describe their opinions using the free text field. They then use cluster analysis to infer how these five contextual parameters affect people's reaction in IoT environments.

An aggregated electronic signals dataset, namely, REDD [44], was established to infer devices behind the smart meter, which is also known as the task of energy disaggregation. The REDD dataset consists of whole-home and circuit/device-specific electricity consumption for a number of real houses over a period of several months. With this data, the researchers were able to identify the correspondence between devices and electricity usage.

5. Challenges and Research Trends

5.1. Research Challenges

(1) *Efficiency.* Resource constraints of IoT devices remain an important impediment towards deploying deep learning models. Memory efficiency and time efficiency would be two core concerns in implementing deep learning in real IoT systems. Although deep learning models can be trained offline, how to deploy the model remains to be a problem.

The power of a deep learning model originates from the large amount of nonlinear, stacked neurons used in the deep learning architecture. Deep learning models consume raw data which pass through layered neurons to inform a decision. How to reduce the storage and computation needed for execution of the deep learning model in resource constraints applications is an ongoing challenge.

With the development of deep learning methods, various new architectures surpass state-of-the-art performance. However, many of them have not necessarily been developed for the IoT setting. To fully adapt these algorithms to an IoT setting would certainly help to improve the performance of recent studies [45, 46].

(2) *Adaptive.* Devices and applications in the IoT ecosystem are evolving every day, and so deep learning must also be adaptable in the same way. In a real-world network, zero-day attacks will occur. New devices are introduced into the IoT system subsequently. Also, the distribution of network traffic or signal frequency would likely change as new devices join the network. A static trained model cannot easily adapt to changing conditions and so could result in an increase in false positives and false negatives. Another ever-changing element is the request from the end user. Those changes bring new challenges to deep learning applications in the IoT setting. Deep learning algorithms must cope with the fast-evolving environment both from the macro- and micro-perspective.

Another consideration is that many IoT devices may be deployed in a wide scale of areas. The properties of the environment where IoT are deployed may vary from each other. Retraining a deep learning model for each setting not only costs a lot of time but also requires further labelled training data.

(3) *Heterogeneous Data.* IoT devices produce a lot of data with different type and scale, such as data from signal frequency and network traffic, which although they may originate from the same device, they will have different formats. Even data of same type may differ in scale, such as packets number and bytes number. Although they all belong to network features, they use different scale. How to handle those heterogeneous data is an ongoing problem [47, 48].

5.2. Research Trends

(1) *Resource Efficient Deep Learning.* Here are two ways toward resource efficient deep learning: (1) modification on deep learning model itself, compressing or pruning the original deep learning model and (2) result cache, preventing duplicate computation by sharing result among devices.

Previous illuminating studies on neural network focused on compressing dense parameters matrices into sparse matrices. One possible approach to reduce model complexity would be to convert parameters into a set of small dense matrices. A small dense matrix does not require additional storage for element indices and is efficiently optimized for processing.

The ultimate goal of the deep learning model is to inform decisions. One question is if it is needed to make decision for every event in system. One observation shows that device with more computation power would convey richer services, while computation limited devices would be inclined to do a limited set of jobs. So instead, would it be possible to cache the result instead of repeatedly calculating the same decision? Similar ideas have been widely used in computer architecture and operation system design. Methods such as latest recently used (LRU) have long been used in the operating system to avoid duplicate storage access, which could reduce large amount of unnecessary computation.

(2) *Lifelong Learning.* Human and animals have the ability to quickly adapt to new environments; they can continually acquire, fine-tune, and transfer knowledge and skills throughout their lifespan. This ability, known as the ability of lifelong learning, is mediated by a rich set of neuron cognitive mechanisms that together contribute to the development and specialization of our sensorimotor skills as well as to long-term memory consolidation and retrieval. Consequently, lifelong learning capabilities are crucial for computational learning systems and autonomous agents interacting in the real world and processing continuous streams of information. In the IoT setting, with dynamically changing environments and low-powered devices, lifelong learning is needed to create more intelligent and efficient agents.

However, lifelong learning remains a long-standing challenge in machine learning. The most common phenomenon in lifelong learning using traditional machine learning algorithms is called catastrophic forgetting, which means with continual acquisition of incrementally available data from unknown nonstationary data distributions will decrease the performance of learning algorithms. This breaks the basic assumption of deep learning or other machine learning, which needs a stationary data distribution in training data. To improve scaling of deep learning algorithms in IoT setting, lifelong learning is needed to co-operate with information incrementally available over time.

6. Conclusion

In this survey, it is seen that deep learning offers significant potential across the IoT setting. This survey focuses primarily on the use of deep learning technology to investigate the security features of devices in the context of IoT. Specifically, deep learning-based device profiling and fingerprinting were comprehensively discussed. An approach for semantically meaningful device modelling was proposed using a functional layer to improve feature mapping for device identification. Finally, we discussed challenges and research trends that we intend to explore in our future research.

Data Availability

No data can be shared publicly.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] G. Egham, "5.8 billion enterprise and automotive IoT: endpoints will be in use in 2020," 2020, https://www.gartner.com/en/newsroom/pressreleases/2019-08-29-gartner-says-5-8-billion-enterprise-and-automotiveio.

[2] F.ortune Business Insights, "COVID-19 impact: high dependency on novel technology to bode well for market," 2020, https://www.fortunebusinessinsights.com/industry-reports/internetof-things-iot-market-100307.

[3] Wikipedia, "2016 dyn cyberattack," 2020, https://en.wikipedia.org/w/index.php?title=2016%20Dyn%20cyberattack&oldid=763071700.

[4] A. Greenberg, "Hackers remotely kill a jeep on the highway—with me in it," 2020, https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/.

[5] R. Langner, "Stuxnet: dissecting a cyberwarfare weapon," *IEEE Security & Privacy Magazine*, vol. 9, no. 3, pp. 49–51, 2011.

[6] R. Minerva, A. Biru, and D. Rotondi, "Towards a definition of the internet of things (IoT)," *IEEE Internet Initiative*, vol. 1, no. 1, pp. 1–86, 2015.

[7] S. Zhao, S. Li, L. Qi, and L. D. Xu, "Computational intelligence enabled cybersecurity for the internet of things," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 4, no. 5, pp. 666–674, 2020.

[8] S. Li, T. Qin, and G. Min, "Blockchain-based digital forensics investigation framework in the internet of things and social systems," *IEEE Transactions on Computational Social Systems*, vol. 6, no. 6, pp. 1433–1441, 2019.

[9] M. Mohammadi, A. Al-Fuqaha, S. Sorour, and M. Guizani, "Deep learning for IoT big data and streaming analytics: a survey," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 2923–2960, 2018.

[10] A. Tuama, F. Comby, and M. Chaumont, "Camera model identification with the use of deep convolutional neural networks," in *Proceedings of the 2016 IEEE International workshop on information forensics and security (WIFS)*, pp. 1–6, IEEE, Abu Dhabi, UAE, 2016.

[11] L. Bondi, L. Baroffio, and D. Guera, "Etc. first steps toward camera model identification with convolutional neural networks," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 259–263, 2016.

[12] S. Qi, Z. Huang, and Y. Li, "Etc. audio recording device identification based on deep learning," in *Proceedings of the 2016 IEEE International Conference on Signal and Image Processing (ICSIP)*, pp. 426–431, IEEE, Singapore, 2016.

[13] J. Yu, A. Hu, and F. Zhou, "Etc. Radio frequency fingerprint identification based on denoising autoencoders," in *Proceedings of the 2019 international conference on wireless and mobile computing, networking and communications (WiMob)*, pp. 1–6, IEEE, Barcelona, Spain, 2019.

[14] J. Bassey, D. Adesina, and X. Li, "Etc. Intrusion detection for IoT devices based on RF fingerprinting using deep learning," in *Proceedings of the 2019 fourth international conference on fog and mobile edge computing (FMEC)*, pp. 98–104, IEEE, Rome, Italy, 2019.

[15] Y. Meidan, M. Bohadana, and A. Shabtai, "Etc. ProfilioT: a machine learning approach for iot device identification based on network traffic analysis," in *Proceedings of the Proceedings of the Symposium on Applied Computing*, Article ID 506509, Pisa, Italy, 2017.

[16] B. Bezawada, M. Bachani, and J. Peterson, "etc. iotsense: behaviour al fingerprinting of iot devices," 2018, http://arxiv.org/abs/1804.03852.

[17] M. Miettinen, S. Marchal, and I. Hafeez, "Etc. Iot sentinel: automated devicetype identification for security enforcement in iot," in *Proceedings of the 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pp. 2177–2184, IEEE, Atlanta, GA, USA, 2017.

[18] S. Aneja, N. Aneja, and M. S. Islam, "IoT device fingerprint using deep learning," in *Proceedings of the 2018 IEEE international conference on internet of things and intelligence system (IOTAIS)*, pp. 174–179, IEEE, Bali, Indonesia, 2018.

[19] L. Bai, L. Yao, and S. S. Kanhere, "etc. Automatic device classification from network traffic streams of internet of things," in *Proceedings of the 2018 IEEE 43rd Conference on Local Computer Networks (LCN)*, pp. 1–9, IEEE, Chicago, IL, USA, 2018.

[20] H. Gao, C. Liu, Y. Li, and X. Yang, "V2VR: reliable hybrid-network-oriented V2V data transmission and routing considering RSUs and connectivity probability," *IEEE Transactions on Intelligent Transportation Systems*, p. 1, 2020.

[21] X. Li, F. Xiao, and L. Li, "Etc. detection method of hardware trojan based on wavelet noise reduction and neural network," in *Proceedings of the International Conference on Cloud Computing and Security*, pp. 256–265, Springer, Cham, Switzerland, 2018.

[22] H. Gao, L. Kuang, Y. Yin, B. Guo, and K. Gou, "Mining Consuming behaviour s with temporal evolution for personalized recommendation in mobile marketing apps," *ACM/Springer Mobile Networks and Applications (MONET)*, vol. 25, no. 4, pp. 1233–1248, 2020.

[23] K. Reshma, M. Priyatharishini, and M. Nirmala Devi, "Hardware trojan detection using deep learning technique," *Advances in Intelligent Systems and Computing*, Springer, Singapore, pp. 671–680, 2019.

[24] K. Hasegawa, M. Yanagisawa, and N. Togawa, "Hardware trojans classification for gate-level netlists using multi-layer neural networks," in *Proceedings of the 2017 IEEE 23rd international symposium on on-line testing and robust system design (IOLTS)*, pp. 227–232, IEEE, Thessaloniki, Greece, 2017.

[25] Y. Meidan, M. Bohadana, Y. Mathov et al., "N-BaIoT-Network-Based detection of IoT botnet attacks using deep autoencoders," *IEEE Pervasive Computing*, vol. 17, no. 3, pp. 12–22, 2018.

[26] H. Gao, W. Huang, and Y. Duan, "The cloud-edge based dynamic reconfiguration to service workflow for mobile ecommerce environments: a QoS prediction perspective," *ACM Transactions on Internet Technology*, 2020.

[27] R. Shire, S. Shiaeles, and K. Bendiab, "Etc. malware squid: a novel iot malware traffic analysis framework using convolutional neural network and binary visualisation," *Smart Spaces, and Next Generation Networks and Systems*, p. 6576, Springer, Cham, Switzerland, 2019.

[28] R. Doriguzzi-Corin, S. Millar, and S. Scott-Hayward, "Etc. LUCID: a practical, lightweight deep learning solution for DDoS attack detection," *IEEE Transactions on Network and Service Management*, vol. 17, 2020.

[29] J. Liu, S. Liu, and S. Zhang, "Detection of IoT botnet based on deep learning," in *Proceedings of the 2019 Chinese control conference (CCC)*, pp. 8381–8385, IEEE, Guangzhou, China, 2019.

[30] H. Lee and A. Kobsa, "Understanding user privacy in internet of things environments," in *Proceedings of the 2016 IEEE 3rd world forum on internet of things (WF-IoT)*, pp. 407–412, IEEE, Reston, VA, USA., 2016.

[31] H. Yu, J. Lim, K. Kim et al., "Pinto: enabling video privacy for commodity IoT cameras," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1089–1101, Toronto Canada, 2018.

[32] S. A. Osia, A. Taheri, A. S. Shamsabadi et al., "Deep private-feature extraction," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 1, pp. 54–66, 2018.

[33] Q. Zhang, L. T. Yang, Z. Chen et al., "Privacy-preserving double-projection deep computation model with crowdsourcing on cloud for big data feature learning," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2896–2903, 2017.

[34] Y. Zhao, M. Li, L. Lai et al., "Federated learning with non-iid data," 2018, http://arxiv.org/abs/1806.00582.

[35] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-edge AI: intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Network*, vol. 33, no. 5, pp. 156–165, 2019.

[36] S. Wang, T. Tuor, T. Salonidis et al., "Adaptive federated learning in resource constrained edge computing systems,"

IEEE Journal on Selected Areas in Communications, vol. 37, no. 6, pp. 1205–1221, 2019.

[37] Y. He, G. J. Mendis, and J. Wei, "Real-time detection of false data injection attacks in smart grid: a deep learning-based intelligent mechanism," *IEEE Transactions on Smart Grid*, vol. 8, no. 5, pp. 2505–2516, 2017.

[38] H. Jafari, O. Omotere, and D. Adesina, "Etc. IoT devices fingerprinting using deep learning," in *Proceedings of the MILCOM 2018-2018 IEEE military communications conference (MILCOM)*, pp. 1–9, IEEE, Los Angeles, CA, USA, 2018.

[39] IEEE Signal Processing Society, "IEEE's signal processing society camera model identification," 2020, https://www.kaggle.com/c/sp-society-camera-modelidentification.

[40] S. Li, T. Tryfonas, G. Russell, and P. Andriotis, "Risk assessment for mobile systems through a multilayered hierarchical bayesian network," *IEEE Transactions on Cybernetics*, vol. 46, no. 8, pp. 1749–1759, Aug. 2016.

[41] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset," *Future Generation Computer Systems*, vol. 100, pp. 779–796, 2019.

[42] Stratosphere Laboratory, *A Labeled Dataset with Malicious and Benign IoT Network Traffic*, Stratosphere Laboratory, Prague, Czechia, 2020.

[43] C. Kolias, "Intrusion detection in 802.11 networks empirical evaluation of threats and a public dataset," *IEEE Communications Surveys & Tutorials*, vol. 18, pp. 184–208, 2015.

[44] J. Z. Kolter and M. J. Johnson, "REDD: A public data set for energy disaggregation research," in *Proceedings of the Workshop on data mining applications in sustainability (SIGKDD)*, vol. 25, pp. 59–62, San Diego, CA, USA, 2011.

[45] R. Mahmoud, T. Yousuf, F. Aloul, and I. Zualkernan, "Internet of things (IoT) security: current status, challenges and prospective measures," in *Proceedings of the 2015 10th International Conference for Internet Technology and Secured Transactions (ICITST)*, pp. 336–341, IEEE, London, UK, 2015 December.

[46] K. Ashton, "That internet of things thing," *RFID Journal*, vol. 22, no. 7, pp. 97–114, 2009.

[47] H. Wang, J. Ruan, G. Wang et al., "Deep learning-based interval state estimation of AC smart grids against sparse cyber attacks," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 11, pp. 4766–4778, 2018.

[48] A. Ukil, S. Bandyopadhyay, and A. Pal, "Privacy for IoT: involuntary privacy enablement for smart energy systems," in *Proceedings of the 2015 IEEE international conference on communications (ICC)*, pp. 536–541, IEEE, London, UK, 2015.

Hindawi
Security and Communication Networks
Volume 2020, Article ID 8884924, 14 pages
https://doi.org/10.1155/2020/8884924

Research Article

Improving Topic-Based Data Exchanges among IoT Devices

Fu Chen,[1,2] **Peng Liu**,[2] **Jianming Zhu**,[1] **Sheng Gao**,[1] **Yanmei Zhang**,[1] **Meijiao Duan**,[1] **Youwei Wang**,[1] **and Kai Hwang**[3]

[1]*Department of Computer Science, Central University of Finance and Economics, Beijing102206, China*
[2]*College of Information Sciences and Technology, The Pennsylvania State University, State College 16801, PA, USA*
[3]*Chinese University of Hong Kong (CUHK), Hong Kong, China*

Correspondence should be addressed to Fu Chen; chenfu@cufe.edu.cn

Received 11 June 2020; Revised 12 September 2020; Accepted 16 October 2020; Published 31 December 2020

Academic Editor: Honghao Gao

Data exchange is one of the huge challenges in Internet of Things (IoT) with billions of heterogeneous devices already connected and many more to come in the future. Improving data transfer efficiency, scalability, and survivability in the fragile network environment and constrained resources in IoT systems is always a fundamental issues. In this paper, we present a novel message routing algorithm that optimizes IoT data transfers in a resource constrained and fragile network environment in publish-subscribe model. The proposed algorithm can adapt the dynamical network topology of continuously changing IoT devices with the rerouting method. We also present a rerouting algorithm in Message Queuing Telemetry Transport (MQTT) to take over the topic-based session flows with a controller when a broker crashed down. Data can still be communicated by another broker with rerouting mechanism. Higher availability in IoT can be achieved with our proposed model. Through demonstrated efficiency of our algorithms about message routing and dynamically adapting the continually changing device and network topology, IoT systems can gain scalability and survivability. We have evaluated our algorithms with open source Eclipse Mosquitto. With the extensive experiments and simulations performed in Mosquitto, the results show that our algorithms perform optimally. The proposed algorithms can be widely used in IoT systems with publish-subscribe model. Furthermore, the algorithms can also be adopted in other protocols such as Constrained Application Protocol (CoAP).

1. Introduction

The Internet of Things (IoT) has emerged as the next evolution of the technology which covers a wide variety of devices and system platforms, such as embedded systems, networked sensors, actuators, and smart home devices. Inevitably, the resource starved nature in most IoT devices makes it error-prone to efficiently manage and maintain a reliable IoT system, not to mention the security issues associated with some scarcely attended IoT devices [1, 2]. One of the biggest challenges in IoT is the management and monitoring of an IoT system, especially when heterogeneous devices are networked in geographically distributed environments. And IoT devices are often used to monitor industrial production, energy consumption, agriculture condition, and business operation, even in sensitive medical application. Therefore, the cross-platform and cross-application data flows among IoT devices are often extremely important. The protocols that allow machine-to-machine, device-to-device, and device-to-server communication to communicate with each other become very important in IoT system. In general, wireless network protocols have evolved in the form of WiFi, ZigBee, Bluetooth, and routing protocols from multihop to ad hoc and mobile ad hoc networks (MANET); the data protocols play an important role in IoT protocol stack, which includes Message Queuing Telemetry Transport (MQTT) [3], Constrained Application Protocol (CoAP) [4], AMQP [5], and WebSocket [6]. Comparatively MQTT seems to be more promising as seen in Figure 1 showing the trends comparison of MQTT, COMP, and AMQP, according to Google trends dataset [7]. This paper includes the communication efficiency of MQTT. All the information in the MQTT is organized with Tree structure and topics-based naming [8].

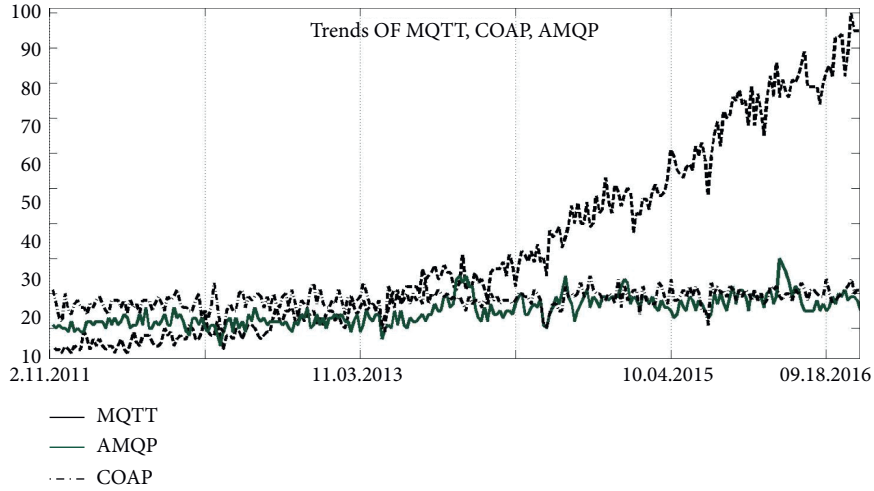


Figure 1: Trends comparison of MQTT, CoAP, and AMQP.

The history of MQTT dates back to 1999, which was invented by IBM. MQTT uses the so-called "publish/subscription" model to transmit data among the IoT things. MQTT is more mature and stable than CoAP and AMQP. Due to the low-cost and power-constrained nature of most IoT devices, the communication efficiency is extremely important. For example, if we can improve the power efficiency, the IoT device can be operated for even longer time period than low power efficiency without having to replace batteries. This is one of the fundamental motivations for our focus on IoT data communication protocol efficiency in this paper. An IoT system consists of a large number of internet-connected resource-constrained and dynamic nature devices [9]. Therefore achieving data transmission efficiency is very important in this type of distributed networks. Data transmission in many IoT systems use publish-subscribe model, as in MQTT. In AMQP protocol, clients send messages to the AMQP server with topics, and server listens for messages with those topics and routes the messages to the applications subscribed to the AMQP server [5, 10]. MQTT uses a similar approach where message brokers use topics to route messages from publishing clients to subscribing clients [11]. To exchange data between devices with publish-subscribe model, matching, filtering, and routing of messages based on topics, as the bridge of publisher and subscriber, have extreme importance in IoT system. Especially when there are plenty of messages produced by a large number of IoT devices, improving the efficiency and effect of topic matching, filtering, and routing is vital. The work in this paper aims to address this important need of data communication in IoT system. To address this important need, we propose two algorithms to improve data exchange efficiency in IoT system. The two important contributions of this paper are as follows:

(i) Development of a topic routing algorithm to communicate data between IoT devices

(ii) Development, implementation, and evaluation of a Broker Network to improve the survivability and robustness of connected-device data exchange in IoT

We evaluate our algorithms with MQTT protocol implementation Mosquitto [3, 12]. We have performed extensive experiments and results show that with our algorithms we can dramatically improve data exchange efficiency while significantly reducing transmission time. The organization of the paper is as follows. In Section 2, we present background and motivation of this work highlighting main concerns that arise in the data communication in IoT and state of the art in the field. Section 3 presents our algorithms in detail. The implementation and evaluation of our algorithms are presented in Section 4. Finally, conclusions are drawn in Section 5.

2. Background

2.1. Need of the Data Exchange Efficiency Problem in IoT. Devices in IoT system often have limited processing capabilities, low bandwidth, and small memory capacities and are powered by batteries with limited working hours. Therefore, improving the usage efficiency of the network and reducing power usage in IoT scenarios are extremely important. As a promising data communication protocol in IoT, MQTT is a machine-to-machine (M2M), lightweight publish-subscribe messaging transport protocol. In publish-subscribe model, subscriber registers its interest topics to a broker and publisher sends data to a broker [13–15]. Publish clients and subscribe clients exchange data through a broker with topic matching. When there are huge message flows exchanged by broker, issue of latency may be inevitable [16]. The latency not only decreases the speed of data transmission but also wastes device battery to decrease the overall device life-span [17–20]. In this paper, we focus on data exchange efficiency in an environment where there is a limited Internet access or there are serious network jitter problems for several hundred miles around. The emphasis is transferring data based on subject-based route to a data center. Essentially, it is about a new gateway and topic-based routing. First, we need to classify necessity of transferring data between gateways. MQTT in IoT is used in data exchange which is different from corresponding sensor network about data integration.

We depict an agriculture processing scenario to clarify the necessity for data transmission with topic-based routing. Consumers today often put significant value on fresh, high quality food. Farmers need to get information from fields, take actions to crop depending on weather conditions, and show the information about the produce to the end consumers with public access interface. This information includes identifying the drought and weather conditions, periodic measurements of monitoring data with gateway, depending on intensity of drought decide about irrigation and watering, monitoring pest disasters, and then taking actions about the spraying pesticides with drone, without distinguishing nodes from the broker's point of view, which brings difficulty due to large number of devices. If there are plenty of things in one IoT system, it is necessary to identify nodes having the corresponding information and to identify these nodes dramatically and efficiently.

If this information is configured manually, while not the automatically, it is a huge challenge to manage thousands of nodes. When the network structure is changed, the configurations need to be done. Therefore, automatic discovering and configuring the IoT network are a huge challenge. And agriculture processing system scenario has the following features: data from farm in fields to table on Internet, field data spanning up to one year, wide geographical areas of crops, and fragile network conditions.

As noted earlier, MQTT used in IoT system emphasizes data exchange, not mainly for data integration. The question arises, why in the agriculture process system we need data exchange? There are several reasons to make it necessary for data exchange in agriculture processing system and data exchange about storm, hurricanes, cyclones bad weather. It is known to all that storm may influence several hundreds of miles. It is very important to spread related information about agriculture. In fact, weather conditions have a heavy influence on the crop management. Data exchanges among different farms are very important. For example, in Pennsylvania, there may have been dozens of farms located in a region where pests and diseases spread very quickly. Similarly, farms in remote and regional areas in Australia are spread over on thousands of acres. If the information around one regain can exchange on time, precautions can be taken to prevent or reduce loss from any natural disaster which may otherwise affect drastically the agriculture processing system.

2.2. Essence of the Problem Studied in This Paper. The IoT things are spread around hundreds of miles in field or plain. According to the mechanism of MQTT protocol, data exchanges among different nodes and brokers should be configured according to the IP addresses and topic names. Only after manual configuration or by configuration files, the data can be exchanged along specific path to another node, which gets damaged crushes down, we will not know until proactive testing is performed. Neighboring nodes might have information about the damaged nodes but the important question here is how to dynamically figure out which nodes have the corresponding data and how to arrive at that node with the shortest path automatically. In our proposed algorithms, we have focused on this problem. On the contrary, if the reconfiguration is done manually or through CLI scripts, data exchange with MQTT in IoT is not reliable and is not easy to recover. Maintenance and management of data exchange in this model is a hard task which can be costly. Therefore, a reliable, automatic data discovery and data exchange model with a shortest path in a fragile, dynamic network environment is very important. With our algorithms, we can find the specific nodes for specific data with one topic and get the shortest path to that nodes dynamically and automatically. We need a smarter recovery program, which can save money (automatically) to avoid remote login and can perform recovery efficiently. Generally speaking, through the broadcast method and the way of active subscription, we can achieve data exchange automation. The existing practices often involve manual configuration where messages are read or processed manually. In our proposed model, we aim for automatic data exchange.

3. Broker Network

Data collection is one of the key functions in IoT system. However, battery powered devices in IoT often work in low bandwidth and fragile network environments. One of the major challenges is to provide efficient, stable, and scalable services in IoT. For example, to extend a device's effective working life-span as much as possible, efficiency should be considered to save power. At the same time, to ensure the availability and scalability of data communication services, broker network is a natural choice in IoT system [21, 22]. The broker network can provide stable connections among the devices to handle the data traffics in IoT. In this section, we describe our proposed novel topic-based data transmission algorithm to efficiently exchange data in IoT system.

3.1. Topic-Based Data Routing in MQTT. When servers in the cloud system gather data from things as messages in IoT system, we will route the messages flow through broker network in IoT to cloud. For example, client 1 subscribes to broker 1 with topic "top" and client 2 publishes to broker 2 with topic "top." Broker 2 should route the message published by client 2 to broker 1, so that client 1 can get the corresponding data. This is the so-called topic-based message routing problem in IoT. The key point here is the efficiency when broker nodes synchronize data among brokers, which include the following:

(i) Forwarding all messages to the other bridges

(ii) Routing of messages to the brokers

For the CPU, bandwidth, and memory constrained devices, the routing decision algorithm should be simple enough to save resource. Normally, we can divide the nodes into two types: one is routing node and the other is terminal subscriber node or publisher node. Our data routing decision algorithm is depicted in Algorithm 1 called TBRouting.

In summary, there are three scenarios:

(i) One to one directly

(ii) One to many directly

(iii) One to remote nodes through routers

In one to remote nodes through routers scenario, the cloud servers gather data from devices as messages transmit through the local IoT network. We address the issue of collecting specific data and how to transfer it efficiently in our algorithm through finding the shortest path based on topics for message delivery to cloud or server. That is to say, a routing protocol is considered. Our message routing decision algorithm is depicted in Figure 2.

In Figure 2, we depict message routing scenario. There are 7 routers, router ①~router ⑦ connected with each other wirelessly and to the Internet or cloud shown in Figure 2. ① is connected to Internet or cloud, and ⑥ is connected to sensor network. When messages are sent to router ⑥ through sensor network, the message will be routed from router ⑥ to router ①. Because the messaging router here is not the typical router in a typical network, therefore, we define the message routing algorithm below and routing table in the following sections. For the fragile network and constrained hardware devices, the topology may be changed constantly [23]. Therefore, the path and router table may change with the topology. Just as the scenario shown in Figure 2, the routing algorithm is introduced in the following steps:

(1) Figuring out the shortest path between router ⑥ and router ①, it is a single-source shortest path topic-based problem in graph theory. We can address it, for example, with Dijkstra's algorithm. Here, the shortest path is router serial ⑥ $\Longrightarrow$ ④ $\Longrightarrow$ ② $\Longrightarrow$ ①.

(2) In reverse sequence, router ① subscribes topics from router ②, router ② subscribes topics from router ④, and router ④ subscribe topics from router ⑥.

(3) When sensors publish topics to router ⑥, the messages will be pushed back to hop by hop until router ①.

When the topology structure of router network changes, we can figure out the new shortest path from the source to the end as described in Step (1) dynamically. The "routing tables" are critical to the system because IoT network may not be using the traditional TCP/IP protocol stack which makes real-time data transmission with the shortest path without considering the network protocol even more challenging. That is to say, regardless of TCP/IP or Zigbee, the IoT system can still find the data source and the shortest path transparently based on topics. In the next section, we will describe this algorithm in detail.

3.2. Topic-Based Data Routing Protocol. In order to facilitate problem description, first we provide some relevant definitions. Then algorithm is depicted in detail with an example. We present efficiency evaluation of our algorithm in Section 4.

3.2.1. Definitions

Definition 1. Routing table structure: in order to make nodes find each other and in the shortest path, some information should be stored in every nodes. Table 1 provides the structure of the routing table.

Definition 2. Subscribing node: node 1 subscribes to node 2; node 1 is the so-called subscribing node.

Definition 3. Subscribing node queue: A subscribes to B, B subscribes to C, C subscribes to D, and "ABC" is the so-called subscribing node queue.

Definition 4. Weight (T): the number of node in a string.

Definition 5. Host_Topic (T): the ultimate data consumer and corresponding topic. For example, "Host_Topic (ABCD/TOP)" = "D/TOP".

Definition 6. Searched_Set: a node set, in which every node has finished building the routing table.

Definition 7. NotSearched_Set: a node set, in which every node has not finished building the routing table.

Definition 8. Adjacent (X): a set of adjacency of node X.

Definition 9. Match (top): determining whether the node can provide the corresponding topic data.

3.2.2. Routing Algorithm. The algorithm essentially completes the following tasks. First, the algorithm determines the node where the topic and corresponding data reside. Second, the algorithm dynamically finds the shortest path. Then the data can be transported to Internet. That is to say, the algorithm solves that which node has the topic and data and how the data can be transported to the ultimate user. We solve this problem with our new routing algorithm named Topic-Based Routing Algorithm (TBRG). The rerouting algorithm is given as follows. In the algorithm, "**R**" is the ultimate data consumer in Algorithm 2.

Next, we elaborate our algorithm through a concrete example illustrated in Figure 3, which shows an example of nodes structure of IoT system. We assume Ⓐ is a node in data center, data ultimate consumer, and Ⓘ is the data source connecting the sensor.

① Initialization of node Ⓐ in Table 2.

② A subscribing to nodes Ⓑ, Ⓓ, and Ⓔ, that is, Ⓐ ⇒ Ⓑ, Ⓐ ⇒ Ⓓ, Ⓐ ⇒ Ⓔ, as shown in Tables 3–5.

③ Next, B and E subscribing to node Ⓒ and node Ⓕ, respectively, that is, node Ⓑ ⇒ node Ⓒ and node Ⓔ ⇒ node Ⓕ in Tables 6 and 7.

④ Next, node Ⓒ and node Ⓕ subscribing to nodes Ⓘ and Ⓖ, respectively, that is, node Ⓒ ⇒ node Ⓘ and node Ⓕ ⇒ node Ⓖ as shown in Table 8, VX.

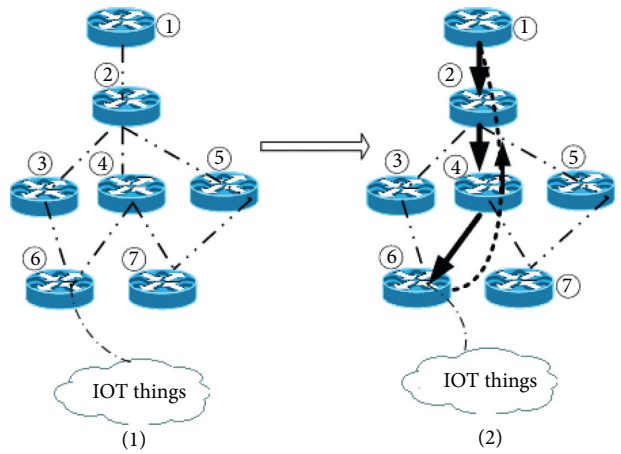


Figure 2: Message routing decision algorithm IoT network.

(1) **procedure TBR**outing.
(2) When a broker receives data with topics from one client, the broker matches topics in its topic tree.
(3) If the node is just the terminal subscriber, then broker forwards it to that node, and then go to 5. If the node is not the terminal consumer of the topic, go to 4.
(4) If the node is not the terminal consumer of the topic, the data will be forwarded to the other broker nodes, which is the next hop of the routing.
(5) Continuing with 3, with next hop, data eventually will be sent to cloud or terminal nodes. The messages are sent with the shortest path to the final destination.
(6) The topic may be sent to many different receivers who subscribe the topics. All the receivers will receive the same messages.
(7) The broker will decide whether to store the topic or not.

Algorithm 1: TBRouting.

Table 1: Structure of the routing table.

Node	Path and topics	Weight
Subscribing node	Subscribing node queue	Number of nodes in queue

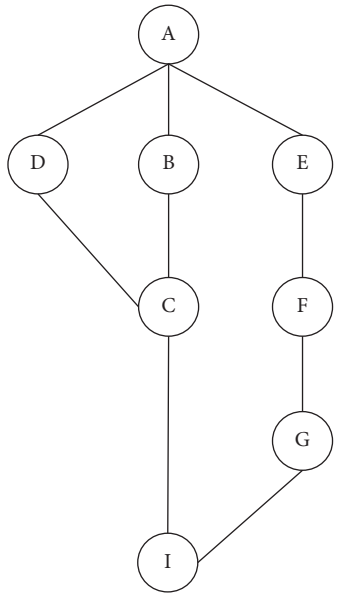


Figure 3: Example of topic routing protocol.

⑤ Node ⑥ subscribing to Node I, that is, node ⑥ ⇒ node I'. In the algorithm, this step is less likely to happen because of the value of weight (CBA/top) in Step ④; the routing information is "CBA/top", as shown in Tables 9 and 10.

Figure 4 shows a bipartite graph. Set (1) is a topic set $x1 \sim xn$ corresponding to published data. Set (2) is a topic set $y1 \sim yn$ corresponding to subscribed data. The next algorithm is to match topics from one node in set (1) to another node in set (2) step by step according to the shortest path in reverse order. In addition, when we upgrade system or update the status about things in field, that is to say, delivering the instruction set from cloud to IoT devices, the instruction dataset is routed reversely with a series of publish and then subscribing among routers.

3.3. Rerouting Algorithm in MQTT. Each broker may serve for thousands of clients' connections. In case one broker crashes down, clients should have the ability to connect to another broker and work continually and smoothly, which is called the rerouting problem. To address this problem, we propose a rerouting algorithm to take over the topic-based session flows with a controller when a broker crashes down. Although it is almost impossible to communicate when broker crashes down, data can still be communicated by another

```
(1) Procedure TBRG
(2)   RoutingTable ⟵ initially, ultimate consumer node R;
(3)   Searched_Set ⟵ R;
(4)   NotSearched_Set ⟵ R;
(5)   while NotSearched_Set ≠ ∅ do
(6)     X ∈ NotSearched_Set;
(7)     for all Adjacent(X) do
(8)       NodeX ∈ Adjacent(X);
(9)       if NodeX ∉ Searched_Set then
(10)        NotSearched_Set ⟵ NodeX;
(11)        X⇒subscribing to NodeX;
(12)        Node X ⟵ subscriping request P;
(13)        i ⟵ Weight(P);
(14)        top ⟵ Host_Topic(P);
(15)        if !Match(top) then
(16)          addItemToRoutingTable(NodeX);
(17)          Searched_Set ⟵ Nodex;
(18)        else if Weight(Route_NodeX(Top)) > i then
(19)          Rewrite(Route_NodeX(P));
(20)     NotSearched_Set − X;
(21)     Searched_Set ⟵ X;
```

Algorithm 2: TBRG.

Table 2: Routing table of node A.

Node	Path and topics	Weight
0	0/top	0

Table 3: Routing table of node B.

Node	Path and topics	Weight
A	A/top	1

Table 4: Routing table of node D.

Node	Path and topics	Weight
A	A/top	1

Table 5: Routing table of node E.

Node	Path and topics	Weight
A	A/top	1

Table 6: Routing table of node C.

Node	Path and topics	Weight
B	BA/top	2

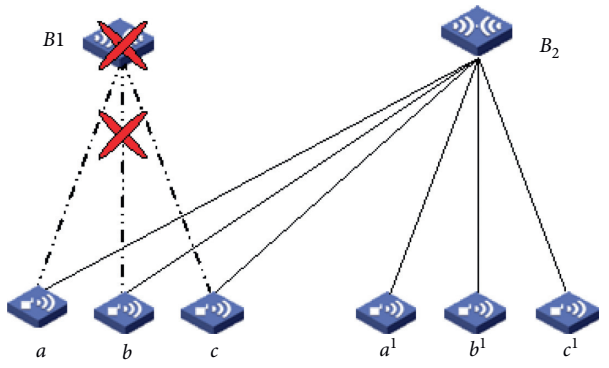


Figure 4: Routing tables in message router.

broker with rerouting mechanism. Therefore, high availability can be gained with this model in IoT, as shown in Figure 5.

3.3.1. Definitions. As in Section 3.2, we first provide some relevant definitions to facilitate the node replacing problem statement.

Definition 10. Topic_Session: when node X publishes a topic "t" to a broker named "Brk," and node Y subscribes the same topic "t" to the same broker "Brk" just illustrated in Figure 6(a) shown; we call this binary relation *Topic_Session* with notation $X\mathrm{TSR}(t, \mathrm{brk})Y$. If node Y only can get topic "t" from node X, we use notation $X!\mathrm{TSR}(t, \mathrm{brk})Y$ to express this relation. We depict the relation with Structural Operational Semantics as follows:

$$\text{Node } X \xrightarrow{\text{Pub}(t)} \text{Brk},$$
$$\text{Node } Y \xrightarrow{\text{Sub}(t)} \text{Brk},$$
$$\text{Node } X \xrightarrow{\text{Pub}(t)} \text{Brk},$$

Normally, we can use process expression to depict communication behavior. Actually, from the perspective of mathematics, this is a binary partial order relation. That is to

Table 7: Routing table of node F.

Node	Path and topics	Weight
E	EA/top	2

Table 8: Routing table of node of G.

Node	Path and topics	Weight
F	FEA/top	3

Table 9: Routing table of node I.

Node	Path and topics	Weight
C	CBA/top	3

Table 10: Possible routing table of node I'.

Node	Path and topics	Weight
G	GFEA/top	4

Table 11: Topic session flow.

No.	Things	Broker	Sub/Pub	Topic	On/off
1	a	B_1	0	t_1	1
2	a	B_1	1	t_6	0
3	a	B_2	1	t_2	1
4	b	B_1	1	t_3	1
5	a	B_1	0	t_4	1
6	c	B_2	1	t_5	0
7	c	B_1	0	t_6	11

say, $X \xrightarrow{\mathrm{TSR}(t)} Y$ and $Y \xrightarrow{\mathrm{TSR}(t)} X$ are different pair set. We use this concept to indicate the data flow.

Definition 11. $Pub(t) \cdot X$: processing node X has an action Pub with a parametric t. In this paper, behavior *Pub* is published.

Definition 12. $Sub(t) \cdot X$: processing node X has an action Sub with a parametric t. In this paper, behavior *Sub* is subscription.

Definition 13. Broke Strong Behavior Equivalence$\sim$: there are two brokers A and B. If the following relation exists, we call this relation $A \backsim B$:

(i) $X\,\mathrm{TSR}(t1, A)Y$ with topic "$t1$"

(ii) $X\,\mathrm{TSR}(t2, B)Y$ with topic "$t2$"

Definition 14. Broker Weak Behavior Equivalence$\approx$: there are two brokers A and B. If the following relations exist, we call this binary relation weak behavior equivalence $A \approx B$:

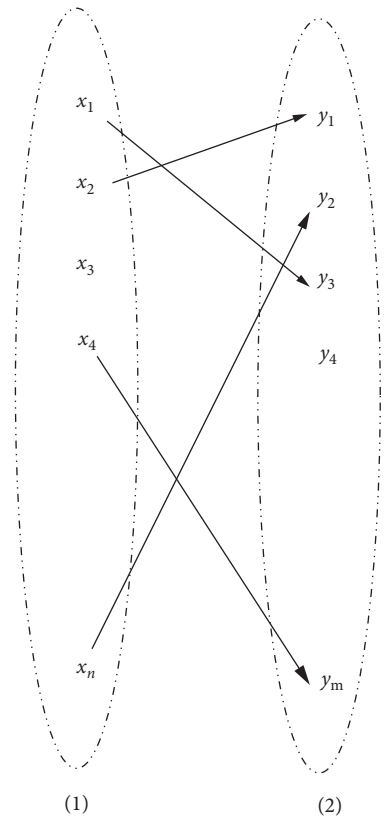


Figure 5: Fault migrating.

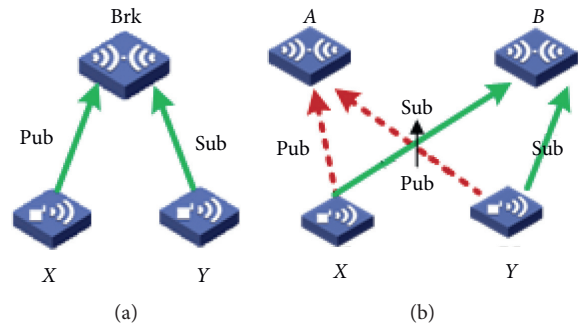


Figure 6: Topic_Session.

(i) $X\,\mathrm{TSR}(t1, A)Y$ or $Y\,\mathrm{TSR}(t1, A)X$ with topic "$t1$"

(ii) $X\,\mathrm{TSR}(t2, B)Y$ or $Y\,\mathrm{TSR}(t2, B)X$ with topic "$t2$"

If broker A crashes down, we want to find another broker B to take over the topic session flows in broker A, in which brokers A and B at least have the Strong Behavior Equivalence relation A$\backsim$B, as Figure 7 showed. In this section, we will focus on how to find an appropriate broker to take over the topic session flow in the crashed broker. With this behavior, the Topic_Session can continue.

3.3.2. Problem Analysis. In normal wireless network, if one node loses its connection, the node may update its status and broadcast its identification information to all neighbor nodes, just as shown in Figure 6(b). Then other sensors or actuators

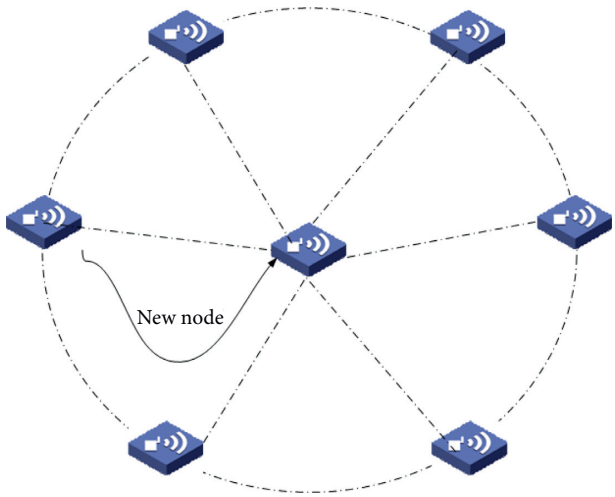


Figure 7: Wireless network broadcast.

may connect to this node. In this model, for example, in Figure 6(b), node X and node Y may find the other nodes.

The IoT network node is different from the normal wireless network node as follows.

(i) Node Y gets the topic "t" from node X through broker A with subscription behavior. However, when broker A crashes down, nodes X and Y must communicate with another broker with the same behavior. Or else node Y cannot get the topic "t" from node X. The situation may be worse when nodes X and Y have the relation $X!(t, \mathrm{brk})Y$.

(ii) In MQTT, one broker is specified when nodes publish or subscribe to a broker. That is to say, the subscriber or publisher cannot automatically reconfigure the broker. For example, MQTT protocol with QoS 0 and the node even are not aware when the broker crashes down.

The problem here is that nodes and brokers have an equal status. So nodes almost know nothing about each other. Because of mutual independence to each other, one node cannot change another node's behavior.

To sum up, when node X and node Y have the relation $X!(t, \mathrm{brk})Y$, nodes X and Y must synchronize migration to other nodes. Publisher may continue to publish data to broker A even when A has crashed down with QoS 0. Therefore, we must clarify the substitution conditions that how a broker can be replaced by another broker partly or totally. This raises the following questions:

(i) What conditions should be met when a broker substitutes one or a few brokers to take over that broker?

(ii) Who will we find a crashed down broker?

(iii) How can we take measures on the nodes or brokers to take over that broker?

(iv) Where should we store the relation $X\,\mathrm{TSR}(t, \mathrm{brk})Y$ set in that crash down broker? So we can keep topic related sessions to continue the data communication.

3.3.3. Broker Substitution Conditions. According to process behavior equivalence, if brokers A and B have a weak behavior equivalence relation $A \approx B$, then nodes X and Y have the approximate communication ability with A and B as shown in Figure 8.

The process expression of broker A is P_A, which is defined by the following syntax:

$$
\begin{aligned}
&P_A ::= \textstyle\sum_{i\in I} \prod \cdot A \,|\, ! \prod \cdot A,\\
&\textstyle\prod ::= Pub(t_i);\\
&Sub\langle t_i \rangle;
\end{aligned}
$$

The notation $!\prod \cdot A$ is loop execution behavior. And t_i is a topic $\in$ Topic_Set, which is a topic set to be published or subscribed. $\prod$ is an action prefix. Then the broker substitution conditions are

$$
\begin{aligned}
&\overline{Pub<t_i>} \cdot X \,|\, Pub(t_i) \cdot A \,|\, Sub(t_i) \cdot A \,|\, \overline{Sub<t_i>} \cdot Y;\\
&\overline{Pub<t_j>} \cdot X \,|\, Pub(t_j) \cdot B \,|\, Sub(t_j) \cdot B \,|\, \overline{Sub<t_j>} \cdot Y;\\
&\Rightarrow A \sim B, X\,\mathrm{TSR}(t, \mathrm{brk})Y;\\
&\begin{pmatrix} \overline{Pub<t_i>} \cdot X \,|\, Pub(t_i) \cdot A \vee Sub(t_i) \cdot A \,|\, \overline{Sub<t_i>} \cdot X \\ \vee \overline{Pub<t_i>} \cdot A \,|\, Pub(t_i) \cdot X \vee Sub(t_i) \cdot X \,|\, \overline{Sub<t_i>} \cdot A \wedge \end{pmatrix},\\
&\begin{pmatrix} \overline{Sub<tj>} \cdot Y \,|\, Sub(tj).B \vee Pub(tj) \cdot B \,|\, \overline{Pub<tj>} \cdot Y \\ \vee \overline{Sub<t_j>} \cdot B \,|\, Sub(t_j).Y \vee Pub(t_j) \cdot Y \,|\, \overline{Pub<t_j>} \cdot B \end{pmatrix},\\
&\Rightarrow A \approx B;\\
&\alpha \cdot X \,|\, \overline{\alpha} \cdot A \,|\, \beta \cdot B \,|\, \overline{\beta}.Y + \overline{\alpha} \cdot X \,|\, \alpha \cdot A \,|\, \beta \cdot B \,|\, \overline{\beta}.Y +\\
&\alpha \cdot X \,|\, \overline{\alpha} \cdot A \,|\, \overline{\beta} \cdot B \,|\, \beta \cdot Y + \overline{\alpha} \cdot X \,|\, \alpha \cdot A \,|\, \overline{\beta} \cdot B \,|\, \beta \cdot Y;\\
&\alpha \in \{Pub, Sub\},\\
&\Rightarrow A \approx B;\\
&\text{where,} \quad t_i, t_j \in Topic\_Set.
\end{aligned}
\tag{1}
$$

Essentially, Strong Behavior Equivalence $\backsim$ is a binary homomorphism relation. And Weak Behavior Equivalence $\approx$ is roughly approximate relation. If brokers A and B meet following conditions,

$A \backsim B$ or $A \approx B$, $XTSR(t, A)$, Y, $t \in Topic\_Set$, topic sessions on A can be transferred to broker B.

3.3.4. Pub/Sub Flow Controller and Measures. As noted in the previous section, publishers are not aware if a broker crashes down. Even the publisher knows the absence of broker, the topic-based session cannot go on without outside help, not to mention finding one or a few brokers to take over the shutdown broker. Therefore, the extra controller is necessary to ensure the survivability of IoT system, just as Software Defined Networks (SDN) do [24]. As shown in Figure 9, there is a controller over brokers to schedule the whole IoT network. The main purposes of the controller are as follows:

(i) Reviving of dead broker through heart beating

(ii) Figuring out the substitute node

(iii) Transferring the topic-based session flow from crash down broker to alternative broker

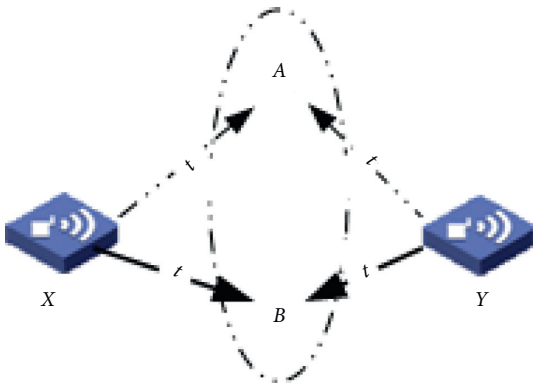


Figure 8: $A \approx B$.

But the controller will never be used as data communication. Its main usage is to manage the nodes in IoT network.

Through heart beating, for example, subscribing "hello world" to managed node on a regular timer with Quality of Service (QoS), it is easy to get the information in broker node in order to determine a substitute node. To figure out substitute node, the controller should collect the following information in regular time. For example, as shown in Table 11, in regular intervals.

In Table 11, if the action is subscribe then 1, else then 0. And if the broker is just performing normal then 1, else then 0.

Through the content in Table 11, we can deduce nodes that can be replaced with each other. For example, B1 and B2 can replace each other, as shown in the proof below:

$$\begin{aligned}
&\because \mathrm{Sub}(t)\cdot a|\,\mathrm{Sub}(t6)\cdot \overline{\mathrm{Pub}(t6)}B1|\,\mathrm{Pub}(t6)\cdot c,\\
&\therefore \Rightarrow c\,\mathrm{TSR}(t6,\ B1)a;\\
&\because \begin{cases} \mathrm{Sub}(t2)\cdot a|\,\mathrm{Sub}(t2)\cdot \overline{\mathrm{Sub}(t5)}B2|\,\mathrm{Sub}(t5)\cdot c, \\ \mathrm{Pub}(t1)\cdot a|\,\mathrm{Pub}(t1)\cdot \overline{\mathrm{Pub}(t6)}B1|\,\mathrm{Pub}(t6)\cdot c, \end{cases} \quad (2)\\
&\therefore \Rightarrow B1 \approx B2.
\end{aligned}$$

This comes to the conclusion $c!(t, B2)a$. This means when broker B1 crashes, the B2 can replace B1, with Structural Operational Semantics depicted as follows:

$$c \xrightarrow{\mathrm{Pub}(t6)} B1 \xleftarrow{\mathrm{Sub}(t6)} a \approx c \xleftarrow{\mathrm{Sub}(t6)} B2 \xrightarrow{\mathrm{Pub}(t6)} a. \quad (3)$$

The Algorithm 3 is showed below.

4. Implementation and Evaluation with Mosquitto or IoTivity

We implement a working prototype of the above algorithm using C and MQTT on Ubuntu system with Mosquitto and its client libmosquitto.lib as shown in Figure 10. MQTT is a machine to machine open source protocol originally developed by IBM. MQTT is particularly suitable for resource constrained IoT devices on fragile networks environment. As discussed in Section 1, MQTT is the most popular protocol used in IoT system now. And Mosquitto is an open source implementation of the MQTT protocol.

In our evaluation system, Figure 10, we deployed 25 nodes and divided nodes into two groups, one as broker and another as client. On the clients, we deployed a shell program to continuously publish data. And the nodes in broker network collect the data with subscription.

4.1. Efficiency with Scale of Data Transmission. Intuitively, in resource constrained device, the efficiency will decrease dramatically with data scale increasing. The trend of data transmission with data scale is one of the experimental contents of in this section. Especially to the specific protocol, the data transmission laws are more worthy of attention. In this section, we will evaluate the average transmission time with increased data scale. In Figure 11, vertical axis is time cost, and horizontal axis is the amount of data transmitted. From Figure 11, we can observe that average transfer completion time will increase dramatically from a specific point.

In broker network, a huge number of topics will be transmitted through the nodes. However, according to the above experiments, we know that the efficiency will decrease dramatically with increasing topics transmitted. Therefore, reducing the number of transmission nodes can reduce propagation delay time, and improve the transmission efficiency. This is just our primary goal in this paper. As the evaluation system shows with our algorithms proposed in this paper we can reduce the number of transmission nodes, which can improve the data transmission efficiency. At the same time, according to the experiment results, whole system efficiency will be improved as evident from the experiment results in the next section.

4.2. Efficiency Improvements. As Figure 12 showed, we used 25 virtual machines as a testbed to test algorithms proposed in this paper. Resources allocated to a virtual machine are limited. For example, memory in each node is less than 300 M. And the nodes are divided into two groups: one group is mainly for broker network and the other group is used for generating data. We deployed a Mosquitto broker in each node. In order to test any scale of the data, we used an endless loop shell script to publish a topic per second continuously; the script is as follows:

```
#!/bin/sh
while true; do
    echo hello.
    sleep 1
done | mosquitto_pub –l –t "Topic"
```

The other nodes can get the data through Mosquitto_sub with a counter. The counter is used to control the data scale. In the program, except subscription and counter, we calculate the time of data transmission. Through a series of subscription behaviors, the contents with corresponding topics can be passed to the final data consumer indirectly node by node and step by step. Each node automatically becomes a data generator pump. Just as described above, we have built an IoT simulation network with libmosquitto library and virtual machines as shown Figure 12, where

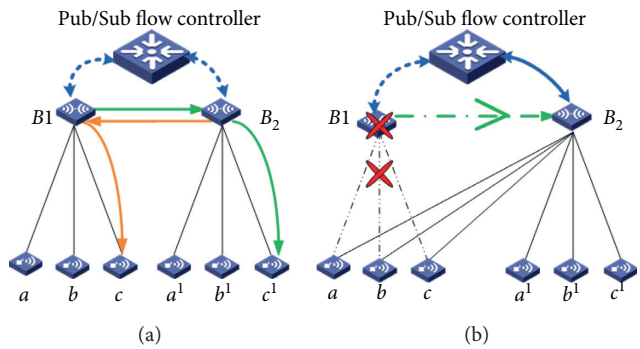


Figure 9: IoT controller.

(1) **Procedure** ReRouting
(2) *Session_Flow_Table* ⟵ *initialization*;
(3) **While** true **do**
(4) *Heartbeating broadcast*;
(5) *On/off* ⟵ *Heartbeatingbroadcast*;
(6) **for all** *No de X* ⟵ *Broker Set* **do**
(7) *PullingthePub/Subinformation*;
(8) *derive Broke Strong Behavio Equivalence*~;
(9) *deriveWeak Behavior Equivalence*≈
(10) *derive Topic_Sessionrelationship*;
(11) *Update Topic_SessionSet*;

Algorithm 3: ReRouting.

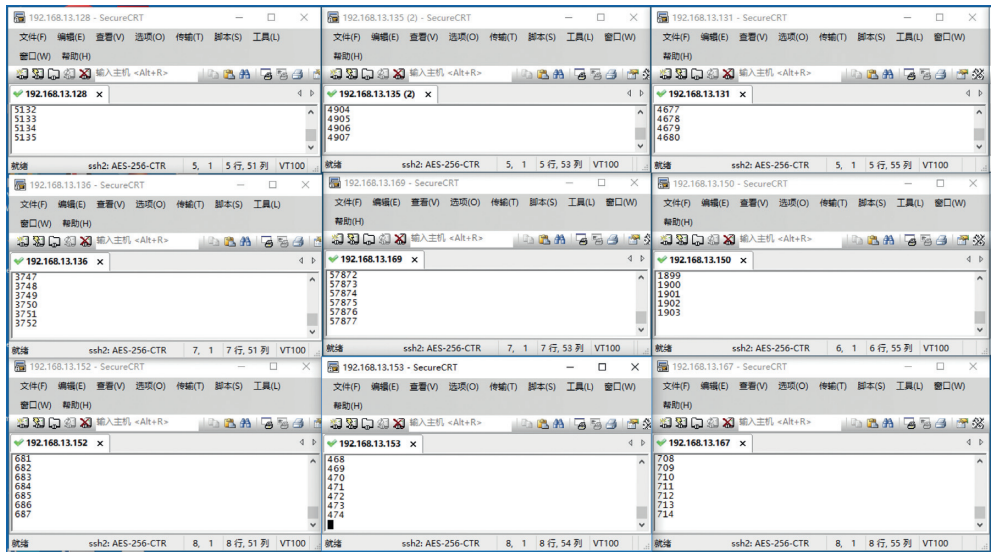


Figure 10: Evaluation system.

nodes 1 ~ 7 are broker network nodes; the rest of the nodes are data generator. Node 1 is the final data consuming node. Node 6 is the edge routing node. Data collected from the IoT will be transmitted to node 6 first and then transmitted to node 1 through broker networks.

Just as described above, our algorithm can determine nodes where the topic and corresponding data reside, and it dynamically adopts the shortest paths between nodes in data consumer and edge routers. The shortest paths mean fewer nodes needed for data transmission. Fewer transmission nodes can save data transmission time. We use the network topology shown in Figure 12 to evaluate the algorithms presented in this paper. In Figure 12, the data may be transmitted through ⑥, ⑦, ④ ③, ②, and ①. And the shortest path is ⑥, ④, ②, and

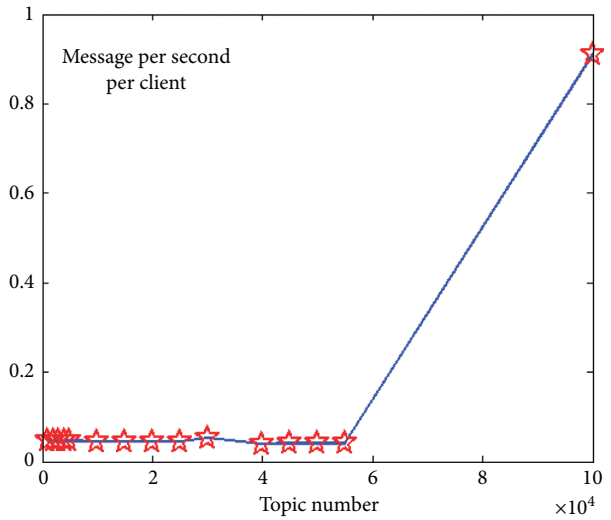


Figure 11: Transmission time with increased data scale.

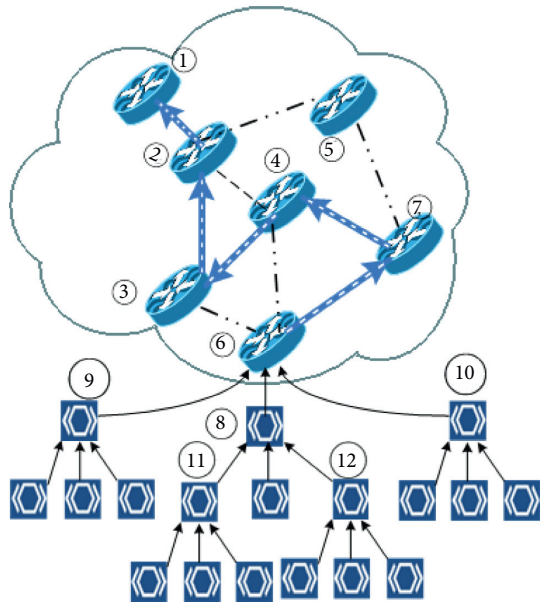


Figure 12: Data transmission without the shortest path.

①, as shown in Figure 13. That is to say, the path in Figure 13 is just the path we want to use to transmit the data.

To get the influence of the data transmission path with the data scale, we obtain the evaluation results with different workload to test the algorithm. The primary performance metrics are data transmission completion time. We get the data from 500 to 10000 items with 500 as a step. We run a Mosquitto on each node in transmission network as a broker. At the same time, we develop a program based on libmosquitto as a client to get the data and calculate the transmission completion time on each node. Figure 14 shows the performance comparison of our algorithm with different amounts of time. At the same time, to compare the entire system performance influence, the average transmission completion times of all nodes are measured, which is shown in Figure 15. In fact fewer transmission nodes can reduce energy consumption for whole system, which is not the scope of this paper.

We compare the data transmission with and without the shortest path in Figure 14. First, we observe that almost in each data scale the transmission completion time with the shortest path is far less than without it. Second, the amount of data transmitted and the transmission completion time is not a simple linear relationship. The comparison of transmission completion time about edge router is depicted in Figure 16. We illustrate the influence to the transmission node under these two different situations using Figure 16. In Figure 16, we observe that the data transmission path has an important influence on the whole system performance. The solution in edge router and in final data consumer has a similar trend. It is sufficient to show that data transmission

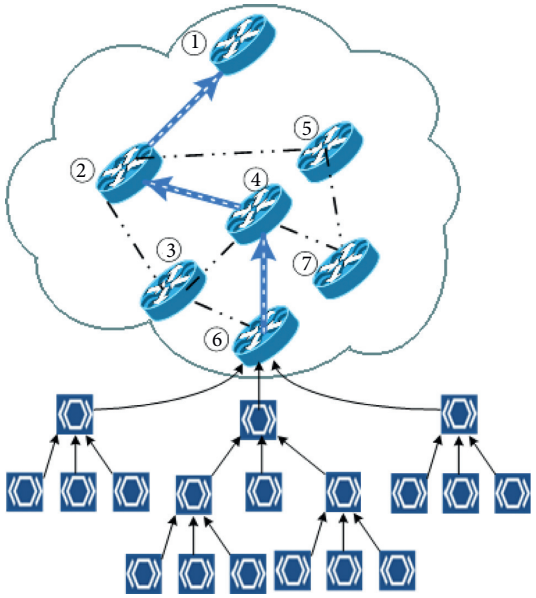


Figure 13: Data transmission with the shortest path.

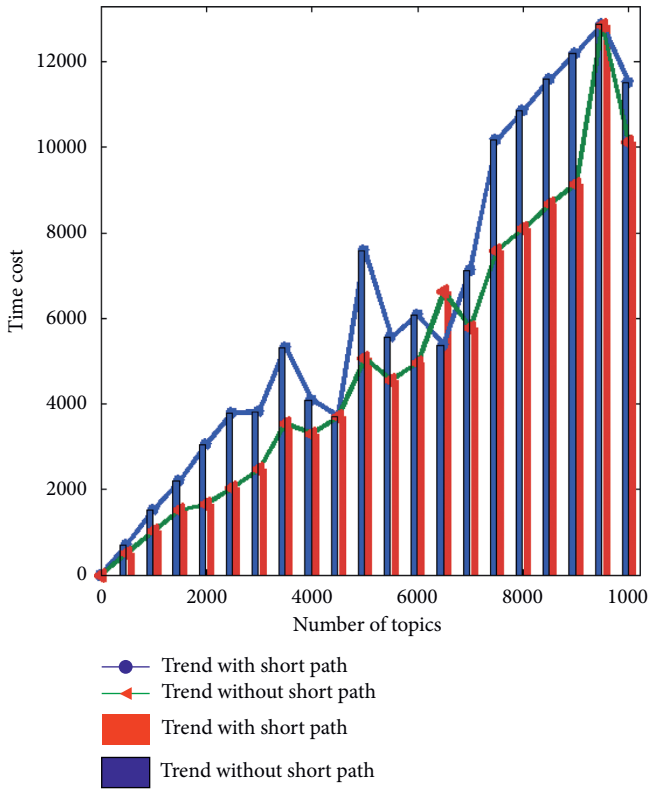


Figure 14: Comparison of data transmission completion time to the data consumer.

path has a significant impact on the network system. We can see from the results that the shortest path has high efficiency.

In order to further verify this view, we compute the average transmission completion time of all the nodes in the path shown in Figure 15 and observe similar trend as in Figure 16.

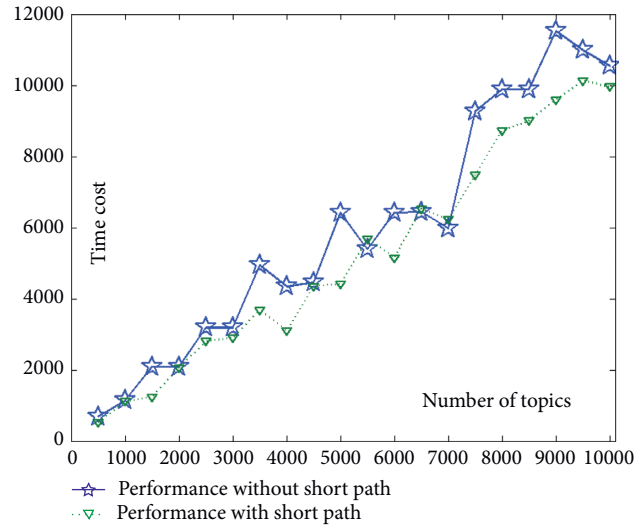


Figure 15: Comparison of the overall performance on two path.

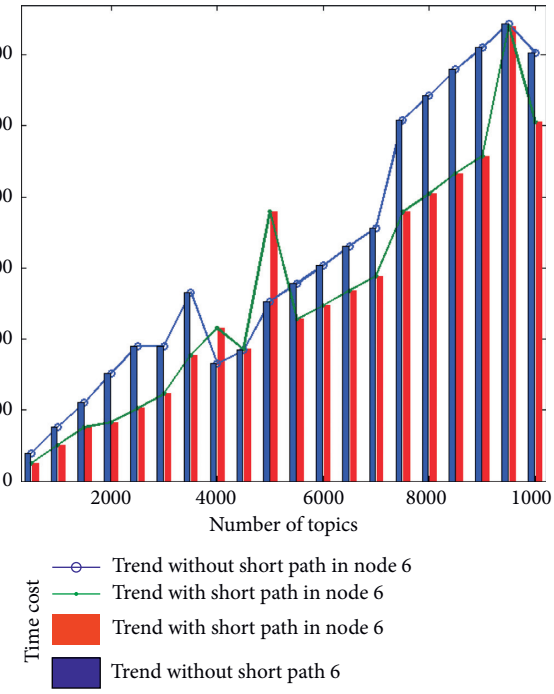


Figure 16: Comparison of data transmission completion time to the edge router nodes.

5. Conclusions

We have presented novel topics-based data routing protocols and algorithms in MQTT. The routing algorithms optimize IoT data transfers with constrained resource and fragile network environments in publish-subscribe model. Our proposed algorithms can adapt the dynamical network topology and are capable of adopting continuously changing devices with rerouting method in IoT system. Through experimental results, we have demonstrated efficiency of our algorithms for message routing, and dynamically adapting the continually changing device and network topology. We

have evaluated our algorithms with a prototype with a Mosquitto based on MQTT protocol. With the experiments performed and simulations on Mosquitto, the results show that our algorithms work well. The proposed algorithms can be widely used in IoT system with publish-subscribe model. Moreover, the algorithms have the capability to be tested in other protocols such as CoAP [25]. From security point of view, identity and authorization of nodes are critical problems [26–30]. Threats such as avoiding data to be published or subscribed by attackers will have adverse effects on the broker in IoT system. In this work, we have focused on the data exchange efficiency in IoT, mainly under the MQTT protocol; in our future work, we aim to study security issues in IoT [31–34].

From security view, identity and authorization of nodes are critical problems avoiding data to be published or subscribed by hacker or crack down the broker in IoT. In this paper, we focus on the data exchange efficiency in IoT, mainly under the MQTT protocol and not involving the security problem, which we will deeply research in future work.

Data Availability

Any data can be obtained from the dropbox url (https://www.dropbox.com/s/vjm06c4z5kbg79i/%E9%87%87%E9%9B%86%E6%95%B0%E6%8D%AE3.txt?dl=0).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors thank Professor Zia Tanveer for his kind review and help. This work was supported in part by National Science Foundation of China under grant nos. 61672104, 61170209, 61702570, and 61602537 and U1509214, Program for New Century Excellent Talents in University under grant no. NCET-13-0676, and Shenzhen Institute of Artificial Intelligence and Robotics for Society.

References

[1] S. Deng, Z. Xiang, P. Zhao et al., "Dynamical resource allocation in edge for trustable internet-of-things systems: a reinforcement learning method," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 9, p. 6103, 2020.

[2] M. Fischer, D. Kümper, and R. Tönjes, "Towards improving the privacy in the MQTT protocol," *GIoTS*, vol. 19, pp. 1–6, 2019.

[3] IoTivity, "IoTivity," 2017, https://www.iotivity.org/.

[4] J. Sathish Kumar and M. A. Zaveri, "Clustering for collaborative processing in IoT network," in *Proceedings of the Second International Conference on IoT in Urban Space (Urb-IoT 16)*, pp. 95–97, ACM, New York, NY, USA, 2016.

[5] CoAP, "Constrained application protocol (CoAP)," 2017, http://coap.technology/.

[6] AMQP, "Advanced message queuing protocol (AMQP)," 2017, https://www.amqp.org/.

[7] WebSocket, "WebSocket," 2017, https://www.websocket.org/index.html.

[8] Y. Zhao, K. Kim, and N. Venkatasubramanian, "DYNATOPS: a dynamic topic-based publish/subscribe architecture," in *Proceedings of the 7th ACM International Conference on Distributed Event-Based Systems (DEBS '13)*, pp. 75–86, ACM, New York, NY, USA, 2013.

[9] M. Stolpe, "The internet of things: opportunities and challenges for distributed data analysis," *ACM SIGKDD Explorations Newsletter*, vol. 18, p. 1, 2016.

[10] T. Luo, H.-P. Tan, and T. Q. S. Quek, "Sensor OpenFlow: enabling software-defined wireless sensor networks," *IEEE Communications Letters*, vol. 16, no. 11, pp. 1896–1899, 2012.

[11] AllSeen Alliance, "Linux foundation," 2017, https://allseenalliance.org/framework/documentation/learn/core/standard-core.

[12] MQTT, "Message queuing telemetry transport (MQTT)," 2017, https://mqtt.org/.

[13] B. Karakostas and N. Bessis, *Intelligent Brokers in an Internet of Things for Logistics*, ACM, New York, NY, USA, 2016.

[14] H. Gao, W. Huang, and Y. Duan, "The cloud-edge based dynamic reconfiguration to service workflow for mobile ecommerce environments: a QoS prediction perspective," *ACM Transactions on Internet Technology*, vol. 36, 2020.

[15] B. Aziz, "A formal model and analysis of an IoT protocol," *Ad Hoc Networks*, vol. 36, p. 49, 2016.

[16] W.-J. ChenRahul, "Responsive mobile user experience using MQTT and IBM MessageSight," *IBM Redbooks*, vol. 3, 2014.

[17] H. Gao, Y. Xu, Y. Yin, W. Zhang, R. Li, and X. Wang, "Context-aware QoS prediction with neural collaborative filtering for internet-of-things services," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4532–4542, 2020.

[18] S. Sakulin, A. Alfimtsev, E. Tipsin, V. Devyatkov, and D. Sokolov, "User interface distribution method based on pi-calculus," *International Journal of Distributed Systems and Technologies*, vol. 10, no. 3, pp. 1–20, 2019.

[19] F. Xiong, A. Li, H. Wang, and L. Tang, "An SDN-MQTT based communication system for battlefield UAV swarms," *IEEE Communications Magazine*, vol. 57, no. 8, pp. 41–47, 2019.

[20] P. Kumar and B. Dezfouli, "Implementation and analysis of QUIC for MQTT," *Computer Networks*, vol. 150, pp. 28–45, 2019.

[21] H. Ziekow, *In-Network Event Processing in a Peer To Peer Broker Network for the Internet of Things*, Springer-Verlag, Berlin, Germany, 2007.

[22] S. D. Madhu Kumar and U. Bellur, "An underlay aware, adaptive overlay for event broker networks," in *Proceedings of the 5th Workshop on Adaptive and Reflective Middleware (ARM '06)*, ACM, New York, NY, USA, 2006.

[23] M. A. Jaeger, H. Parzyjegla, G. Mühl, and K. Herrmann, "Self-organizing broker topologies for publish/subscribe systems," in *Proceedings of the 2007 ACM Symposium on Applied Computing (SAC '07)*, pp. 543–550, ACM, New York, NY, USA, 2007.

[24] G. De Luca and Y. Chen, "Visual IoT/robotics programming language in pi-calculus," *ISADS*, vol. 16, pp. 23–30, 2017.

[25] A. Larmo, A. Ratilainen, and J. Saarinen, "Impact of CoAP and MQTT on NB-IoT system performance," *Sensors*, vol. 19, no. 1, p. 7, 2019.

[26] D. J. Wu, A. Taly, A. Shankar, and D. Boneh, "Privacy, discovery, and authentication for the internet of things," *Computer Security-ESORICS 2016*, vol. 23, pp. 301–319, 2016.

[27] S. P. Mahambre and U. Bellur, "An adaptive approach for ensuring reliability in event based middleware," in

Proceedings of the Second International Conference on Distributed Event-Based Systems (DEBS '08), pp. 157–168, ACM, New York, NY, USA, 2008.

[28] A. J. Hintaw, S. Manickam, S. Manickam, S. Karuppayah, and M. F. Aboalmaaly, "A brief review on MQTT's security issues within the internet of things (IoT)," *Journal of Communications*, vol. 14, no. 6, pp. 463–469, 2019.

[29] H.-Y. Chien, X.-A. Kou, M.-L. Chiang, and C. Su, "Secure and efficient MQTT group communication design," *CSII (Selected Papers)*, vol. 19, pp. 177–186, 2019.

[30] R. Singh Bali, F. Jaafar, and P. Zavarsky, "Lightweight authentication for MQTT to improve the security of IoT communication," *ICCSP*, vol. 19, pp. 6–12, 2019.

[31] H. Gao, C. Liu, Y. Li, and X. Yang, "V2VR: reliable hybrid-network-oriented V2V data transmission and routing considering RSUs and connectivity probability," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, 2020.

[32] K. Mladenov, *Formal Verification of the Implementation of the MQTT Protocol in IoT devices*, Master Thesis, University of Amsterdam, Amsterdam, Netherlands, 2017.

[33] A. P. Haripriya and K. Kulothungan, "Secure-MQTT: an efficient fuzzy logic-based approach to detect DoS attack in MQTT protocol for internet of things," *EURASIP J. Wireless Comm. and Networking*, vol. 2019, p. 90, 2019.

[34] G. Kim, S. Kang, J. Park, and K. Chung, "An MQTT-based context-aware autonomous system in oneM2M architecture," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8519–8528, 2019.

Hindawi
Security and Communication Networks
Volume 2020, Article ID 8830310, 12 pages
https://doi.org/10.1155/2020/8830310

Research Article

Seam-Carved Image Tampering Detection Based on the Cooccurrence of Adjacent LBPs

Dengyong Zhang[1,2] **Xiao Chen**,[1,2] **Feng Li**[1,2] **Arun Kumar Sangaiah**,[3] **and Xiangling Ding**[4]

[1]*College of Computer and Communication Engineering, Changsha University of Science and Technology, Changsha 410114, China*
[2]*Hunan Provincial Key Laboratory of Intelligent Processing of Big Data on Transportation, Changsha University of Science and Technology, Changsha 410114, Hunan, China*
[3]*School of Computing Science and Engineering, Vellore Institute of Technology (VIT), Vellore 632014, India*
[4]*School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan 411004, China*

Correspondence should be addressed to Feng Li; lif@csust.edu.cn

Received 26 August 2020; Revised 11 November 2020; Accepted 11 December 2020; Published 21 December 2020

Academic Editor: Honghao Gao

Seam carving has been widely used in image resizing due to its superior performance in avoiding image distortion and deformation, which can maliciously be used on purpose, such as tampering contents of an image. As a result, seam-carving detection is becoming crucially important to recognize the image authenticity. However, existing methods do not perform well in the accuracy of seam-carving detection especially when the scaling ratio is low. In this paper, we propose an image forensic approach based on the cooccurrence of adjacent local binary patterns (LBPs), which employs LBP to better display texture information. Specifically, a total of 24 energy-based, seam-based, half-seam-based, and noise-based features in the LBP domain are applied to the seam-carving detection. Moreover, the cooccurrence features of adjacent LBPs are combined to highlight the local relationship between LBPs. Besides, SVM after training is adopted for feature classification to determine whether an image is seam-carved or not. Experimental results demonstrate the effectiveness in improving the detection accuracy with respect to different scaling ratios, especially under low scaling ratios.

1. Introduction

As image processing technologies, such as scene graphs prediction [1]and image tampering technology, have steadily developed, the content-aware image retargeting techniques have emerged and have attracted increasing attention. One of these, seam carving [2], is able to avoid image distortion and deformation when applied to an image, creating no obvious difference in visual effect. However, this method can also be used for malicious tampering purposes, for instance, removing specific objects in the image or modifying with the semantic content conveyed by the original image, which creates great obstacles for image forensics tasks. Therefore, seam-carving detection has become an important issue, and designing a method for detecting images that may have been subjected to seam carving is of vital importance.

Over the past years, some approaches have been proposed to aim at the detection of seam carving. Sarkar et al. [3] proposed a forensic method based on the Markov feature, which exploits the 324-dimensional Markov feature for classification in the block-based Discrete Cosine Transform domain. However, it achieves a less-than-ideal detection accuracy with a small scaling ratio. As described in [4], Lu and Wu proposed a detection method based on forensic hash to research the problems of seam-carving estimation and tampering localization. However, a forensic hash must be built ahead of time, so it is active; furthermore, it can be detected so easily that counterfeiters can remove it in some

ways. Wei et al. [5] proposed dividing the image into small squares of size 2×2 and looking for the patch that could possibly recover the small squares from seam carving. This method eliminates the patch transition probability among the three small connected squares. Yin et al. [6] first used local binary pattern (LBP) to preprocess the image data and then defined six new features based on the half-seam to reveal the energy change in half of an image; subsequently, these were combined with the existing 18 energy features, after which those features were finally classified using support vector machine (SVM). However, this method is still not ideal for image detection accuracy with low scaling ratios. Wattanachote and Shih [7] proposed a forensic approach based on Blocking Artifact Characteristics Matrix (BACM). In an original JPEG image, regular symmetrical data are presented in a BACM block matrix. Following seam carving, these symmetrical data are reconstructed so that their symmetry is destroyed. From this, 22 features are proposed and a high recognition rate is obtained. However, this method can be easily affected by the quality factor (QF). Ke et al. [8] put forward a forensic method based on the additional seam-carving operation. First, an additional seam-carving operation is performed on the testing image. The approach then extracts 11-dimensional features by calculating the similarity, energy relative error, and seam distance difference between an image and the version of the image undergoing seam carving; this facilitates identifying whether the image has been tampered using seam carving or not. It can be observed that this method has wide applicability. Ye and Shi [9] proposed a method that incorporates a local derivative pattern, Markov transition probability, and the subtractive pixel adjacency model. They also utilized recursive feature elimination based on the linear support vector machine to reduce the feature dimensionality, which greatly improves detection accuracy. Liu et al. [10] combined calibrated adjacent joint density with a rich model-based method originally used for steganalysis and exploited the feature selection algorithm to reduce the feature dimensionality of the combined feature set by using a smaller and more optimized feature set, enabling further improvements on the detection accuracy for the forensic task. Subsequently, as discussed in [11], Liu developed a hybrid large feature mining-based method. As there are many types of large features, ensemble learning is utilized to process high-dimensional features, effectively solving the problem of differentiating between seam-carved and untouched JPEG images. However, a lot of work is involved. Han et al. [12] further proposed a blind detection method based on the block artifact grid (BAG) mispairing characteristic, which firstly extracts BAG from a JPEG image and then constructs 10-dimensional features of the BAG chart. Clustering technology is thus used to extract these features and obtain the classification results. This method is not only able to detect whether the image has been seam-carved or not but also can go a step further and locate the object removed from an image by seam carving. Cieslak et al. [13] proposed a forensic approach based on the combination of convolutional neural networks (CNNs) and local binary pattern (LBP). This method first transforms images to the LBP domain and then inputs them to the CNN in order to determine whether or not these images have undergone a seam-carving operation. The experimental results reach a detection accuracy of more than 81%. Similarly, Ye et al. [14] proposed a CNN-based deep learning architecture. This approach makes use of the joint optimization of feature extraction and pattern classification for employing more effective features, thereby improving the classification speed and accuracy. Zhang et al. [15] proposed a detection approach based on uniform local binary patterns (ULBP). In this approach, ULBP is exploited to decrease the species of binary patterns without losing any information, thereby reducing the dimensionality of features and mitigating the effects of high-frequency noise. Lu and Niu [16] combined the histogram features of the local neighborhood magnitude occurrence pattern (LNMOP) with a histogram of oriented gradient (HOG) and selected the final features for the classifier from the extracted LNMOP features. However, this approach does not consider the postprocessing of tampered images.

A large number of approaches exist for the detection of seam-carved images, which has strongly promoted the development of the seam-carving forensics task. However, even if some images (such as higher-resolution biopsy slice images [17] with many details) are rarely tampered under low scaling ratio conditions, the content expressed may be changed. Thus, some scope for further improving the detection accuracy still exists. As we know, in an image that undergoes seam carving, the minimum cumulative energy seams defined by the energy function are removed, resulting in the change of not only the energy but also the local texture. Most existing methods employing this characteristic directly use the local binary pattern (LBP) features of the image for forensic detection; however, these methods cannot completely represent the change of local texture information.

Accordingly, we introduce the cooccurrence of adjacent LBPs into the forensic task. Cooccurrence is defined as the simultaneous occurrence of all forms of adjacent LBPs, which are generally used to extract information that is bound up with the global structure in various characteristics based on local region; as described in [18], we obtain this information through the use of autocorrelation matrices calculated from two adjacent LBPs. The advantage of this feature is that it comprises both the original LBPs and the cooccurrence of adjacent LBPs; therefore, the location relationship of adjacent LBP values can be more explicitly determined. In this paper, the cooccurrence feature is combined with energy bias and noise-based features, as well as half-seam-based features, for feature extraction purposes. Subsequently, a trained support vector machine (SVM) classifier is used to determine whether seam carving has been applied to the image or not. Our experimental results prove that this forensic method achieves superior detection performance relative to existing methods. The contributions of the proposed approach are threefold: (1) we analyse the distortion that affects image resizing by seam carving and conclude that the relationship between adjacent pixels is an important clue for the forensics of seam-carving operation;

inspired by [18], we extract the cooccurrence feature of adjacent LBPs; (2) we combined the cooccurrence feature of adjacent LBPs with the existent energy features to form a new feature set; (3) we apply PCA to reduce the dimensionality of the features to get an identified feature set; a series of experiments have verified that the proposed method has higher detection accuracy in identifying whether the image has undergone seam carving or not. Moreover, the experimental results demonstrate that the proposed approach has better robustness for TIFF and JPEG images.

The remainder of this paper is organized as follows. Section 2 briefly introduces the theory of the seam-carving algorithm. Section 3 introduces some related backgrounds and discusses the seam-carving detection method based on the cooccurrence of adjacent LBPs. The experimental results are reported in Section 4. Finally, Section 5 summarizes the paper.

2. Seam Carving

The seam-carving algorithm preserves important areas of the image and performs operations on unimportant areas. Because of the smoothness of unimportant areas in an image, it is not easy to cause visual perception if these areas are modified. As discussed in [2], the implementation of the seam-carving algorithm assigns different energy values to each pixel of the image on the basis of the importance of the pixels in said image and then calculates the parts that can be removed according to these energy values. The type of algorithm can bypass the limitations of traditional image processing technology, such as excessive deleting deformation [19] or distortion caused by image scaling, which is the reason why it can be so widely utilized.

A seam is an eight-connected path, consisting of a set of connected pixels that pass through the image from top to bottom or from left to right. The least important parts, namely, the lowest energy pixel paths (called "optimal seams" in the image), are deleted as possible by using a seam-carving algorithm to reduce the image scale and achieve the desired width and height, while retaining the important content of the image in vision. In the calculation of the seam, the energy function of each path is defined as follows:

$$e(I) = \left|\frac{\partial}{\partial x} I\right| + \left|\frac{\partial}{\partial y} I\right|, \tag{1}$$

where I is an image of $n \times m$. There are two types of seams, namely, vertical seam and horizontal seam. Taking the vertical seam as an example, its definition is as provided in equation (2), while the definition of the horizontal seam is similar:

$$\mathbf{S} = \{s_i\}_{i=1}^{n} = \{i, p(i)\}_{i=1}^{n}, \quad \text{s.t. } \forall i, |p(i) - p(i-1)| \le 1, \tag{2}$$

where i denotes the row coordinates and $p(i)$ denotes the corresponding column coordinates of pixels. On the basis of the above formula, the pixel value set of a seam S can be obtained as follows:

$$\mathbf{I}_s = \{I(s_i)\}_{i=1}^{n} = \{I(i, p(i))\}_{i=1}^{n}. \tag{3}$$

It should be noted that, for a vertical seam, there is only one pixel in each row of the seam. In light of the energy function $e(I)$ provided above, the optimal seam that minimizes this seam cost is defined as follows:

$$s^* = \min_s E(s) = \min \sum_{i=1}^{n} e(I(s_i)). \tag{4}$$

Dynamic programming is exploited to select the optimal seam, after which the optimal seam is continuously eliminated in order to reduce the image size or remove the target object. Taking the vertical seam as an example, the step traverses the image from the second row to the last row and computes the possible minimum energy matrix $M(i, j)$ for each pixel (i, j).

$$M(i, j) = e(i, j) + \min(M(i-1, j-1), M(i-1, j), M(i-1, j+1)). \tag{5}$$

Once the cumulative minimum energy M has been constructed for all possible seams, the optimal seams are found continuously by backtracking from the minimum value of the last row of M. The image content that has undergone seam carving is visually stable. Any possible visual artifacts appear only near the removed seams, while the remainder of the image remains unchanged. This is because, when an optimal seam is removed, the pixel paths at the right of the seam will move to the left to compensate for the missing pixel path. Generally speaking, it is difficult to visually detect such weak changes.

An example of seam carving is presented in Figure 1. Here, we perform seam carving on an image; the vertical seams to be eliminated are displayed and labeled in red. By removing these vertical seams, the width of the image is reduced by 15%. As can be seen from Figure 1(c), there is no obvious deformation or distortion of the refrigerator, and the most important content remains stable. This is because the smooth area in the image is mainly removed when seam carving is performed, which makes the change of image unable to be detected visually. Therefore, we study texture information of the image and then introduce the cooccurrence of adjacent LBPs and combine it with the energy features to detect this visually imperceptible tampering behavior.

3. Proposed Method

The energy feature can be used to describe the inherent characteristics of the image. In the process of seam carving, the lowest energy seams as defined by the energy function in the image are removed, bringing about changes in energy information of the image. At the same time, local texture change is important information for the operation of seam carving, which reflects the characteristics of the image surface; more specifically, it depicts the repeated local patterns and their arrangement rules in the image and additionally exhibits rotation invariance as well as excellent noise resistance performance. This combination of energy features and local texture features enables image information to be more comprehensively embodied, meaning that the

(a) (b) (c)

Figure 1: An image resized by seam carving. (a) An original image. (b) The image with vertical seams. (c) The seam-carved image.

proposed method can achieve higher detection accuracy. Therefore, in this paper, we briefly extract the energy features and local texture features related to the global structure in the LBP domain, where LBP is used to report the change of local texture.

There are several steps involved in our proposed approach. First, the LBP values of the suspicious image are calculated with the goal of transforming the image to the LBP domain. Second, we extract the above-mentioned features in the LBP domain involving energy-based features, seam-based features, noise-based features, half-seam-based features, and the cooccurrence feature of adjacent LBPs. Among those features, as Ryu and Lee's paper has introduced, energy-based features include average energy, average row energy, average column energy, and average energy difference, for a total of four features. The seam-based features refer to the minimum value, maximum value, mean value, standard deviation, and the difference between the maximum value and the minimum value in both directions (i.e., for both column and row directions) following the construction of the cumulative minimum energy matrix M for all possible seams. There are 10 features in total. In order to remove the noise from the candidate image I, the image is filtered using a Wiener filter F with windows; moreover, the noise is computed by $n = I - F(I)$, from which the average value, standard deviation, skewness, and kurtosis based on the noise can be calculated. As described in Yin's paper, half-seam-based features are defined with reference to the half seams rather than the whole seams of the image; in other words, these are calculated according to the accumulated minimum energy matrix M of half of the image. Subsequently, three statistics (called min, max, and mean) are calculated for the specified column and row, for a total of six features. Based on the 24 features obtained at the end of this process, we propose to apply the cooccurrence feature of adjacent LBPs to seam-carving detection. As a result, we obtain a grand total of 25 LBP-based features totally. Following feature extraction, the SVM classifier is adopted for classification task; because of its supervised nature, it needs to be trained before the testing stage. In the final classification phase, we input the features extracted from the image under study into the trained SVM classifier. Finally, we obtain the detection results under different scaling ratios and compare them with the data derived through other methods. Figure 2 presents the general framework of our proposed method.

3.1. Local Binary Pattern (LBP). As described in [20], LBP is a kind of operator applied to describe the local texture characteristics of an image. It is defined as having a window size of 3×3, with the center pixel value of the window used as the threshold value. Compared with the adjacent 8 pixel values, if the adjacent pixel value is bigger than the center pixel value, the position of this adjacent pixel is represented by 1; if not, it is represented by 0. The formula can be expressed as follows:

$$\begin{aligned} \mathrm{LBP}(x_c, y_c) &= \sum_{p=1}^{8} s(I(p) - I(c)) * 2^p, \\ s(x) &= \begin{cases} 1, & x \geq 0, \\ 0, & \text{otherwise.} \end{cases} \end{aligned} \tag{6}$$

Here, $I(c)$ is the center pixel value at location (x_c, y_c) of the image, $I(p)$ denotes the value of its adjacent pixel, and p indicates the total number of adjacent pixels.

In this way, we can obtain an 8-bit binary number, which is generally converted into a decimal number. This value then is retained as the LBP value of the window center pixel in order to express the texture information of the 3×3

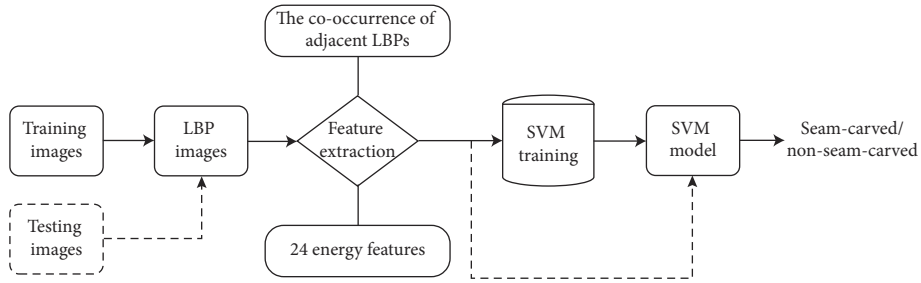


Figure 2: The framework of the proposed method.

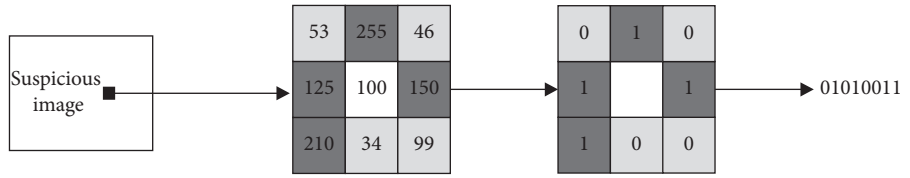


Figure 3: An example of a basic LBP operator.

region. By recursively calculating the LBP value of each pixel, the input image can be transformed into LBP domain. Figure 3 presents an example of a basic LBP operator. It can be seen from the figure that the LBP value of the center pixel is 83.

3.2. The Co-occurrence of Adjacent LBPs. In this paper, the cooccurrence characteristic of adjacent LBPs is applied to the detection of image seam-carving operation. Figure 4 illustrates the difference between the LBP histogram and the histogram of special cooccurrence of adjacent LBPs.

Figure 4(a) displays two example images composed of two LBP patterns (X and Y); here, the numbers of Xs and Ys in each image are the same. The gray and white surrounding squares represent the values of adjacent pixels that are larger and smaller than the center pixel value, respectively. As can be seen in Figure 4(b), when the numbers of different LBP values in two disparate images are identical, the same results are obtained utilizing the original LBP histograms. However, the histograms of the cooccurrence of adjacent LBPs extracted from the two images are different, as shown in Figure 4(c). In this case, it is clear that the cooccurrence of adjacent LBPs can more efficiently give expression to the local texture characteristics of an image.

In terms of the procedure for extracting the cooccurrence feature of adjacent LBPs, the encoding method proposed in [18] is exploited. When calculating the LBP value of an image (as shown in Figure 5), there are two sparser configurations of LBPs, including four pixels in the diagonal direction and four pixels in the cross direction of eight adjacent pixels; one of these is selected to reduce the computational complexity, enabling all possible LBP values to be obtained with a total number of N ($N \leq 16$).

There are four location cases from each reference LBP to its neighbor LBP, as shown in Figure 6. Thereby, the configurations patterns of adjacent LBPs are combined in these four ways to facilitate continuously calculating the autocorrelation matrices of adjacent LBPs.

The autocorrelation matrix $C(k)$ is calculated continuously according to these four patterns, which can be expressed as

$$C_{i,j}(k) = \sum_{a \in I} f_i(a) f_j(a+k)^T,$$
$$f_i(a) = \begin{cases} 1, & \text{if LBP}(a) \\ 0, & \text{otherwise,} \end{cases} \tag{7}$$

where a is the position of every pixel in the image. k is the distance from the reference LBP to its adjacent LBP. i and j are two LBP values. After calculating all the autocorrelation matrices, eventually, these matrices are vectorized and combined to create $4N^2$ dimensional features.

In order to achieve better detection performance, we utilize the Principal Component Analysis (PCA) [21] method, the purpose of which is to extract important information from a data table and represent this information as a new set of orthogonal variables, called the principal component. This approach is exploited to reduce the dimensionality of the high-dimensional data, allowing the introduced feature to be better combined with other features (Algorithm 1).

3.3. SVM Classifier. After extracting the twenty-five features, a classifier is required to determine whether the image has undergone seam carving or not. As we know, support vector machine (SVM) can be utilized not only in medical scenarios such as prediction of the extubation failure [22] but also in forensic tasks such as image security.

In this paper, first, we adopt the downloaded LibSVM [23] in MATLAB and svm-scale function to normalize the experimental data (such as features and labels) to be trained. The RBF kernel function is then selected. Grid search and 5-fold cross validation is exploited for parameter optimization selection in order to choose the best parameters c (the penalty factor) and g (parameter of gamma function), which are selected to train the entire training sets using SVM-train

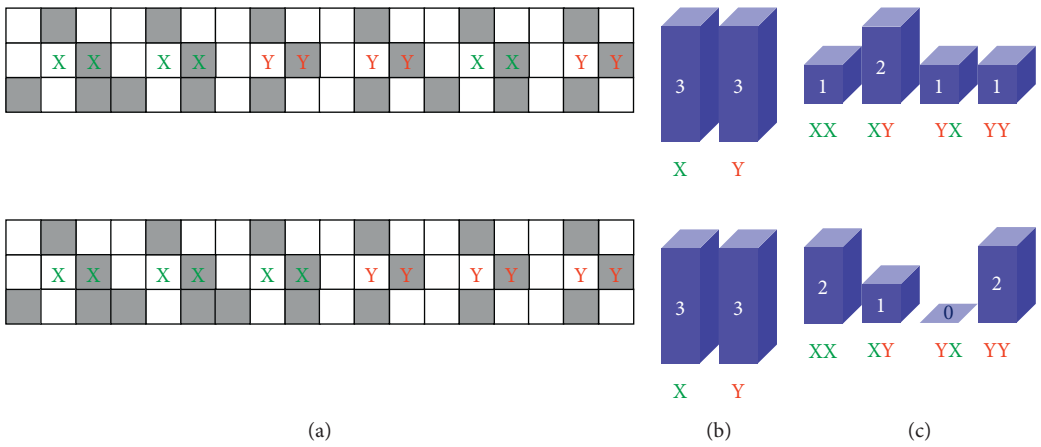


Figure 4: The comparison of LBP histogram and histogram of the cooccurrence of adjacent LBPs. (a) Two images where the center pixel LBP values are X, Y. (b) LBP histograms of the two images. (c) Histograms of the cooccurrence of adjacent LBPs of the two images.

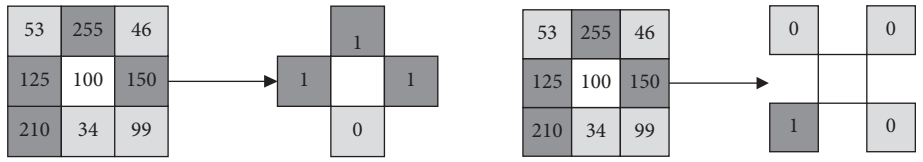


Figure 5: Two sparser configurations of LBPs. (a) LBP (+). (b) LBP (×).

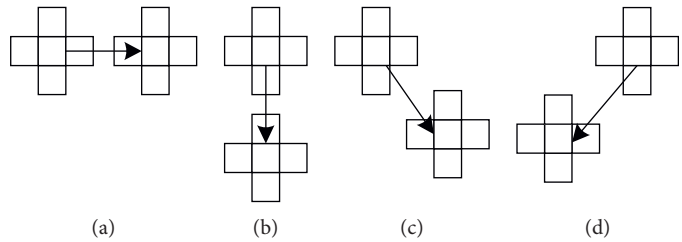


Figure 6: The configuration patterns of the cooccurrence of adjacent LBPs.

function to obtain a SVM model. Finally, according to the SVM model, the svm-predict function is used to detect image under investigation, while the confusion matrix is used to determine the classification result and the classifier accuracy.

4. Experimental Results and Discussion

The hardware configuration used for experiments is a personal computer (Intel(R) Core(TM) i5-7400 CPU @ 3.00 GHz 8.00 GB RAM). We employ MATLAB 2016b as the experimental tool. Moreover, LibSVM [23] is used for the classification task. Python and Gnuplot are used for the parameter optimization.

In order to prove the robustness of our proposed method in different format images, experiments are carried out on TIFF and JPEG images. In addition, our method focuses on the detection of low scaling ratio images and strives to improve their detection accuracy.

As there is currently no complete image database for seam-carving forensics available, the Uncompressed Color Image Database (UCID) is utilized here due to its abundant content (e.g., people, plants, goods, etc.). We obtain 1338 original images without compression from the UCID image set and then implement seam-carving technology to resize these images in the following two cases: (1) after compressing images, under the condition that the quality factor (QF) is 10, 20, 50, 75, and 100; (2) exploiting uncompressed TIFF format images. In the former cases, the scaling ratios used for seam carving are 1%, 2%, 5%, 10% , 20%, 30%, and 50%. For example, consider an image with a size of 512×384 where the scaling ratio is 10% in the vertical direction; this means that the width is reduced by 10%, resulting in the image size changing to be 461×384. Thus, 1338 compressed images with QF of 10, 20, 50, 75, and 100 at scaling ratios of 1%, 2%, 5%, 10%, 20%, 30%, and 50% can be obtained, for a total of 46,830 images; there are also 1338

uncompressed images at scaling ratios of 1%, 2%, 5%, 10%, 20%, 30%, and 50%, respectively, for a total of 9366 images.

Therefore, based on the adding of the original images, we divide the entire image set into identical sized training and testing sets. For compressed image sets at every QF and for uncompressed image sets, we have several training sets, the scaling ratios of which are 1%, 2%, 5%, 10%, 20%, 30%, and 50%. In addition, we have several testing sets at scaling ratios of 1%, 2%, 5%, 10%, 20%, 30%, and 50%. Furthermore, we also extract images from the uncompressed image sets with scaling ratios of 10%, 20%, 30%, and 50% to create a mixed set. The number of images in different scaling ratio is equal, and then we divide these sets equally into training and testing sets. Since the UCID image set is divided equally, the number of images in each subset is the same. To summarize, in the training and the testing sets, each subset contains 669 images with a specific scaling ratio. During the experiment, we firstly extract the cooccurrence feature of adjacent LBPs and twenty-four energy features of the images under investigation. We then use the SVM classifier after training to complete the classification tasks of these features, inferring how many images have been tampered (while the others have not been tampered with) under each subset of different scaling ratios, and then use these experimental data to evaluate the detection performance of our proposed approach in different situations and to compare it with other existing methods.

Generally speaking, the quality factor (QF) has a certain influence on the detection operation. However, it can also be seen from Table 1 that, at the condition of each specific QF, the method proposed in this paper can achieve extremely high accuracies for images with different scaling ratios; specifically, all images with large scaling ratios can be correctly detected. Furthermore, this also demonstrates the wide applicability of this method under different QFs (i.e., the method is stable and less affected by quality factors). When the value of QF is 0, meaning that the images are not compressed, it can be observed that this approach is also suitable and can achieve considerable accuracy.

In order to demonstrate more clearly that the combination of the cooccurrence features of adjacent LBPs and 24 energy features yields the best experimental results, we also carried out a comparison experiment in which the cooccurrence features were added separately for the detection without the 24 energy features. EFCOFAL is used to describe the combination features of the cooccurrence feature of adjacent LBPs and 24 energy features, and COFAL is adopted to describe the cooccurrence features of adjacent LBPs.

As can be seen from Table 2, the effect of using the cooccurrence features of adjacent LBPs alone is not ideal compared with the effect when it they are combined with the 24 energy features, especially when scaling ratios are low.

Table 3 summarizes the comparison results of the six forensic methods. As the image scaling ratio increases, the detection accuracy of the six approaches evidently also increases as across the board. In general, the proposed method achieves the best accuracy, which is higher on average by 35.74%, 30.75%, 13.27%, and 9.87% than the other four methods; particularly under small scaling ratios of 1%, 2%, and 5%, this method achieves more outstanding performance. Compared with the method proposed by Ye et al., it can be seen that the average accuracy of the proposed method is 0.25% higher when the scaling ratios are among 5% and 50%.

The method proposed by Wei et al. does not consider the alternation of the internal nature (such as energy change) in an image caused by seam carving. Ryu and Lee [24] proposed a method based on energy bias and noise features in the LBP domain. Compared with the former method, this latter method takes advantage of the inherent change in the characteristics of the image following seam carving. On this basis, Yin et al. added six half-seam-based features that more comprehensively reflect the change of energy and local texture. However, the energy distribution changes when seams are inserted in a seam-carved image, which can offset the alternation of energy distribution caused by seam carving and make the forensic task more difficult. The change of local texture is also unable to reflect the location relationship of the adjacent LBP values in the local region, leading to the detection accuracy not being ideal when the scaling ratio is low. The method proposed by Wattanachote et al. involves displaying regular symmetrical data in the original JPEG image in the Blocking Artifact Characteristics Matrix (BACM), while the symmetrical data in the block reconstructed by the seam carving are destroyed. Accordingly, 22 features are proposed and used for the feature classification, and considerable accuracy is obtained as a result. However, due to the influence of the quality factor (QF), the accuracy exhibits a large wave motion, which is also (in brief) the reason why the performance of this method is not inadequate enough. Ye et al. proposed a deep-learning-based method; essentially, the CNN is exploited, after which more effective features are used to substantially boost the classification rates and obtain high detection accuracy.

In essence, the method proposed in this paper, which is based on the local texture features and energy features of an image in the LBP domain, introduces the cooccurrence feature of adjacent LBPs, which can reflect the location relationship information of the LBP values corresponding to adjacent pixels in an image in order to improve the detection performance and thereby reduce the detection difficulty caused by the variety of changes to energy features. The experimental results reveal that, in fact, our proposed approach also achieves better detection accuracy.

The receiver operating characteristic (ROC) curves for the four approaches are plotted in Figure 7. The subfigures, that is, Figures 7(a)–7(f), represent the ROC performance under different scaling ratios from 1% to 50%, respectively. It can be observed that the corresponding area under ROC curves (AUC) obtained by the proposed method is significantly larger than the other three methods, whether the scaling ratio is large or small, demonstrating that our method can achieve higher accuracy and confirming its robustness.

Input: an image, I
Output: co-occurrence feature of adjacent LBPs of the image, CoALBP
(1) Transform I into LBP domain based on Figure 5
(2) Define four configurations of adjacent LBPs shown in Figure 6
(3) Calculate the auto-correlation matrices based on formula (7)
(4) Vectorize those matrices and obtain the CoALBP feature
(5) Reduce the dimensionality of the feature using PCA

Algorithm 1: Extraction of cooccurrence feature of adjacent LBPs.

Table 1: Detection accuracy of different scaling ratios under different QF.

Scaling ratio (%)	Quality factor (QF)					
	0 (%)	10 (%)	20 (%)	50 (%)	75 (%)	100 (%)
1	91.26	92.52	94.96	93.26	96.14	92.26
2	92.10	93.55	94.25	95.95	97.95	93.80
5	94.33	95.31	97.31	97.01	97.01	97.01
10	96.11	96.85	96.85	99.70	97.80	99.70
20	98.83	98.95	99.10	98.65	98.65	98.80
30	99.94	100	100	99.85	99.85	100
50	100	100	100	100	100	100

Table 2: Comparison of the accuracy of EFCOFAL and COFAL.

Scaling ratios (%)	Accuracy (%)	
	EFCOFAL	COFAL
1	91.26	81.54
2	92.10	85.63
5	94.33	91.55
10	96.11	94.14
20	98.83	98.30
30	99.94	99.24
50	100	99.98

In addition, we also exploit the cross experiment to test the effect of our proposed approach alongside the other four methods on different training and testing sets. The cross experiment image sets are conducted with five different scaling ratios, specifically 10%, 20%, 30%, 50%, and a mixed scaling ratio, where the images with scaling ratios of 10%, 20%, 30%, and 50% are uniformly distributed in the mixed image set.

The detection accuracies under various experimental situations are listed in Table 4, which displays the accuracy of cross experiment obtained using the UCID database. Table 4(a) presents the results of Wei et al.; the method achieves higher accuracy in the detection of mixed sets than the other three methods but not our proposed method. Table 4(b) demonstrates the results obtained by Ryu and Lee; it can be seen that when the mixed set is used as test set and training set, the experimental accuracy is generally not high. Table 4(c) lists the results of Yin et al., and Table 4(d) presents the results obtained by Wattanachote et al.; neither of the two methods has good robustness. Finally, Table 4(e) shows the results of our proposed approach. These results indicate that, overall, the detection accuracy of our proposed method is higher than that of the other four methods. Furthermore, our proposed method is more robust than the other four methods in the cross experiments.

Besides, according to Section 3.2, we know that even though the dimensionality of the cooccurrence feature has been reduced in order to reduce the computational complexity in the process of generating the cooccurrence feature of adjacent LBPs, the feature dimensionality is still too large compared with other energy features. As shown in Table 5, when we use the feature directly without further dimension reduction, the detection accuracy is very low when testing the image with low scaling ratios.

In this paper, we propose an image forensic approach based on the cooccurrence of adjacent LBPs, which can effectively report the location relationship of adjacent LBP values. From the above experimental results, it is evident that

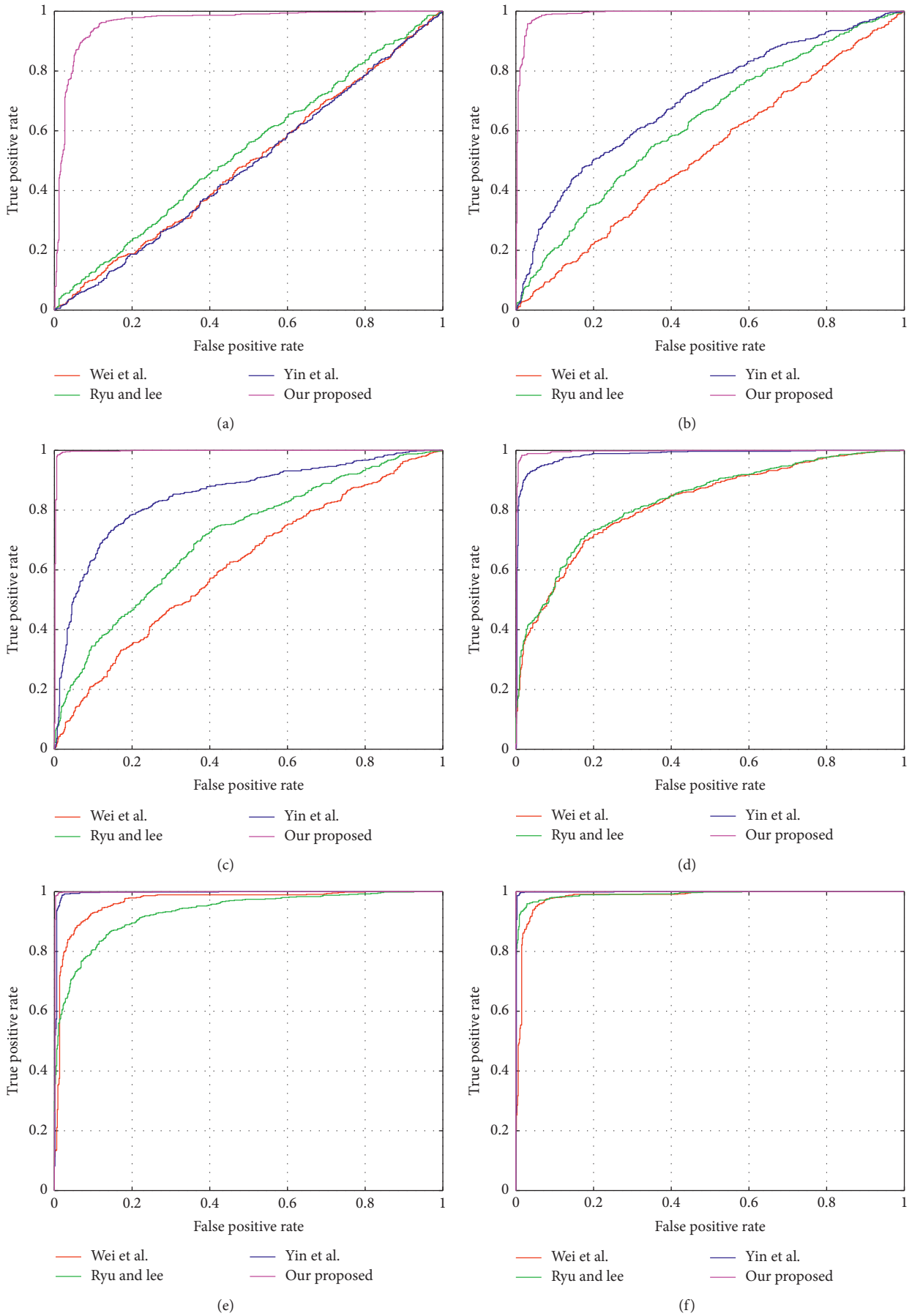


Figure 7: Comparison of ROC curves among the proposed method and three existing approaches. (a) ROC curve for 1% carved image. (b) ROC curve for 5% carved image. (c) ROC curve for 10% carved image. (d) ROC curve for 20% carved image. (e) ROC curve for 30% carved image. (f) ROC curve for 50% carved image.

Table 3: Comparison of the accuracy of seam-carving detection with different scaling ratios.

Scaling ratios	Accuracy (%)					
	Wei et al. [5]	Ryu and Lee [24]	Yin et al. [6]	Wattanachote et al. [7]	Ye et al. [14]	Ours
1%	50.07	52.39	51.12	87.59	—	91.26
2%	50.97	52.77	52.47	87.86	—	92.10
5%	50.20	58.30	63.83	88.54	93.99	94.33
10%	57.91	65.22	80.00	89.50	96.71	96.11
20%	74.18	75.37	94.48	89.70	98.55	98.83
30%	91.34	85.52	98.66	90.13	99.08	99.94
50%	94.93	96.27	99.85	94.90	99.60	100
Average	63.88	68.87	86.35	89.75	97.59	96.08

Table 4: The detection accuracies under cross experimental situations of different methods.

Test	Train				
	10%	20%	30%	50%	Mixed
(a) Wei et al. [5]					
10%	—	64.86	63.75	61.58	72.47
20%	65.47	—	82.48	62.37	80.67
30%	61.65	92.03	—	71.92	91.92
50%	72.14	94.46	94.83	—	94.35
Mixed	66.21	84.99	83.15	72.57	84.85
(b) Ryu and Lee [24]					
10%	—	62.18	62.93	62.48	59.49
20%	72.27	—	69.51	61.06	67.64
30%	79.15	79.75	—	72.94	77.73
50%	85.21	88.42	76.01	—	90.21
Mixed	62.03	61.88	60.62	59.49	62.33
(c) Yin et al. [6]					
10%	—	60.24	56.05	51.64	63.83
20%	83.56	—	75.04	56.20	87.52
30%	92.60	92.23	—	84.68	86.47
50%	99.03	99.48	99.48	—	98.51
Mixed	69.28	67.94	66.52	59.04	76.01
(d) Wattanachote et al. [7]					
10%	—	55.19	54.67	49.63	54.19
20%	58.52	—	64.05	56.95	63.45
30%	53.70	70.96	—	66.18	72.35
50%	38.15	76.94	82.81	—	75.97
Mixed	52.69	67.00	68.83	65.88	69.74
(e) Ours					
10%	—	85.28	66.82	66.37	85.43
20%	99.78	—	80.34	89.39	87.59
30%	99.85	98.03	—	98.32	96.71
50%	100	100	100	—	99.48
Mixed	72.80	70.93	68.54	62.33	85.13

Table 5: Detection accuracy of different scaling ratios using features without dimension reduction.

Scaling ratios (%)	1	2	5	10	20	30	50
Accuracy (%)	29.45	77.73	89.54	93.42	95.67	95.07	100

our method achieves good detection accuracy under different scaling ratios. When the scaling ratio is high, the detection accuracy of our method is almost 100%. Meanwhile, when the scaling ratio is low, it also achieves higher accuracy than other methods; this is of great significance for those images that have been tampered in a nonobvious way. The detection results of TIFF and JPEG images also show that our proposed approach is robust.

5. Conclusions and Future Work

Seam carving is widely utilized due to its ability to protect the important areas of an image from a visual perspective, meaning that the pivotal contents of the image are not distorted or deformed. Moreover, this technology may also be utilized maliciously, which can result in change to the semantic contents of the image; however, this situation may not be perceivable by the naked eye, meaning that it is more likely that people will be misled or that some harmful behaviors will occur, which endanger society. Therefore, despite the challenges, it is necessary to develop and improve seam-carving detection research. In this paper, a forensic method designed for the seam-carving detection task and based on the cooccurrence of adjacent LBPs is proposed. Experimental results demonstrate that our method has better detection performance and good robustness under different QF values and scaling ratios. This is of great significance for forensic work in the field of image security. However, the proposed approach only detects whether or not the image has been seam-carved and cannot determine the specific place at which seam carving has occurred within the image. In the future, we will continue to research location detection [25] of the seam-carved image. Moreover, many video/image processing methods [26–31] will be adopted to extract the identified features. We will also try to apply deep learning [32–37] methods to identify whether the image is seam-carved or not.

Data Availability

The software code and data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

All authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This project was supported by the National Natural Science Foundation of China under Grant nos. 61972057 and 62072055, Hunan Provincial Natural Science Foundation of China under Grant no. 2020JJ4626, Scientific Research Fund of Hunan Provincial Education Department of China under Grant no. 19B004, "Double First-class" International Cooperation and Development Scientific Research Project of Changsha University of Science and Technology under Grant no. 2018IC25, and the Young Teacher Growth Plan Project of Changsha University of Science and Technology under Grant no. 2019QJCZ076.

References

[1] W. Gao, Y. Zhu, W. Zhang, K. Zhang, and H. Gao, "A hierarchical recurrent approach to predict scene graphs from a visual-attention-oriented perspective," *Computational Intelligence*, vol. 35, no. 3, pp. 496–516, 2019.

[2] S. Avidan and A. Shamir, "Seam carving for content-aware image resizing," *ACM Transaction on Graphics*, vol. 26, no. 3, 2007.

[3] A. Sarkar, L. Nataraj, and B. Manjunath, "Detection of seam carving and localization of seam insertions in digital images," in *Proceedings of the 11th ACM Workshop on Multimedia and Security*, pp. 107–116, Berkeley, CA, USA, 2009.

[4] W. Lu and M. Wu, "Seam carving estimation using forensic hash," in *Proceedings of the Thirteenth ACM Multimedia Workshop on Multimedia and Security*, pp. 9–14, San Francisco, CA, USA, 2011.

[5] J.-D. Wei, Y.-J. Lin, and Y.-J. Wu, "A patch analysis method to detect seam carved images," *Pattern Recognition Letters*, vol. 36, pp. 100–106, 2014.

[6] T. Yin, G. Yang, L. Li, D. Zhang, and X. Sun, "Detecting seam carving based image resizing using local binary patterns," *Computers & Security*, vol. 55, pp. 130–141, 2015.

[7] K. Wattanachote and T. K. Shih, "Tamper detection of JPEG image due to seam modifications," *IEEE Transaction Information Forensics and Security*, vol. 10, 2015.

[8] Y. Ke, Q. Q. Shan, F. Qin et al., "Detection of seam carved image based on additional seam carving behavior," *Image Processing and Pattern Recognition*, vol. 9, no. 2, pp. 167–178, 2016.

[9] J. Ye and Y.-Q. Shi, "An effective method to detect seam carving," *Journal of Information Security and Applications*, vol. 35, pp. 13–22, 2017.

[10] Q. Liu, P. A. Cooper, and B. Zhou, *An Improved Approach to Detecting Content-Aware Scaling-Based Tampering in Jpeg Images*, IEEE ChinaSIP, Beijing, China, 2013.

[11] Q. Liu, "An improved approach to exposing JPEG seam carving under recompression," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 7, 2019.

[12] R. Han, Y. Ke, L. Du, F. Qin, and J. Guo, "Exploring the location of object deleted by seam-carving," *Expert Systems with Applications*, vol. 95, pp. 162–171, 2018.

[13] L. F. S. Cieslak, K. A.P. Da Costa, and J. P. Papa, "Seam carving detection using convolutional neural networks," in *Proceedings of the IEEE SACI 2018*, Timisoara, Romania, 2018.

[14] J. Ye, Y. Shi, G. Xu et al., "A convolutional neural network based seam carving detection scheme for uncompressed digital images," *Lecture Notes in Computer Science*, pp. 3–13, Springer Science Business Media, Berlin, Germany, 2019.

[15] D. Zhang, G. Yang, F. Li, J. Wang, and A. K. Sangaiah, "Detecting seam carved images using uniform local binary patterns," *Multimedia Tools and Applications*, vol. 79, no. 13-14, pp. 8415–8430, 2020.

[16] M. Lu and S. Niu, "Detection of image seam carving using a novel pattern," *Computational Intelligence and Neuroscience*, vol. 201915 pages, Article ID 9492358, 2019.

[17] J. Chen, H. Ying, X. Liu et al., "A transfer learning based super-resolution microscopy for biopsy slice images: the joint

methods perspective," *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, vol. 14, 2020.

[18] R. Nosaka, Y. Ohkawa, and K. Fukui, "Feature extraction based on co-occurrence of adjacent local binary patterns PSIVT 2011 part II," *Lecture Notes in Computer Science*, pp. 82–91, Springer Science Business Media, Berlin, Germany, 2011.

[19] Z. Zhang, "The comparison of image retargeting algorithms based on seam carving," in *Proceedings of the 2015 International Conference on Test, Measurement and Computational Methods*, Atlantis Press, Chiang Mai, Thailand, 2015.

[20] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern Recognition*, vol. 29, no. 1, pp. 51–59, l, 1996.

[21] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 2, no. 4, pp. 433–459, 2010.

[22] T. Chen, J. Xu, H. Ying et al., "Prediction of extubation failure for intensive care unit patients using light gradient boosting machine," *IEEE Access*, vol. 7, no. 1, pp. 150960–150968, 2019.

[23] C.-C. Chang and C.-J. Lin, "Libsvm," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 1–27, 2011.

[24] S. J. Ryu and H. Y. Lee, "Detecting trace of seam carving for forensic analysis," *IEICE Transaction Information System*, vol. 97, no. 5, pp. 1304–1311, 2014.

[25] J. Li, B. Yang, and X. Sun, "Segmentation-based image copy-move forgery detection scheme," *IEEE Transaction Information Forensics Security*, vol. 10, no. 3, pp. 507–518, 2015.

[26] Z. Bi, L. Yu, H. Gao et al., "Improved VGG model-based efficient traffic sign recognition for safe driving in 5G scenarios," *International Journal of Machine Learning and Cybernetics*, pp. 1–12, 2020.

[27] B. Lin, S. Deng, H. Gao et al., "A multi-scale activity transition network for data translation in EEG signals decoding," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2020.

[28] J. Qin, H. Li, X. Xiang et al., "An encrypted image retrieval method based on harris corner optimization and LSH in cloud computing," *IEEE Access*, vol. 7, no. 1, pp. 24626–24633, 2019.

[29] Y. Tan, J. Qin, X. Xiang, W. Ma, W. Pan, and N. N. Xiong, "A robust watermarking scheme in YCbCr color space based on channel coding," *IEEE Access*, vol. 7, no. 1, pp. 25026–25036, 2019.

[30] J. Zhang, Y. Wu, W. Feng, and J. Wang, "Spatially attentive visual tracking using multi-model adaptive response fusion," *IEEE Access*, vol. 7, pp. 83873–83887, 2019.

[31] Y. Chen, W. Xu, J. Zuo et al., "The fire recognition algorithm using dynamic feature fusion and IV-SVM classifier," *Cluster Computing*, vol. 22, no. 3, pp. 7665–7675, 2019.

[32] J. Wang and J. H. Qin, J. Xiang, Y. Tan, and N. Pan, "CAPTCHA recognition based on deep convolutional neural network," *Mathematical Biosciences and Engineering*, vol. 16, no. 5, pp. 5851–5861, 2019.

[33] Y. Luo, J. Qin, X. Xiang, Y. Tan, Q. Liu, and L. Xiang, "Coverless real-time image information hiding based on image block matching and dense convolutional network," *Journal of Real-Time Image Processing*, vol. 17, no. 1, pp. 125–135, 2020.

[34] L. Xiang, G. Guo, G. Yu, V. Sheng, and P. Yang, "A convolutional neural network-based linguistic steganalysis for synonym substitution steganography," *Mathematical Biosciences and Engineering*, vol. 17, no. 2, pp. 1041–1058, 2020.

[35] S. He, Z. Li, Y. Tang, Z. Liao, F. Li, and S.-J. Lim, "Parameters compressing in deep learning," *Computers, Materials & Continua*, vol. 62, no. 1, pp. 321–336, 2020.

[36] A. K. Sangaiah, D. V. Medhane, T. Han, M. S. Hossain, and G. Muhammad, "Enforcing position-based confidentiality with machine learning paradigm through mobile edge computing in real-time industrial informatics," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4189–4196, 2019.

[37] L. Xiang, G. Zhao, Q. Li, W. Hao, and F. Li, "TUMK-ELM: a fast unsupervised heterogeneous data learning approach," *IEEE Access*, vol. 6, pp. 35305–35315, 2018.

Hindawi
Security and Communication Networks
Volume 2020, Article ID 8891035, 13 pages
https://doi.org/10.1155/2020/8891035

Research Article

Visual Object Multimodality Tracking Based on Correlation Filters for Edge Computing

Guosheng Yang and Qisheng Wei

School of Information Engineering, Minzu University of China, No. 27 Zhongguancun South Avenue, Beijing, China

Correspondence should be addressed to Qisheng Wei; 18301393@muc.edu.cn

Received 30 June 2020; Revised 23 September 2020; Accepted 6 November 2020; Published 15 December 2020

Academic Editor: Honghao Gao

In recent years, visual object tracking has become a very active research field which is mainly divided into the correlation filter-based tracking and deep learning (e.g., deep convolutional neural network and Siamese neural network) based tracking. For target tracking algorithms based on deep learning, a large amount of computation is required, usually deployed on expensive graphics cards. However, for the rich monitoring devices in the Internet of Things, it is difficult to capture all the moving targets in each device in real time, so it is necessary to perform hierarchical processing and use tracking based on correlation filtering in insensitive areas to alleviate the local computing pressure. In sensitive areas, upload the video stream to a cloud computing platform with a faster computing speed to perform an algorithm based on deep features. In this paper, we mainly focus on the correlation filter-based tracking. In the correlation filter-based tracking, the discriminative scale space tracker (DSST) is one of the most popular and typical ones which is successfully applied to many application fields. However, there are still some improvements that need to be further studied for DSST. One is that the algorithms do not consider the target rotation on purpose. The other is that it is a very heavy computational load to extract the histogram of oriented gradient (HOG) features from too many patches centered at the target position in order to ensure the scale estimation accuracy. To address these two problems, we introduce the alterable patch number for target scale tracking and the space searching for target rotation tracking into the standard DSST tracking method and propose a visual object multimodality tracker based on correlation filters (MTCF) to simultaneously cope with translation, scale, and rotation in plane for the tracked target and to obtain the target information of position, scale, and attitude angle at the same time. Finally, in Visual Tracker Benchmark data set, the experiments are performed on the proposed algorithms to show their effectiveness in multimodality tracking.

1. Introduction

Visual object tracking (VOT), the subfield of computer vision, is a process of continuously estimating the target state through video image sequence. In recent years, VOT has become a very active research domain due to its extensive applications in many sorts of fields such as intelligent surveillance [1], automatic driving [2], and traffic flow monitoring [3], to name a few.

In fields such as security monitoring and control, the traditional network architecture is difficult to deal with in terms of network delay and security reliability, and thus edge computing technology was born. Tasks with different attributes can be passed to different levels for processing. Zhan [4] shows that the first few feature extraction layers could run on edge device, and the others run on the cloud. And Gao [5, 6] divides tasks into different levels according to the business applications and using edge devices in one level.

As Figure 1 shows, for nonsensitive areas, video streams with lower resolutions can be processed on the local device; in the medium area, ordinary-resolution video streams can be used on the edge device; and in high-risk areas, high-resolution video streams can be used on the core cloud server, thereby reducing network bandwidth and improving the overall operating efficiency of the system. This article mainly explores the processing of video streams on edge clouds, and the tracking algorithm used is based on filtering.

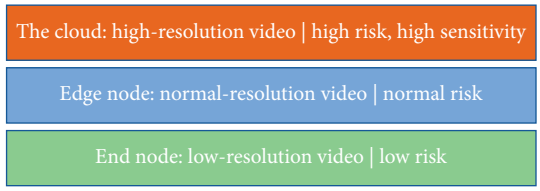


Figure 1: End-edge-cloud computing graphic.

A number of robust tracking algorithms have been proposed and developed to deal with the problems resulted from occlusion, illumination variation, background clutter, motion blur, and so on [7–14]. Such these algorithms are divided into deep learning-based category (DLC) and correlation filter- (CF-) based category [9].

For DLC, since the papers written by Geoff Hinton et al. [15, 16] were published, deep learning has become especially popular in the context of deep neural networks and has achieved impressive success on many applications, especially on feature extraction in computer vision. Inspired by such success, various deep-learning-based trackers [13, 14, 17–22] have been proposed and developed to cope with the problems encountered in tracking. Although most of trackers based on deep neural networks demonstrated the potential advantages for significantly improving the tracking performance which was testified by world VOT competitions [17], there are still some obvious limits. For example, there are fewer or even no training data available for the tracker because the prior information of the tracked object or the object bounding box is usually available only in the first frame. Even if the offline pretraining is employed to learn the target features for constructing a feature set of many targets, it is very possible for tracking a particular object whose features are not contained in the feature set. Nowadays, zero-shot and one-shot learning, as well as Siamese region proposal network, may be the most effective measures to cope with these problems [23–27].

And correlation filter-based tracking is also a solution. From its beginning of the minimum output sum of squared error (MOSSE) method [10] to the discriminative scale space tracker (DSST) method [12], a lot of improvements have continuously been made, which makes tracking based on CF achieve some highlighted tracking performances, such as lower computational load, being robust to the appearance variations of targets, and high tracking accuracy. However, there are still some improvements that need to be further studied for the tracking based on CF. One is that the algorithms do not consider the target rotation on purpose. The other is that the real-time property of DSST cannot always be ensured because it has a very heavy computational load to extract the histogram of oriented gradient (HOG) features from so many patches centered at the target position in order to ensure the scale estimation accuracy. This inspired us to think of an idea: on the premise of ensuring the tracking accuracy, appropriately decrease the number of patches in order to save the time for the introduction of the target rotation into DSST to form a multimodality tracking. It means that the tracker should simultaneously cope with translation, scale, and rotation in plane for the tracked target, which leads us to propose the visual object multi-modality tracking based on correlation filters (MTCF), to figure out these two problems, and at the same time to obtain the target information of position, scale, and attitude angle simultaneously.

In this paper, we design a correlation filter-based tracker aiming at tracking the target accurately and robustly with the tracking speed at 25 frames per second and tracking the rotation of target.

2. Related Materials

In this section, centering on tracking based on CF, we briefly list some relevant research works which have contributions to the tracking based on CF to highlight our motivations.

MOSSE method is taken as the earliest real-time CF-based tracker [28], which is an improved version of average synthetic exact filter (ASEF) [29] trained offline to detect objects. MOSSE tracker has strong robustness to target appearance and environment change, which can achieve very fast tracking speed. This is because that the correlation convolution of image in time domain is transformed into the multiplication of image in frequency domain, which greatly reduces the computation complexity and load. However, MOSSE method uses only grayscale samples to train CF and mainly focuses on translation without considering scale and rotation.

Based on the MOSSE, the circulant structure kernel CSK method [30] constructs a circulant matrix of training data by using cyclic shift of target window to maintain dense sampling around the target, rather than random sampling. On the other hand, CSK method maps ridge regression of linear space to nonlinear space through a kernel function and simplifies the calculation via solving a dual problem in nonlinear space to avoid inverse matrix operation, which leads to reducing the computation complexity and improving the tracking speed. The kernelized correlation filter (KCF) method [11] is an improved version of CSK. It introduces multichannel HOG features into CSK to enhance the feature representation ability and to improve the tracking performance significantly. Nevertheless, there exists a major imperfection for KCF method; i.e., it is not robust to the scale variation of the target. In addition, for the KCF-based tracking, the authenticity of negative samples will decrease along with the increase of cyclic displacement, which results in the tracker being trained on a portion of unreal samples. To address this issue, Danelljan et al. [18] introduce a spatial regularized term in the goal function of KCF-based tracker to penalize the filter coefficients near the margins of the bounding box. Based on [18], Dai et al. [28] propose a novel adaptive spatial regularized CF to make the tracker learn more reliable filter coefficients by fully exploiting the diversity information of different objects in different frames during the tracking process. However, just as the standard KCF-based trackers do, these two trackers are still not robust to the scale variation of the target.

DSST [12, 31] trackers address the scale adaptation problem using multiscale searching strategies. It divides tracking into translation prediction and scale prediction.

Firstly, translation prediction is performed by applying a standard translation filter on the current frame to get the position of the target. Secondly, the target size is estimated by employing trained scale filter at the target location obtained from the translation filter. Translation filter and scale filter are two independent filters, and both are based on MOSSE. Although DSST tracker has improved the tracking performance and is robust to target scale variation, there exist some obvious limitations to be further perfected. One is that DSST does not consider the target rotation on purpose, which has strong negative impacts on the tracking performance. The other is that it is not necessary for guaranteeing the tracking speed to spend a lot of operation time on sampling too many patches centered on the target location.

Besides of the tracking method, features of the tracked target are also key components of a tracker, which has a very heavy influence on the tracking performance. Generally speaking, the richer the features are, the better the performance of the tracker is. The simplest feature is intensity matrix of the search image, which is used in MOSSE [10]. And SIFT features [32] and HOG features [33] are used in object tracking afterwards. In recent years, deep features [34] are widely used in object tracking. In this paper, HOG features combined with grayscale features rather than deep features are adopted because our focus is on the CF-based tracking. And we do not adopt SIFT because SIFT is scale-invariant and we need to explicitly capture the size change of the object.

Summarizing the analysis stated above, we propose the MTCF to alleviate the imperfections of the relevant CF-based trackers stated above. Aiming at tracking the target accurately and robustly with the tracking speed at 25 frames per second at least for practical visual object tracking, MTCF consists of 4 tasks. Firstly, based on the standard CF-based translation tracker, determine the target location in the current frame. Secondly, based on DSST, sample several patches (with alterable number of patches) with different resolutions, centered at the tracked target location determined by translation CF, figure out the feasible scale for patches, and seek out an optimal decision policy to find the final scale among feasible scales. Thirdly, based on the standard CF-based translation tracker, design a rotation tracker using space searching. Lastly, integrate the previous 3 tasks to form MTCF.

3. Methodology of the Tracking Design

3.1. Variable Symbols Used in This Paper. In this paper, f denotes the "feature" of one image patch cropped with specific bounding box, h denotes the correlation filter, and g denotes the response map of correlation. In this way, $f_{\text{trans},i}$ denotes the feature of i^{th} frame used to correlate with translation filter h_{trans} and we get the translation response map $g_{\text{trans},i}$.

And s_i denotes the scale of the target the tracker got after i^{th} frame, and r_i denotes the rotation angle of the target after tracking i^{th} frame.

In terms of the convolution theorem, the correlation in spatial domain can be transformed to element-wise manipulation, which will dramatically reduce the correlation computation load. Thus, for computation efficiency, correlation manipulation is proposed to use Fast Fourier Transform (FFT) method in frequency domain. So, let the uppercase variables be the Fourier transforms of their lowercase counterparts, i.e., $F_{\text{trans},i}$, $G_{\text{trans},i}$, and H_{trans} corresponding to $f_{\text{trans},i}$, $g_{\text{trans},i}$, and h_{trans}, respectively.

3.2. Standard Translation Tracker Based on Correlation Filter. As Figure 2 shows, given a video sequence, draw a rectangular bounding box (the very close same size as the target, the red one) around the target in first frame and extract a feature map $f_{\text{trans},i}$ from the chosen region (the green rectangle, two times the size of the red one). And then train a correlation filter h_{trans} to correlate with $f_{\text{trans},1}$ to get an ideal response $g_{\text{trans},1}$. In the next frames, use the correlation filter h_{trans} to correlate with extracted feature map from the chosen region and get a response map $g_{\text{trans},i}$ as follows:

$$g_{\text{trans},i} = f_{\text{trans},i} * h_{\text{trans}}, \tag{1}$$

where $*$ represents convolution operation.

In normal tracking process, there should be one peak in the response map. And the peak position is considered as the center of target (and in this sense tracking executes). The key of tracking is to find a robust feature extractor and maintain the correlation filter h_{trans} to counter a variety of adverse effects such as target appearance transformation, occlusion, and so on, using appropriate updating strategy.

3.3. Scale Tracker Based on Correlation Filter. Being different from that in the original DSST, the number of scales S (or the number of image patches) in this paper is an optional positive integer determined by the trade-off between tracking speed and tracking accuracy (i.e., smaller S is selected if tracking speed takes priority to tracking accuracy, vice versa). Let M, N be the shape of the target, and construct image patches centered on the target position p_i with different scales to form an image patch set

$$B_{\text{scale},i} = \left\{ \beta^n M \times \beta^n N \mid \ n = 0, \ \pm 1, \ \pm 2, \ldots, \ \pm \text{round}\left(\frac{S}{2}\right) \right\}, \tag{2}$$

where β is scale step. Resize each $\beta^n M \times \beta^n N$ from $B_{\text{scale},i}$ into the same shape to form a bounding box set $\text{patch}_{\text{scale},i}$. As Figure 3 shows, instead of extracting one feature map from a bounding box with fixed scale, the tracker extracts a feature map $f^n_{\text{scale},i}$ for each patch from the bounding box set $\text{patch}_{\text{scale},i}$ (the number of feature maps is 33 in Figure 3). Each feature map $f^n_{\text{scale},i}$ is concatenated into a vector, and all these vectors are combined into a feature map $f^n_{\text{scale},i}$. And we design a scale correlation filter to correlate the feature map $f^n_{\text{scale},i}$, and the scale where maximum response taking place is the predicting scale to match current scale of target.

3.4. Rotation Tracker Based on Correlation Filter. The target may rotate during tracking, so we use rotated bounding box

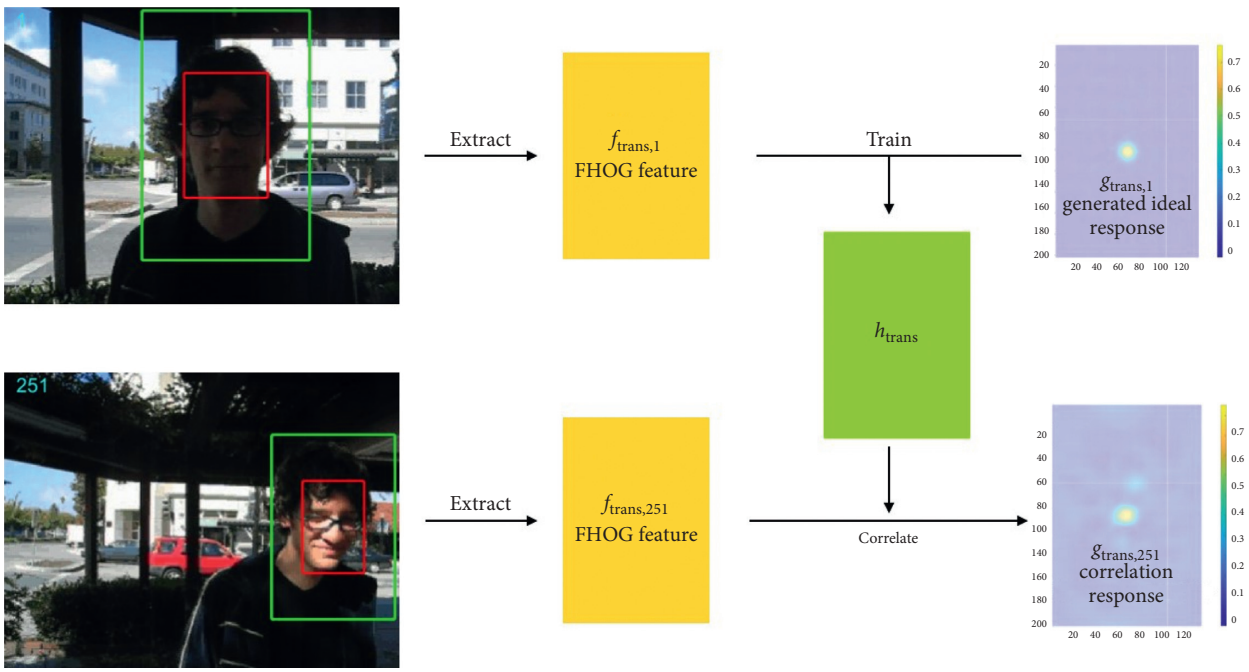


Figure 2: Correlation filter-based tracking diagram (the two frames are from sequence "Trellis" in [35]).

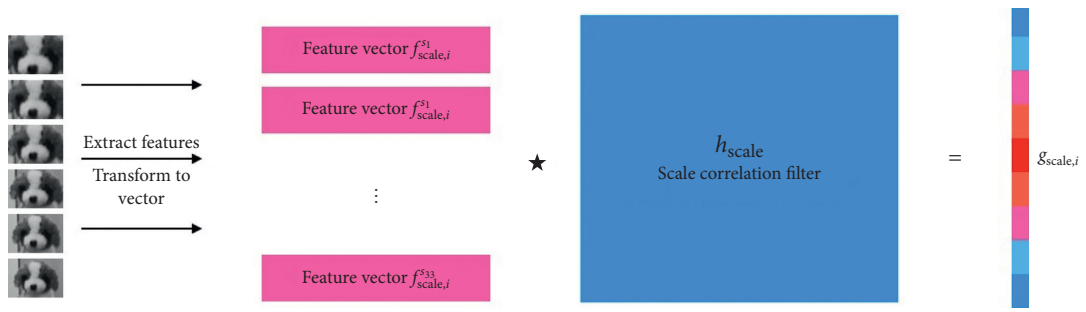


Figure 3: Scale tracking diagram.

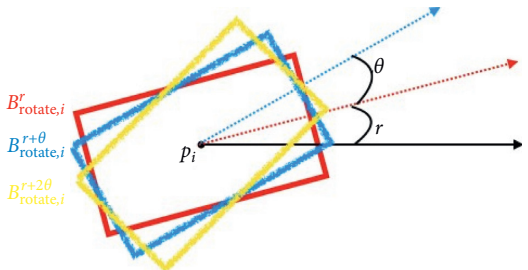


Figure 4: Bounding boxes rotate around the target center p_i.

centered on target to crop every frame. As shown in Figure 4, let p_i be the center of the target and r the current angle the target rotated, and $B^{r}_{\text{rotate},i}$ denotes the bounding box with the rotation in frame i. And using the rotation around target center, we construct a set of bounding boxes

$$B_{\text{rotate},i} = \left\{B^{r-\Theta}_{\text{rotate},i}, \ldots, B^{r-\theta}_{\text{rotate},i}, B^{r}_{\text{rotate},i}, B^{r+\theta}_{\text{rotate},i}, \ldots, B^{r+\Theta}_{\text{rotate},i}\right\}, \tag{3}$$

with the same size of $B^{r}_{\text{rotate},i}$; here Θ is the given maximum rotation angular displacement and θ is rotation step.

For each bounding box $B^{r}_{\text{rotate},i}$ from $B_{\text{rotate},i}$, extract feature map $f^{\text{rot}}_{\text{rotate},i}$, and correlate with the rotation correlation filter to get a maximum response value, where

$$\text{rot} \in \{r - \Theta, \ldots, r + \Theta\}. \tag{4}$$

Compare those values and find the largest one to get the predicting rotation angle r. Let r be the tracking result of frame i, as follows:

$$r_i = \max_{\text{rot}\in\{r-\Theta,\ldots,r+\Theta\}} \max g^{\text{rot}}_{\text{rotate},i}. \tag{5}$$

In addition to the methods we used here, we also envisioned the "1-dimensional correlation rotation tracking" in the Supplementary Materials. However, after testing, it shows that this method requires too much calculation and is not suitable for use at edge nodes.

3.5. Multimodality Tracking Based on Correlation Filter. Integrate translation, scale, and rotation stated in previous section to form MTCF whose iteration procedure at the i^{th}

frame is briefly outlined with the known parameters obtained in the $(i+1)^{\text{th}}$ frame, including target position p_{i-1}, translation filter h_{trans}, scale filter h_{scale}, scale s_{i-1}, rotation filter h_{rotate}, and rotation angle r_{i-1}.

3.5.1. Translation Estimation

(1) Construct bounding box $B_{\text{trans},i}$ with the scale s_{i-1}, centered at p_{i-1} in the i^{th} frame.

(2) Extract feature map $f_{\text{trans},i}$ from $B_{\text{trans},i}$.

(3) Calculate the correlation map $g_{\text{trans},i}$ using $f_{\text{trans},i}$ and h_{trans}.

(4) Obtain the target new position p_i corresponding to the position where the largest correlation value of $g_{\text{trans},i}$ taking place.

3.5.2. Scale Estimation

(1) Construct image patches $\text{patch}_{\text{scale},i}$ of different scales centered on the target position p_i in the i^{th}frame

(2) Extract feature map patches $\mathbf{f}_{\text{scale},i}$ from image patches $\text{patch}_{\text{scale},i}$, and concatenate each feature map $f_{\text{scale},i} \in \mathbf{f}_{\text{scale},i}$ to form an vector, and then combine those vectors to form a feature matrix $f_{\text{scale},i}$

(3) Calculate the correlation map $g_{\text{scale},i}$ using $f_{\text{scale},i}$ and h_{scale}

(4) Update the target scale with the optimal s corresponding to the position where the largest-scale correlation value is located

3.5.3. Rotation Estimation

(1) Construct image patches $\text{patch}_{\text{rotate},i}$ from the bounding box set $B_{\text{rotate},i}$ centered on target position p_i with rotation angle r_{i-1}

(2) Extract feature maps $f^{\text{rot}}_{\text{rotate},i}$ for every patch from $\text{patch}_{\text{rotate},i}$

(3) For every feature map $f^{\text{rot}}_{\text{rotate},i}$, make the correlation with the original rotation filter, and get a maximum response value $\text{score}_{\text{rotate},i}$

(4) Update the target rotation angle r_i with the optimal rot corresponding to the best $\text{score}^{\text{rot}}_{\text{rotate},i}$

3.5.4. Model Update

(1) Construct the bounding box $B_{\text{trans},i}$ centered on target position $p_i = (x_i, y_i)$ with scale s and rotation angle r

(2) Extract $f_{\text{trans},i}$, $f_{\text{scale},i}$, and $f_{\text{rotate},i}$

(3) Update translation model

(4) Update scale model

(5) Update rotation model

3.5.5. Keep Tracking. Output the tracking results of the i^{th} frame and return to the next frame tracking.

4. MTCF: The Entire Model

4.1. Translation Tracking Procedure. The simplest correlation-based tracking only focuses on translation of the target. In the first frame, we label a rectangular region $B_{\text{trans},1}$ centered on the target. So, the tracker can extract the feature map of target appearance. The feature map must maintain a spatial mapping because the tracker uses the position where the maximum response happens to predict new target position.

The simplest feature map is gray intensity matrix transformed from the specific region (for example, $B_{\text{trans},1}$) of original frame. Many researchers use 2-dimensional Hanning window (see Figure 5) to preprocess the primitive intensity matrix. After being processed by the Hanning window, the intensity matrix focuses on the central region of target and weakens the background information near the bounding box edge. Because in the first frame we draw the bounding box tightly around the target, the tracker may lose some features and behave unstable.

To address this issue, the simple way is to expand the search region. Define a parameter bb to determine how many times the size of $B_{\text{trans},1}$ the search window is. As a rule, greater parameter bb will contribute to extracting more features of the target and making the tracker stable. But much more time will be spent on the extracting features from the large search region.

This is clearly demonstrated by "S1" presented in the Supplementary Materials. In this paper, we adopt a trade-off policy and select the bb = 2.

From now on, we will use $B_{\text{trans},1}$ to represent the translation search windows.

From $B_{\text{trans},1}$ we get the $f_{\text{trans},1}$, and because we need to train the translation correlation filter h_{trans}, an initial $g_{\text{trans},1}$ is required. In prior papers, most of researchers take the Gauss-shaped response map as initialization, as follows:

$$\begin{aligned} g_{\text{expl}} &= e^{-(d/\sigma)}, \\ d &= \sqrt{(x - x_{\text{center}})^2 + (y - y_{\text{center}})^2}, \\ \sigma &= 1, \\ (x, y) &\in B_{\text{trans},1}, \end{aligned} \tag{6}$$

and Figure 6 shows an example of $g_{\text{trans},1}$.

Though the intensity feature is cheap in computation, it is unstable. Because it only takes advantage of little information in the frame. Recently, lots of deep features (for example, convolution neural network feature) are introduced to object tracking and behave well in accuracy and robustness. However, it is too computational expensive, and in this paper, we focus on FHOG [36] feature.

We use an FHOG feature extractor to get the feature map $f_{\text{trans},i}$. In translation step, 27 dimensions in FHOG and 1 dimension of intensity feature are taken into account. According to DSST [12], discriminative correlation filters for multidimensional features are applied as follows.

Minimize the cost function,

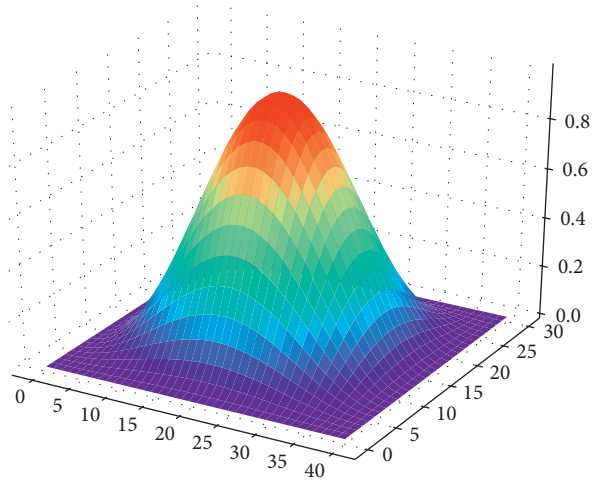


Figure 5: The 2-dimensional Hanning window (the shape is 40×30).

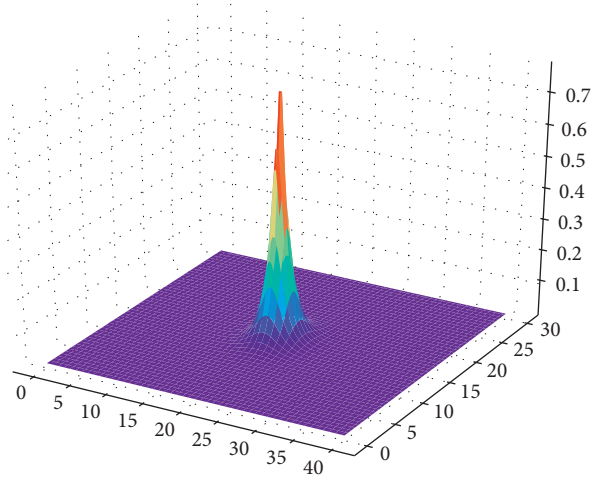


Figure 6: The Gauss-shaped response map ($\sigma = 1$, shape of B is 40×30).

$$\varepsilon = \left\| \sum_{l=1}^{d} f^l * h^l - g \right\| + \lambda \left\| \sum_{l=1}^{d} h^l \right\|. \tag{7}$$

Here, g is the ideal response about the correlation between feature map and filter, and the parameter $\lambda \geq 0$ is the regularization term. In FFT domain, the solution [12] can be written as

$$H^l = \frac{F^l \odot G^*}{\sum_{k=1}^{d} F^k \odot F^{k*} + \lambda}, \tag{8}$$

where $*$ indicates the complex conjugation, $\odot$ is for element-wise multiplication, and $l \in \{1, \ldots, d\}$ is the dimension number.

The translation filter H_{trans} can be solved as below:

$$H^*_{\text{trans}} = \frac{G_{\text{trans},i} \odot F^*_{\text{trans},i}}{\sum_l F^l_{\text{trans},i} \odot F^{l*}_{\text{trans},i} + \lambda}. \tag{9}$$

Equation (9) is employed in offline learning to obtain the correlation filter. In the practical tracking, the tracker (for example, MOSSE, KCF, and DSST) takes the target position in the $(i-1)^{\text{th}}$ frame as the center of bounding box B_i in the i^{th} frame, extracts feature map $f_{\text{trans},i}$ from B_i, and then calculates the correlation map

$$g_{\text{trans},i} = F^{-1}\left\{G_{\text{trans},i} = F_{\text{trans},i} H^*_{\text{trans},(i-1)}\right\}, \tag{10}$$

to determine the target new position p_i corresponding to the element with maximum value in $s_{\text{trans},i}$; here, F^{-1} indicates the inverse Fourier transform.

$$p_i = \max s_{\text{trans},i}(k, l). \tag{11}$$

Afterwards, reconstruct bounding box B_i centered on the target new position p_i from which feature map $f_{\text{trans},i}$ is extracted, and then update the translation CF to get $H^*_{\text{trans},i}$. Lastly, an iterative formula for equation (9) is presented as the following equations from equations (9)–(12) according to [10, 12]:

$$G_{\text{trans},i} = F_{\text{trans},i} \odot H^*_{\text{trans},(i-1)}, \tag{12}$$

$$A_{\text{trans},i} = \eta G_{\text{trans},i} \odot (1-\eta) A_{\text{trans},(i-1)}, \tag{13}$$

$$D_{\text{trans},i} = \eta F_{\text{trans},i} \odot F^*_{\text{trans},(i-1)} + (1-\eta) D_{\text{trans},(i-1)}, \tag{14}$$

$$H^*_{\text{trans},i} = \frac{A_{\text{trans},i}}{D_{\text{trans},i}}, \tag{15}$$

where η is the learning rate.

4.2. Scale Tracking Procedure. As for scale tracking procedure, two methods are commonly used. One is called "exhaustive scale tracking" and the other is "1-dimensional correlation filter scale tracking." In this paper, we use "1-dimensional correlation" method.

In the previous frame, we got the target position p_{i-1} and scale s_{i-1}.

Let M, N be the shape of the target, construct image patches centered on the target position p_i in terms of the method presented in Section 3, and resize to form a bounding box set $\text{patch}_{\text{scale},i}$. FHOG extractor is applied to extract a feature map $f^n_{\text{scale},i}$ for each patch from the bounding box set $\text{patch}_{\text{scale},i}$. Each feature map $f^n_{\text{scale},i}$ is concatenated into a vector, and all of these vectors are combined into an integrated vector $f_{\text{scale},i}$. Estimating the target scale can be solved by learning a separate 1-dimensional correlation filter. Design a 1-dimensional filter h_{scale} to correlate with $f_{\text{scale},i}$. The initial ideal response $g_{\text{scale},i}$ is a Gauss-shaped peak, as Figure 7 shows.

The scale with the largest correlation response value is taken as the optimal scale s_i.

Afterwards, extract feature map $f_{\text{scale},i}$ from the $\text{patch}^n_{\text{scale},i}$ centered on the target new position p_i with the

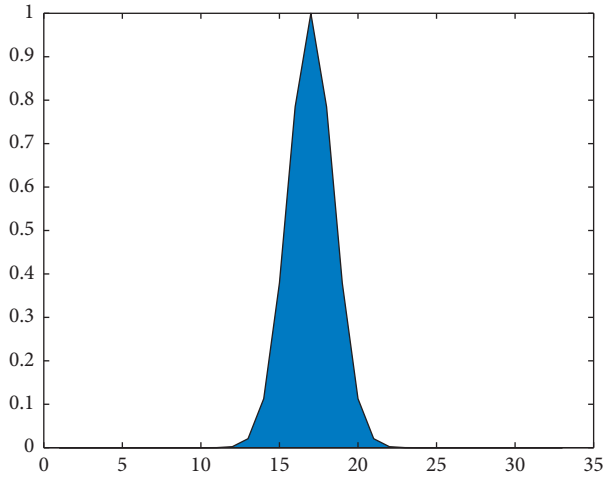


Figure 7: The 1-dimensional Gauss-shaped peak as the initial correlation response.

target final scale s_i, and then update the scale correlation filter to get $H^*_{\text{scale},i}$ using equations (13)–(17).

In this process, set the parameter "spatial bin size" to 4 to save time in the next process and use all FHOG dimensions. So, the length of S feature vector is $(M/4) \times (N/4) \times 31$.

Estimating the target scale can be solved by learning a separate 1-dimensional correlation filter. Treat the feature vector as multidimensional features and S vectors turn into 1-dimensional feature $f_{\text{scale},i} = \left\{f^{\text{sca}_1}_{\text{scale},i}, \ldots, f^{\text{sca}_S}_{\text{scale},i}\right\}$.

$$g_{\text{scale},i} = f_{\text{scale},i} * h_{\text{scale}}, \tag{16}$$

$$G_{\text{scale},i} = F_{\text{scale},i} \odot H^*_{\text{scale},(i-1)}, \tag{17}$$

$$A_{\text{scale},i} = \eta G_{\text{scale},i} \odot F^*_{\text{scale},(i-1)} + (1-\eta) A_{\text{scale},(i-1)}, \tag{18}$$

$$D_{\text{scale},i} = \eta F_{\text{scale},i} \odot F^*_{\text{scale},(i-1)} + (1-\eta) D_{\text{scale},(i-1)}, \tag{19}$$

$$H^*_{\text{scale},i} = \frac{A_{\text{scale},i}}{D_{\text{scale},i}}. \tag{20}$$

Construct different groups containing different number of patches. The number of patches varies from small to large (e.g., from 10 to 55), and all patches are centered at the tracked target location determined by translation CF in current frame. Let the basic DSST [12] perform on the visual track data set [35], and calculate the tracking speed and the tracking accuracy which is characterized by the Euclidean distance between tracking window center and ground truth center for each group. The experiment results are shown as in Figure 8.

From Figure 8, it can be seen that the tracking accuracy and speed are different with different numbers of patches. The larger number of the patches corresponds to a low tracking speed, and vice versa. Thus, on the premise of ensuring the tracking accuracy, appropriately select the number of patches in order to save the time for the introduction of the target rotation into DSST to form a multimodality tracking. After experiments, it is found that most of the time is spent in the feature extraction module.

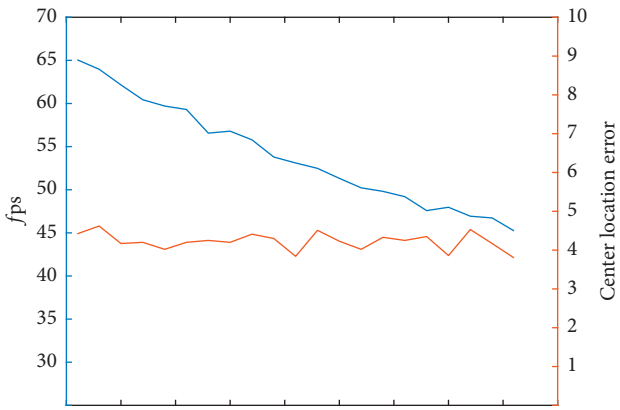


Figure 8: The tracking accuracy and speed results for groups with different scales.

4.3. Rotation Space Tracking Procedure. Set the target attitude with $r = 0$ in the first frame; construct a set of bounding boxes $B_{\text{rotate},i}$ as described in the previous section in successive frame. FHOG extractor is applied to extract a feature map $f^{\text{rot}}_{\text{rotate},i}$ for each patch $B^{\text{rot}}_{\text{rotate},i}$ from the bounding box set $B_{\text{rotate},i}$. Estimating the target rotation can be solved by learning a separate 1-dimensional correlation filter. Train a 1-dimensional a single rotation filter h_{rotate} as the similarity function to compute the maximum correlation response $\max g^{\text{rot}}_{\text{rotate},i}$ for each feature map $f^{\text{rot}}_{\text{rotate},i}$. Therefore, the best tracking angle r_i is calculated by using the following equation:

$$r_i = \max_{\text{rot}} \max g^{\text{rot}}_{\text{rotate},i}. \tag{21}$$

Afterwards, extract feature map $f_{\text{rotate},i}$ from the $B^{\text{rot}}_{\text{rotate},i}$ centered on the target new position p_i with the target final

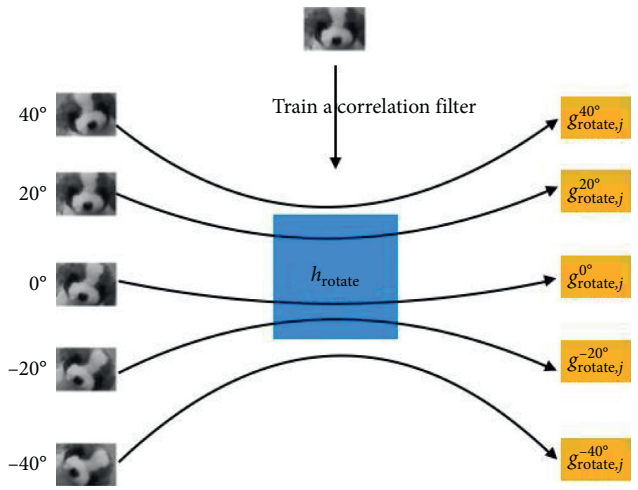


Figure 9: Exhausted method for tracking rotation.

rotation angle r_i, and then update the rotation correlation filter to get $H^*_{\text{rotate},i}$ using the following equations:

$$g_{\text{rotate},i} = f_{\text{rotate},i} * h_{\text{rotate}}, \tag{22}$$

$$G_{\text{rotate},i} = F_{\text{rotate},i} \odot H^*_{\text{rotate},(i-1)}, \tag{23}$$

$$A_{\text{rotate},i} = \eta G_{\text{rotate},i} \odot F^*_{\text{rotate},i} + (1-\eta) A_{\text{rotate},(i-1)}, \tag{24}$$

$$D_{\text{rotate},i} = \eta F_{\text{rotate},i} \odot F^*_{\text{rotate},(i-1)} + (1-\eta) D_{\text{rotate},(i-1)}, \tag{25}$$

$$H^*_{\text{rotate},i} = \frac{A_{\text{rotate},i}}{D_{\text{rotate},i}}. \tag{26}$$

We take Figure 9 as an example to demonstrate our search rotation angle. As Figure 9 shows, $r = 0°$, $\theta = 20°$, $\Theta = 40°$, androt $\in \{-40, -20, 0, 20, 40\}$. Construct a set of bounding boxes $B_{\text{rotate},i}$ with 5 patches $B^{\text{rot}}_{\text{rotate},i}$; train the rotation correlation filter h_{rotate} using the samples in the first frame. And Figure 10 shows the correlation response with each patch, where the $r + 20°$ corresponds to the highest response. Thus, we can make a conclusion that $r + 20°$ is the best predicting rotation angle in Figure 9, which demonstrate the effectiveness of our proposed search rotation method.

In this process, how to set parameters of θ and Θ is very important to obtain the good tracking performance including tracking speed and tracking accuracy. Greater Θ and smaller θ will contribute to the good tracking performance, but much more time will be spent on the extracting features of the tracked target, which has a negative influence on the tracking speed. This is clearly demonstrated by "S2" presented in the Supplementary Materials. As a rule, parameters of θ and Θ are fixed by experiments according to the requirements of tracking tasks.

In this paper, we also adopt such a policy.

5. Experiment

5.1. Experiment Setup. In this paper, our method is implemented in MATLAB R2019a on Windows 10 system. The experiments are conducted on a PC with Intel Xeon® 2.4 GHz and 63.9 GB RAM. The data set is selected from the visual track data set [35]. Our experiment is divided into 3 groups with different parameters. All of them are used to testify our proposal method: on the premise of ensuring the tracking accuracy, appropriately decrease the number of patches in order to save the time for the introduction of the target rotation into DSST to form a multimodality tracking, to verify the effectiveness of our proposed rotation tracking algorithm, and to demonstrate the whole tracking performance of our proposed visual object multimodality tracking algorithm based on correlation filter.

In the experiment of each group, the visual track data set is selected to have target translation, scale, and rotation simultaneously. And the number of scales S, the scale factor β, and the learning rate η are kept unchanged in each group and are fixed as (33, 1.02, and 0.015) and (27, 1.0247, and 0.015), respectively, which means that maximum and minimum scale field of two groups are the same, as Figure 11 shows. And we test the influence of different size (bb) of searching window in the Supplementary Materials.

5.2. Experiment of the First Group. In this group experiment, Θ is selected to be 10°, and θ is selected to be 5°. As a result, the tracking speed is 31fps, and the experiment results are shown in Figures 12 and 13 consisting of some typical tracking frames.

From Figure 11, it can be seen that appropriately decreasing the number of patches completely can save the time for the introduction of the target rotation into DSST to form

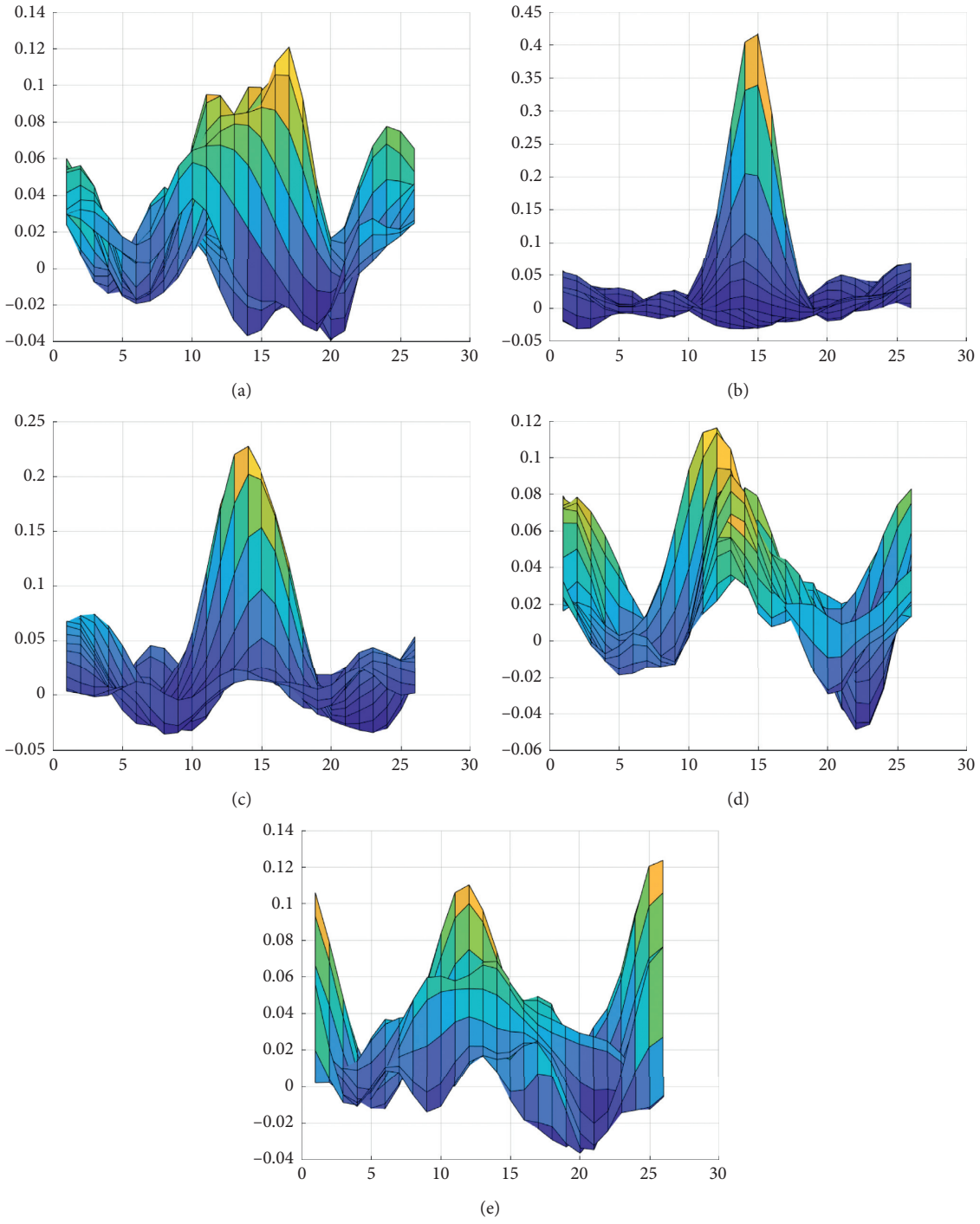


FIGURE 10: The correlation response with image patches cropped by different-rotation bounding box. (a) $r + 40^{\circ}$, (b) $r + 20^{\circ}$, (c) r, (d) $r - 20^{\circ}$, and (e) $r - 40^{\circ}$.

a multimodality tracking on the premise of ensuring the tracking accuracy and that our proposed rotation tracking algorithm can work well.

5.3. Rotation Tracking Performance Test. In this group experiment, θ is selected to be 12, and τ is selected to be 4. As a result, the tracking speed is 29 fps, and the experiment

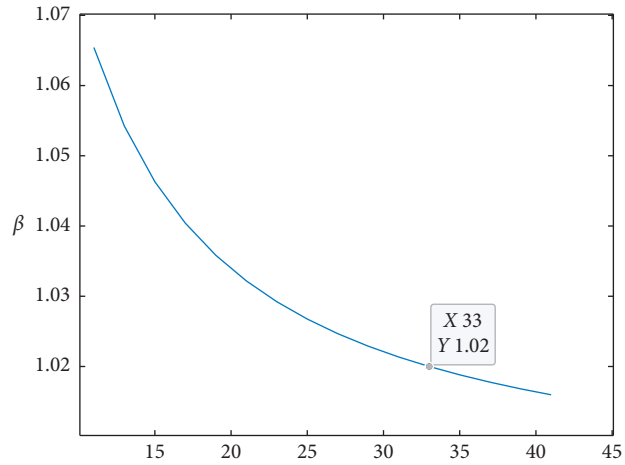
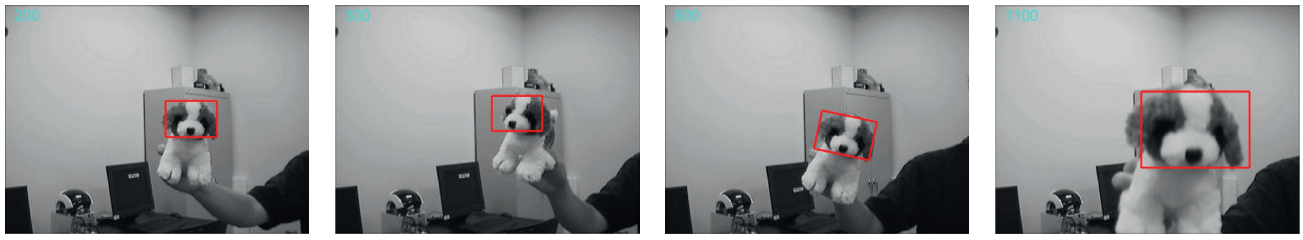
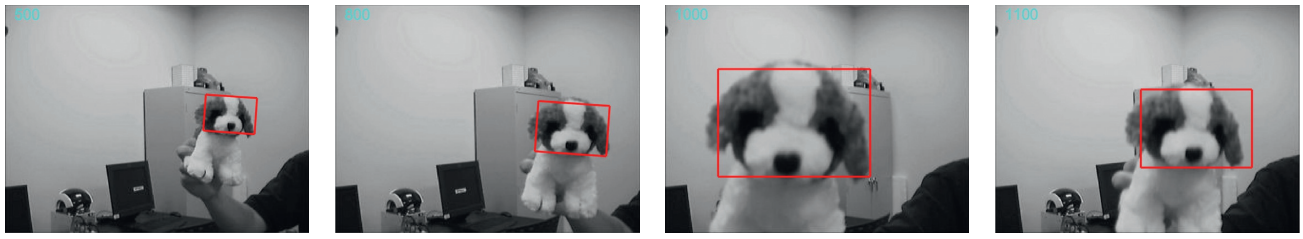


Figure 11: β is the function of S to guarantee fixed maximum and minimum scale field.



(a) (b) (c) (d)

Figure 12: The tracking results with tracking speed 31fps. The red window represents that the target is tracked completely. (a) represents the original figure, (b) represents object translation, (c) represents object rotation, and (d) represents object scale.



(a) (b) (c) (d)

Figure 13: The tracking results with tracking speed 29 fps. The red window represents that the target is tracked completely. (a) Rotation, (b) scale, (c) scale, and (d) translation.

results are shown in Figure 13 consisting of some typical tracking frames.

In this group experiment, the tracking speed is 29 fps which is lower than that in the first group experiment because τ is selected to be 4 which means the number of $B^{\rho}_{(\text{rotate},i)}$ is increased, resulting in much more time being spent on extracting feature map $f^{\rho}_{(\text{rotate},i)}$ from $B^{\rho}_{(\text{rotate},i)}$. But our proposal visual tracker still can work well in tracking the target with translation, scale, and rotation. This can be shown by Figure 13. From this perspective, we can say that the rotation step can be appropriately increased if tracking accuracy is preferred, and vice versa.

5.4. Multimodality Tracking Performance Test. In both of the two group experiments, our proposed MTCF algorithm is performed on the visual track data set [35] to demonstrate the multimodality tracking performance; the tracking results are shown in Figure 14.

From Figure 14, it can be seen that our proposed MTCF has good multimodality tracking performance, which can enable us to obtain the position, scale, and attitude angle of the tracked target simultaneously.

The generalization ability of this algorithm still maintains the same level as DSST and is very dependent on the HOG extraction algorithm.

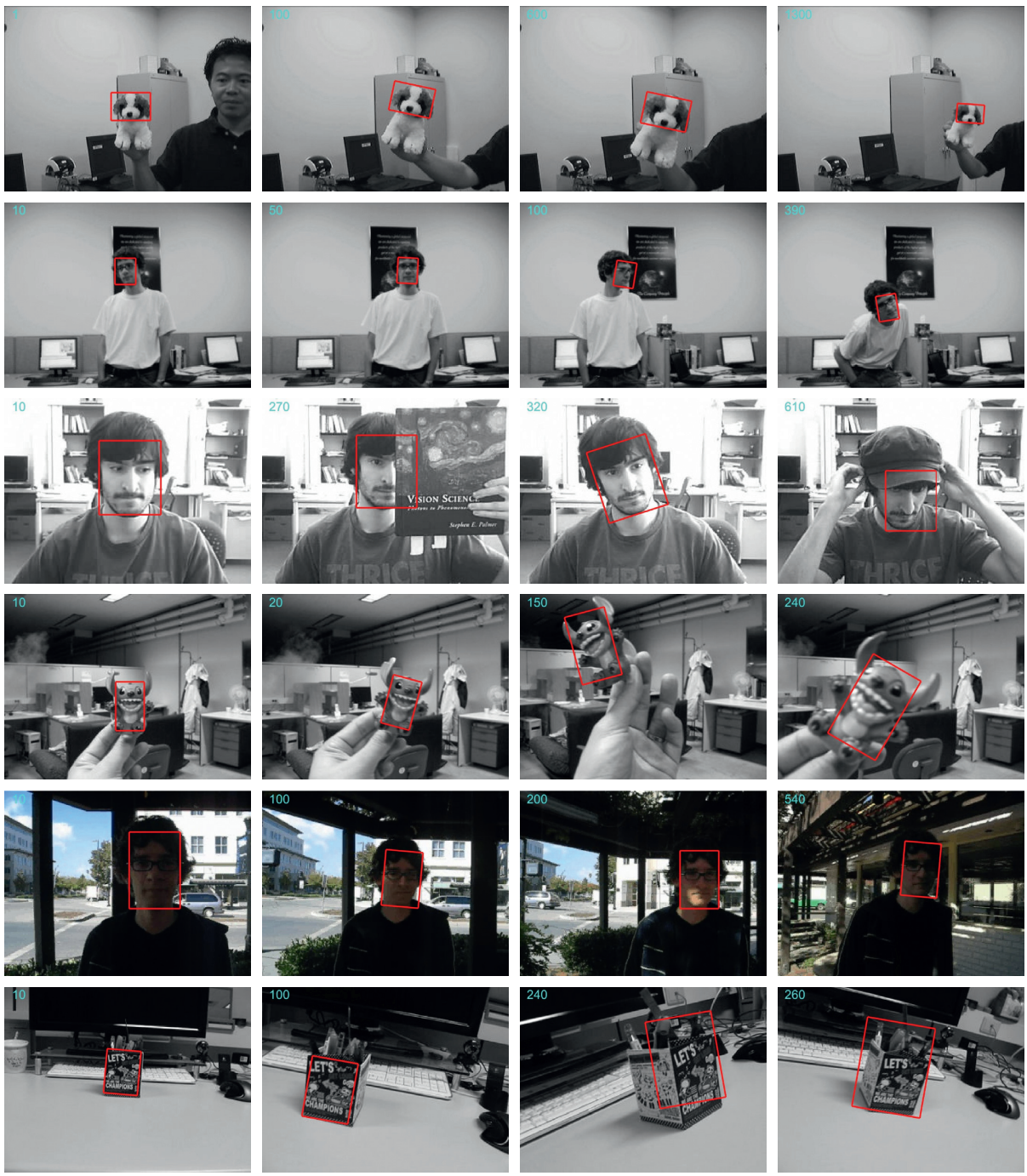


Figure 14: Some tracking results using MTCF, where the first five rows show that our tracker could track the target in scale variation, partially occlude occasion, and illumination variation, and the last row shows that the rotation tracking performance depends on scale tracking.

6. Conclusion and Future Work

In this paper, on the premise of ensuring the tracking accuracy, we introduce the alterable patch number for target scale tracking and the space searching for target rotation tracking into the standard DSST tracking method and propose a multimodality tracking MTCF to simultaneously cope with translation, scale, and rotation in plane for the tracked target and to obtain the target information of position, scale, and attitude angle at the same time. Experimental results demonstrate that the proposed multimodal target tracking algorithm MTCF (1) can reach the approving tracking speed which is largely exceeded 25 fps at least for practical visual object tracking by appropriately decreasing the number of patches for target scale tracking and (2) can obtain good tracking performance for translation, scale, and rotation simultaneously. In the future, our work will focus on the distributed hardware and software

implementation of the proposed multimodal comprehensive tracking algorithm.

For terminal devices not equipped with GPU units, low-resolution video is used to reduce the computational pressure on target features. For edge devices with certain computing capabilities, they are responsible for the main target tracking tasks. Finally, for a few critical and high-risk areas, the network bandwidth saved by the above two is used to upload to the central cloud processor for calculation to achieve hierarchical governance coordination.

Data Availability

All the source codes and related pictures will be uploaded to GitHub and will be available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 61772575, National Key R&D Program of China under Grant 2017YFB1402101, and Independent Research Projects of Minzu University of China.

Supplementary Materials

S1: influence of different size of searching window and analysis of the results. S2: 1-dimensional correlation rotation tracking, $\theta = 30^{\circ}$. (*Supplementary Materials*)

References

[1] G. Liu, S. Liu, K. Muhammad, A. K. Sangaiah, and F. Doctor, "Object tracking in vary lighting conditions for fog based intelligent surveillance of public spaces," *IEEE Access*, vol. 6, 2018.

[2] Y. Nishida, T. Sonoda, S. Yasukawa et al., "Underwater platform for intelligent robotics and its application in two visual tracking systems," *Journal of Robotics and Mechatronics*, vol. 30, no. 2, pp. 238–247, 2018.

[3] A. De Bruin and M. J. Booysen, "Drone-based traffic flow estimation and tracking using computer vision," 2015.

[4] Z. Zhang, X. Q. Zhang, D. C. Zuo, and G. D. Fu, "Research on target tracking application deployment strategy for edge computing," *Ruan Jian Xue Bao/Journal of Software*, vol. 31, no. 9, 2020.

[5] H. Gao, L. Kuang, Y. Yin, B. Guo, and K. Dou, "Mining consuming behaviors with temporal evolution for personalized recommendation in mobile marketing apps," *Mobile Networks and Applications*, vol. 25, no. 4, pp. 1233–1248, 2020.

[6] X. Ma, H. Gao, H. Xu, and M. Bian, "An IoT-based task scheduling optimization scheme considering the deadline and cost-aware scientific workflow for cloud computing," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, no. 1, 2019.

[7] X. Sheng, Y. Liu, H. Liang, F. Li, and Y. Man, "Robust visual tracking via an improved background aware correlation filter," *IEEE Access*, vol. 7, 2019.

[8] X. Dong, J. Shen, D. Yu, W. Wang, J. Liu, and H. Huang, "Occlusion-aware real-time object tracking," *IEEE Transactions on Multimedia*, vol. 19, no. 4, pp. 763–771, 2016.

[9] M. Fiaz, A. Mahmood, S. Javed, and S. K. Jung, "Handcrafted and deep trackers," *ACM Computing Surveys*, vol. 52, no. 2, pp. 1–44, 2019.

[10] D. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2544–2550, IEEE, New York, NY, USA, 2010, http://ieeexplore.ieee.org/document/5539960/.

[11] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, 2014.

[12] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Discriminative scale space tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 8, pp. 1561–1575, 2016.

[13] R. Tao, E. Gavves, and A. W. M. Smeulders, "Siamese instance search for tracking," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1420–1429, IEEE, New York, NY, USA, 2016, http://ieeexplore.ieee.org/document/7780527/%20.

[14] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan, "SiamRPN++-evolution of siamese visual tracking with very deep networks," *CoRR*, vol. 1812, 2018.

[15] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 502–504, 2006.

[16] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[17] P. Li, D. Wang, L. Wang, and H. Lu, "Deep visual tracking: review and experimental comparison," *Pattern Recognition*, vol. 13, no. 2, pp. 117–126, 2017.

[18] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Learning Spatially Regularized Correlation Filters for Visual Tracking," in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 4310–4318, IEEE, New York, NY, USA, 2015, http://ieeexplore.ieee.org/document/7410847/.

[19] S. Yun, J. Choi, Y. Yoo, K. Yun, and J. Y. Choi, "Action-decision networks for visual tracking with deep reinforcement learning," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1349–1358, IEEE, New York, NY, USA, 2017, http://ieeexplore.ieee.org/document/8099631/.

[20] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, "Fully-convolutional siamese networks for object tracking," in *Proceedings of the Computer Vision-ECCV 2016 Workshops*, pp. 850–865, Springer, Berlin, Germany, 2016.

[21] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4293–4302, IEEE, Berlin, Germany, 2016, http://ieeexplore.ieee.org/document/7780834/.

[22] H. Fan and H. Ling, "SANet: structure-aware network for visual tracking," in *Proceedings of the 2017 IEEE Conference on*

Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 2217–2224, IEEE, Berlin, Germany, May 2017, http://ieeexplore.ieee.org/document/8015009/.

[23] Z. Al-Halah and R. Stiefelhagen, "How to transfer? Zero-shot object recognition via hierarchical transfer of semantic attributes," 2015.

[24] O. Vinyals, C. Blundell, T. Lillicrap, and D. Wierstra, "Matching networks for one shot learning," in *Advances in Neural Information Processing Systems*, pp. 3630–3638, IEEE, Berlin, Germany, 2016.

[25] X. Dong, J. Shen, D. Wu, K. Guo, X. Jin, and F. Porikli, "Quadruplet network with one-shot learning for fast visual object tracking," *IEEE Transactions on Image Processing*, vol. 28, no. 7, pp. 3516–3527, 2019.

[26] H. Zhang, W. Ni, W. Yan, J. Wu, H. Bian, and D. Xiang, "Visual tracking using siamese convolutional neural network with region proposal and domain specific updating," *Neurocomputing*, vol. 275, pp. 2645–2655, 2018.

[27] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8971–8980, New York, NY, USA, 2018.

[28] K. Dai, D. Wang, H. Lu, C. Sun, and J. Li, "Visual tracking via adaptive spatially-regularized correlation filters," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4670–4679, London, UK, 2019.

[29] D. S. Bolme, B. A. Draper, and J. R. Beveridge, "Average of synthetic exact filters," in *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2105–2112, IEEE, Berlin, Germany, 2009.

[30] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *Proceedings of the European Conference on Computer Vision*, pp. 702–715, Springer, Berlin, Germany, 2012.

[31] M. Danelljan, G. Häger, F. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *Proceedings of the British Machine Vision Conference*, BMVA Press, Nottingham, UK, 2014.

[32] H. Zhou, Y. Yuan, and C. Shi, "Object tracking using sift features and mean shift," *Computer Vision and Image Understanding*, vol. 113, no. 3, pp. 345–352, 2009.

[33] C. Gárate, P. Bilinsky, and F. Bremond, "Crowd event recognition using hog tracker," in *Proceedings of the 2009 Twelfth IEEE International Workshop on PerforMance Evaluation of Tracking and Surveillance*, pp. 1–6, Berlin, Germany, 2009.

[34] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, "Convolutional features for correlation filter based visual tracking," in *Proceedings of the the IEEE International Conference on Computer Vision (ICCV) Workshops*, Berlin, Germany, December 2015.

[35] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: a benchmark," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, London, UK, 2013.

[36] R. B. Girshick, P. F. Felzenszwalb, and D. McAllester, "Discriminatively trained deformable part models," 2012.

Hindawi
Security and Communication Networks
Volume 2020, Article ID 6647562, 12 pages
https://doi.org/10.1155/2020/6647562

Research Article

A Privacy-Protection Model for Patients

Wenzhi Cheng, Wei Ou, Xiangdong Yin, Wanqin Yan, Dingwan Liu, and Chunyan Liu

School of Electronics and Information Engineering, Hunan University of Science and Engineering, Yongzhou 425199, Hunan, China

Correspondence should be addressed to Wei Ou; ouwei@huse.edu.cn

Received 2 October 2020; Revised 12 November 2020; Accepted 25 November 2020; Published 10 December 2020

Academic Editor: Honghao Gao

The collection and analysis of patient cases can effectively help researchers to extract case feature and to achieve the objectives of precision medicine, but it may cause privacy issues for patients. Although encryption is a good way to protect privacy, it is not conducive to the sharing and analysis of medical cases. In order to address this problem, this paper proposes a federated learning verification model, which combines blockchain technology, homomorphic encryption, and federated learning technology to effectively solve privacy issues. Moreover, we present a FL-EM-GMM Algorithm (Federated Learning Expectation Maximization Gaussian Mixture Model Algorithm), which can make model training without data exchange for protecting patient's privacy. Finally, we conducted experiments on the federated task of datasets from two organizations in our model system, where the data has the same sample ID with different subset features, and this system is capable of handling privacy and security issues. The results show that the model was trained by our system with better usability, security, and higher efficiency, which is compared with the model trained by traditional machine learning methods.

1. Introduction

Medical information contains personal information, which includes name, gender, age, and address, and the leakage of these information may cause serious risks to patients. Hathaliya and Tanwar [1] stated that patients' data is stored on cloud servers, and attackers can launch various security attacks, such as data confidentiality, privacy, and integrity attacks, which will pose a serious threat to patients. Prakash and Singaravel [2] stated that many organizations or companies collect personal privacy or sensitive data on the Internet, which means that personal information of sensitive data must be protected, and the individual has the right of management to access data or information for himself or herself. Papaioannou et al. [3] introduced that Internet of medical things (IoMT) provides e-health service for patients to help them improve their quality of life, but there are some attacks in the IoMT system (e.g., eavesdropping attacks, spoofing attacks, masquerading attacks, and Denial-of-Service attacks), which cause serious issues for patients. Bernd et al. [4] introduced that the advanced medical technology improves the efficiency of medical system and brings convenience to patients, but it also poses huge threats to the privacy of patients.

The collection and analysis of patient cases can effectively help researchers to extract case feature and to achieve the objectives of precision medicine, but it may cause privacy issues for patients. Hulsen et al. [5] informed that machine learning can identify patterns in biomedical data which can provide information for the development of clinical biomarkers or indicate unsuspected treatment targets with accelerating the goals of precision medicine, and patient information must be protected. Azencott [6] indicated that machine learning plays an important role in the development of precision medicine, which can tailor treatment plans based on the clinical or genetic features of patients, but these developments require the collection and sharing of a large amount of patient data or information, which will cause a crisis of patient privacy leakage. Gao et al. [7] and Yang et al. [8] proposed a sparsity alleviation recommendation approach that can achieve a better user information sharing and recommendation. Price et al. [9] stated that big data in health can help to measure hospital quality, to develop scientific hypothesis, and to monitor drug and device safety,

but there is little to protect patients from health privacy threats.

The medical data has also the risk of privacy data leakage in the communication channel, and attackers may obtain or tamper with user information from the channel. Gao et al. [10, 11] presented a Manhattan mobility model, which can improve and optimize the channel transmission quality to achieve the expected information transmission. Ma et al. [12] solved communication problems by optimized scheduling of servers. Gnad et al. [13] stated that vulnerabilities may cause channel attacks in Internet of Things, which will pose a serious threat to users.

Encryption is a good way to protect privacy, but it is not conducive to the sharing and analysis of medical cases. Tian et al. [14] explained that if a patient dies during treatment and the secret key is lost, it will inevitably lead to the unavailability of the data, which is not conducive to the further study of the case. Meanwhile, if the patient falls into a coma and cannot share the secret key to access the case, it will also affect the patient's treatment, which may delay treatment and cause death. In this situation, encryption will limit the rescue work or medical research of doctors, which will lead to serious issues.

In order to protect patient information privacy, many researchers have taken different measures to solve privacy issues. Sun et al. [15] discussed that existing solutions of privacy use data encryption, access control, and trusted third-party auditing in Internet of Medical Things (IoMT), but there is still Information Island of sharing. Li et al. [16] stated that a patient can encrypt and transmit medical information to a third-party trusted cloud service to authorize doctors to access it, and they presented two Secure and Efficient Dynamic Searchable Symmetric Encryption (SEDSSE) schemes for solving patient privacy. Cano and Cañavate-Sanchez [17] addressed an approach with a dual signature in ECDSA (Elliptic Curve Digital Signature Algorithm), which can protect the privacy of patient data transmitted from IoMT (Internet of Medical Things) devices to cloud servers. Liu et al. [18] presented two distinct RSSs (redactable signature schemes) with flexible release control (RSSs-FRC) to protect the privacy of the release of authenticated medical documents, which proves to have great advantages in security and efficiency. Hamza et al. [19] proposed a chaotic encryption password system based on privacy protection to protect the privacy of patients, which can protect the patient's image from being compromised by the broker.

Although encryption can solve the issues of patient information privacy, it is not conducive to the sharing of case information and its disadvantage of promoting the development of medical technology. In addition, if the patient's secret key is lost and the medical data cannot be shared, it will be detrimental to the research of treatment plans. Therefore, we propose a privacy-protection model with technology of blockchain, homomorphic encryption, and federated learning, which will be discussed in Section 2.

The rest structure of this article is as follows. Section 2 describes related work, which includes blockchain, homomorphic encryption, and federated learning. Section 3 proposes a privacy model and describes its algorithm. Section 4 validates the model with experiments and discusses the results. Section 5 concludes this paper and proposes directions for future research.

2. Related Work

2.1. Blockchain. Blockchain technology originated from Bitcoin, and Satoshi Nakamoto uses blockchain technology to solve the problem of maintaining the order of transactions and avoid double spending in Bitcoin [20]. Crosby et al. [21] stated that the underlying blockchain technology of the digital currency Bitcoin has worked perfectly and is widely used in the financial and nonfinancial fields, and the core of the blockchain is the distributed ledger, decentralization, smart contracts, and consensus mechanism. Bashir [22] defined distributed ledger, decentralization, smart contracts, and consensus mechanism; distributed ledger means that the ledger is distributed across the entire network among all peers in the network, and each peer has a copy of the complete ledger; the basic idea of decentralization is to assign control and authority to the periphery of the organization, rather than having a central agency completely control the organization; the smart contract is a secure and uninterruptible computer program that represents an agreement that can be automatically executed and enforced; the consensus mechanism is a set of steps taken by most or all nodes in the blockchain to reach a consensus on the state or value of the proposal.

Blockchain has advantages in solving privacy security, so some researchers use blockchain to solve medical privacy issues. Liu et al. [23] presented a blockchain-based privacy-preserving data sharing (BPDS) for EMRs (Electronic Medical Record), and the patient predefined access authorization to share secure data automatically through the smart contract of the blockchain. Zhang and Lin [24] proposed a blockchain-based secure and privacy-preserving PHI (Personal Health Information) sharing (BSPP) scheme which based on blockchain to improve the diagnosis of e-health systems. Ji et al. [25] presented a blockchain-based multilevel privacy-preserving location sharing (BMPLS) scheme, which can realize the sharing of exclusive protected locations based on blockchain in the telemedicine information system. Al Omar et al. [26] proposed a patient-centric medical data management system by using blockchain as storage to obtain privacy, and they can ensure security by using encryption to protect patient data.

Although blockchain can solve the privacy of patients, there are still some problems in the sharing of medical records between different hospitals in different regions. Meanwhile, some medical institutions even strictly prohibit patient record sharing in order to protect patient information, which is not conducive to the development of medical technology. Therefore, this paper intends to use blockchain technology as the bottom layer and introduce homomorphic encryption and federated learning to facilitate data sharing between different hospitals in different regions, which can guarantee patient privacy.

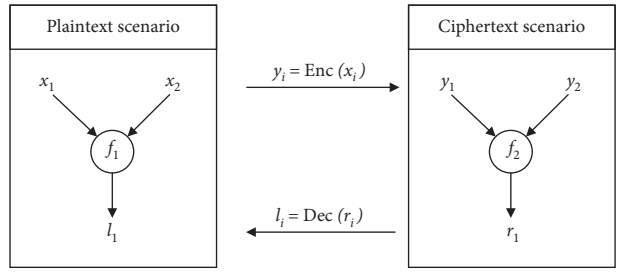


Figure 1: The process of homomorphic encryption. The function operations f_1 and f_2 are equivalent.

2.2. Homomorphic Encryption. Homomorphic encryption (HE) can perform meaningful calculations on encrypted data without decrypting the data, and the homomorphic encryption system can be used to encrypt the data to solve the privacy problem in a satisfactory manner [27]. Alloghani et al. [28] stated that homomorphic encryption can allow encrypted data to be calculated in the cloud server to avoid user privacy leakage. Acar et al. [29] discussed that the servers or the users with the key have high authority in the legacy encryption system, which leads to privacy and security issues, and homomorphic encryption (HE) is a solution of a special encryption scheme that can solve these issues for allowing any third party to operate on encrypted data without prior decryption.

Homomorphic encryption has good privacy-protection features, which is extremely suitable for sharing medical records, and some researchers apply homomorphic encryption to the medical field for privacy protection. Vengadapurvaja et al. [30] proposed an effective homomorphic encryption algorithm to encrypt medical images and perform useful operations on them without breaking confidentiality. Yang et al. [31] presented a safe and high-vision medical image framework to protect the privacy and security of medical images, and they adopted a novel Reversible Data Hiding (RDH) technology to embed private data into medical images and used homomorphic encryption which based on chaotic maps to ensure the security of medical images.

According to Acar et al. [29] and Gentry and Boneh [32], this paper describes the process of homomorphic encryption, which is shown in Figure 1.

As shown in Figure 1, both x_1 and x_2 are plaintext, and they can perform operation f_1 to get result l_1 on the client, which is no encryption operation in this process. In addition, $x_1$and x_2 are encrypted by function $y_i = \text{Enc}(x_i)$ (encryption process) to obtain ciphertexts $y_1$and y_2, and the ciphertexts can be sent to third-party application scenarios (e.g., cloud computing) for operations (such as calculations and search), which the result is ciphertext r_1 after $c_1$and c_2 are operated by function f_2. Finally, after the ciphertext R_1 passes through the decryption function $l_i = \text{Dec}(r_i)$(decryption process), the result is equal to the plaintext l_1, which is $l_1 = \text{Dec}(r_1)$.

We assume that an encryption scheme G is represented as (M, C, K, E, D), where M is the plaintext space; C is the ciphertext space; K is the key space; E is the encryption algorithm; D is the decryption algorithm; and $\oplus$ is the ciphertext-related operator. Therefore, we define some definitions of homomorphic encryption as follows.

Definition 1. Assuming P and L are operations, when the plaintext dataset $M = \{m_1, m_2, \ldots, m_n\}$, $k \in K$ if it satisfies$P(E_k(m_1), E_K \quad (m_2), \ldots, E_K(m_n)) = E_k(L(m_1, m_2, \ldots, m_n))$, then the encryption scheme is homomorphic for operation L. The basic idea of homomorphic encryption is to achieve the same effect as the corresponding plaintext operation by performing certain operations on multiple ciphertexts.

Definition 2. For any plaintext $m_i, m_j \in M$, the corresponding ciphertext is $c_i = E(m_i)$, $c_j = E(m_j)$ and $c_i, c_j \in C$. If $E(m_i + m_j) = E(m_i) \oplus E(m_j)$ or $D(E(m_i) \oplus E(m_j)) = m_i + m_j$ is true, then the encryption scheme G has additive homomorphism.

Definition 3. For any plaintext $m_i, m_j \in M$, the corresponding ciphertext is $c_i = E(m_i)$, $c_j = E(m_j)$, and $c_i, c_j \in C$. If $E(m_i \cdot m_j) = E(m_i) \oplus E(m_j)$ or $D(E(m_i) \oplus E(m_j)) = m_i \cdot m_j$ is true, then the encryption scheme G has multiplicative homomorphism.

Definition 4. For any plaintext $m_i, m_j \in M$, the corresponding ciphertext is $c_i = E(m_i)$, $c_j = E(m_j)$, and $c_i, c_j \in C$. If $E(m_i \cdot m_j) = E(m_i) \oplus m_j$ or $D(E(m_i) \oplus m_j) = m_i \cdot m_j$ is true, then the encryption scheme G has a homomorphic property with mixed multiplication.

Definition 5. If scheme G has both additive homomorphism and multiplication homomorphism properties and it can satisfy the finite numbers of addition and multiplication ciphertext operations, then the encryption scheme G is called a somewhat homomorphic encryption scheme.

Definition 6. If scheme G has both additive homomorphism and multiplication homomorphism properties and it can satisfy the any numbers of addition and multiplication ciphertext operations, then the encryption scheme G is called a fully homomorphic encryption scheme.

The homomorphic encryption process allows two or more different organizations to update the algorithm model without contact, which means that it can be performed on plaintext on encrypted data without decryption, and it can well protect user privacy date. Moreover, the above definitions will be applied in subsequent encryption algorithms, which are shown in Section 3.2.

With the technical support of the blockchain in the previous section, it can ensure that data can be safely and accurately transmitted in the communication channel. Meanwhile, homomorphic encryption can guarantee that users' private data participate in calculations or statistics in encrypted form, which avoids the risk of data leakage. In subsequent sections, this article will briefly introduce federated learning, which guides user privacy data from different institutions or organization on how to participate in training.

2.3. Federated Learning. Federated learning (FL) involves training statistical models on remote devices or isolated data centres with keeping data localized, which can protect user privacy as much as possible [33]. Li et al. [34] stated that federated learning enables multiple parties to jointly train machine learning models without exchanging local data, which covers technologies from multiple research fields, such as distributed systems, machine learning, and privacy. According to Li et al. [33, 34] and Yang et al. [35], we could draw an illustration of federated learning system, which is shown in Figure 2.

In Figure 2, federated learning is divided into central training and local training. The federated learning system identifies k clients, and each client trains its private dataset ρ_k in L-SGD (Local Stochastic Gradient Descent). Then, the final results of the entire dataset are aggregated to obtain the final training results in the central server. Next, the w_i is the trained parameter, which is calculated and generated by the client, and the central server is responsible for collecting parameter w_i and coordinating processing. Finally, the server will obtain the latest model parameters and assign them to each client for completing the parameter update of the entire federated learning system after all rounds of data training.

In addition, federated learning can be applied in the medical field to solve the privacy problem of patients. Xu et al. [36] surveyed the applications of federated learning in the medical field, and federated learning keeps sensitive data locally, which provides an effective solution for the data source and privacy protection of medical institutions. Sheller et al. [37] stated that federated learning can promote multiagency and cross-field cooperation without sharing medical data, which can well protect the privacy of patients and improve the development of precision medicine. Yang et al. [38] described federated learning into the following three categories, which are shown in Figure 3.

As shown in Figure 3, federated learning is divided into three categories by Yang et al., and the detailed explanation is as follows:

(1) *Horizontal Federated Learning.* When the features of different users of the two datasets more overlap, horizontal federated learning can extract the same feature data of different users for training. For instance, the user groups of the two banks in different regions have less intersections, but their businesses are very similar; therefore, the recorded user characteristics are the same, and horizontal federated learning can be used to construct a federated model. In this case, the bank can get a better training set or data result without uploading user information, which can protect user privacy.

(2) *Vertical Federated Learning.* Vertical federated learning is similar to horizontal federated learning, but is different in data feature extraction, where vertical federated learning extracts data of the same user in different dimensions. For example, two organizations with different businesses conduct data analysis on users in the same area, which can analyse the characteristic values of the same users to achieve better data results.

(3) *Federated Transfer Learning.* When the user characteristics and samples overlap in the two datasets are less, federated transfer learning can be used to solve the lack of data for analysis. For example, banks in two different countries have different user characteristics and samples, and federated transfer learning can be used for effective data analysis.

To sum up, federated learning can easily solve user data analysis problems without uploading user information, which can protect user privacy. In order to understand the application of federated learning in this article, we have made the following definition.

The training process can be regarded as an optimization problem in machine learning, and its formula can be expressed as follows:

$$\min_{\omega \in R^d} f(\omega), \tag{1}$$

$$f(\omega) \triangleq \frac{1}{n} \sum_{i=1}^{n} f_i(\omega). \tag{2}$$

In formulas (1) and (2), $f(\omega)$ stands for the loss function and $f_i(\omega)$ is the loss corresponding to the prediction of the data point with index i. For client k, there are n_k training points in the dataset on client k, where $n_k = |\rho_k|$. Therefore, the optimization problem of federated learning can be reexpressed as

$$\begin{aligned} f(\omega) &= \sum_{k=1}^{k} \frac{n_k}{n} \cdot F_k(\omega), \\ F_k(\omega) &= \frac{1}{n_k} \sum_{i \in \rho_k} f_i(\omega). \end{aligned} \tag{3}$$

In addition, it is difficult for federated learning to satisfy the independent and identical distribution assumption (IID Assumption), in which the training data is uniformly and randomly distributed on each client.

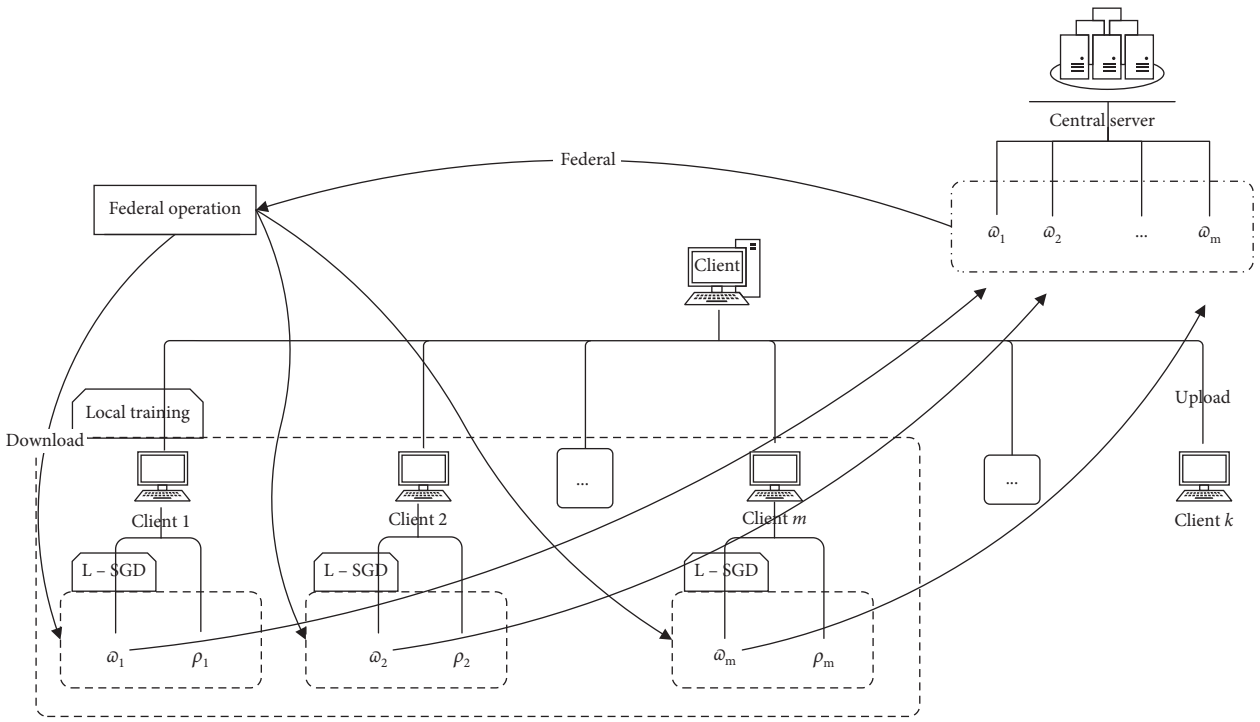


Figure 2: Federated Learning System. L-SGD means Local Stochastic Gradient Descent.

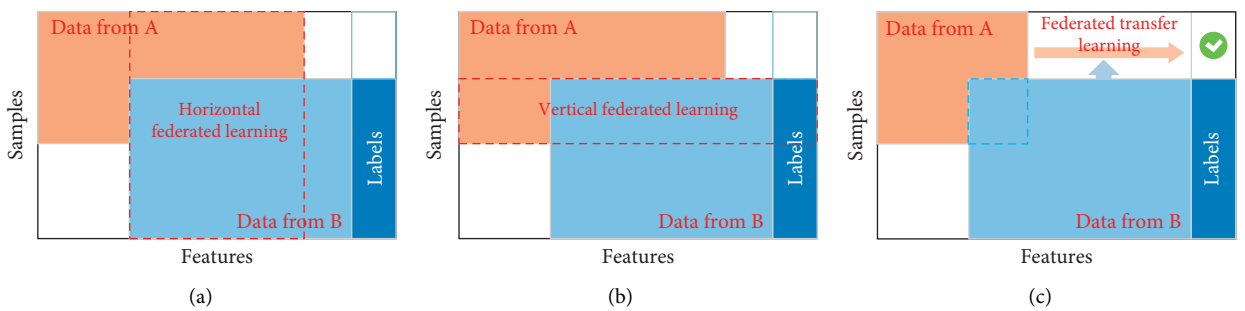


Figure 3: Categorization of federated learning. (a) Horizontal federated learning. (b) Vertical federated learning. (c) Federated transfer learning.

3. Models and Algorithms

3.1. Federated Learning Verification Model. This part will explain the design and deployment of the model and gives the following two assumptions.

Normal Distribution Assumption. Assume that each assumption sample is a D-dimensional normal distribution drawn from k samples, and each is Normal (μ_k, σ_k^2)

Independence Assumption. Assuming that the two datasets are independent, for each Normal (μ_k, σ_k^2), satisfy Normal $(\mu_k, \sigma_k^2) \propto$ Normal $(\mu 1_k, \sigma 1_k^2) \cdot$ Normal $(\mu 2_k, \sigma 2_k^2)$

We established the following model scheme, which is based on the above assumptions, and the model scheme obeys the following steps:

(1) Establish a Gaussian mixture model based on the clustering problem, and use the EM (Expectation Maximization) algorithm to update the parameters.

(2) Build FL (federated learning) server. The main job of the server is to send the FL scheme to the clients, to receive the trained parameters, and to integrate the federated distribution tasks to the engineers for analysis.

(3) The clients preprocess the data. Each client preprocesses its own data, which uses encryption

algorithms to encrypt sensitive data and stores it in a fixed area.

(4) The server assigns tasks to the clients. At this stage, the server sends a signal to the client to propose the conditions required for the training plan, such as memory, container, and size of the collected data.

(5) The server receives the response from the clients. After the client receives the signal from the server, it responds to the server and returns information (e.g., data size, time, etc.) to the server for collecting data.

(6) The server initializes the parameters to the client. The server initializes the parameters: probability matrix ϕ, category probability θ, and $\mu 1_0$, $\sigma 1_0^1$, $\mu 2_0$, $\sigma 2_0^1$, $\ldots\ldots$ $\mu 1_n$, $\sigma 2_0^1$.

(7) The server sends the training plan to the clients. For example, the server sends probability matrices ϕ, $\mu 1_0$, and $\sigma 1_0^1$ to client A and sends probability matrices ϕ, $\mu 2_0$, and $\sigma 2_0^1$ to client B.

(8) The clients start to train the data based on the requirement of the server. At the first step, client A locally updates $\mu 1_{t-1}, \sigma 1_{t-1}^2$ and the given probability matrix ϕ, which obtains an updated D1-dimensional normal distribution Normal$(\mu 1_t, \sigma 1_t^2)$. Meanwhile, client B will get an updated D2-dimensional normal distribution Normal$(\mu 2_t, \sigma 2_t^2)$ with the similar steps.

(9) After finishing the training, the result will be sent to the server (e.g., client A sent $\mu 1_t$ and $\sigma 1_t^2$, and client B sent $\mu 2_t$ and $\sigma 2_t^2$), and data is encrypted by Paillier, which is a homomorphic encryption algorithm.

(10) The server updates the parameters from the client and starts a new cycle towards the end.

We built a federated learning verification system for conducting subsequent simulation experiments, which is shown in Figure 4.

In Figure 4(a), it is a federated model of data alignment, which is a local client training model, and this model makes sure that privacy data cannot be exchanged or updated. In Figure 4(b), we have designed a vertical federated learning model, which conducted "noninteractive" connection for data platforms with similar samples and different indicator dimensions, and the model can realize the collaborative calculation of the expanded sample size with similar index data, which can improve the overall safety of the system and the integrity and comprehensiveness of the analysis results. Next, the system is divided into the user layer and edge service layer. The user layer is composed of Internet of Things (IoT) equipment, mobile terminals, etc., and the server layer is equipped with mobile edge computing servers, which are composed of base stations with storage and computing capabilities. Meanwhile, the federated learning local training runs on the client of user side (e.g., Agency A and Agency B), which learns the local model parameters with based on the data of user side.

In addition, we have designed an operation flowchart of the system, and it helps the following experimental work, which is shown in Figure 5. As shown in Figure 5, it is an operation flowchart of the system, which is from Figure 4. According to Figures 4 and 5, the operation of the model system is divided into two parts: client and server.

(1) The client participates in local data training, which uses an algorithm based on gradient descent to find model parameters for minimizing the loss function.

(2) The server collects the model parameters with being trained by the client, which collects parameters from various clients and federated model parameters and updates the overall model, and the server sends the new model to the clients for starting a new round of training and learning.

As shown in Figure 5, the client will select a model and an agency from the server for starting, which follows the system in Figure 4. Next, if the client contains an existing model, it will update the model from the selected model, or it will use the selected model. Moreover, if the client needs to update the model, it will request communication to train the model, or it will send the existing model to the central server for obtaining results.

In the process of data training, the user privacy data will not be shared or calculated in the form of plain text, and they will be securely encrypted in the communication channel by using blockchain and homomorphic encryption technology. In addition, the client does not need to upload or share the private data of the local user and gets the final result after the training of the model. Therefore, our model shows that it can protect user privacy data.

3.2. A FL-EM-GMM Algorithm. In the previous section, we designed a federated learning verification model, which will be made an experiment in the next section. In this section, we have completed the model algorithm, which is FL-EM-GMM Algorithm (Federated Learning Expectation Maximization Gaussian Mixture Model Algorithm), and it is shown in Algorithm 1.

As shown in Algorithm 1, it is a FL-EM-GMM algorithm, which displays the detail of federated learning process. At the server steps, the algorithm will complete the initialization work which is based on the client's data, and it will assign task parameters to the client. Next, at the client steps, the algorithm will finish the training process which is based on server parameters and local model parameters to obtain the best training dataset, and it will share the trained parameters to the server. Finally, the clients will update the latest model according to the server's parameters after the

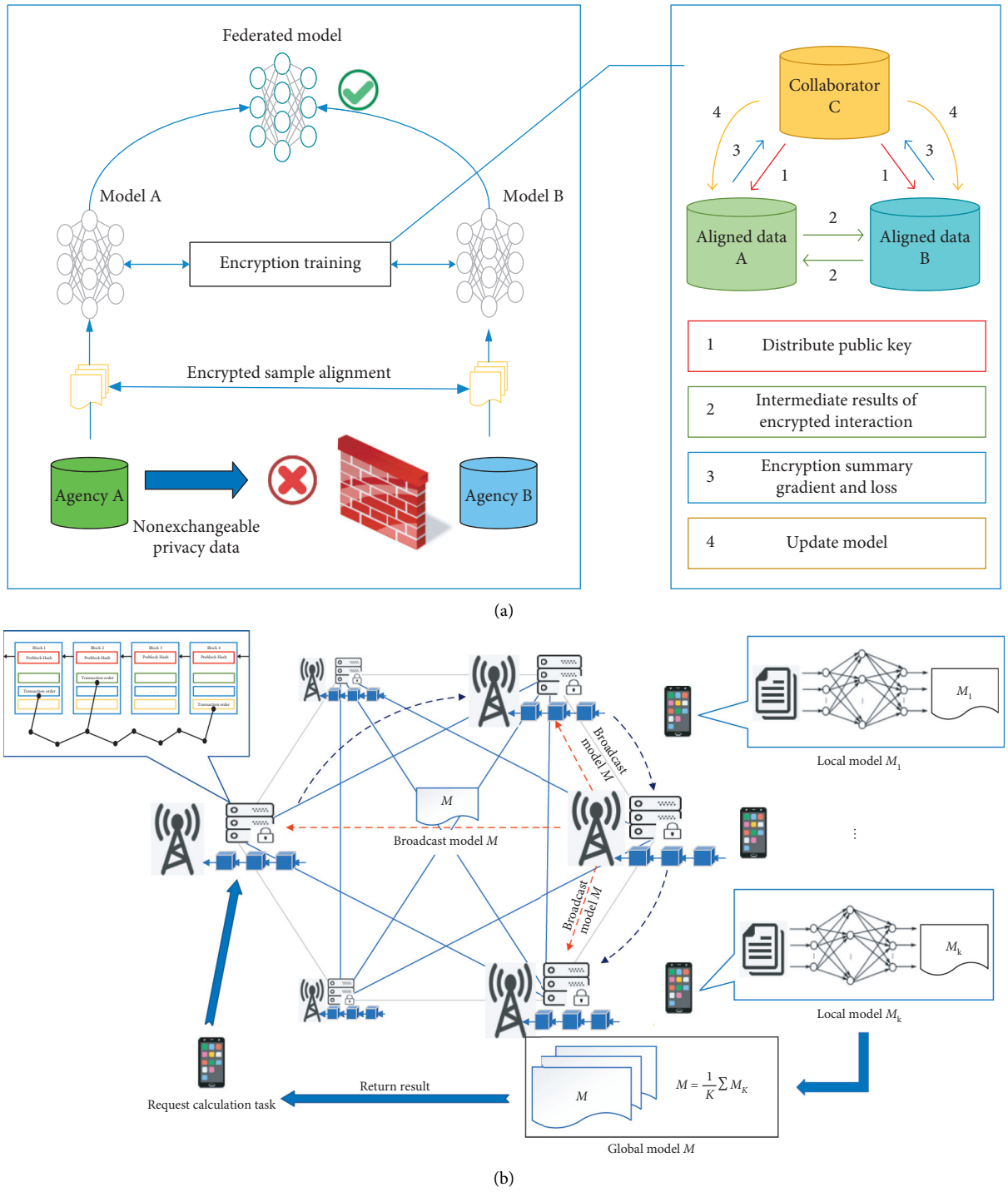


Figure 4: (a) Federated model of data alignment. (b) Federated learning verification model system.

cycle ends, and the parameters are trained in local, which avoids the risk of privacy data leakage.

4. Experiment and Simulation

In experiment, we used FATE official components [39] and three servers for testing, and the data source is provided by FATE official, which is breast cancer patients' data (<569, 30>). Next, we use the previously designed model and system to ensure that privacy data is not leaked, which achieves open patients' records and collaborative diagnosis. The experiment deploys FATE on two machines, and each machine runs a FATE instance. In order to judge whether the patient is benign or malignant, the basic information of the patient is obtained from the database, such as ID number, standard deviation of grey values, and average size

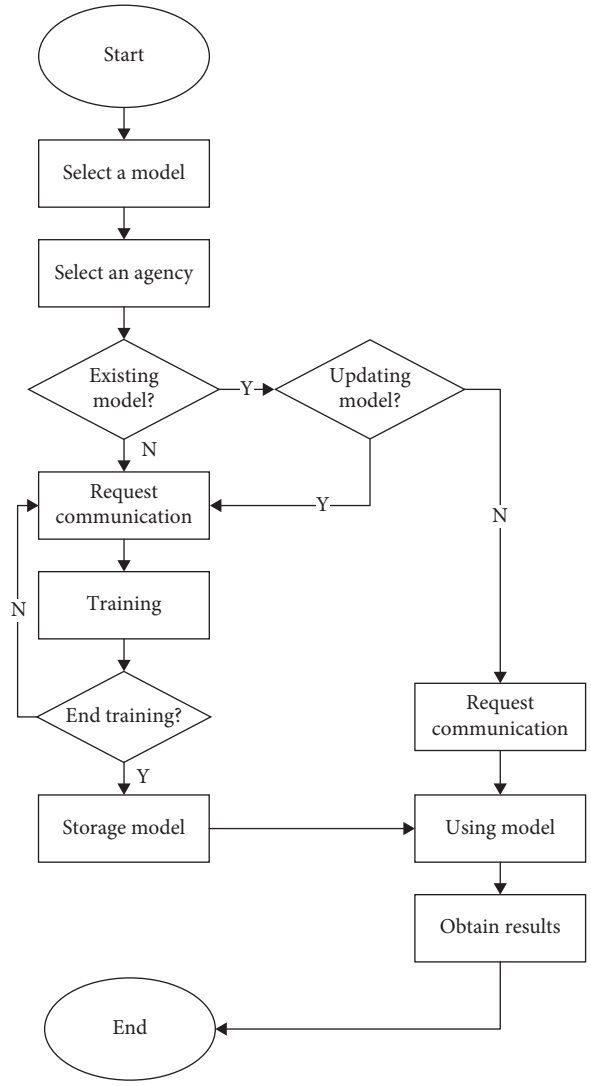


Figure 5: An operation flowchart of system.

of core tumour. While uploading these source data to the server, the server will normalize and process them to desensitize the data and enter it into the system. In data analysis, some key terms are as follows:

(1) *Feature selection*. Feature selection is to find the correlation between X and Y labels, which is called IV (Information Value). Since not every X and Y are correlated, we exclude the X with lower correlation for reducing noise.

(2) *Feature binning*. As the normalized data, most of them become very small values, so the regressed model is too sensitive to the requirements of parameter accuracy. Therefore, we perform simple binning, which change the continuous value into several segment types and make the data into 1–10 (10 discrete values of labels), and it helps to improve variable expression ability and model discrimination.

The feature binning parameters are shown in Table 1.

In Table 1, WoE means Weight of Evidence and IV means Information Value. Meanwhile, according Table 1, we draw Figures 6 and 7.

In Figure 6, we have differentiated and compared the data of Event Count and Nonevent Count, and it shows that the value is related to the choice of feature binning parameters. Figure 6 shows that, as the number of training increases, the Event Count has decreased, which means the parameters are effectively trained.

Meanwhile, we draw Figure 7 based on WoE (Weight of Evidence) in Algorithm 1. The greater the WoE value is, the greater the possibility that the cancer is benign. Conversely, the smaller the WoE value is, the greater the possibility that the cancer is malignant. Figure 7 indicates that cancer patients can be screened by the value of WoE for better disease prevention and achieve the goal of precision medicine.

Input: Data x and y from dataset A and dataset B, number of clusters K.
Output: GMM parameters θ, μ and σ^2 clusters assignment distribution ϕ_i.
(1) The server initializes parameters θ, μ_0 and σ_0^2.
(2) for $t = 1, \ldots, T$//Iteration
(3) for $i = 1, \ldots, N$
(4) for $j = 1, \ldots, K$
(5) //#The server:
(6) //At E-step, calculate the formula below.
(7) $\phi_i(j)_{(t)} = \theta_{j_{(t-1)}} \cdot \text{Normal}(x_i|\mu 1_{j_{(t-1)}}, \sigma 1^2_{j_{(t-1)}}) \cdot \text{Normal}(y_i|\mu 2_{j_{(t-1)}}, \sigma 2^2_{j_{(t-1)}}) / \sum_{k=1}^{K} \pi_{k_{(t-1)}} \text{Normal}(x_i|\mu 1_{k_{(t-1)}}, \sigma 1^2_{k_{(t-1)}}) \cdot \text{Normal}(y_i|\mu 2_{k_{(t-1)}}, \sigma 2^2_{k_{(t-1)}})$
(8) //At M-step, calculate the formula below.
(9) $n_{j_{(t)}} = \sum_{i=1}^{n} \phi_i(j)_{(t)}$
(10) $\theta_{j_{(t)}} = n_{j_{(t)}}/n$
(11) //# Client A: At M-step, calculate the formula below, the output parameter data is encrypted with partial homomorphic cryptographic algorithm paillier.
(12) $A(\mu_{j_{(t)}}) = 1/n_{j_{(t)}} \sum_{i=1}^{n} \phi_i(j)_{(t)} x_i$
(13) $A(\sigma^2_{j_{(t)}}) = (1/n_{j_{(t)}} \sum_{i=1}^{n} \phi_i(j)_{(t)} (x_i - \mu_{i(t)})(x_i - \mu_{i(t)})^T)^{-1}$
(14) //# Client B: At M-step, calculate the formula below, the output parameter data is encrypted with partial homomorphic cryptographic algorithm paillier.
(15) $B(\mu_{j_{(t)}}) = 1/n_{j_{(t)}} \sum_{i=1}^{n} \phi_i(j)_{(t)} y_i$
(16) $B(\sigma^2_{j_{(t)}}) = (1/n_{j_{(t)}} \sum_{i=1}^{n} \phi_i(j)_{(t)} (y_i - \mu_{i(t)})(y_i - \mu_{i(t)})^{T-1})$
(17) end for
(18) end for
(19) end for

Algorithm 1: FL-EM-GMM algorithm.

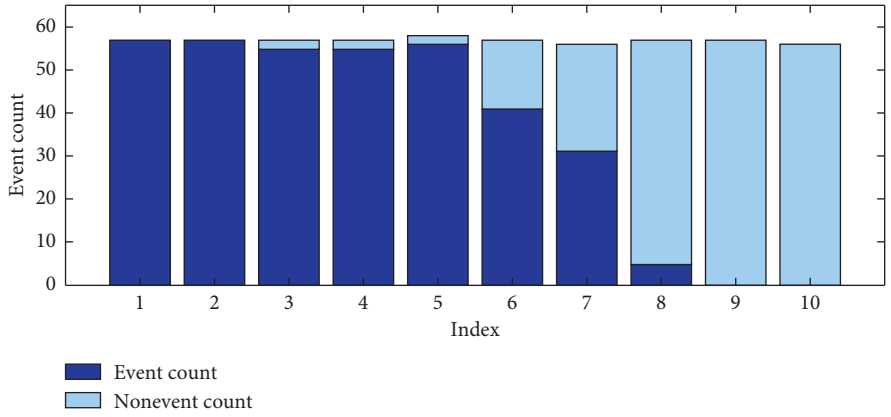


Figure 6: Data difference of event count and nonevent count.

We have divided the machines into guest and host for data privacy protection, and guest machine is the data application side, while host machine is the data provider. The guest machine uses the IV (Information Value) to check the binning effect, but it cannot determine the characteristic values X and Y, which can protect patient's privacy. We obtained a set of values after training, and we only displayed the first 10 results for the reason that there are too many training results, which are shown in Table 2.

In Table 2, "predict_score" is the probability of logistic regression prediction output. When the probability of "predict_score" is greater than 0.5, "predict_result" is 1, or it is 0. In addition, when "predict_result" is 0, the result is benign, and when it is 1, the result is malignant. According

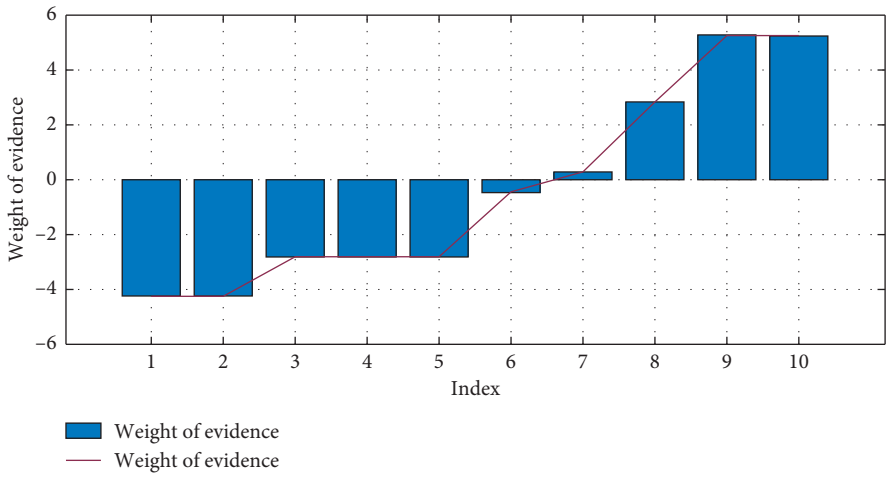


Figure 7: Weight of evidence.

Table 1: The result of feature binning parameters.

Index	Binning	IV	WoE	Event_Count	Event_Ratio	Non_Event_Count	Non_Event_Ratio
1	$x0 \leq -1.047670$	0.670339	−4.223783	57	0.161064	0	0.002358
2	$-1.047670 < x0 \leq -0.784675$	0.670339	−4.223783	57	0.161064	0	0.002358
3	$-0.784675 < x0 \leq -0.612797$	0.403950	−2.793036	55	0.154062	2	0.009434
4	$-0.612797 < x0 \leq -0.469910$	0.403950	−2.793036	55	0.154062	2	0.009434
5	$-0.469910 < x0 \leq -0.269040$	0.414430	−2.811055	56	0.156863	2	0.009434
6	$-0.269040 < x0 \leq -0.053674$	0.016531	−0.419834	41	0.114846	16	0.075472
7	$-0.053674 < x0 \leq 0.232100$	0.009515	0.306038	31	0.086835	25	0.117925
8	$0.232100 < x0 \leq 0.840923$	0.662137	2.862955	5	0.014006	52	0.245283
9	$0.840923 < x0 \leq 1.536720$	1.420925	5.266082	0	0.001401	57	0.271226
10	$X0 > 1.536720$	1.391434	5.248537	0	0.001401	56	0.266509

Table 2: The result of training.

Index	Label	Predict_result	Predict_score	Predict_detail	Type	Other
1	0	0	0.078641	{'0': 0.9213587838889536	'1': 0.07864121611104637}	Train
2	0	0	0.242046	{'0': 0.7579539730347816	'1': 0.2420460269652184}	Train
3	0	0	0.295851	{'0': 0.7041491549790799	'1': 0.29585084502092}	Train
4	0	1	0.953959	{'0': 0.04604114486783373	'1': 0.9539588551321663}	Train
5	0	1	0.598822	{'0': 0.401177819540372	'1': 0.598822180459628}	Train
6	0	1	0.980817	{'0': 0.019183466307048036	'1': 0.980816533692952}	Train
7	0	0	0.049818	{'0': 0.95018217382329	'1': 0.04981782617671007}	Train
8	1	0	0.003503	{'0': 0.9964965574508085	'1': 0.00350344254919155}	Train
9	1	0	0.02129	{'0': 0.978709874597958	'1': 0.021290125402041987}	Train
10	0	0	0.35347	{'0': 0.6465295143293885	'1': 0.3534704856706114}	Train

to statistics, the proportion of "1" is 60.61%, and the proportion of "0" is 39.39% during this experiment.

5. Conclusions

The emergence of federated learning provides new ideas for artificial intelligence to break the data barrier and further develop, and it allows data owners to jointly establish a common model with the premise of protecting local data, which solves the privacy and data security issues of medical institutions. In order to solve the problem of privacy leakage of medical patients, we designed a federate learning verification model and an FL-EM-GMM Algorithm (Federated Learning Expectation Maximization Gaussian Mixture Model Algorithm), and we have verified the algorithm, which shows that medical privacy data training can be completed in local to avoid the risk of data leakage. Moreover, we conducted experiments on the federated task of datasets from two organizations in our model system, in which the data has the same sample ID with different subset features, and this system is capable of handling privacy and security issues. The results show that the model trained by our system has better usability, security, and higher efficiency, compared with the model trained by traditional machine learning methods.

In the future, we will build an improved federated learning system with customizable password algorithms, which will make the federated learning system more flexible and easier to implement. Meanwhile, we will improve the compatibility of homomorphic encryption algorithms and federated learning systems to make them more efficient and stable.

Data Availability

In the experiment, we used FATE official components and three servers for testing, and the data source is provided by FATE official, which is breast cancer patients' data, in https://github.com/FederatedAI/FATE.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors would like to express appreciation for the support of the Construct Program of Applied Characteristic Discipline and the Research Project (Project No. 19XKY053) in Hunan University of Science and Engineering, Key Research and Development Project of Hunan Provincial Science & Technology Department (Project No. 2017NK2390), and Yongzhou Guiding Science and Technology Program (Project No. 2019-yzkj-10).

References

[1] J. J. Hathaliya and S. Tanwar, "An exhaustive survey on security and privacy issues in healthcare 4.0," *Computer Communications*, vol. 153, pp. 311–335, 2020.

[2] M. Prakash and G. Singaravel, "An approach for prevention of privacy breach and information leakage in sensitive data mining," *Computers & Electrical Engineering*, vol. 45, pp. 134–140, 2015.

[3] M. Papaioannou, M. Karageorgou, G. Mantas et al., "A survey on security threats and countermeasures in internet of medical things (IoMT)," *Transactions on Emerging Telecommunications Technologies*, vol. e4049, 2020.

[4] B. Bernd, D. M. Lopez, and C. Gonzalez, "Patient privacy and security concerns on big data for personalized medicine," *Health and Technology*, vol. 6, no. 1, pp. 75–81, 2016.

[5] T. Hulsen, S. S. Jamuar, A. R. Moody et al., "From big data to precision medicine," *Frontiers in Medicine*, vol. 6, no. 34, 2019.

[6] C.-A. Azencott, "Machine learning and genomics: precision medicine versus patient privacy," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 376, no. 2128, Article ID 20170350, 2018.

[7] H. Gao, L. Kuang, Y. Yin, B. Guo et al., "Mining consuming behaviors with temporal evolution for personalized recommendation in mobile marketing apps," *ACM/Springer Mobile Networks and Applications (MONET)*, vol. 25, no. 4, pp. 1233–1248, 2020.

[8] X. Yang, S. Zhou, and M. Cao, "An approach to alleviate the sparsity problem of hybrid collaborative filtering based recommendations: the product-attribute perspective from user reviews," *Mobile Networks and Applications*, vol. 25, no. 2, pp. 376–390, 2020.

[9] W. N. Price, I. G. Cohen, and I. Glenn Cohen, "Privacy in the age of medical big data," *Nature Medicine*, vol. 25, no. 1, pp. 37–43, 2019.

[10] H. Gao, C. Liu, Y. Li et al., "V2VR: reliable hybrid-network-oriented V2V data transmission and routing considering RSUs and connectivity probability," *IEEE Transactions on Intelligent Transportation Systems*, vol. 99, pp. 1–14, 2020.

[11] H. Gao, W. Huang, and Y. Duan, "The cloud-edge based dynamic reconfiguration to service workflow for mobile ecommerce environments: a QoS prediction perspective," *ACM Transactions on Internet Technology*, vol. 2020, Article ID 8843584, 14 pages, 2020.

[12] X. Ma, H. Gao, H. Xu et al., "An IoT-based task scheduling optimization scheme considering the deadline and cost-aware scientific workflow for cloud computing," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019249 pages, 2019.

[13] D. R. E. Gnad, J. Krautter, and M. B. Tahoori, "Leaky noise: new side-channel attack vectors in mixed-signal IoT devices," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 3, pp. 305–339, 2019.

[14] H. Tian, J. He, and Y. Ding, "Medical data management on blockchain with privacy," *Journal of Medical Systems*, vol. 43, no. 2, p. 26, 2019.

[15] W. Sun, Z. Cai, Y. Li et al., "Security and privacy in the medical internet of things: a review," *Security and Communication Networks*, vol. 2018, Article ID 5978636, 9 pages, 2018.

[16] H. Li, Y. Yang, Y. Dai et al., "Achieving secure and efficient dynamic searchable symmetric encryption over medical cloud data," *IEEE Transactions on Cloud Computing*, vol. 991 page, 2017.

[17] M. D. Cano and A. Cañavate-Sanchez, "Preserving data privacy in the internet of medical things using dual signature ECDSA," *Security and Communication Networks*, vol. 2020, Article ID 4960964, 9 pages, 2020.

[18] J. Liu, J. Ma, Y. Xiang et al., "Authenticated medical documents releasing with privacy protection and release control," *IEEE Transactions on Dependable and Secure Computing*1 page, 2019, https://ieeexplore.ieee.org/document/8611220.

[19] R. Hamza, Z. Yan, K. Muhammad, P. Bellavista, and F. Titouna, "A privacy-preserving cryptosystem for IoT E-healthcare," *Information Sciences*, vol. 527, pp. 493–510, 2020.

[20] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," *Manubot*, 2019, https://git.dhimmel.com/bitcoin-whitepaper/.

[21] M. Crosby, P. Pattanayak, S. Verma et al., "Blockchain technology: beyond bitcoin," *Applied Innovation*, vol. 2, no. 6–10, 2016.

[22] I. Bashir, *Mastering Blockchain: Distributed Ledger Technology, Decentralization, and Smart Contracts Explained*, Packt Publishing Ltd, Birmingham, UK, 2018.

[23] J. Liu, X. Li, L. Ye et al., "BPDS: a blockchain based privacy-preserving data sharing for electronic medical records," in *Proceedings of the 2018 IEEE Global Communications Conference (GLOBECOM)*, December 2018.

[24] A. Zhang and X. Lin, "Towards secure and privacy-preserving data sharing in e-health systems via consortium blockchain," *Journal of Medical Systems*, vol. 42, no. 8, p. 140, 2018.

[25] Y. Ji, J. Zhang, J. Ma et al., "BMPLS: blockchain-based multi-level privacy-preserving location sharing scheme for telecare medical information systems," *Journal of Medical Systems*, vol. 42, no. 8, p. 147, 2018.

[26] A. Al Omar, M. S. Rahman, A. Basu et al., "Medibchain: a blockchain based privacy preserving platform for healthcare data," in *Proceedings of the International Conference on Security, Privacy and Anonymity in Computation, Communication and Storage*, pp. 534–543, Springer, Cham, Switzerland, December 2017.

[27] L. Zhang, Y. Zheng, and R. Kantoa, "A review of homomorphic encryption and its applications," in *Proceedings of the 9th EAI International Conference on Mobile Multimedia Communications*, pp. 97–106, Xi'an China, June 2016.

[28] M. Alloghani, M. Alani, D. Al-Jumeily et al., "A systematic review on the status and progress of homomorphic encryption technologies," *Journal of Information Security and Applications*, vol. 48, Article ID 102362, 2019.

[29] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, "A survey on homomorphic encryption schemes," *ACM Computing Surveys*, vol. 51, no. 4, pp. 1–35, 2018.

[30] A. M. Vengadapurvaja, G. Nisha, R. Aarthy, and N. Sasikaladevi, "An efficient homomorphic medical image encryption algorithm for cloud storage security," *Procedia Computer Science*, vol. 115, pp. 643–650, 2017.

[31] Y. Yang, X. Xiao, X. Cai, and W. Zhang, "A secure and high visual-quality framework for medical images by contrast-enhancement reversible data hiding and homomorphic encryption," *IEEE Access*, vol. 7, pp. 96900–96911, 2019.

[32] C. Gentry and D. Boneh, *A fully homomorphic encryption scheme*, Stanford University, Stanford, CA, USA, 2009.

[33] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[34] Q. Li, Z. Wen, Z. Wu et al., "A survey on federated learning systems: vision, hype and reality for data privacy and protection," 2019, https://arxiv.org/abs/1907.09693.

[35] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 2, pp. 1–19, 2019.

[36] J. Xu, S. Benjamin, S. Chang et al., "Federated learning for healthcare informatics," 2019, https://arxiv.org/abs/1911.06270.

[37] M. J. Sheller, B. Edwards, G. A. Reina et al., "Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data," *Scientific Reports*, vol. 10, no. 1, pp. 1–12, 2020.

[38] Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, and H. Yu, "Federated learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 13, no. 3, pp. 1–207, 2019.

[39] https://github.com/FederatedAI/FATE.

Hindawi
Security and Communication Networks
Volume 2020, Article ID 6697640, 11 pages
https://doi.org/10.1155/2020/6697640

Research Article

A Novel Coevolutionary Approach to Reliability Guaranteed Multi-Workflow Scheduling upon Edge Computing Infrastructures

Zhenxing Wang,[1] **Wanbo Zheng**,[2] **Peng Chen**,[3] **Yong Ma**,[4] **Yunni Xia**,[1] **Wei Liu**,[5] **Xiaobo Li**,[6] **and Kunyin Guo**[1]

[1]*Software Theory and Technology Chongqing Key Lab, Chongqing University, Chongqing, China*
[2]*School of Mathematics, Kunming University of Science and Technology, Kunming, Yunnan 650500, China*
[3]*School of Computer and Software Engineering, Xihua University, Chengdu, Sichuan 610065, China*
[4]*School of Computer and Information Engineering, Jiangxi Normal University, Nanchang 330022, China*
[5]*Shanghai Jiaotong University Chongqing Research Institute, Chongqing 401121, China*
[6]*Chongqing Animal Husbandry Techniques Extension Center, Chongqing 401121, China*

Correspondence should be addressed to Peng Chen; chenpeng@mail.xhu.edu.cn, Yong Ma; may@jxnu.edu.cn, and Yunni Xia; xiayunni@hotmail.com

Received 14 October 2020; Revised 4 November 2020; Accepted 23 November 2020; Published 8 December 2020

Academic Editor: Honghao Gao

Recently, mobile edge computing (MEC) is widely believed to be a promising and powerful paradigm for bringing enterprise applications closer to data sources such as IoT devices or local edge servers. It is capable of energizing novel mobile applications, especially the ultra-latency-sensitive ones, by providing powerful local computing capabilities and lower end-to-end delays. Nevertheless, various challenges, especially the reliability-guaranteed scheduling of multitask business processes in terms of, e.g., workflows, upon distributed edge resources and servers, are yet to be carefully addressed. In this paper, we propose a novel edge-environment-based multi-workflow scheduling method, which incorporates a reliability estimation model for edge-workflows and a coevolutionary algorithm for yielding scheduling decisions. The proposed approach aims at maximizing the reliability, in terms of success rates, of services deployed upon edge infrastructures while minimizing service invocation cost for users. We conduct simulative experimental case studies based on multiple well-known scientific workflow templates and a well-known dataset of edge resource locations as well. Simulative results clearly suggest that our proposed approach outperforms traditional ones in terms of workflow success rate and monetary cost.

1. Introduction

Edge computing is an evolving computing paradigm offering a more efficient alternative: data is processed and analyzed closer to the point where it is created. It enables computation as a service model and prepares a proximity-based and mobility-aware resource provisioning model of virtualized resources applicable on demand [1, 2] The edge service providers are equipped with computational facilities, which allow them to provide necessary spaces required by commercial and noncommercial users. Recently, the edge computing paradigm has evolved as an increasingly popular force for supporting and enabling business process and scientific workflow execution [3–5]. A workflow is a set of dependent or independent tasks illustrated as a directed acyclic graph (DAG) [6–8], in which the nodes indicate the tasks and a directed arch represents the interdependency among the corresponding tasks. Workflow scheduling involves mapping workflow tasks to computational resources for execution, and the resulting optimization problem is well acknowledged to be NP-hard.

Recently, as novel bioinspired and genetic algorithms are becoming increasingly versatile and powerful, a great deal of research efforts are paid to applying them in dealing with edge-environment-oriented workflow scheduling problem [9–11]. However, it remains a great challenge to develop efficient scheduling algorithm with good scheduling performance, low service-level-agreement (SLA) violation rate, and high user-perceived quality of service.

In this paper, we propose a novel edge-environment-based multi-workflow scheduling approach by leveraging a multi-workflow-reliability estimation model and preference-inspired coevolutionary algorithms, i.e., PICEA-g, for yielding scheduling decisions. We show through simulative studies as well that our proposed method clearly outperforms traditional ones in terms of multiple metrics.

2. Literature Review

2.1. Related Work. It is widely believed that to arrange multitask business processes or workflows upon distributed nodes or computing resources with Quality of Service (QoS) constraints, e.g., reliability, is an NP-hard problem [12, 13]. It is therefore extremely time-consuming to yield optimal schedules through traversal-based algorithms. Fortunately, heuristic and metaheuristic strategies with polynomial complexity are capable of producing approximate or near-optimal solutions at the cost of acceptable optimality loss.

For example, Wang et al. [14] proposed a look-ahead genetic algorithm (LAGA), which utilized reliability-based reputation scores for optimizing the makespan and the reliability of a workflow application. Wen et al. [15] aimed at solving the problem of deploying workflow applications over federated clouds while meeting the reliability, security, and cost requirements. Wu et al. [16] proposed a soft error-aware and energy-efficient task scheduling method for workflow applications in DVFS-enabled cloud infrastructures under reliability and completion time constraints. Cao et al. [17] proposed a soft error-aware VM selection and the task scheduling approach to minimize the execution cost of cloud workflows under makespan, reliability, and memory constraints while considering soft errors in cloud data centers. Garg et al. [18] proposed a new scheduling algorithm called the reliability and energy-efficient workflow scheduling algorithm, which jointly optimized lifetime reliability of application and energy consumption and guaranteed the user-specified QoS constraint. Nik et al. [19] proposed a scheduling approach, which included four algorithms for minimizing the workflow execution cost while also meeting the user-specified deadline and reliability.

To minimize the overall error probability in a multi-server mobile edge computing (MEC) network, where the wireless data transmission/offloading was carried by finite blocklength (FBL) codes, Zhu et al. [20] characterized the FBL reliability of the transmission phase and investigated the extreme event of queue length violation in the computation phase by applying extreme value theory and provided an optimal framework for deciding time allocation and server selection. Peng et al. [8] proposed a novel method to evaluate the resource reliability in mobile edge computing environment and addressed the workflow scheduling problem by using a Krill-based algorithm. Kouloumpris et al. [21] considered an architecture consisting of an edge node, an intermediate node (hub), and the cloud infrastructure and then used a mathematical programming-based framework to derive an application-reliability-optimal task allocation based on multiple operational constraints. Wang et al. [22] developed a reinforcement-learning-based approach to the multi-workflow scheduling method. However, they considered the centralized cloud environment as the underlying infrastructure and thus ignored the overhead for inter-edge-node data transmission. For a similar optimization objective, Wang et al. [23] and Saeedi et al. [24] employed an immune-based PSO algorithm for scheduling workflows over centralized clouds.

3. Models and Systems

3.1. System Architecture. An edge computing system usually consists of an edge computing agent (ECA) and multiple edge servers. The edge computing agent manages all resources and each edge server owns several virtual machines (VMs), each of which can usually handle a workflow task that a user offloads at a time. An edge server usually has limited capacity for storage and computation. Due to the requirement of signal strength and channel stability, as illustrated in Figure 1, it is usually believed that an edge server can cover a limited circular range and thus users can only offload their tasks to the reachable edge servers in terms of such coverage ranges.

As can be seen in Figure 2, instead of considering the monolithic task configurations, we consider that user requests can be structured and process-like requests can be expressed as workflows with different constructs. A workflow refers to a directed acyclic graph (DAG), $G = (T, E)$. T denotes the task set $T = \{t_1, t_2, \ldots t_n\}$, E is the set of edges between tasks, and $e_{ij} = (t_i, t_j)$ is a priority constraint, indicating that t_i is the precedent task of t_j.

The notations used in this paper are shown in Table 1.

3.2. Problem Formulation. In engineering, reliability is the probability of a system or component to perform its required functions under the stated conditions and with dependable outcomes. Guaranteeing reliability of computing systems and applications is a challenging problem due to the fact that faults are hard to avoid due to hardware failure, software bugs, transient faults, devices that work in high temperature, and so on. The reliability issue of edge-environment-based multi-workflow can be further complicated due to the fact that structured and process-based task flows are more susceptible to varying types of faults, especially transmission errors and faults occurring when wireless communications between edge nodes and users are required.

As shown in Figure 3, the reliability of a workflow is usually structure dependent as follows:

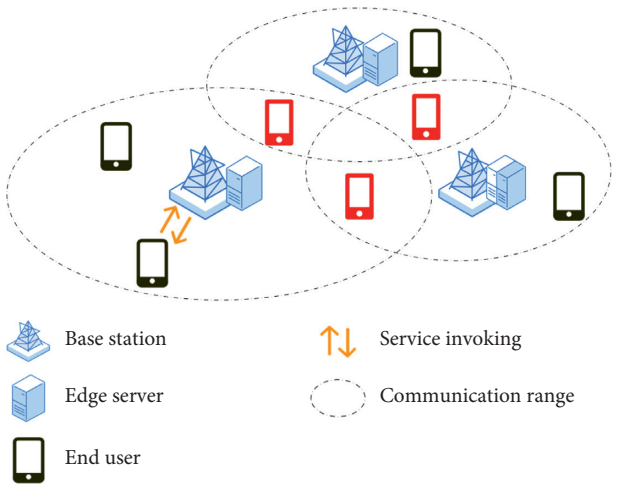


Figure 1: Proximity constraint example.

$$R_\alpha(r_j, n_\alpha) = \prod_{j=1}^{n_\alpha} r_j,$$
$$R_\beta(r_j, n_\beta) = 1 - \prod_{j=1}^{n_\beta}(1 - r_j), \quad (1)$$
$$R_\gamma(r_j, n_\gamma) = \sum_{j=1}^{n_\gamma} \lambda_j \cdot r_j,$$

where n_α denotes the number of tasks in a sequential routing, n_β is the number of tasks succeeded by a split point in a parallel routing, and n_γ is that of a selective routing, respectively. For a task executed on the edge server p, its reliability can be estimated as its success rate of execution, i.e., the probability that its time-to-failure (TTF_p) exceeds its completion time:

$$r_j = \sum_{p=1}^{m} \sum_{k=1}^{m_p} x_{pk} \cdot \mathrm{Prob}(\mathrm{TTF}_p > \mathrm{FT}(T_{pk})), \quad (2)$$

where

$$x_{pk} = \begin{cases} 1, & \text{if } \mathrm{VM}_{pk} \text{ is selected for the task,} \\ 0, & \text{otherwise,} \end{cases}$$
$$\mathrm{FT}(T_{ijpk}) = \mathrm{FT}(\mathrm{prior}(T_{ijpk}), \mathrm{FT}(\mathrm{pred}(T_{ijpk}))) + t_{ijpk}. \quad (3)$$

To estimate the monetary cost of workflows, we first have to estimate the cost for renting server p:

$$V_p^{\mathrm{rent}} = \sum_{k=1}^{m_p} \max[\mathrm{FT}(T_{pk})] \cdot C_p^r, \quad (4)$$

where $\max[\mathrm{FT}(T_{pk})]$ is the completion time of the task executing queue on VM_{pk} and C_p^r is the charge per unit time for renting server p.

The transmission time for the task i can be estimated as $\Delta_i = \Delta_i^{\mathrm{ul}} + \Delta_i^{\mathrm{dl}} + \Delta_i^{\mathrm{bh}}$, which is composed of three parts [25], where Δ_i^{ul} indicates the uplink communication time, Δ_i^{dl} is the the downlink time, and Δ_i^{bh} is the the backhaul link time. According to [26, 27], Δ_i^{bh} can be infinitesimal, and the downlink time Δ_i^{dl} can usually be a constant ξ. Therefore, Δ_i can be expressed as

$$\Delta_i = \frac{d_i}{\omega_p \cdot \eta_p^i} + \xi, \quad (5)$$

where $\eta_p^i = \lambda/\mathrm{disk}(i, p)$ is decided by the distance between the task (user) and the server; as the distance increases, the bit error rate increases and the average transmission speed decreases [27]. And, ω_p indicates the averaged bandwidth of the server p and d_i is the the data size of task i. If the transmission price per unit time of the server p as C_p^t, then the transmission fees can be estimated as

$$V_p^{\mathrm{commu}} = \frac{d_i}{\omega_p \cdot \eta_p^i} \cdot C_p^t. \quad (6)$$

Based on the described system configuration, the problem that we are interested in is thus, for given proximity constraints of server-user communications and deadline, how to schedule workflows with higher reliability and lower cost. The resulting formulation is thus

$$\begin{aligned} &\mathrm{Min}\ f_1 = \mathrm{reliability} = \prod_{i=1}^{n} R_i, \\ &\mathrm{Min}\ f_2 = \mathrm{cost} = V_p^{\mathrm{rent}} + V_p^{\mathrm{commu}}, \end{aligned} \quad (7)$$

subject to

$$\begin{aligned} &\mathrm{ST}(T_{ij}) \geq \max[\mathrm{FT}(T_{il})], \quad T_{il} \in \mathrm{pred}(T_{ij}) \text{ and } l \in \{1, \ldots, n_i\}, \\ &T(W_i) \leq D(W_i), \\ &\mathrm{dist}_{ip} \leq \mathrm{cov}_p, \\ &x_{ijpk} \leq 1. \end{aligned} \quad (8)$$

4. PICEA-g for Multi-Workflow Scheduling

4.1. Preference-Inspired Coevolutionary Algorithms Using Goal Vectors. It has long been known that preference-based approaches are useful for the generation of trade-off surfaces in objective subspaces of interest to the decision maker. Wang et al. [28] offered one realization of such approach named preference-inspired coevolutionary algorithm using goal vectors (PICEA-g), which had been testified to outperform four other best-in-class multiobjective evolutionary algorithms, e.g., NSGA-II, MOEA, HypE, and MOEA/D.

PICEA-g is a coevolutionary approach in which the usual population of candidate solutions is considered evolvable with a set of goal vectors during the search. In this algorithm, optimality of candidate solutions is decided by a Pareto-dominance model. To be specific, a family of goal vectors and a population of candidate solutions coevolved during the search process. A candidate solution gains fitness by meeting a set of goal vectors in the objective space, but the fitness contribution must be shared with other solutions satisfying those goal vectors. Goal vectors only gain fitness

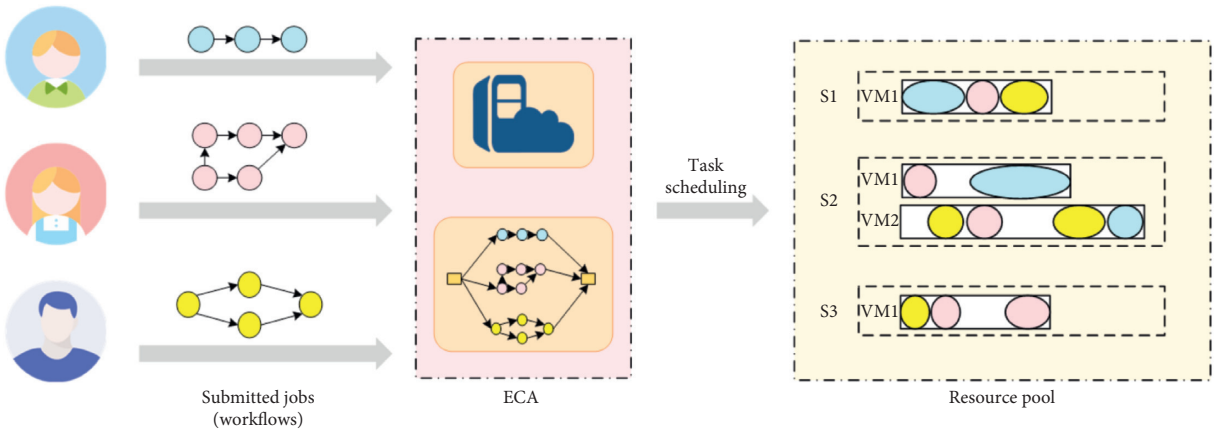


Figure 2: Edge computing deployment example.

Table 1: Notations and description.

Notation	Description
N	The total number of workflows
M	The total number of edge servers
R_i	The reliability of the workflow
λ_j	The selection probability for a task
x_{pk}	A boolean variable indicating whether VM_{pk} is selected for a task
ST (T_{pk})	The start time of the task on VM_{pk}
$MTTF_p$	The mean time-to-failure of the server p
D (W_i)	User-defined deadline of the workflow i
pred (T_{ij})	All predecessor node tasks of T_{ij} in the workflow i
cov_p	The coverage area of the server p
n_i	The total number of tasks in the workflow
m_p	The total number of virtual machines in the server p
r_j	The success rate of the task
T_{pk}	The task executed on VM_{pk}
t_{pk}	The execution time of the task on VM_{pk}
FT (T_{pk})	The completion time of the task on VM_{pk}
prior (T_{pk})	The prior task of T_{pk} in the execution queue of VM_{pk}
T (W_i)	The finish time of the workflow i
$dist_{ip}$	The distance between the server p and device i
–	–

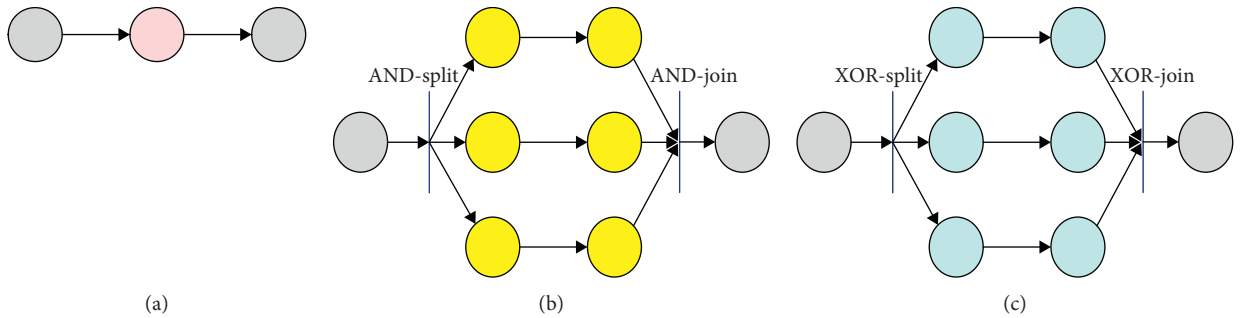


Figure 3: Routing patterns: (a) sequential routing; (b) parallel routing; (c) selective routing.

by being satisfied by a candidate solution, but the fitness is reduced the more the time a goal being satisfied by other solutions in the population. Ultimately, the population of candidate solutions and the goal vectors coevolve toward the Pareto optimal front. The fitness F_s of a candidate solution s and the fitness F_g of a preference g can be calculated by (9)–(11) as follows:

$$F_s = 0 + \sum_{g \in G \cup G_C | s \leq g} \frac{1}{n_g}, \tag{9}$$

where n_g denotes the number of solutions that satisfy preference g. In this formulation, when s fails to satisfy any g, the fitness F_s is defined as 0. And,

$$F_g = \frac{1}{1+\alpha}, \tag{10}$$

where

$$\alpha = \begin{cases} 1, & n_g = 0, \\ \dfrac{n_g - 1}{2N_S - 1}, & \text{otherwise}, \end{cases} \tag{11}$$

where N_S is the population size of candidate solutions.

A $(\mu + \lambda)$ elitist framework is usually used for implementing the above model as shown in Figure 4. As can be seen, a population of N_S candidate solutions and a set of N_G preferences, denoted by S and G, respectively, are evolved for a fixed number of generations, maxGen. In each generation t, genetic variation operators are implemented on parents $S(t)$ to produce N_S offspring, $Sc(t)$. Meanwhile, N_G new goal vectors, $Gc(t)$, are randomly regenerated based on the predefined bounds. Then, $S(t)$ and $Sc(t)$ and $G(t)$ and $Gc(t)$ are pooled, respectively, whereafter the combined population is sorted according to the fitness. Finally, a truncation-selection is applied to select the best N_S candidate solutions and N_G vectors as the new population, $S(t+1)$ and $G(t+1)$.

4.2. Encoding. For a workflow application, a chromosome is a data structure in which a scheduling solution is encoded. We use a two-dimensional string to represent a scheduling solution. One dimension of the string represents the index of resources, which depicts the task-resource mapping, while the other dimension denotes the order between tasks.As illustrated in Figure 5, in this solution, there are tasks from three workflows, namely, w_1, w_2, and w_3, which are assigned to virtual machines on two edge servers. For instance, VM_{21} is executing four tasks with the processing sequence of $t_{12} \longrightarrow t_{13} \longrightarrow t_{21} \longrightarrow t_{24}$. The decoding scheme can be described as the reverse of encoding.

4.3. Initialization. Two constraints are applied here to generate uniformly feasible chromosomes to improve the quality of the initial population, meanwhile, accelerating the convergence rate, i.e., the topological constraint and the proximity constraint. Based on the constraints, the initial population is generated as follows:

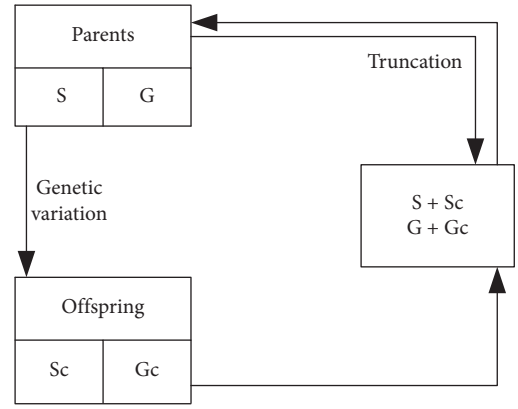


Figure 4: $(\mu + \lambda)$ elitist framework.

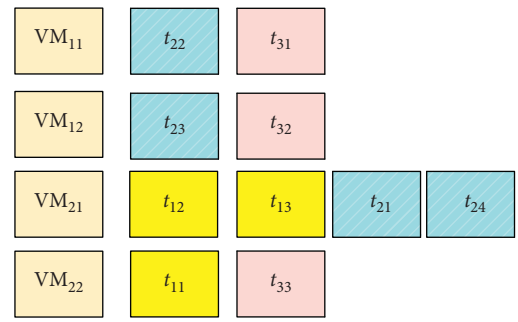


Figure 5: Solution coding.

(1) Firstly, each workflow is converted into a task list $T = \{t_0, \ldots, t_{j-1}, t_j\}$ after topological sort.

(2) Secondly, a resource r_i from $R = \{r_0, \ldots, r_{i-1}, r_i\}$ is selected as the computing resource VM, only if r_i is available for t_j. Then, t_j is assigned to a VM.

(3) Repeat the above steps until all workflow tasks are assigned. Then, a chromosome is generated.

When the population size reaches the defined value, the initialization process stops.

The initial goal vectors are randomly generated as objective vectors in the objective space within predefined bounds. In practice, the bounds are estimated via preliminary single-objective optimizations.

4.4. Population Update. The iterative update of population consists of discrete steps described below, until the termination condition is satisfied.

4.4.1. Genetic Variation. The genetic variation changes the workflow task allocation information to maintain diversity in the population. In our proposed genetic variation operation, a solution is mutated intelligently based on a resource priority heuristic. To generate a promising offspring solution, Dongarra et al. [29] have proven that the resource, which has the minimal multiplication value of some key

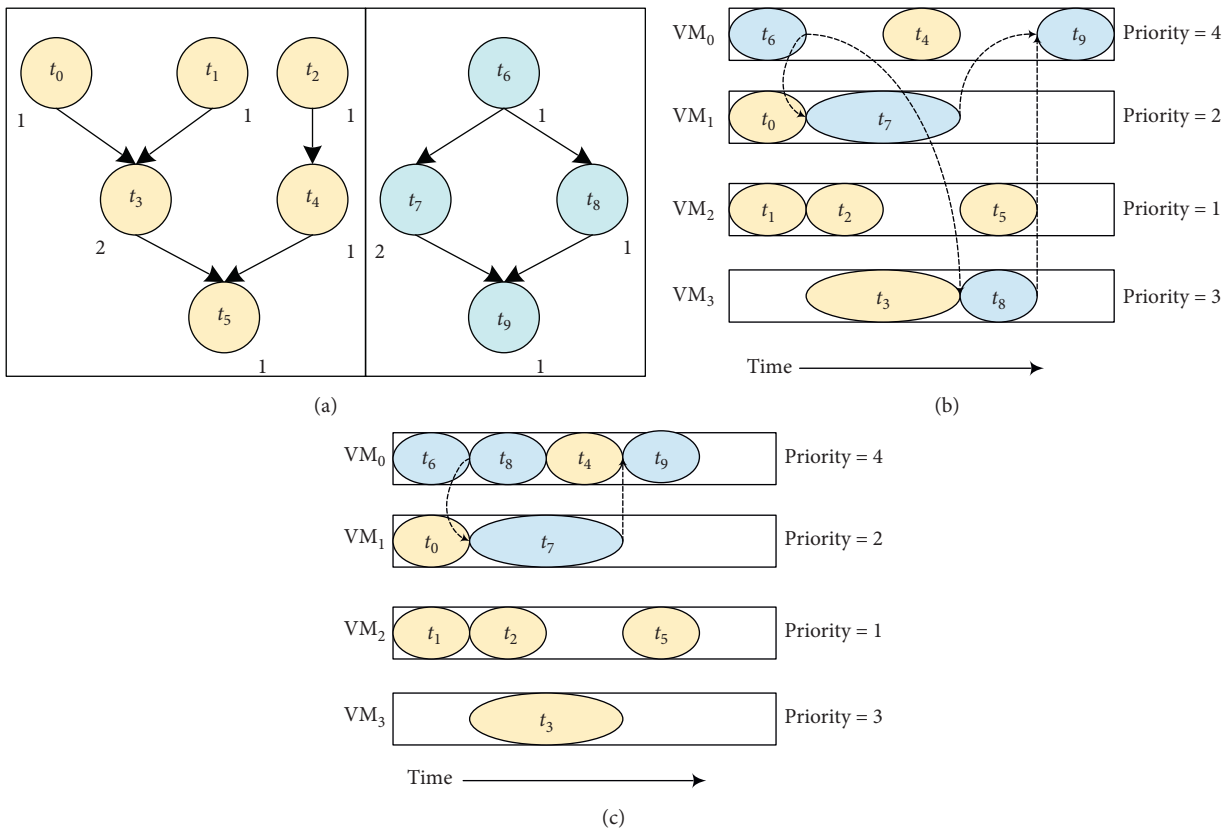


Figure 6: Genetic variation: (a) workflow example; (b) original scheduling; (c) scheduling after mutation.

performance indicators, should have a higher priority to be selected in the scheduling. Hence, we have

$$\Gamma_p = \frac{C_p^r \cdot C_p^t \cdot \mathrm{dist}(i, p)}{\mathrm{MTTF}_p \cdot \omega_p}. \tag{12}$$

Then, we let $1/\Gamma$ indicate the priority of the server p. The genetic variation operation randomly selects one task in the solution and reassigns it to any available server with a higher priority. As an example shown in Figure 6(b), task t_8 is originally scheduled to VM_4, whose priority is 3. Thus, the genetic variation reassigns it to VM_1 with a higher priority of 4.

According to the precedence constraint, we insert t_8 into the position behind t_6, as shown in Figure 6(c).

Simultaneously, N_G are new preference sets and $Gc(t)$ are randomly regenerated based on the initial bounds.

4.4.2. Fitness Calculation. Fitness calculation is based on the distribution of function value vectors and goal vectors in the target space. Assume that there are two candidate solutions s_1 and s_2, their offspring s_3 and s_4, two existing preferences g_1 and g_2, and two new preferences g_3 and g_4 (i.e., $N_S = N_G = 2$) as shown in Figure 7.

The process to calculate the fitness F_s of a candidate solution s and fitness F_g of a preference g is shown in Table 2.

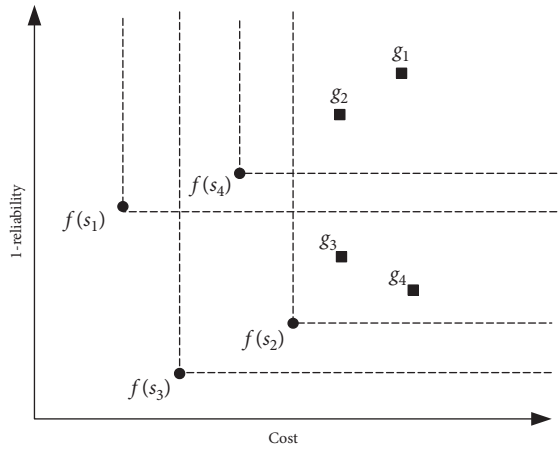
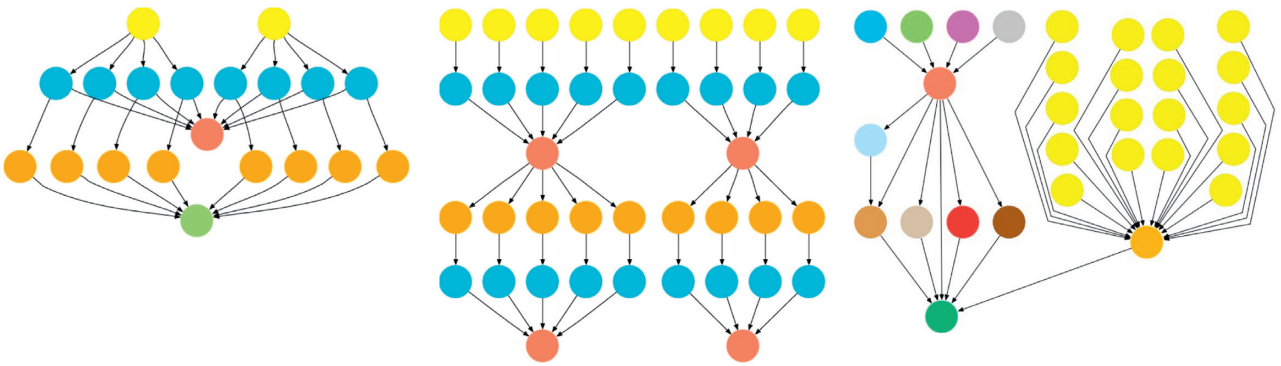


Figure 7: Example for Pareto-dominance relation.

Table 2: Fitness calculation in Figure 7.

Notation	$s1$	$s2$	$s3$	$s4$
$G\{g \mid s \le g\}$	$g1$, $g2$	$g1$, $g2$, $g3$, $g4$	$g1$, $g2$, $g3$, $g4$	$g1$, $g2$
F_s	1/2	3/2	3/2	1/2
	$g1$	$g2$	$g3$	$g4$
n_g	4	4	2	2
α	1	1	1/3	1/3
F_g	1/2	1/2	3/4	3/4



(a) (b) (c)

Figure 8: The case templates of workflows: (a) CyberShake; (b) LIGO; (c) SIPHT.

Objective function: $F(x)$;
Algorithm-related parameters: N_S, N_G, itermax;
Generate initial population S and initial goal vectors G;
$F_S = F(S)$;
while $t <$ itermax **do**
  Generate new population Sc from S by genetic variation
  $F_{Sc} = F(Sc)$
  Merge S and Sc into Joint S
  Merge F_S and F_{Sc} into Joint F
  Find Pareto Nondominated ndJoint S from Joint S
  Generate a new goal vector Gc
  Merge G and Gc into Joint G
  Evaluate the fitness of Joint S and Joint G
  if size(ndJoint S) $< N_S$ **then**
    Set the fitness of ndJoint S as max value
    Updtae S by truncation selection from Joint S
  else
    Update S by truncation selection from ndJoint S
  end
  Update G by truncation selection from Joint G
  if terminate condition satisfied, **then**
    Break
  end
end

Algorithm 1: PICEA using goals (PICEA-g).

Table 3: Resource configurations and the price-per-minute of edge servers.

Edge server types	Vcpu	Memory (g)	Unit-price/minute
tp1	1 core	1	0.0558 cents
tp2	1 core	2	0.1262 cents
tp3	2 core	4	0.1675 cents

4.4.3. Truncation Selection. Truncation selection aims to select the best N_S candidate solutions from the union population according to their fitness. However, some solutions with higher fitness may be Pareto-dominated. Therefore, we identify all nondominated solutions before the selection. If the number of nondominated solutions does not exceed the population size, then we assign the maximum fitness to all the nondominated solutions. However, if more than N_S nondominated solutions are found, we then disregard the dominated solutions prior to applying truncation selection (implicitly, their fitness is set to zero).

4.4.4. Termination Conditions. This phase is a major part of the proposed algorithm, which can specify the final solutions. In this article, the termination condition is examined in two

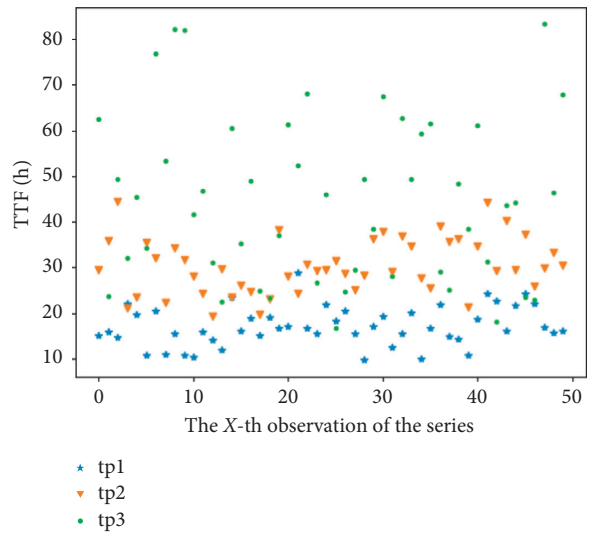


Figure 9: The TTF data collected at different types of edge servers.

Figure 10: Edge servers and users deployment.

stages: (1) as soon as a maximum iteration criterion is met, the proposed algorithm terminates and (2) T is a threshold value for terminating algorithm, set to 0.9 in our study. In every generation, after calculating the fitness of the populations, if the fitness function value is less than T, the algorithm continues; otherwise, it terminates. Whenever the algorithm ends, a set of optimal solutions is presented to the user. According to all levels presented in this article, the final solution is the best solution for all objectives including reliability and cost.

Algorithm 1 presents all the operations of the PICEA-g algorithm.

5. Performance Evaluation

To evaluate the effectiveness and correctness of our proposed method, we conduct extensive simulative experiments and show through simulative results that our proposed method outperforms traditional ones. We actually intended in the beginning to employ a real-world edge-workflow-scheduling environment to test our developed algorithms. However, we found out that such an edge environment for executing real-world scientific workflow is yet to come. Consequently, we have to rely on simulations and simulative datasets in for the model validation and comparison purpose.

These simulative experiments are based on three well-known workflow templates [30], namely, CyberShake, LIGO, and SIPHT, as shown in Figure 8.

We consider that all edge servers are with 3 different types of resource configurations and charging plans, i.e., tp1, tp2, and tp3, as shown in Table 3. We collected historical time-to-failure (TTF) records of three types as illustrated in Figure 9 as the input reliability data for edge servers. Then, the MTTF of each type of edge servers can be estimated by a Monte Carlo method [31].

We assume as well that edge servers and users are located according to the EUA dataset [2] as shown in Figure 10.

We compare our proposed method with three existing approaches, namely, NSGA-II [32], MOEA/D [33], and SPEA-II [34]. Figure 11 shows the solutions obtained by abovementioned approaches for different workflow cases where the x and the y axes represent the resulting success rate and cost. Figure 12 shows the comparison of Pareto optimal solutions of different methods with varying numbers of edge servers.

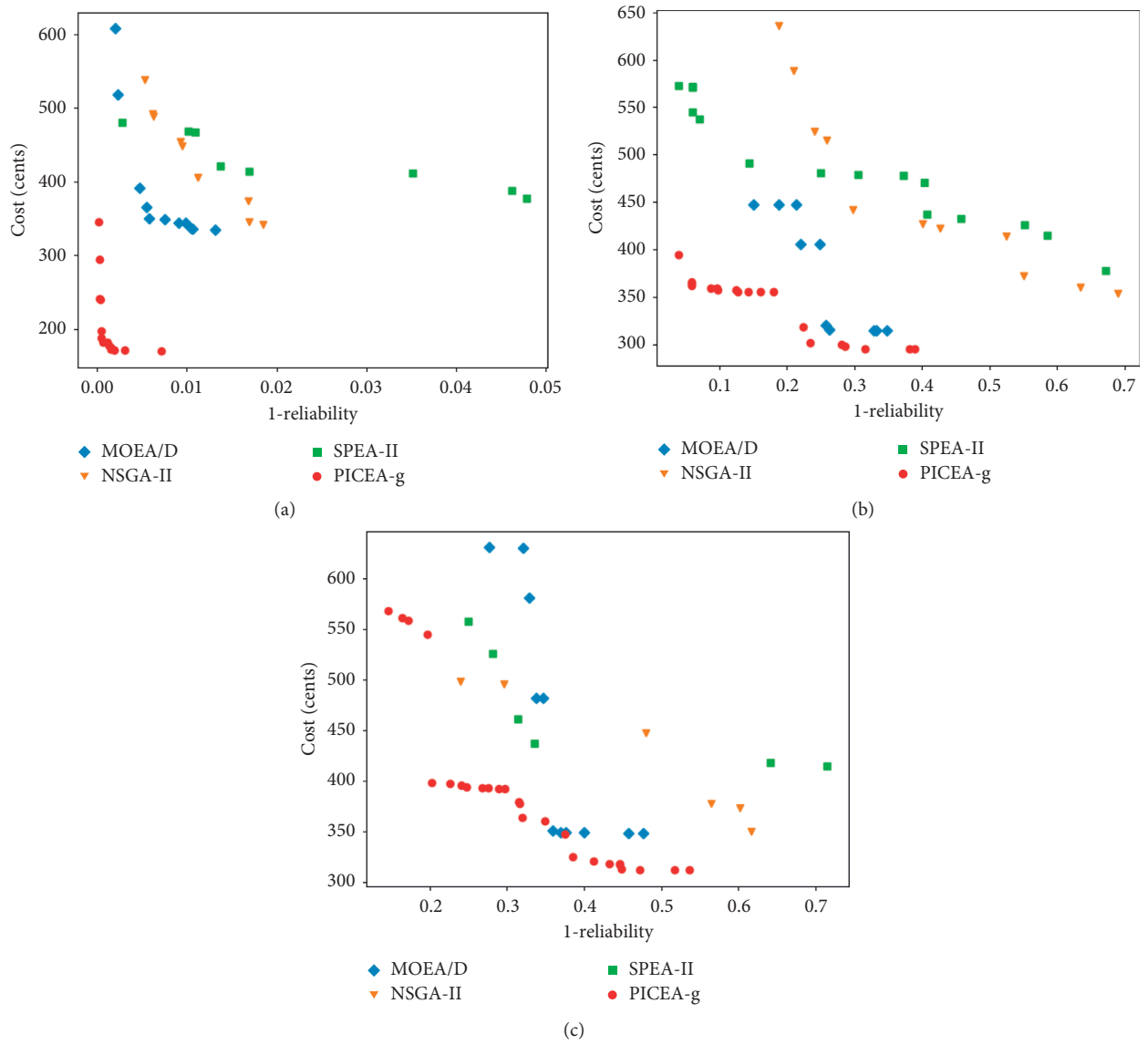


Figure 11: Experiment results with different workflow structures: (a) CyberShake; (b) LIGO; (c) IPHT.

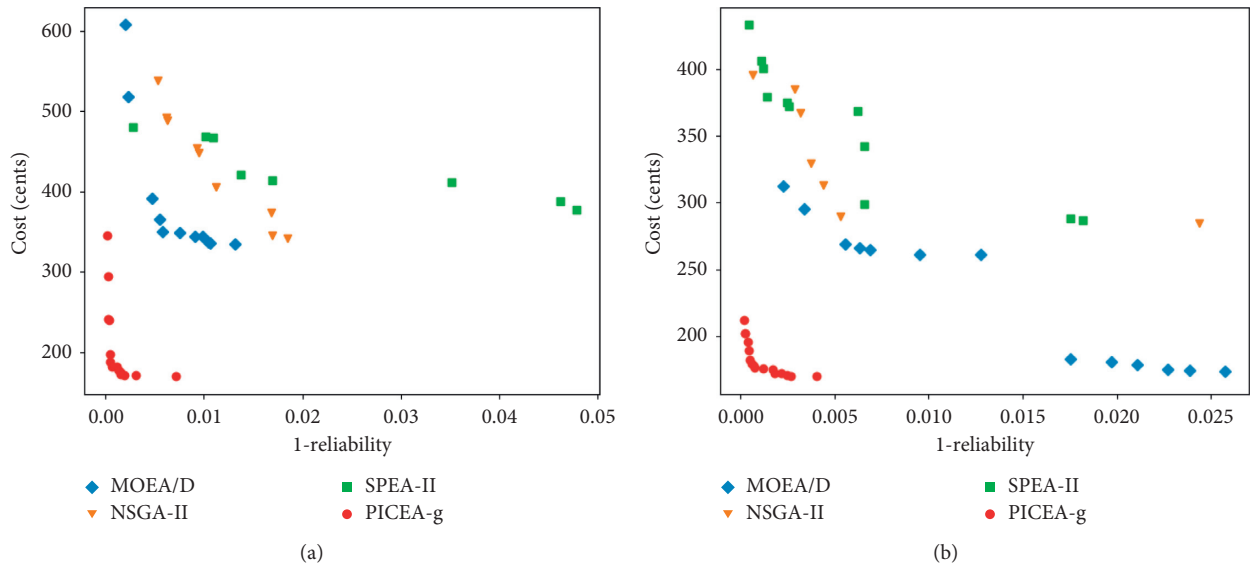
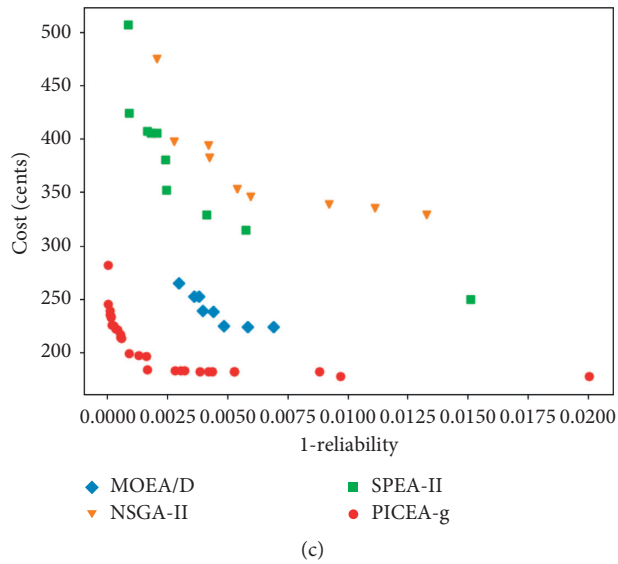


Figure 12: Continued.



(c)

Figure 12: Experiment results with the number of edge servers (a) 8 servers; (b) 10 servers; (c) 12 servers.

As can be seen from Figures 11 and 12, (1) it is evident that our method achieves better Pareto optimal fronts than its peers, regardless of workflow cases or the number of edge servers and (2) our method acquire more feasible solutions than its peers due to the fact that multiple goal vectors help to identify the solution population toward the Pareto front.

6. Conclusion

In this paper, we address the problem of reliability-guaranteed multi-workflow scheduling in the edge computing environment. We develop a reliability-driven scheduling strategy based on the PICEA-g algorithm. Extensive simulations based on several well-known workflow templates and a real-world edge-server-location dataset clearly indicate that our proposed method outperforms its counterparts in terms of different performance metrics.

Data Availability

The EUA dataset used to support the findings of this study is available at https://github.com/swinedge/eua-dataset.

Disclosure

Zhenxing Wang and Wanbo Zheng are co-first authors.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Zhenxing Wang and Wanbo Zheng contributed equally to this work.

Acknowledgments

This work was in part supported by the Chongqing Research Program of Technology Innovation and Application under Grants cstc2019jscx-msxm0652 and cstc2019jscx-fxyd0385; Key Research and Development Plan of Jiangxi Province (No. 20181ACE50029); Science and Technology Program of Sichuan Province under Grant 2020JDRC0067/2020YFG0326; and the Talent Program of Xihua University under Grant Z202047.

References

[1] H. Gao, L. Kuang, Y. Yin, B. Guo, and K. Dou, "Mining consuming behaviors with temporal evolution for personalized recommendation in mobile marketing apps," *Mobile Networks and Applications*, vol. 25, no. 4, pp. 1233–1248, 2020.

[2] Q. He, G. Cui, X. Zhang et al., "A game-theoretical approach for user allocation in edge computing environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 3, pp. 515–529, 2020.

[3] H. Liu, Y. Ma, P. Chen et al., "Scheduling multi-workflows over edge computing resources with time-varying performance, A novel probability-mass function and DQN-based approach," Edited by W. Ku, Y. Kanemasa, M. A. Serhani, and L. Zhang, Eds., in *Proceedings of the Web Services–ICWS 2020-27th International Conference, Held as Part of the Services Conference Federation, SCF 2020*, vol. 12406, pp. 197–209, pp. 197–209, Honolulu, HI, USA, September, 2020.

[4] Y. Ma, J. Zhang, S. Wang et al., "A novel approach to cost-efficient scheduling of multi-workflows in the edge computing environment with the proximity constraint," Edited by S. Wen, A. Y. Zomaya, and L. T. Yang, Eds., in *Proceedings of the Algorithms and Architectures for Parallel Processing-19th International Conference, ICA3PP 2019*, vol. 11944, pp. 655–668pp. 655–, Melbourne, Australia, December, 2019.

[5] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[6] C. Hoffa, G. Mehta, T. Freeman et al., "On the use of cloud computing for scientific workflows," in *Proceedings of the 2008 IEEE Fourth International Conference on EScience*, pp. 640–645, IEEE, Indianapolis, IN, USA, December, 2008.

[7] G. Juve, E. Deelman, G. B. Berriman, B. P. Berman, and P. Maechling, "An evaluation of the cost and performance of scientific workflows on amazon EC2," *Journal of Grid Computing*, vol. 10, no. 1, pp. 5–21, 2012.

[8] Q. Peng, H. Jiang, M. Chen, J. Liang, and Y. Xia, "Reliability-aware and deadline-constrained workflow scheduling in mobile edge computing," in *Proceedings of the 2019 IEEE 16th International Conference on Networking, Sensing and Control (ICNSC)*, pp. 236–241, IEEE, Banff, Canada, May, 2019.

[9] X. Lyu, W. Ni, H. Tian et al., "Optimal schedule of mobile edge computing for internet of things using partial information," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2606–2615, 2017.

[10] Y. Mao, J. Zhang, S. H. Song, and K. B. Letaief, "Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems," *IEEE Transactions on Wireless Communications*, vol. 16, no. 9, pp. 5994–6009, 2017.

[11] L. Tong, Y. Li, and W. Gao, "A hierarchical edge cloud architecture for mobile computing," in *Proceedings of the 35th Annual IEEE International Conference on Computer Communications, INFOCOM 2016*, pp. 1–9, IEEE, San Francisco, CA, USA, April, 2016.

[12] X. Sun, S. Wang, Y. Xia, and W. Zheng, "Predictive-trend-aware composition of web services with time-varying quality-of-service," *IEEE Access*, vol. 8, pp. 1910–1921, 2020.

[13] X. Yang, S. Zhou, and M. Cao, "An approach to alleviate the sparsity problem of hybrid collaborative filtering based recommendations: the product-attribute perspective from user reviews," *Mobile Networks and Applications*, vol. 25, pp. 376–390, 2019.

[14] X. Wang, C. S. Yeo, R. Buyya, and J. Su, "Optimizing the makespan and reliability for workflow applications with reputation and a look-ahead genetic algorithm," *Future Generation Computer Systems*, vol. 27, no. 8, pp. 1124–1134, 2011.

[15] Z. Wen, J. Cala, P. Watson, and A. Romanovsky, "Cost effective, reliable and secure workflow deployment over federated clouds," *IEEE Transactions on Services Computing*, vol. 10, no. 6, pp. 929–941, 2017.

[16] T. Wu, H. Gu, J. Zhou, T. Wei, X. Liu, and M. Chen, "Soft error-aware energy-efficient task scheduling for workflow applications in DVFS-enabled cloud," *Journal of Systems Architecture*, vol. 84, pp. 12–27, 2018.

[17] E. Cao, S. Musa, J. Zhang et al., "Reliability aware cost optimization for memory constrained cloud workflows," in *Proceedings of the 19th International Conference, ICA3PP 2019*, S. Wen, A. Y. Zomaya, and L. T. Yang, Eds., vol. 11945, pp. 135–150, Springer, Melbourne, Australia, December, 2019.

[18] R. Garg, M. Mittal, and L. H. Son, "Reliability and energy efficient workflow scheduling in cloud environment," *Cluster Computing*, vol. 22, no. 4, pp. 1283–1297, 2019.

[19] S. S. M. Nik, M. Naghibzadeh, and Y. Sedaghat, "Cost-driven workflow scheduling on the cloud with deadline and reliability constraints," *Computing*, vol. 102, no. 2, pp. 477–500, 2020.

[20] Y. Zhu, Y. Hu, T. Yang, and A. Schmeink, "Reliability-optimal offloading in multi-server edge computing networks with transmissions carried by finite blocklength codes," in *Proceedings of the 17th IEEE International Conference on Communications Workshops, ICC Workshops 2019*, pp. 1–6, IEEE, Shanghai, China, May, 2019.

[21] A. Kouloumpris, M. K. Michael, and T. Theocharides, "Reliability-aware task allocation latency optimization in edge computing," in *Proceedings of the 25th IEEE International Symposium on On-Line Testing and Robust System Design, IOLTS 2019*, D. Gizopoulos, D. Alexandrescu, P. Papavramidou, and M. Maniatakos, Eds., pp. 200–203, IEEE, Rhodes, Greece, July, 2019.

[22] Y. Wang, H. Liu, W. Zheng et al., "Multi-objective workflow scheduling with deep-q-network-based multi-agent reinforcement learning," *IEEE Access*, vol. 7, pp. 39974–39982, 2019.

[23] P. Wang, Y. Lei, P. R. Agbedanu, and Z. Zhang, "Makespan-driven workflow scheduling in clouds using immune-based PSO algorithm," *IEEE Access*, vol. 8, pp. 29281–29290, 2020.

[24] S. Saeedi, R. Khorsand, S. Ghandi Bidgoli, and M. Ramezanpour, "Improved many-objective particle swarm optimization algorithm for scientific workflow scheduling in cloud computing," *Computers & Industrial Engineering*, vol. 147, Article ID 106649, 2020.

[25] A. Al-Shuwaili, O. Simeone, A. Bagheri, and G. Scutari, "Joint uplink/downlink optimization for backhaul-limited mobile cloud computing with user scheduling," *IEEE Transactions on Signal and Information Processing Over Networks*, vol. 3, no. 4, pp. 787–802, 2017.

[26] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, 2016.

[27] Z. Lan, W. Xia, W. Cui et al., "A hierarchical game for joint wireless and cloud resource allocation in mobile edge computing system," in *Proceedings of the 10th International Conference on Wireless Communications and Signal Processing, WCSP 2018*, pp. 1–7, IEEE, Hangzhou, China, October, 2018.

[28] R. Wang, R. C. Purshouse, and P. J. Fleming, "Preference-inspired coevolutionary algorithms for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 4, pp. 474–494, 2013.

[29] J. J. Dongarra, E. Jeannot, E. Saule, and Z. Shi, "Bi-objective scheduling algorithms for optimizing makespan and reliability on heterogeneous systems," in *Proceedings of the SPAA 2007: 19th Annual ACM Symposium on Parallelism in Algorithms and Architectures*, P. B. Gibbons and C. Scheideler, Eds., pp. 280–288, ACM, San Diego, CA, USA, June, 2007.

[30] G. Juve, A. Chervenak, E. Deelman, S. Bharathi, G. Mehta, and K. Vahi, "Characterizing and profiling scientific workflows," *Future Generation Computer Systems*, vol. 29, no. 3, pp. 682–692, 2013.

[31] https://wiki.mbalib.com/wiki/Monte_Carlo_method.

[32] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[33] Q. Zhang and H. Li, "MOEA/D: a multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[34] E. Zitzler, M. Laumanns, and L. Thiele, "Spea2: improving the strength pareto evolutionary algorithm multiobjective optimization," in *Proceedings of the Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems EUROGEN'2001*, Athens, Greece, September, 2001.

Hindawi
Security and Communication Networks
Volume 2020, Article ID 8871588, 13 pages
https://doi.org/10.1155/2020/8871588

Research Article

A Novel Approach to Transform Bitmap to Vector Image for Data Reliable Visualization considering the Image Triangulation and Color Selection

Zhike Han,[1] **Xiuchao Wu**,[1] **Meng Chi**,[1] **Jun Tang**,[2] **and Lijing Yang**[2]

[1]*College of Computer Science and Technology, Zhejiang University, 38 Zheda Road, Hangzhou, China*
[2]*Hangzhou Health Information Center, Hangzhou, China*

Correspondence should be addressed to Meng Chi; chimeng089@gmail.com

Received 5 July 2020; Revised 22 September 2020; Accepted 21 October 2020; Published 2 November 2020

Academic Editor: Honghao Gao

Vector image is a type of image composed of many geometric primitives. Compared with bitmaps, vector images have the ability to save memory as well as to enlarge without distortion. Meanwhile, it has been commonly adopted in data visualization (image data) because it can be scaled to multiple sizes to fit different scenes. For instance, it can be applied for the illustrations in newspapers and magazines, the logo on the web, the background for poster, the design of text, and traffic signs. However, transforming a bitmap to vector image is still a challenging problem because of the complicated content of a bitmap, which tends to consist of more than just simple geometry. Aiming at this issue, there is a new approach proposed to transform from bitmaps to vector images, which is based on triangle units and consists of three steps. In detail, firstly, there is an initial mesh constructed for one image in pixel level after detecting features. Then, the initial mesh will be simplified by collapsing two vertices as the initial mesh is too dense to represent one image. Specifically, there are two main parts in this step, which are collapse conditions and collapse influences. In the step of collapsing, issues such as overlap and sharp triangles can be conquered by a sort-edge method (which will be illustrated specifically later). The final step is to select one color for each triangle, since it is helpful to save the memory and speed up the process of this method. In addition, one color will represent one triangle; hence, in the final step, the four-triangle sample method will be applied in order to prevent a vector image from generating too large color discontinuity. Once the pretest proceeds without mistake, the method above is able to work for the general bitmaps. Note that our method can be applied to information security and privacy, since one image can be encoded to some triangles and colors.

1. Introduction

Visualization of image data tends to be spread into many aspects in today's industries such as the illustrations in magazines, the logo on the web, and the images of posters. To take a poster as an example (especially a big size of poster), usually, there might be a large number of scales needed for various scenes if there are more than one image in a poster. Then generally there are two options for poster production. One is to prepare bitmaps of multiple scales to meet diverse needs, while it costs large storage space and larger bitmaps might be distorted. Therefore, in order to prevent visual distortion in enlarging, the second method of applying vector images to a poster tends to be preferred instead. In the field of information security and privacy, image vectorization can be used to represent one image by a lot of triangles, which can encrypt the image. In the edge computing mode, some images may need to be processed into a vector, which can ensure the reliability of data.

To transform bitmaps into vector images, some methods have been proposed [1–3]. Xia et al. [1] introduced a vectorization algorithm, in which they constructed the initial mesh in the subpixel level and used TPS [4] to do the color fitting. In this way, the representing feature in subpixel can pinpoint the location of features; meanwhile, Thin-Plate Splines (TPS) seem to perform well on the surface smooth. However, critically it requires large memory and computation and a huge number of parameters for one triangle.

Instead of using curved triangles like [1], Liao et al. [2] provided an iterated method for image triangulation. They subdivided each triangle into four small triangles to make the surface smooth. With the increasing number of iterations, the surface becomes smoother. However, it is time-consuming and might generate a large number of small triangles. Therefore, both the TPS and subdivided method will result in vector images with large memory.

In order to avoid potential problems in previous experience, there is a new method about transforming bitmap to vector image proposed in this paper. The main difference of this current method is in the aspects of pixel-level image triangulation and color selection (i.e., allocating one color for one triangle). The new method can be divided into the three following steps: initial mesh construction, mesh simplification, and color selection. Different from [1] whose initial mesh is constructed in the subpixel level, the current method's initial mesh is created in pixel level. In this way, less cost of memory is required and the process is able to be run faster.

Before constructing the mesh, feature detection is necessary, since color discontinuity is significant in these features' area, which should be treated specifically. The initial mesh is very dense after creation; thus, a simplification of initial mesh tends to be essential. In particular, collapse is a way of mesh simplification, which can be defined as a merge of one vertex and another, followed by some influence on each of the vertices' neighborhood's changing (Figure 1). In this step, two aspects need to be considered, collapse condition and collapse influence. The former refers to the requirement to perform different operations on the collapse in different situations. The latter refers to the effect on the structure of the vertices after the collapse is completed. At the same time, in order to prevent overlap (Figure 2) and sharp angle in the results of collapse, the sort-edge method is applied. Once the mesh is simplified, one image can be represented by a certain number of triangles, which is called image triangulation. Image triangulation has extensive usages such as image vectorization [1] and image editing [2]. As for the last step of color selection, there are many interpolation functions such as nearest interpolation, linear interpolation, and Thin-Plate Splines [4]. This paper will use one color as an example to fill a triangle. Even though it is challenging to fill a triangle with just one color considering it is a discrete fitting, using one color fitting still takes advantage of reducing memory costs and speeding up the program process. Furthermore, beneficially the four-triangle sample method used for color selection is able to prevent vector from generating overdiscontinuity. After color selection for each triangle, every triangle's coordinates of three vertices and RGB colors will be recorded, which can be saved as an SVG format vector image. Our contributions are summarized as follows:

(i) We propose a method to convert bitmaps into vector figures. This method includes three steps: initial mesh construction, mesh simplification, and color selection.

(ii) In initial mesh construction, we first constructed a square mesh at the pixel level instead of the subpixel level to improve efficiency. Then we used feature detection and its results to divide the square mesh into the triangular mesh. The use of feature detection allows the segmentation process to consider the significance of color discontinuity.

(iii) In initial mesh construction, we mainly simplify the generated triangular mesh through the collapse operation. In this process, various situations are regulated by constructing collapse conditions, and the occurrence of overlap and the sharp triangle is avoided through the sort-edge method.

(iv) In color selection, to improve efficiency, one color is used for one triangle, and a quantitative method is used to measure the quality of the color selection. At the same time, four-triangle sampling is used to reduce color discontinuity.

The rest of this paper is organized as the following sections. Section 2 will elaborate related work about image vectorization. In Section 3, the method of image vectorization will be introduced, which includes three steps of initial mesh construction, mesh simplification, and color selection. Section 4 is going to illustrate the results of vector images, the discussion of the current method, and future work.

2. Related Work

The vector image is represented by geometric base primitives. It has wide applications such as [5] and may apply to some other domains like image captioning [6–8]. Recently, some new methods for image vectorization are proposed [9–12]. In general, there are three geometric base primitives for transforming a bitmap into a vector image, which are diffusion curve, gradient mesh, and triangles. The diffusion curve [13, 14] is the base unit for a smooth vector image creation. For example, Xie et al. [3] and their team have suggested a method that can automatically generate sparse diffusion curve vectorizations of raster images by fitting curves in the Laplacian domain. Besides, there are also some studies about the diffusion curve [15, 16]. Image vectorization based on gradient mesh [17, 18] is also available. To prove it, there are some studies that discovered color operations by gradient mesh [19–22].

Triangle is used as based primitives for image vectorization in this paper. Image triangulation has wide applications such as image vectorization [1], image editing [2], stylized image [23], image compression [24, 25], and some other cases like [26–28]. Image triangulation usually has two steps, initial mesh construction and mesh simplification. As has been stated previously, the initial mesh construction of Xia et al. [1] is overcomplicated and memory-consuming, since they use subpixel features to construct the initial mesh. As for mesh simplification, a method of [29, 30] using quadric error as a metric can determine the order of simplification. In the current method, there tends to be no particular order for collapse; meanwhile, the collapse is only limited by collapse conditions. For each iteration, the passive vertices of one collapse are randomly chosen. Besides, Liao

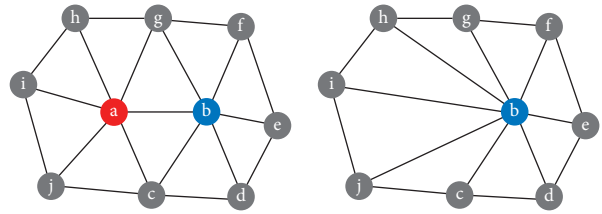


Figure 1: (a) Before collapse. (b) After collapse. We can see that there are 5 types of vertices $\{a, b, C_{ab}, O_a, O_b\}$.

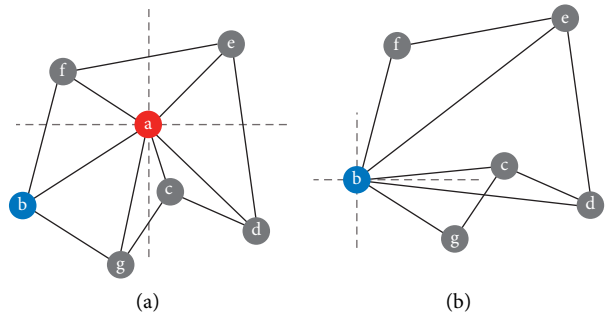


Figure 2: (a) Before collapse. (b) Overlap after collapse. We can see that before collapse the sequence of neighbors of a except b is $\{g, c, d, e, f\}$ but after collapse the sequence becomes $\{g, d, c, e, f\}$ which means overlap happens.

et al. [2] and their team provide a different way for image triangulation. They add an additional step that subdivides the simplified mesh to smooth the surface in a further way. Specifically, one triangle can be subdivided into four small triangles for a smoother surface based on certain algorithms for triangulation such as classical triangulation algorithms like Delaunay [31]. Recently, TriWild [32] is suggested as a preferred one, which is a robust 2D meshing algorithm and generates curved triangles reproducing smooth feature curves.

Color selection is a suitable method. In the case of [1], it uses TPS [4] to fit color. Thin-Plate Splines (TPS) are a spline-based technique for data interpolation and smoothing. Recently, Chen et al. [33, 34] discovered image vectorization by adopting TPS. Applying TPS to color fitting requires saving a huge number of parameters for one triangle, though it can achieve a smooth surface. In [2], the color can be smooth and natural by subdividing one triangle into four small triangles. This paper's method may generate too large color discontinuity, since one color for one triangle is a discrete fitting. Fortunately, conducting a four-triangle sample method is able to address this problem.

In conclusion, "one color for one triangle" can maintain its advantages of saving storage and easy calculation; on the other hand, it can avoid disadvantages of generating large color discontinuity by using the four-triangle sample method.

3. Image Vectorization

In this section, the process of constructing the initial triangular mesh and simplifying the mesh and color fitting (or color selection) will be illustrated specifically. The first two steps are renamed as image triangulation. The third step achieves image vectorization by color selection. The total steps of image vectorization are summarized as follows:

(1) The first step is the initial mesh construction. It is about how to construct the mesh based on the original pixels after feature detection. Feature pixels are treated specifically, since color discontinuity is much larger compared with general pixels.

(2) The second step is mesh simplification. It is necessary to simplify the mesh to get a smaller number of triangles because the initial mesh tends to be too dense to represent one image. In particular, collapse is used to simplify the mesh, which contains collapse conditions and collapse influences. Problems like overlap and sharp triangles are able to be addressed by the sort-edge method.

(3) The final step is color selection, which is about how to determine the color for the triangle. The error function is defined in this step to get the best color for each triangle. It is recommended to adopt a four-triangle sample method to prevent vector image from generating color overdiscontinuity.

3.1. Initial Mesh Construction. The bitmap is composed of pixels with colors (gray or RGB). Image vectorization begins with constructing the initial base units from bitmap because image vectorization needs to use geometric units to represent one image. Triangles are used as base units in this method; thus, this beginning step is about constructing

initial triangular mesh on a bitmap. It can be subdivided into three steps, which are square mesh construction, feature detection, and transferring square mesh to triangular mesh.

Firstly, for each pixel in an image, the nearest four pixels (up, down, left, and right) are linked, which will result in a square mesh. In this step, for general pixels, they have four neighbor pixels. For boundary pixels, they have only three neighbor pixels. In addition, for four vertexes of one image, they have only two neighbor pixels.

Secondly, neighbor feature pixels belonging to the same feature should have one edge to link them together. Canny [35] is used to detect features, which consists of feature pixels. Detected features are thinned to 1-pixel wide and linked by small gaps. Features shorter than 10 pixels should be discarded. Different features are marked for different flags. Once the feature pixels in the image are identified, they will be divided into different types. Then, the square mesh can be transferred to triangular mesh.

Finally, the square mesh is transferred to a triangular mesh. For each square mesh, there are four pixels. If the diagonal of square mesh has two feature pixels, the two feature pixels must be linked by one edge as neighbor pixels. Otherwise, either side of diagonal can be chosen as an edge (Figure 3), which is not significantly influenced by the features. All square meshes should be operated referring to the above statements for generating triangular mesh in the final. Initial mesh construction tends to be completed here.

3.2. Simplify Mesh. A large number of triangles are generated by initial mesh construction. Mesh simplification can reduce the number of triangles by collapse. As noted in this paper, the two vertices related to one collapse should be distinguished. As is shown in Figure 1, vertices a collapse into vertices b. a is called active vertices in this collapse; and b is called passive vertices (i.e., the position of passive vertices in one collapse is fixed). In the process of collapse, specifically, there are two points that need to be concerned. One is the condition of collapse, and the other is the influence of collapse. Garland and Heckbert [29] and their team simplify mesh by quadric error metrics. Differently, collapse conditions are used to control collapse in the current method. The collapse condition is composed of some rules to limit the collapse. For example, collapse conditions should prevent collapse from generating overlap and sharp triangles. Moreover, collapse conditions can only be conducted when one vertex collapses into another, which needs the user's personal judgement. Collapse influence is about the changes in the topology of the triangular mesh when collapse happens. There are five kinds of vertices type. The influence of each type of vertices will be explained in detail later.

3.2.1. Collapse Conditions. The whole image has four vertices that should be fixed in the process of collapse. In the same way, one image has four boundaries. The vertices in the same boundary can collapse to each other. In one collapse, if one vertex is in boundary and another is not, then it can be passive vertex but cannot be active vertex, since boundary

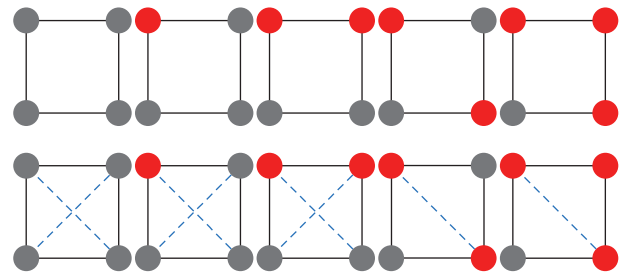


Figure 3: Initial mesh construction. Red point represents feature pixel, and gray point represents general pixel.

vertices collapsing into nonboundary vertices will destroy the shape of the image. The rest of the vertex is divided into feature vertices and general vertices (nonfeature vertices). Feature vertices are detected by Canny. Two feature vertices with the same feature can collapse to each other (in the condition that is not concerned about the shape of the feature, which will be talked about below). However, if collapse happens between feature vertices and general vertices, feature vertices cannot be active vertices. As for general vertices, they can be either active vertices or passive vertices.

For one feature (here feature refers to the set of feature vertices that belong to the same feature), the end of it should be fixed; otherwise, this feature may disappear. Suppose one feature $f = \{v_1, v_2, \ldots, v_n\}$, where $v_i, i \in n$ denotes the vertices in order which belong to feature f, which has n vertices. If v_1 and v_n are not fixed, then v_1 can collapse to v_2 when meeting the collapse conditions. Additionally, it may appear that v_{n-1} collapses to v_n and, finally, only v_n remains. It may show that the feature $f = \{v_n\}$ if the end vertices of f are not fixed. There is another constriction of collapse between the same feature vertices, which is the shape of one feature. One collapse should not be allowed if it influences the shape of the feature a lot. As shown in Figure 4, a collapses to b (same as a is active vertices and b is passive vertices in this collapse), and angle(cab) is used to measure the extent of change to the shape of the feature, which can be computed as

$$\theta = \arccos\left(\frac{L(a,c)^2 + L(a,b)^2 - L(b,c)^2}{2L(a,c)L(a,b)}\right), \quad (1)$$

where $L(\cdot)$ denotes the Euclidean distance of two vertices and θ represents angle(cab).

Collapse does not allow two-edge cross which is called overlap; hence, the collapse will be rolled back once overlap happens. The way of judging whether there is overlap or not is based on the following criteria (sort-edge method). As shown in Figure 4, the authors suppose that vertex a collapses to vertex b. Before the collapse, a coordinated system is created with a as the origin. The anticlockwise order of the neighbors of a (except passive vertices) is $\{g, c, d, e, f\}$. Note that only the order is concerned (i.e., the first one of this order is arbitrary); $\{g, c, d, e, f\}$ is equivalent to $\{f, g, c, d, e\}$. After the collapse, the authors use vertex b as the origin to create a coordinate system and to judge if the anticlockwise

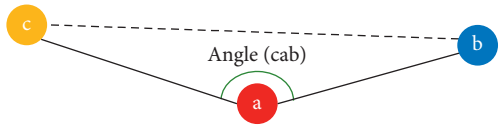


Figure 4: Keep feature shape. If vertices a want to collapse into vertices b, then angle(cab) should be larger than threshold.

order of the neighbors of b is the same as that of a. If the order before the collapse is different from that after the collapse, then roll back this collapse; otherwise allow this collapse.

The sharp triangle is defined as the triangle that has a very small angle. A stricter limit for the collapse is to avoid sharp triangles due to their many disadvantages. For example, compared with the equilateral triangle, the distribution of sampling points for a sharp triangle is more uneven in the process of color selection, which may cause a large error. Besides, the sharp triangle is not friendly to subsequent collapse. Fortunately, the sort-edge method can be adopted to avoid sharp triangles. In order to avoid sharp triangles, it is allowed to directly calculate the angle of two adjacent edges and then roll back the collapse that generates a small angle because the sequence of edges has already been known.

3.2.2. Collapse Influences. A collapse will be accepted if it meets collapse conditions. The vertices in one collapse can be identified as five types. It is supposed that vertex a collapses to vertex b (i.e., a is the active vertex and b is the passive vertex). Then five types of vertices can be defined as $\{a, b, C_{ab}, O_a, O_b\}$, where C_{ab} denotes the vertices that are neighbors of both a and b. O_a denotes the vertices that only link with a and O_b denotes the vertices that only link with b. Take Figure 3 for an example; $C_{ab} = \{c, g\}$, $O_a = \{h, i, j\}$, $O_b = \{d, e, f\}$. In actual implementation, the authors create a struct for vertices that have a neighbor list, also called the edge list. Specifically, the changes of five types of vertices were actually accompanied by the changes of neighbor list, which are explained as follows.

For vertex a, it is the active vertices in this collapse, which just requires clear identification of all neighbors in the neighbor list, since vertices a will disappear after the collapse. For passive vertices b, they delete the neighbor a and add all vertices in O_a as new neighbors. When a disappears after the collapse, all vertices in O_a should delete a in their neighbor list and include b, since, with a collapsing to b, the vertices linked to a will now link to b. As for vertices in C_{ab}, they delete a in their neighbor list because vertices a collapsing to vertices b can be described as that a and b are now the same. Vertex in O_b is kept unchanged, which means the collapse has no influence on O_b. As shown in Figure 1, the collapse can change the triangular mesh structure while making the number of triangles less, which is the aim of mesh simplification.

3.2.3. Algorithm Design. In code implementation, a structure named *Vertices* for vertices is designed. In the structure *Vertices*, there are some members such as *position* (the position of this vertex), *Neighbors* (the neighbor vertices of this vertex, it is a list), and *color* (the color of these vertices). Therefore, the list of *Vertices* can be used to represent the triangular mesh. Actually, the step of mesh simplification is also operating *Neighbors* in *Vertices*.

Algorithm 1 shows the symbolic description of the algorithm based on the assumption that vertices a collapse to vertices b. The structure *Vertices* of a is as v_a, and similarly, v_b denotes the structure *Vertices* of b. N_a, N_b denote the *Neighbors* of a, b, and n_a, n_b denote the items of N_a, N_b, which are also the structure. The function **remove (x, y)** means deleting its neighbor **y** by removing **y** from its *Neighbors* (remove **y** in **x**'s neighbor list) in the structure *Vertices* of **x**. Function **create (x, y)** means adding **y** into **x**'s neighbor list and function **clear (x)** means clearing all neighbors of vertices **x**.

As is discussed in collapse influences, the vertices in one collapse can be divided into five types. For each type, the impact tends to be different. Specifically, Algorithm 1 starts from the loop through the neighbor vertices of active vertices in one collapse. As for each of the neighbor vertices, their neighbor list tends to be updated according to their own type. The time complexity of Algorithm 1 is $O(1)$ because of the constant number of neighbors of one vertex.

Given that the triangular mesh can be represented by the list of *Vertices*. The steps of algorithm for mesh simplification are shown in Algorithm 2. Function **random (x)** means to randomly choose one neighbor of vertices **x**; meanwhile, function **collapse_condition (x, y)** returns true if this collapse is allowed, and function **collapse (x, y)** in this algorithm refers to the functionality of Algorithm 1.

As is shown in Algorithm 2, it loops through the entire vertices list. As for each vertex, one of its neighbor vertices is randomly chosen. Then this vertex is collapsed to its neighbor vertices. Algorithm 2 will be iterated T times. In most of authors' experiments, T is set to 40. Therefore, the time complexity of Algorithm 2 is $O(M)$, where M denotes the total number of vertices in one image.

3.3. Color Selection. After mesh simplification, image triangulation is completed, which means the bitmap is represented by many triangles. In order to distinguish image vectorization, each triangle should be colored. For triangle coloring, Xia et al. [1] introduced TPS [4] to do color fitting, but TPS [4] has high computation and storage consumption. A different way of color fitting is implemented in this current study. Specifically, only one color will be used to represent one triangle, which needs more accurate color selection as it is a discrete fitting.

Each triangle should be subdivided into n^2 equal parts, which means dividing each side length of the triangle into n equal parts (in this paper n is set to 3). Then the center of each subtriangle is taken as a sample (Figure 5). Since only one color is needed, an error function is defined for each sample like the following formula:

Input: the structure *Vertices* of active vertices a, the structure *Vertices* of passive vertices b.
Output: the structure *Vertices* of $\{a, b, C_{ab}, O_a, O_b\}$

```
(1) for n_a in N_a do
(2)   if n_a == v_b then
(3)     continue
(4)   isCommon == false//judge if n_a is in C_ab
(5)   for n_b in N_b do
(6)     if n_a == n_b then
(7)     isCommon = true
(8)     break
(9)   for n_{n_a} in N_{n_a} do
(10)    if n_{n_a} == v_a then
(11)      remove (n_a, a)//operation for both O_a and C_ab
(12)      if isCommon == false then
(13)        create (v_b, n_a)//operation for b
(14)        create (n_a, v_b)//operation for O_a
(15) clear (v_a)//operation for a
```

Algorithm 1: Collapse algorithm.

Input: the list of Vertices
Output: the list of Vertices (simplified)

```
(1) for v in Vertices do
(2)   n = random (v)
(3)   if collapse_condition (v, n) == true then
(4)     collapse (v, n)
```

Algorithm 2: Mesh simplification algorithm.

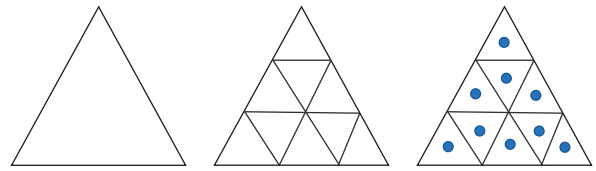


Figure 5: (a) The original triangle. (b) Subtriangles. (c) Samples of subtriangles.

$$\text{error}_i = \sum_{j}^{N} \left\| c_i - c_j \right\|, \tag{2}$$

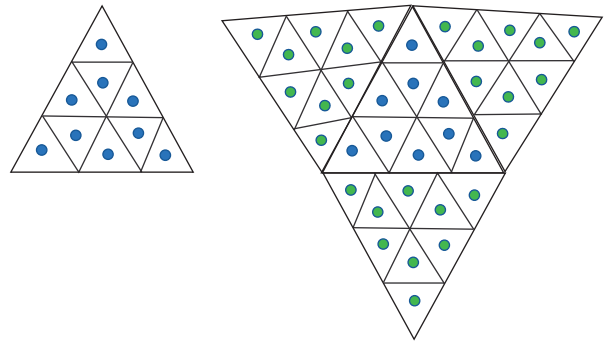
where c_i, c_j denote the colors of the ith and jth samples. N is the number of samples. error_i denotes the error of ith sample.

The best color of the triangle is the ith sample with minimum error_i for $i = 1, 2, \ldots, N$.

$$c_{\text{best}} = \text{color}\left(\arg\min_{i \in N} \text{error}_i \right), \tag{3}$$

where $\text{color}(\cdot)$ denotes the function that gets color by using sampling point as input parameter.

However, sampling from only one triangle may cause large color continuity; hence, each triangle is too independent. Therefore, to address this problem, this paper provides an approach of four-triangle sample (Figure 6) method. Conducting this method can start from defining the triangle that is doing color selection as a reference triangle. For each reference triangle, there are three near triangles and each of them shares one edge with a reference triangle. Sampling points from these four triangles get the best color for the reference triangle using formulas (2) and (3). Nevertheless, there is a consideration that needs to be concerned which is if one side of the reference triangle is feature edge, the triangle related to this feature edge should not be sampled. Two triangles sharing one feature edge must have a



(a) (b)

Figure 6: (a) One-triangle sample. (b) Four-triangle sample.

very different color because color discontinuity is significant in the feature area.

4. Results and Discussion

The proposed method about image vectorization consists of three steps, which are initial mesh construction, mesh simplification, and color selection (Figure 7). Some results about our method are shown in Figure 8. All of these steps are described, respectively, as follows:

Step 1: initial mesh construction

(i) *Input Images*. The input images are bitmaps with formats of *png*, *jpg*, etc. The size of input images is arbitrary.

(ii) *Canny Detection*. Using Canny to detect features in the input image, the thresholds of Canny are set to 0.05 and 0.125, respectively. The features shorter than 10 pixels are removed.

(iii) *Square Mesh Construction*. For each pixel in the input image, connect it to its neighbor s with four pixels (up, down, left, and right), which will result in a dense square mesh.

(iv) *Triangular Mesh Construction*. Every square mesh consists of 4 pixels. If both the two pixels on its diagonal are feature pixels (detected by Canny), just connect this diagonal and it will make this square change to 2 triangles. Or either diagonal can be connected to generate 2 triangles. Thus, a dense triangular mesh is got.

Step 2: mesh simplification

(i) *Collapse Vertices*. Each vertex (same as the pixel, also named vertices in this section) randomly chooses one neighbor vertex of it (two vertices with an edge between them are called neighbor vertices) and then collapses itself into its neighbor vertices if this collapse meets collapse conditions. The iterations for all vertices are set to 40. After that, the simplified triangular mesh can be created.

(ii) *Extract Triangles*. BFS algorithm is used to extract all triangles on the triangular mesh. Each triangle is saved as 3 vertices.

Step 3: color selection

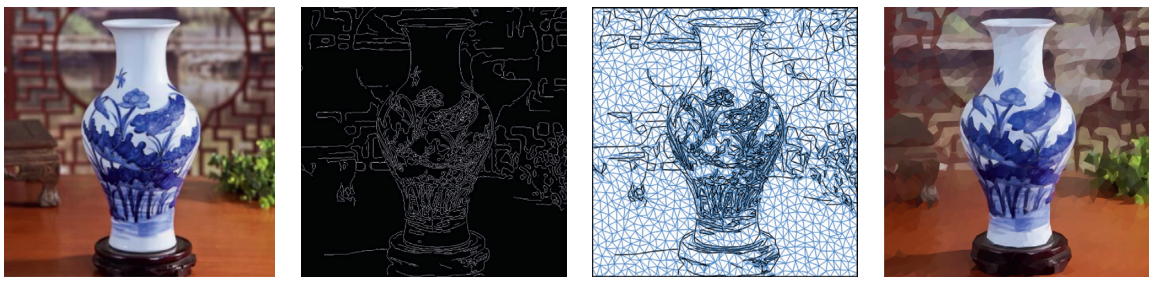
(i) *Point Sampling*. The side of every extracted triangle needs to be equally subdivided into n parts. As shown in Figure 5, linking the points on the side will generate n^2 subtriangles. The center of every subtriangle will be extracted as a sampling point. Thus, n^2 sampling points can be gained for each triangle. In this implementation, n is set to 3.

(ii) *Select Colors for Triangles*. This method uses formulas (2) and (3) to get the best color from the sampling points for each triangle.

(iii) *Saved as SVG Format*. Once the three vertices of each triangle and their corresponding color are finally settled, all the information can be saved as SVG format, which is also the format of the vector image.

4.1. Details and Memory Trade-Off Experiment. This experiment is about adjusting parameters to do details and memory trade-off. The current method about image vectorization starts from Canny detection whose threshold is set to 0.05 and 0.125, respectively. Features shorter than 10 pixels are dropped, since they are considered unimportant. In the step of mesh simplification, for each iteration, all pixels in one image will be enumerated to collapse and the total number of iterations is set to 40. In color selection, each triangle will be subdivided into 9 small triangles for sampling.

The compared parameters here are cos(angle), feature length, and side length. The parameter cos(angle) controls the smallest angle of each triangle; feature length limits the length of feature; side length prevents the side length of each triangle from over length. To test the influences of these parameters, the authors set cos(angle) in $\{-0.8, -0.5, -0.2\}$, feature length in $\{10, 30, 50\}$, and side length = 100. The result is shown in Figure 9. In addition, the authors set side length in $\{10, 30, 50, 100\}$, cos(angle)) = −0.5, and feature length = 30. The result is shown in Figure 10.



(a) (b) (c) (d)

Figure 7: The pipeline of image vectorization. From left to right are input image, Canny detection, simplified triangular mesh, and vector image.

Note that side length is different from feature length, though both of them are related to the side length of the triangle. The feature length just controls the edge whose two vertices are both features, while side length controls the rest of the edge length. The reason for choosing two parameters to control is that it makes the method more flexible. In this way, more details of the foreground of the image can be preserved; meanwhile, more details of the image background can be discarded.

As shown in Figure 9, the petal of the flower of the top-right image is more simplified than the others. This is because the increase of cos(angle(cab)) and feature length will make the feature vertices collapse more freely. Specifically, the increase of cos(angle(cab)) makes the restriction on the sharp triangle in the collapse conditions relax so that the collapse that has a greater impact on the shape of the feature can also occur; and a larger *feature length* parameter value leads to the fact that, after more collapses occur, the *feature length* can still be within the limit. These two changes will make the final number of triangles less, so more details of the picture will be lost. But more triangles and better details will take up more memory. As shown in Table 1, with the decrease of cos(angle(cab)) and *feature length*, the number of triangles increased. Although the details of the picture are better, the memory increases. Therefore, when cos(angle(cab)) is −0.8 and *feature length* is 10, the feature vertices can hardly collapse and may generate many tiny triangles to be close to the curve. So, as can be seen in Figure 9, the details of the image under this setting are better, but its memory is the largest in Table 1.

In Figure 10, when the side length is 10, the simplified triangular mesh is very dense, and the details of the output image are well preserved. However, when the side length becomes 100, a smaller number of triangles can represent this image; meanwhile, the details are not preserved well. Thus, if the *side length* becomes bigger, the number of triangles will decline. Therefore, this parameter can determine the density of triangles. At the same time, it can be seen in Table 2 that as the *side length* increases, the memory also increases. It also shows that better preservation of picture details comes at the cost of memory increase.

The three parameters actually can impact the details of one output vector image by controlling the triangles' number and shape (angle, side length). Thus, details and memory trade-off can be done for different needs.

4.2. K-Medoids for Color Simplification Experiment. This experiment is about the use of K-medoids for reducing the number of colors and about the difference in changing the value K in K-medoids. Usually, after getting the vector image, many colors may not be important to the contribution of image features. If the number of useless colors in one image can be reduced with the most details of the image maintained, the storage cost of this image will be reduced. Therefore, an algorithm for simplifying the number of colors is needed. Fortunately, in data science, K-medoids is an unsupervised method for clustering data and it is robust to noise. It can be exactly used to cluster the colors in images to achieve the purpose of reducing the number of useless colors.

For each triangle, one color is saved as an RGB format. Therefore, each color in RGB format can be regarded as a 3D space point, which can then be clustered by K-medoids. For each cluster, there is one central point making the equal distance between the center points and keeping all member points in this cluster smallest. After the termination of the K-medoids algorithm, all member points in one cluster will be replaced by center points, which means all colors in this cluster are replaced by center color. K-medoids is sensitive to the initial parameters so that sometimes initial colors can be selected manually for further preciseness.

It is set that cos(angle) = −0.5, feature length = 30, side length = 30, and K is in {15, 10, 5}, where K denotes the number of clusters in K-medoids (i.e., the number of colors for the vector image). There are different values of K in {15, 10, 5} tested whose outputs are illustrated in Figure 11.

As shown in Figure 11, with the number of colors descending, the apple just keeps the most important colors (i.e., red and green). This is because K-medoids can find a good color to represent a group of colors. The last apple shows that only 5 colors can represent this apple. It proves that K-medoids can efficiently cluster image colors and keep the most important color for apples. Furthermore, it is obvious that when the number of colors descends, the vector image will become less detailed. For example, compared with the first apple in Figure 11, the last apple loses minor texture significantly. If one image with many similar colors is unable to be distinguished, the adoption of K-medoids can decrease the number of colors and also reduce the storage of this image.

(a) (b)

Figure 8: (a) Input image. (b) Vector image.

4.3. The Advantages of Four-Triangle Sample Experiment. There is an approach called the four-triangle sample method in the process of color selection, which will be described below. One triangle is generally marked as a reference triangle; the remaining three triangles sharing an edge with the reference triangle are also taken into account. Finally, the color fitting will be arranged for sampling points from these four triangles. In this experiment, a comparison of the difference between using and not using this method will be conducted.

It is set that cos(angle) = −0.5, feature length = 30, and side length = 30. The compared parameter here is the number of sampling triangles, which tends to be one and four, respectively.

In Figure 12, the magnification part shows that the four-triangle sample (Figure 12(c)) is more robust than the one-triangle sample (Figure 12(b)). The original image (Figure 12(a)) has many noisy colors such as the ground color in the original image. The one-triangle sample may

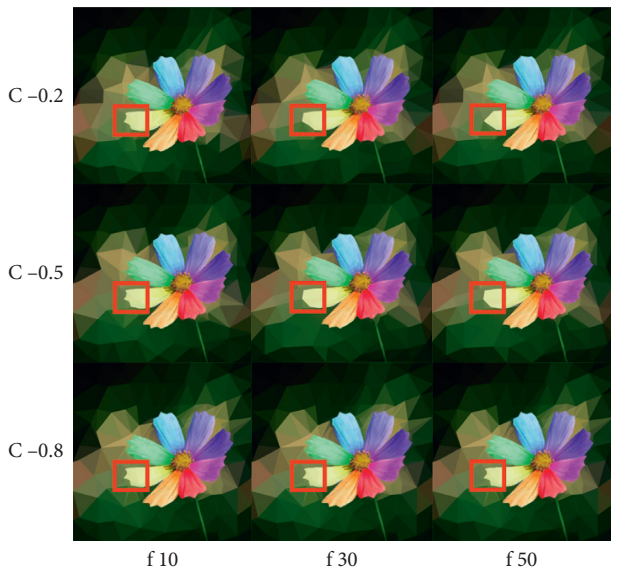


Figure 9: The influence of cos (angle (cab)) and feature length. We can see that the increase of cos (angle (cab)) and feature length will lead to less limitation for feature collapse.

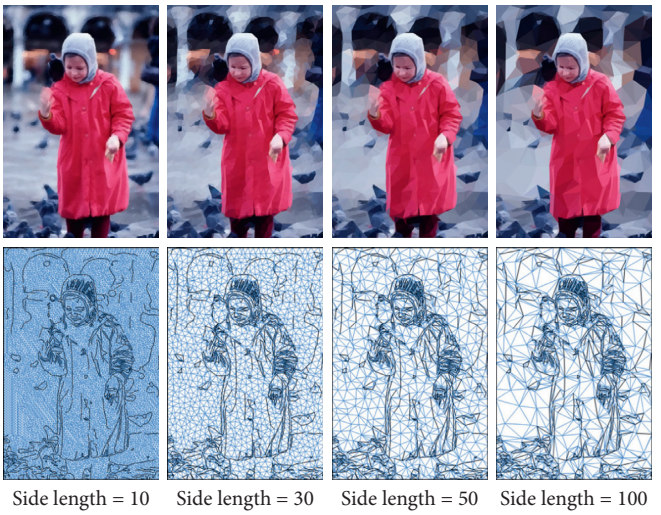


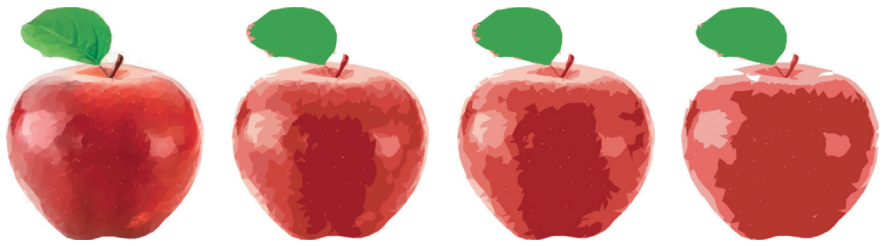
Figure 10: Vector image with different side length.

Table 1: The size of experimental results images of the source image with a size of 32 kB under different parameters.

Image size (kB)	$f = 10$	$f = 30$	$f = 50$
$c = -0.2$	18.1	15.3	12.2
$c = -0.5$	19.8	17.6	14.7
$c = -0.8$	23.7	20.3	17.4

Table 2: The size of experimental results images of the source image with a size of 758 kB under different parameters.

Side length	10	30	50	100
Image size (kB)	636.2	531.6	425.4	239.7



(a) (b) (c) (d)

Figure 11: Simplified color. From left to right, the number of colors is getting less. (a) Stylized image. (b) Stylized image number of color 15. (c) Stylized image number of color 10. (d) Stylized image number of color 5.

(a) (b) (c)

Figure 12: Comparison of one-triangle sample with four-triangle sample. (a) Original image. (b) One-triangle color selection. (c) Four-triangle color selection.

increase the proportion of these noisy colors, which may cause many small discontinuity triangles. Differently, the four-triangle sample takes the neighbor triangles into account, which is able to reduce the proportion of noisy colors. To expand this method, a larger number of triangles can be sampled for a better result. However, computing costs should be concerned in that case. Therefore, users should consider the balance of cost and result when using the four-triangle sample method. Any reference triangle of the four-triangle sample method is unlikely to be influenced by feature triangles which are sharing the same feature edge with reference triangle but on different sides.

5. Conclusions and Future Work

Due to its simplicity and effectiveness, this paper's method has the ability to do image triangulation quickly. In addition, some parameters are provided to adjust the density of triangles or the extent of the collapse. Problems of collapse such as overlap and sharp triangles are fixed by the sort-edge method. Using one color to represent one triangle makes less computation and storage, while it may cause a challengeable task since one color may lead to large error. Thanks to the four-triangle sample method, this problem can be conquered and color selection tends to be more robust. Our method outputs each triangle's three vertices and RGB color, which then can be saved as an SVG format image to get a vector image.

There are still some limitations to this method and recommendations for future work. For example, in textureless areas like the cloud in Figure 12, the performance of this method still has the potential to improve. To be more specific, the number of features is not large enough, and so they are not representative enough in textureless areas. In order to acquire more features, adjusting parameters of Canny is suggested. However, increasing features may cause additional unexpected limitations, which might result in dense triangles. Conclusively, the result of feature detection is a big consideration for the authors. Therefore, it tends to be recommended to explore a method for facilitating feature detection in the future. For the consideration of time and space efficiency, the conversion method proposed in this paper is relatively simple, but the conversion result of some specific types of pictures is completely acceptable. In the future, we plan to develop a quantitative measurement method to measure the quality of the conversion result or even directly measure the original image to determine whether this method is suitable for it.

Data Availability

The software code and samples used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This paper is supported by Medical Health Science and Technology Project of Zhejiang Provincial Health Commission (2018KY645) and Zhejiang Science and Technology Plan Project of China (2020C03091).

References

[1] T. Xia, B. Liao, and Y. Yu, "Patch-based image vectorization with automatic curvilinear feature alignment," in *Proceedings of the ACM SIGGRAPH Asia 2009 papers on—SIGGRAPH Asia*, Yokohama, Japan, December 2009.

[2] Z. Liao, H. Hoppe, D. Forsyth, and Y. Yu, "A subdivision-based representation for vector image editing," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 11, pp. 1858–1867, 2012.

[3] G. Xie, X. Sun, X. Tong, and D. Nowrouzezahrai, "Hierarchical diffusion curves for accurate automatic image vectorization," *ACM Transactions on Graphics*, vol. 33, no. 6, pp. 1–11, 2014.

[4] F. L. Bookstein, "Principal warps: Thin-Plate splines and the decomposition of deformations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 6, pp. 567–585, 1989.

[5] Q. Fu, Y. He, F. Hou, J. Zhang, A. Zeng, and Y.-J. Liu, "Vectorization based color transfer for portrait images," *Computer-Aided Design*, vol. 115, pp. 111–121, 2019.

[6] J. Yu, J. Li, Z. Yu, and Q. Huang, "Multimodal transformer with multi-view visual representation for image captioning," *IEEE Transactions on Circuits and Systems for Video Technology*, 2019.

[7] J. Yu, M. Tan, H. Zhang, D. Tao, and Y. Rui, "Hierarchical deep click feature prediction for fine-grained image recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[8] X. Yang, S. Zhou, and M. Cao, "An approach to alleviate the sparsity problem of hybrid collaborative filtering based recommendations: the product-attribute perspective from user reviews," *Mobile Networks and Applications*, vol. 25, no. 2, pp. 376–390, 2019.

[9] S. Lu, W. Jiang, X. Ding et al., "Depth-aware image vectorization and editing," *The Visual Computer*, vol. 35, no. 6–8, pp. 1027–1039, 2019.

[10] J. Shan and W. Jiang, "Research on vectorization method of complex linear image data," in *Proceedings of the International Conference on Advanced Hybrid Information Processing*, pp. 315–321, Springer, Harbin, China, July 2018.

[11] B. Kerautret, P. Ngo, Y. Kenmochi, and A. Vacavant, "Greyscale image vectorization from geometric digital contour representations," in *Proceedings of the International Conference on Discrete Geometry for Computer Imagery*, pp. 319–331, Springer, Vienna, Austria, September 2017.

[12] M. Yang, H. Chao, C. Zhang et al., "Effective clipart image vectorization through direct optimization of bezigons," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 2, pp. 1063–1075, 2015.

[13] A. Orzan, "Diffusion curves: a vector representation for smooth-shaded images," in *Proceedings of the SIGGRAPH'08*, Los Angeles, CA, USA, August 2008.

[14] S. Jeschke, "Generalized diffusion curves: an improved vector representation for smooth-shaded images," *Computer Graphics Forum*, vol. 35, no. 2, pp. 71–79, 2016.

[15] H. Lin, J. Zhang, and C. Xu, "Diffusion curves with diffusion coefficients," *Computational Visual Media*, vol. 4, no. 2, pp. 149–160, 2018.

[16] S. Zhao, F. Durand, and C. Zheng, "Inverse diffusion curves using shape optimization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 7, pp. 2153–2166, 2017.

[17] B. Price and W. Barrett, "Object-based vectorization for interactive image editing," *Visual Computer*, vol. 22, no. 9–11, pp. 661–670, 2006.

[18] J. Sun, L. Liang, F. Wen, and H.-Y. Shum, "Image vectorization using optimized gradient meshes," *ACM Transactions on Graphics*, vol. 26, no. 3, p. 11, 2007.

[19] Y. Xiao, L. Wan, C. S. Leung, Y.-K. Lai, and T.-T. Wong, "Optimization-based gradient mesh colour transfer," *Computer Graphics Forum*, vol. 34, no. 6, pp. 123–134, 2015.

[20] L. Wan, Y. Xiao, N. Dou, C.-S. Leung, and Y.-K. Lai, "Scribble-based gradient mesh recoloring," *Multimedia Tools and Applications*, vol. 77, no. 11, pp. 13753–13771, 2018.

[21] G. J. Hettinga, R. Brals, and J. Kosinka, "Colour interpolants for polygonal gradient meshes," *Computer Aided Geometric Design*, vol. 74, Article ID 101769, 2019.

[22] H. Gao, C. Liu, Y. Li, and X. Yang, "V2VR: reliable hybrid-network-oriented V2V data transmission and routing considering RSUs and connectivity probability," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[23] K. Lawonn and T. Günther, "Stylized image triangulation," *Computer Graphics Forum*, vol. 38, no. 1, pp. 221–234, 2019.

[24] D. Marwood, P. Massimino, M. Covell et al., "Representing images in 200 bytes: compression via triangulation," in *Proceedings of the 25th IEEE international conference on image processing (ICIP)*, pp. 405–409, IEEE, Athens, Greece, October 2018.

[25] R.-I. Chang and C.-Y. Su, "Color gradient vectorization for SVG compression of comic image," *Journal of Visual Communication and Image Representation*, vol. 33, pp. 235–246, 2015.

[26] S. Roy, P. Shivakumara, U. Pal, T. Lu, and G. H. Kumar, "Delaunay triangulation based text detection from multi-view images of natural scene," *Pattern Recognition Letters*, vol. 129, pp. 92–100, 2020.

[27] X. Zhang, F. Fan, M. Gheisari, and G. Srivastava, "A novel auto-focus method for image processing using laser triangulation," *IEEE Access*, vol. 7, pp. 64837–64843, 2019.

[28] H. Gao, W. Huang, and Y. Duan, "The cloud-edge based dynamic reconfiguration to service workflow for mobile ecommerce environments: a QoS prediction perspective," *ACM Transactions on Internet Technology*, 2020.

[29] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," in *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 209–216, ACM Press/Addison-Wesley Publishing Co., Los Angeles, CA, USA, August 1997.

[30] H. Gao, L. Kuang, Y. Yin, B. Guo, and K. Dou, "Mining consuming behaviors with temporal evolution for personalized recommendation in mobile marketing apps," *Mobile Networks and Applications*, vol. 25, no. 4, pp. 1233–1248, 2020.

[31] D. T. Lee and B. J. Schachter, "Two algorithms for constructing a Delaunay triangulation," *International Journal of*

Computer & Information Sciences, vol. 9, no. 3, pp. 219–242, 1980.

[32] Y. Hu, T. Schneider, X. Gao et al., "TriWild: robust triangulation with curve constraints," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, p. 52, 2019.

[33] K. W. Chen, Y. S. Luo, Y. C. Lai et al., "Image vectorization with real-time thin-plate spline," *IEEE Transactions on Multimedia*, vol. 22, no. 1, pp. 15–29, 2019.

[34] H. Gao, Y. Xu, Y. Yin, W. Zhang, R. Li, and X. Wang, "Context-aware QoS prediction with neural collaborative filtering for Internet-of-Things services," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4532–4542, 2019.

[35] J. F. Canny, "A computation approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, 1986.

Hindawi
Security and Communication Networks
Volume 2020, Article ID 8871626, 9 pages
https://doi.org/10.1155/2020/8871626

Research Article

Cyborgan OS: A Lightweight Real-Time Operating System for Artificial Organ

Pan Lv, Hong Li, Jinsong Qiu, Yiqi Li, and Gang Pan

College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China

Correspondence should be addressed to Jinsong Qiu; jschiu@zju.edu.cn

Received 10 July 2020; Revised 18 September 2020; Accepted 30 September 2020; Published 14 October 2020

Academic Editor: Honghao Gao

The software of artificial organ is more and more complex, but it lacks real-time operating system to manage and schedule its resources. In this paper, we propose a lightweight real-time operating system (RTOS) Cyborgan OS based on the SmartOSEK OS. Cyborgan OS optimizes and improves it from the code size, context switch, low power consumption, and partial dynamic update, making it suitable for the artificial organ control system. Finally, we use the heart blood pump model to analyze the task allocation and execution sequence as well as the code size of the whole program. In this application, the maximum space occupied by the code is only 15 kB, which is suitable for most microcontrollers.

1. Introduction

Cyborg is a biological creature whose functioning has been enhanced through integration of mechanical, electrical, computational, or otherwise artificial components [1]. Among them, the artificial organ and cyborg insect are typical representatives [2, 3]. From a perspective provided by Zheng et al., the human will be enhanced by artificial electromechanical components and become new cyborg intelligent systems [4, 5].

In the field of artificial organs, a lot of research work is focused on organ materials, mechanical structure design, application systems, and so on [6, 7]. Kosaka et al. have proposed a remote artificial heart monitoring system based on Internet and noninvasive sensors, which can manage the physiological state of patients and the working state of artificial heart in real time and effectively [8]. The authors in [9] propose a microcontroller pacemaker for bradycardia pacing, along with a PC-based programmer. Its algorithm is implemented in Arduino microcontrollers, which is programmed using PC and MATLAB. Markovic et al. design a blood pump for a wearable artificial kidney device. Its control system for the operation of the blood pump is composed of a central computer, power cards, and position measurement cards [10]. The power card contains a microcontroller STM32F103, which controls the power switch of the blood pump. Despite all those accomplishments, according to our limited knowledge, there is no special operating system for artificial organs. As the artificial organs are becoming more powerful, especially the application of smartphones in artificial organs, the complexity of artificial organ is greatly increased [11]. It is necessary to resort to unified management and scheduling of hardware and software resources of artificial organs. Therefore, it is advisable to adopt an embedded RTOS as the basic software platform of artificial organs. Limited by the physical conditions of human body, microcontrollers used in artificial organs generally have limited memory and very small battery capacity. This application scenario requires the embedded RTOS to be lightweight and of low power consumption at system level to extend battery life.

This paper designs and implements a lightweight real-time OS for artificial organs and names it Cyborgan OS. The main contributions of this paper are as follows:

(1) Lightweight kernel: we implement a 10 kB level kernel based on priority scheduling, which is suitable for most Microcontroller Units (MCU)

(2) Low power design: by optimizing the system operation and taking the task wake-up alignment method, we reduce the system power consumption

(3) Partial dynamic software update: to make the artificial organ have the ability of function upgrade, we have realized partial dynamic software update, which makes the software upgrade without having to halt the operating system

The rest of this paper is organized as follows. Section 2 introduces the related work. Section 3 outlines the system overview of the Cyborgan OS. Section 4 describes the basic functionality of the Cyborgan OS kernel. Section 5 presents the power management service and the partial dynamic software update service employed in implementing the Cyborgan OS. Section 6 provides a concrete application example to evaluate Cyborgan OS. Finally, the conclusions are drawn in Section 7.

2. Related Work

In the field of Internet of Things (IoT), operating system, communication, security, and cloud task scheduling are the key technologies [12–15]. Artificial organs are a typical application scenario of operating system for low-end IoT devices since artificial organs are generally built around a microcontroller with very constrained memory and small battery capacity. Hence, the same challenges are posed for OS designers for artificial organs when it comes to handling the highly limited hardware resources.

A generic OS for low-end IoT devices should have a small memory footprint, support for heterogeneous hardware, network connectivity, energy efficiency real-time capability, and security [16]. TinyOS is one of the earliest application-specific operating systems that directly deals with the limitations of sensor devices. It has a lightweight event scheduler where all program executions are performed in tasks that run to completion [17]. In contrast to TinyOS, Contiki is a lightweight operating system that supports dynamic loading and replacement of individual programs and services during run-time and without relinking [18]. Contiki has an event-driven kernel. FreeRTOS is a popular RTOS and has been ported to many MCUs [19]. Multithreading is supported by its preemptive microkernel and a tickless mode is provided for low power applications. The application firmware is written using FreeRTOS, and hence the device functionality is guaranteed and is designed to resist failures due to software malfunctions [20]. RIOT is an open-source microkernel-based RTOS that aims for a developer-friendly programming model and API [21]. Developed with the particular requirements of low-power wireless IoT devices in mind, RIOT uses an architecture inherited from FireKernel and provides support for multithreading with standard API [22].

Compared with the previous operating system, the SmartOSEK OS that our design is based on is an embedded RTOS for automotive electronics. It guarantees real-time and reliability through full preemptive task scheduling and static memory allocation [23, 24]. The kernel of SmartOSEK OS is configurable and tailorable, so its code size can be small.

In terms of embedded low-power technology, Dynamic Voltage and Frequency Scaling (DVFS) is an effective energy-saving technology, which is widely used in embedded systems. Classic DVS algorithms include the PAST algorithm proposed by Weiser, the AQRS algorithm proposed by Yao et al. [25, 26]. FreeRTOS applies a tickless idle technique to achieve more power saving during the processor idle periods [27].

For the moment, most dynamic software updates designed for embedded systems are achieved via bootloader, which is a part of the microcontroller [28–30]. But the software update method via bootloader has a drawback that the update involves the whole system and will halt the entire system when the update is underway. When it comes to artificial organs, as they are already implanted in the human body, halting the control system can be dangerous and is catastrophic. Also it is mentioned in [31] that the communication rate of artificial organs is relatively low, and the partial dynamic software update method can reduce the amount of code to be updated and reduce the update time consumption. In this study, we propose a method of partial dynamic software updating, which can update the program without stopping the system.

3. System Overview

The Cyborgan OS consists of kernel, service, and driver as shown in Figure 1. In the OS kernel, task, interrupt, and alarm are the minimum function subsets to realize the operating system functions. The other four modules can be tailored according to the needs. In this way, the memory footprint of the OS kernel can be reduced as much as possible, so as to adapt to more types of microcontrollers.

The service layer abstracts common functions into services and provides them for system or application calls. At present, we have implemented two services: power management service and partial software update service. The driver layer realizes the basic hardware drivers such as microcontroller drivers, communication drivers, and I/O drivers. Given that drivers are already very common technologies, this paper will not introduce them in detail. As mentioned above, the Cyborgan OS is designed to provide a lightweight operating system kernel with low power consumption to provide system level support for the complex functionality of artificial organs. Therefore, its design should follow several principles:

(1) Real-time: In common scenarios of real-time control systems such as blood pump of artificial heart, it is often required that the microcontroller gives a response within a predefined time interval. Hence, the Cyborgan OS should support real-time task scheduling.

(2) Tailorability: The Cyborgan OS needs good tailorability, because when constructing different applications, irrelevant modules can be deleted considering the limited memory resources of the microcontroller chip.

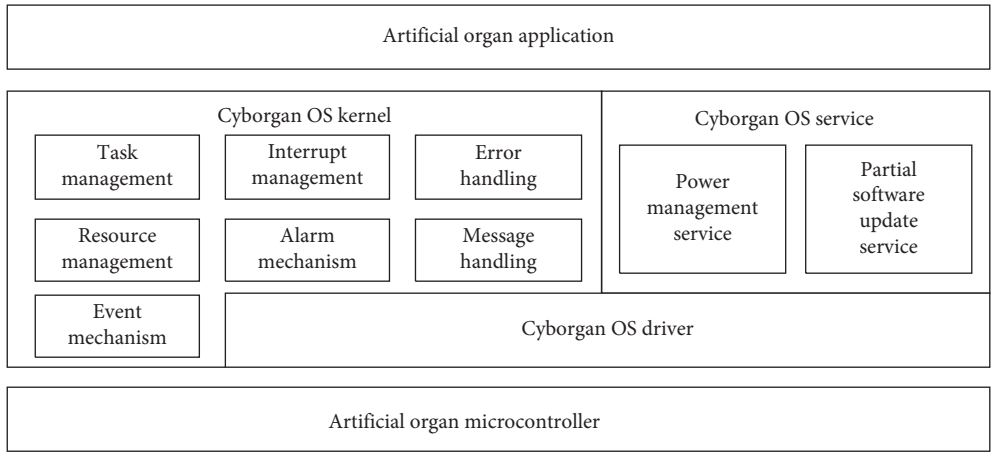


Figure 1: Architecture of Cyborgan OS.

(3) Low power design: Artificial organs installed inside human bodies often cannot obtain stable and continuous power supply. Hence, the operating system should have low power consumption and can switch dynamically between operating modes in accordance with the task status.

(4) Partial dynamic software update: During the software update, the normal functioning of artificial organs cannot be halted, and the update should be done as quickly as possible. So partial dynamic software update is applied to upgrade the functions of artificial organs.

4. Cyborgan OS Kernel

The kernel of Cyborgan OS initially evolved from the SmartOSEK OS [24]. It provides basic functions such as task management, interrupt management, alarm mechanism, resource management, event mechanism, messages handling mechanism, and error handling mechanism. The last four functions can be tailored dynamically according to the requirements of artificial organ applications to reduce the kernel size. All other functional components, such as power management and partial dynamic software update, are kept separate from the kernel and described in the next section.

4.1. Task Management. In common scenarios of real-time control systems such as the blood pump of artificial heart, the control software of artificial organs can be subdivided into functions implemented in the form of tasks. It is often required that the microcontroller give a response within a predefined time interval. Hence, the Cyborgan OS supports fully preemptive scheduling policies. For example, in the application of artificial heart, blood pump must run periodically, its control task should have the highest priority, and it can preempt low priority tasks such as external communication.

In order to save the task switching time of Cyborgan OS, we optimize the task switching according to the method in paper [32]. As shown in Figure 2, the basic idea is to divide

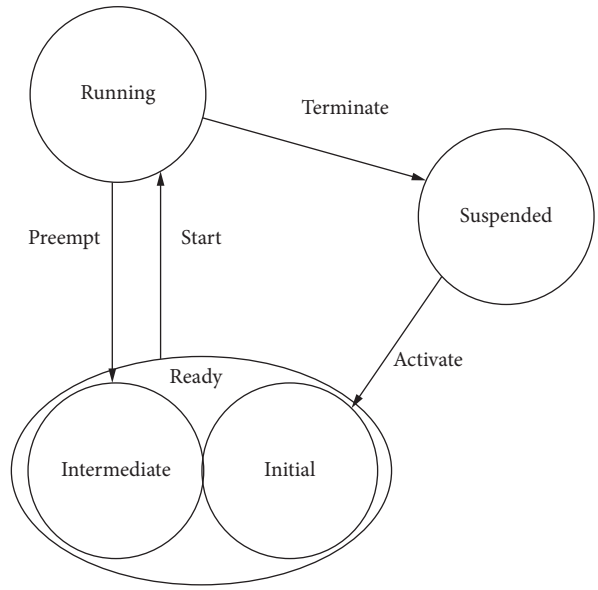


Figure 2: The improved basic task model [32].

the ready state into intermediate state and initial state. According to different states, different task context switching methods are used to improve the task switching speed. When the current running task is terminated into suspended state, it is not necessary to save the context because the task in the suspended state has finished. The task in initial state that is ready to run is a completely new task, so there is no context to recover from. In other words, context switching is needed only when the task enters or leaves the intermediate state.

In the application of artificial organs, the execution of tasks is often serialized, and preemption of normal tasks rarely occurs; that is, tasks rarely enter the intermediate state. Therefore, this context switching method can improve the speed of task switching.

4.2. Messages Handling Mechanism. The Cyborgan OS uses OSEK COM mechanism to realize message handling. OSEK COM supports m:n-communication [33]. It supports two

message types: internal message and external message. The internal message refers to the message transmission between tasks in MCU, which is sent and received using shared memory. The external message is used between MCU through an external physical bus such as SPI.

As shown in Figure 3, when sending an external message, OSEK COM packs the message in protocol data unit (PDU) and then sends it through the transmission layer. Receiving messages is a reverse process. In this way, the OS can shield the difference of communication hardware and achieve good portability.

4.3. Alarm Mechanism. Based on counters, the Cyborgan OS offers alarm mechanisms to handle periodic tasks such as sensor acquisition. A counter tracks the number of ticks and at each tick, the alarm value gets incremented by one. When the alarm value equals the predefined counter value, the OS will activate a task, set an event, or call an alarm callback routine, as shown in Figure 4. More than one alarm can be attached to the same counter. In addition, the OS provides services to cancel alarms and to get the current state of an alarm.

In the application of artificial organs, there are often periodic tasks such as sensor acquisition and actuator control. In this case, the alarm mechanism can be used to activate the corresponding tasks regularly.

4.4. Interrupt Management. The OS encapsulates the interrupt handling mechanism, and the ISR is subdivided into two categories:

(1) ISR category 1: The ISR category 1 does not call any system service. Once the ISR is finished, the systems return to the instruction where the interrupt has occurred. This ISR category does not affect task management.

(2) ISR category 2: The user routine in the ISR category 2 can call system services. When the ISR is done and a preemptable task is interrupted, rescheduling will take place.

In Cyborgan OS, interrupts are handled by hardware and can interrupt tasks. If a task is activated from an interrupt routine, the task will be scheduled at the end of all running or queued interrupt routines.

4.5. Resource Management. The resource management module handles the situation when tasks of different priority visit simultaneously the shared resources such as the memory and hardware devices. When multiple tasks access the same resource, because the low priority task occupies the resource first, the high priority task sharing access to the resource has to wait the resource; it then makes other low priority tasks run before these high priority tasks. To avoid priority inversion, the priority ceiling protocol is used in the resource management module.

In this protocol, each resource is assigned a priority ceiling, which is a priority higher than or equal to the highest priority of any task, which may lock the resource. When a task locks a resource, its priority will be raised immediately to the priority ceiling of the resource. When the task releases the resource, its priority will be dynamically reset to its default priority. A task can lock several resources and is required to release them in the Last In First Out (LIFO) order.

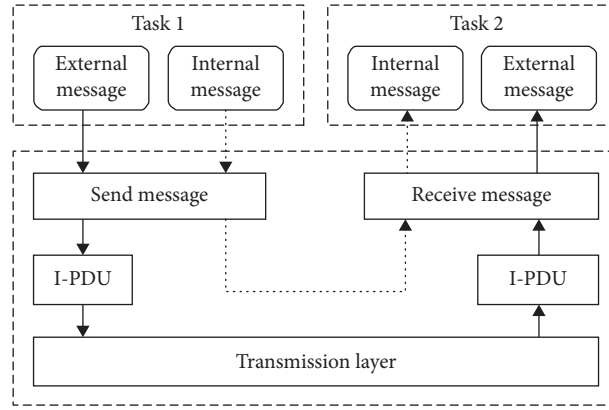


Figure 3: Structure of message handling mechanism.

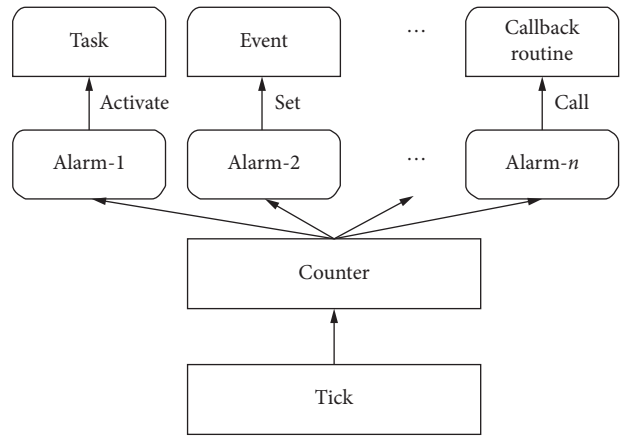


Figure 4: Structure of alarm mechanism.

4.6. Event Mechanism. The event mechanism is used for task synchronization and is provided only for extended tasks to help these tasks to switch between the wait state and other states. In the system configuration, each extended task is assigned one or several events and is called the owner of these events. An event can only be removed by its owner, and only its owner will wait for it to occur. The OS provides interfaces to set, remove, or wait for events. If a running extended task requires an event that has not yet occurred, the task will switch into the wait state. If the event that a blocked extended task is waiting for occurs, the task will enter the intermediate state. And if the event that a running extended task is waiting for has already occurred, the task will continue running.

4.7. Error Handling Mechanism. Besides handling errors by looking at the returned values, the OS also provides a hook routines mechanism in which users can define how to handle

certain errors in hook routines. The priority of error handling function is higher than that of any task so that errors can be handled on time. For example, when a sensor of an artificial organ fails, the application can call hook routines to deal with the fault ahead of all other tasks.

5. Cyborgan OS Service

In order to better meet the needs of artificial organs for low power consumption and program update, we extend the two services of power management and partial dynamic software update based on the operating system kernel. These services can be started or shut down by the application.

5.1. Power Management Service. Based on the low power modes of the microcontroller, Cyborgan OS implements a power management service that enables the operating system to switch between run mode, sleep mode, and stop mode and takes the task wake-up alignment to further alleviate the impact of task switch on system power consumption [34].

5.1.1. System Operation Optimization. In an embedded RTOS, a task is generally activated by event-based triggers or time-based triggers. In the event-based triggering, a task is activated when corresponding event such as an interrupt takes place. After the task is completed, the operating system performs the idle task and waits for the next event to trigger another task. The time-based triggering makes use of the alarm mechanism to periodically activate certain tasks. When the task is executed, the operating system will again perform the idle task. In the SmartOSEK OS, the idle task will do nothing except incrementing an uint8 variable in an infinite loop, which is a waste of system resources.

To avoid this waste, the Cyborgan OS will automatically switch to the sleep mode when the idle task is activated. When it needs to terminate the idle task and to perform other tasks, it will automatically switch to the run mode. Assuming that the idle task occupies 50% of the CPU time and the current in the run mode and in the sleep mode is 4 mA and 0.1 mA, respectively, after taking this mode-switching method, the average current is reduced from 4 mA to 2.05 mA, and the power consumption drops by 48.8%, which are significant advances.

5.1.2. Task Wake-up Alignment. In the implementation of the Cyborgan OS, the system tick is triggered by the alarm interrupt and is triggered every 1 ms in the default setting. When the system enters the sleep mode, even if there is no user interrupts tasks to be handled, the system tick will make the operating system switch from the sleep mode to the run mode every 1 ms. Figure 5(a) shows the interference from the system tick with the system operation. In [27] tickless idle technique is implemented in FreeRTOS to solve this problem. As shown in Figure 5(b), the tickless idle implies disabling the tick sourced from the idle task function, in order to ensure that the processor remains in the low power mode for longer periods of time. At the moment of entering the low power mode, the information of when a user task has to continue execution is obtained from the delayed task list and is used for the configuring of the timer to generate interrupt and wake up processor just before the task has to be deblocked.

(1) No optimization will bring unnecessary interruptions caused by the system tick, since the system tick changes the value of the timer by the interrupt. Even if there is no user task, it will leave the sleep mode due to the system tick interrupt. Taking this strategy, the power consumption (P_c) is

$$P_c = (t1 + t2 + t3 + t\text{Tick} \times \text{tickCnt}) \times a1 + t\text{Sleep} \times a2 + (\text{tickCnt} + 3) \times 2 \times a3. \quad (1)$$

(2) Applying the tickless idle technique, the system tick is temporarily disabled when the system enters the sleep mode and a general-purpose timer clocked with the system clock is used. So unnecessary system tick interrupts will not be generated in the sleep mode, and the power consumption is

$$P_c = (t1 + t2 + t3) \times a1 + t\text{Sleep} \times a2 + 3 \times 2 \times a3. \quad (2)$$

(3) On the basis of the tickless idle technique, the task wake-up alignment further reduces the number of switches between the run mode and the sleep mode. The power consumption is

$$P_c = (t1 + t2 + t3) \times a1 + t\text{Sleep} \times a2 + 2 \times a3. \quad (3)$$

In the scenario of artificial organs, most tasks are periodic, so it suffices to trigger them periodically. For this reason, this paper proposes the task wake-up alignment to optimize the system. The task wake-up alignment implies waking up for execution at the same time those tasks need to be waken up. As shown in Figure 5(c), when the period of one task is multiple that of another, we can select a time interval and wake up the tasks periodically. After each wake-up, the waken up tasks are triggered and executed in batch. After completing these tasks, the operating system performs the idle task and enters the sleep mode. Compared with the tickless idle technique, the wake-up alignment can further reduce the number of wake-ups and hence the power consumption.

As shown in Figure 5, Task1, Task2, and Task3 are needed to be waken up during period T. $t1$, $t2$ and $t3$ represent the execution time of Task1, Task2, and Task3, respectively. tSleep represents the time the system is in sleep mode. tTick represents the execution time of each system tick. tickCnt represents the number of ticks in period T. It is assumed that the current is $a1$ in the run mode, $a2$ in the sleep mode, and $a3$ during the mode switch. The three different strategies are analyzed in detail as follows:

From the above analysis, we conclude that the task wake-up alignment can achieve lower power consumption.

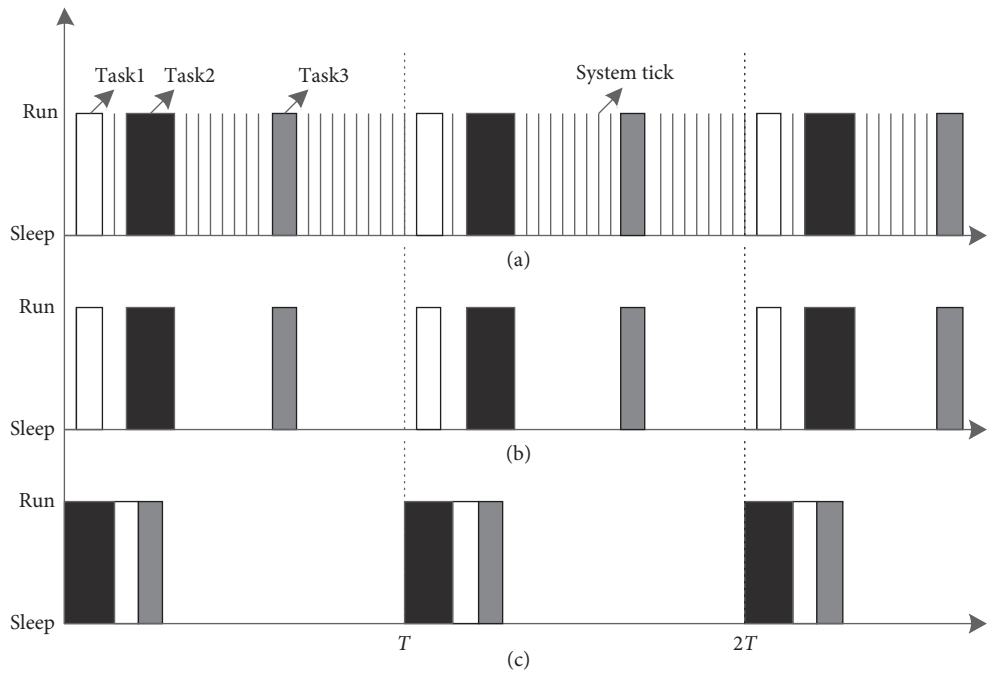


Figure 5: Operating system run/sleep mode switch using three strategies: (a) no optimization, (b) tickless idle technique, and (c) task wake-up alignment.

5.2. Partial Dynamic Software Update Service. The application scenario of artificial organs poses two challenges for designers of software update: (1) how to update the program without affecting the system's normal functioning? (2) If the update fails, how to do the version rollback of the program? To solve these two questions, this paper proposes a partial dynamic software update service.

The key of partial dynamic software update lies in the incremental update and rollback mechanism, which means that every single update targets only one function, burning in the new version of the function to replace the old one. Each function that needs to be updated is placed in a predefined program section, ensuring that the update is performed at the function level.

Partial dynamic software update divides the microcontroller's memory into a fixed code area and updatable code area, as shown in Figure 6. The kernel of the operating system resides in the fixed code area since it is not allowed to modify the kernel of the operating system. In the early stage of designing the application software for the artificial organs, it is necessary to estimate which functions may need to update afterward and place these functions in the updatable code area.

The function is the minimum component that can be updated by the dynamic software update in part. Taking advantage of the compiler's ability to assign the compiled code to a designated address, we put each function that needs update separately in a self-defined program section. The self-defined program section is physically mapped to the updatable code area. According to the estimated number of functions that may need to be updated, we partition the updatable code area to make it convenient to store the updated program.

Once the patch file is received by the microcontroller of artificial organs via Wireless Body Area Networks (WBAN) [35], it will be parsed and safety-checked. After the program is burnt into the designated memory, the corresponding function pointer will be modified so that the application program can use the newly updated function. The framework of the dynamic software update in part is shown in Figure 7.

We designed a rollback mechanism to improve reliability. When the data verification or program burning fails, the update will be stopped and the previous version of the program will be reloaded so that the program continues to run the old version of the program.

6. The Application of Cyborgan OS

In this paper, we instantiate the operating system based on NXP KL02 microcontroller, which only has 32 kB ROM and 4 kB RAM. To verify the feasibility of the Cyborgan OS for artificial organs, this paper tests it in a heart-blood pump model. The heart-blood pump model uses an artificial blood pump to replace the ventricle. The model can perform respiratory monitoring, temperature monitoring and blood pump control, etc. When the patient's respiratory rate changes, the blood pump output is controlled in real-time. Closed-loop control is realized in this heart-blood pump model. Figure 8 is a picture of the heart-blood pump model.

6.1. Hardware Framework. In the heart-blood pump model, the blood pump control node (KL02-A) and the sensor acquisition node (KL02-B) are responsible for the control and data acquisition of all the peripherals. Both nodes run

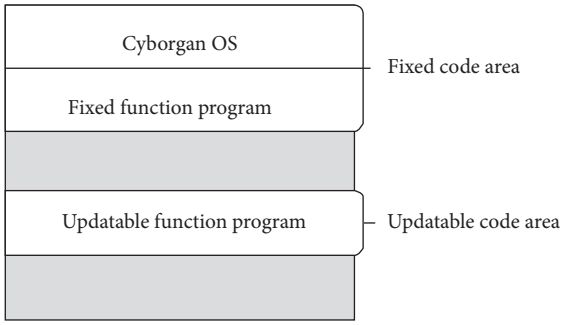


Figure 6: Layout of memory.

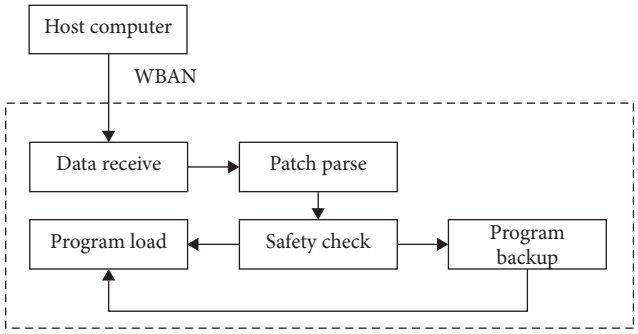


Figure 7: Framework of the partial dynamic software update.

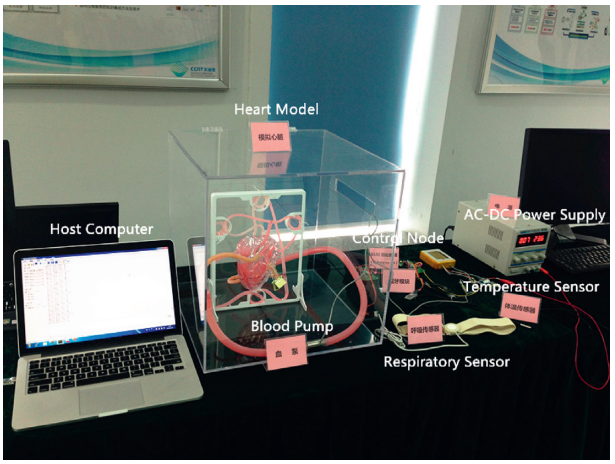


Figure 8: Heart-blood pump model.

Cyborgan OS and communicate with each other by the I^2C bus. The hardware framework is shown in Figure 9.

6.2. Application Software Design. According to the improved method mentioned above, we optimize the task allocation, make it run serially, and reduce the time of context switching. As shown in Figure 10, task OsTaskRecv and OsTaskCtrl run on KL02-A, in which OsTaskRecv receives the sensor data sent from KL02-B and OsTaskCtrl controls the blood pump. They are activated in a sequence of 100 ms cycles. Three tasks named OsTaskTemp, OsTaskRate, and OsTaskSend run on KL02-B, which are responsible for body temperature, respiratory rate acquisition, and sensor data transmission to KL02-A. According to the task wake-up alignment strategy, the OS activates these three tasks in sequence every 100 ms too.

When the blood pump is in normal operation, it pumps about 73 times/minute. If the respiratory rate or body temperature is too high, the blood pump control state will switch into an emergency state and gradually increase its pumping frequency. When exiting from the emergency state, the blood pump will gradually lower its pumping frequency in the same way, finally returning to 73 times/minute.

We used the CodeWarrior development environment to integrate the application code running on KL02-A and KL02-B with the Cyborgan OS source files into a project.

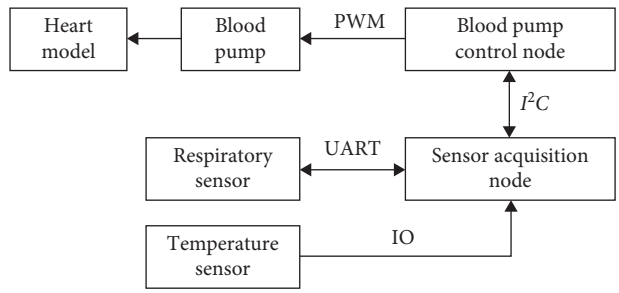


Figure 9: Hardware framework of the heart-blood pump model.

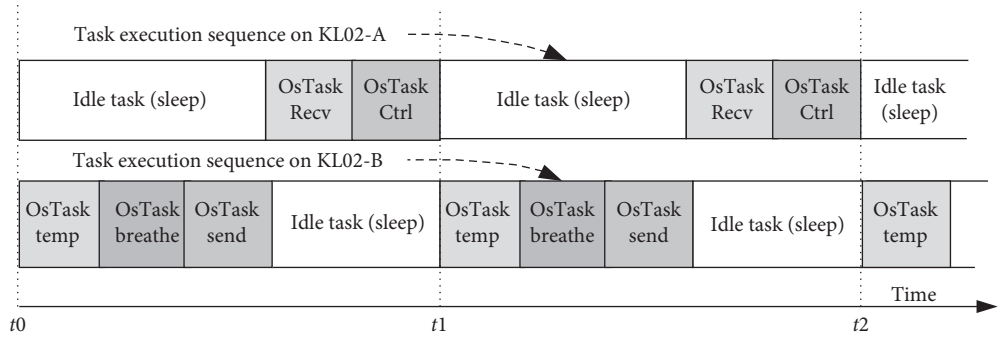


Figure 10: Task execution sequence on two nodes.

Table 1: Sizes of data segments of the ELF files generated in the heart-blood pump model.

Segment	KL02-A (byte)	KL02-B (byte)
.text	13240	11040
.data	56	66
.bss	1932	2508

Then we compile and link this project to generate the object code ELF files. Finally, the ELF file is parsed with the greadelf tool to get the sizes of data segments, as shown in Table 1.

As can be seen from Table 1, Cyborgan OS is lightweight enough to run on the KL02 platform with very limited resources.

7. Conclusion

With the increasing complexity of artificial organs, embedded real-time operating system as its basic software platform becomes inevitable. However, due to the limitations of the physical space of artificial organs, hardware resources are limited. Therefore, code size, low power consumption, and partial dynamic software update have become the three key points of artificial organ operating systems. According to the application scenario of artificial organ, this paper puts forward the design principle and structure of artificial organ operating system. According to these principles, Cyborgan OS reduces the code size of the kernel without sacrificing the basic performance, implements a low-power strategy, and adopts the partial dynamic software update method. The operating system has the characteristics of real-time, low power consumption, and small code space, which will be further extended to the other fields of Internet of things and edge computing in the future.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

Part of the paper has been published in CPSCom-2019. The work has been supported by the National Natural Science Foundation of China NSFC61602404, the National Key Research and Development Program of China 2017YFB1301102, and National Important Science & Technology Specific Projects 2017ZX01038201.

References

[1] D. Mertz, *The International Encyclopedia of Communication*, International Communication Association (ICA), Washington, DC, USA, 2008.

[2] N. Zheng, Q. Ma, M. Jin et al., "Abdominal-waving control of tethered bumblebees based on sarsa with transformed reward," *IEEE Transactions on Cybernetics*, vol. 49, no. 8, pp. 3064–3073, 2019.

[3] H. Hong, X. Wang, Z. Zhu, Q. Ma, N. Guan, and N. Zheng, "An HSR data model for cyborg insect research experiments," *Chinese Journal of Electronics*, vol. 27, no. 4, pp. 680–686, 2018.

[4] N. Zheng, L. Su, D. Zhang, L. Gao, M. Yao, and Z. Wu, "A computational model for ratbot locomotion bsed on cyborg intelligence," *Neurocomputing*, vol. 170, pp. 92–97, 2015.

[5] Z. Wu, N. Zheng, S. Zhang, X. Zheng, L. Gao, and L. Su, "Maze learning by a hybrid brain-computer system," *Scientific Reports*, vol. 6, no. 1, Article ID 31746, 2016.

[6] N. H. Cohrs, A. Petrou, M. Loepfe et al., "A soft total artificial heart-first concept evaluation on a hybrid mock circulation," *Artificial Organs*, vol. 41, no. 10, pp. 948–958, 2017.

[7] C. Fox, S. Chopski, N. Murad et al., "Hybrid continuous-flow total artificial heart," *Artificial Organs*, vol. 42, no. 5, pp. 500–509, 2018.

[8] R. Kosaka, Y. Sankai, R. Takiya, T. Jikuya, T. Yamane, and T. Tsutsui, "Tsukuba remote monitoring system for continuous-flow artificial heart," *Artificial Organs*, vol. 27, no. 10, pp. 897–906, 2003.

[9] M. Sayahkarajy, E. Supriyanto, M. H. Satria et al., "Design of a microcontroller-based artificial pacemaker: an internal pacing device," in *Proceedings of the 2017 International Conference on Robotics, Automation and Sciences*, pp. 1–5, Melaka, Malaysia, November 2017.

[10] M. Markovic, M. Rapin, M. Correvon, and Y. Perriard, "Design and optimization of a blood pump for a wearable artificial kidney device," *IEEE Transactions on Industry Applications*, vol. 49, no. 5, pp. 2053–2060, 2013.

[11] N. Reiss, T. Schmidt, M. Boeckelmann et al., "Telemonitoring of left-ventricular assist device patients-current status and future challenges," *Journal of Thoracic Disease*, vol. 10, no. 15, pp. S1794–S1801, 2018.

[12] H. Gao, C. Liu, Y. Li, and X. Yang, "V2VR: reliable hybrid-network-oriented V2V data transmission and routing considering RSUs and connectivity probability," *IEEE Transactions on Intelligent Transportation Systems*, vol. 99, 2020.

[13] H. Gao, Y. Xu, Y. W. Yin, R. Li, and X. Wang, "Context-aware QoS prediction with neural collaborative filtering for Internet-of-Things services," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4532–4542, 2020.

[14] X. Zhang, H. Gao, H. Xu, and M. Bian, "An IoT-based task scheduling optimization scheme considering the deadline and cost-aware scientific workflow for cloud computing," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, no. 1, 19 pages, 2019.

[15] Y. Xu, L. Li, H. Gao, L. Hei, R. Li, and Y. Wang, "Sentiment classification with adversarial learning and attention mechanism," *Computational Intelligence*, 2020.

[16] O. Hahm, E. Baccelli, H. Petersen, and N. Tsiftes, "Operating systems for low-end devices in the internet of things: a survey," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 720–734, 2016.

[17] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors," *ACM SIGARCH Computer Architecture News*, vol. 28, no. 5, pp. 93–104, 2000.

[18] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki-a lightweight and fiexible operating system for tiny networked sensors," in *Proceedings of the 9th Annual IEEE International Conference on Local Computer Networks*, pp. 455–462, Tampa, FL, USA, November 2004.

[19] R. Barry, *FreeRTOS: A Free Open Source RTOS for Small Embedded Real Time Systems*, Elsevier, Amsterdam, Netherlands, 2003.

[20] V. J. Devi, "Artificial cardiac pacemaker," in *Proceedings of the 2017 Third International Conference on Science Technology Engineering & Management*, pp. 1015–1017, New York, USA,, March 2017.

[21] E. Baccelli, C. Gundogan, O. Hahm et al., "RIOT: an open source operating system for low-end embedded devices in the IoT," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4428–4440, 2018.

[22] H. Will, K. Schleiser, and J. Schiller, "A real-time kernel for wireless sensor networks employed in rescue scenarios," in *Proceedings of the 2009 IEEE 34th Conference on Local Computer Networks*, pp. 834–841, Berlin, Germany, November 2009.

[23] OSEK Group, "OSEK/VDX operating system specification 2.2.3," 2003, http://www.osek-vdx.org.

[24] L. V. Hong, *Design and Implementation of SmartOSEK OS 4.0 Consulting AUTOSAR*, Zhejiang University, Zhejiang, China, 2010.

[25] F. Yao, A. Demers, and S. Shenker, "A scheduling model for reduced CPU energy," in *Proceedings of IEEE 36th Annual Foundations of Computer Science*, pp. 374–382, Milwaukee, WI, USA, October 1995.

[26] L. Chandrasena, P. Chandrasena, and M. Liebelt, "An energy efficient rate selection algorithm for voltage quantized dynamic voltage scaling," in *Proceedings of the 14th International Symposium on Systems Synthesis*, pp. 124–129, Montréal, Canada, October 2001.

[27] M. Simonovic and L. Saranovac, "Power management implementation in freeRTOS on LM3S3748," *Serbian Journal of Electrical Engineering*, vol. 10, no. 1, pp. 199–208, 2013.

[28] W. Yang and L. Luo, "The dynamic loading of the module in the embedded system," *Microcontroller & Embedded System*, vol. 11, pp. 8–10, 2005.

[29] M. Wahler, S. Richter, and M. Oriol, "Dynamic software updates for real-time systems," in *Proceedings of the 2nd International Workshop on Hot Topics in Software Upgrades*, New York City, NY, USA, January 2009.

[30] H. Wang, T. Wang, Q. Wang et al., "Remote update mechanism of reliable embedded Software based on bootloader," *Microcomputer Information*, vol. 23, no. 7, pp. 57–59, 2007.

[31] M. N. Islam and M. R. Yuce, "Review of medical implant communication system (MICS) band and network," *ICT Express*, vol. 2, no. 4, pp. 188–194, 2016.

[32] Z. Wu, H. Li, G. Yang, Z. Gao, and P. Lv, "An improved method of task context switching in OSEK operating system," *International Journal of Pervasive Computing and Communications*, vol. 6, no. 2, pp. 179–191, 2010.

[33] OSEK Group, "OSEK/VDX communication specification 3.0.1," 2003, http://www.osek-vdx.org.

[34] P. Lv, Y. Li, H. Li et al., "Optimization methods of operating system for artificial organs," in *Proceedings of the 2019 IEEE Cyber, Physical and Social Computing*, pp. 222–228, Atlanta, GA, USA, July 2019.

[35] M. Usman, M. R. Asghar, I. S. Ansari, and M. Qaraqe, "Security in wireless body area networks: from in-body to off-body communications," *IEEE Access*, vol. 6, pp. 58064–58074, 2018.

Hindawi
Security and Communication Networks
Volume 2020, Article ID 8897282, 17 pages
https://doi.org/10.1155/2020/8897282

Research Article

An Improved Broadcast Authentication Protocol for Wireless Sensor Networks Based on the Self-Reinitializable Hash Chains

Haiping Huang,[1,2] **Qinglong Huang**,[1,2] **Fu Xiao**,[1,2] **Wenming Wang**,[1,3] **Qi Li**,[1] **and Ting Dai**[4]

[1]*School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210003, China*
[2]*Jiangsu High Technology Research Key Laboratory for Wireless Sensor Networks, Nanjing 210023, China*
[3]*University Key Laboratory of Intelligent Perception and Computing of Anhui Province, Anqing Normal University, Anqing 246011, China*
[4]*Department of Computer Science, North Carolina State University, Raleigh, NC 27695, USA*

Correspondence should be addressed to Haiping Huang; hhp@njupt.edu.cn

Received 20 June 2020; Revised 24 July 2020; Accepted 31 July 2020; Published 1 September 2020

Academic Editor: Honghao Gao

Broadcast authentication is a fundamental security primitive in wireless sensor networks (WSNs), which is a critical sensing component of IoT. Although symmetric-key-based μTESLA protocol has been proposed, some concerns about the difficulty of predicting the network lifecycle in advance and the security problems caused by an overlong long hash chain still remain. This paper presents a scalable broadcast authentication scheme named DH-μTESLA, which is an extension and improvement of μTESLA and Multilevel μTESLA, to achieve several vital properties, such as infinite lifecycle of hash chains, security authentication, scalability, and strong tolerance of message loss. The proposal consists of the (t, n)-threshold-based self-reinitializable hash chain scheme (SRHC-TD) and the d-left-counting-Bloom-filter-based authentication scheme (AdlCBF). In comparison to other broadcast authentication protocols, our proposal achieves more security properties such as fresh node's participation and DoS resistance. Furthermore, the reinitializable hash chain constructed in SRHC-TD is proved to be secure and has less computation and communication overhead compared with typical solutions, and efficient storage is realized based on AdlCBF, which can also defend against DoS attacks.

1. Introduction

With the rapid development of Internet of Things (IoT) and 5G technology, the number of sensing terminals, such as various sensor nodes and tiny IoT devices, has also increased dramatically [1–3]. Edge computing is a new emerging paradigm that overcomes the scalability problem of traditional wireless sensor networks (WSNs) architecture [4–7]. The combination of wireless sensor networks and edge computing can more effectively deploy the network and process a large amount of sensory data from sensor nodes.

In hostile and harsh conditions, such as large-scale agricultural monitoring and homeland border detection, sensor nodes are usually deployed to the monitoring area by aircraft, and the base station may be a temporarily deployed edge server such as mobile weather station or UAVs (unmanned aerial vehicles for agricultural surveillance), whose computation and storage capacities are not always powerful. These sensor nodes are difficult to recycle and need to be replenished after damage or exhaustion. For effectively acquiring and perceiving data from massive sensors, the base station (or edge server) usually sends commands or application updating data packets to vast sensor nodes through broadcasting. It is necessary for sensor nodes to authenticate the identity of the sender, together with the validity and integrity of these messages [8–11]. Thus, the broadcast authentication becomes an essential service in practical and secure wireless sensor networks or IoT. The broadcast authentication protocol in wireless sensor network needs to meet the following three principles [12]: (1) any malicious

receiver being able to hardly forge any packet from the sender; (2) low communication, computation, and storage overheads; and (3) tolerance of message loss or fault.

Based on the three principles, many broadcast authentication protocols applied in WSNs have appeared sequentially. μTESLA [13] is one of the most representative broadcast authentication protocols. The contributions of μTESLA lie in the low computation strength and high authentication speed by using the special asymmetry mechanism, which is realized by delaying the disclosure of symmetric keys. Subsequently, an improved version named Multilevel μTESLA [14] has been proposed to extend the capability of μTESLA. With the continuous concern regarding broadcast authentication, some novel protocols [15–19] sequentially emerge in WSNs, followed by the Multilevel μTESLA protocol.

Although these protocols are distinguished from each other, most of them are the enhanced versions of μTESLA or Multilevel μTESLA, which contribute to the safety and efficiency by improving the organization form of broadcast authentication. However, the disadvantages of them cannot be ignored. First, a traditional asymmetric mechanism, such as non-light-weight public key cryptography (PKC) in [20], is not encouraged to be used in broadcast authentication due to its high computation overheads and impracticability in resource-constrained sensor networks. Second, the light-weight hash key chain is adopted by most of these protocols. However, an overlong hash chain may lead to the increase of storage overhead and information divulging risk on the base station, especially in the edge computing scenario, and a short one will be consumed quickly, because the running time of the network is uncertain and the entire lifetime of a large-scale network is hard to predict. Third, a long hash chain has security risks itself. Håstad and Näslund [21] show that, in a hash chain composed of the same hash functions, the attackers inverting the k-th iteration are actually k times easier than inverting a single hash function. Kwon and Hong [22] find some future keys of μTESLA by utilizing time-memory-data-tradeoff technique in a 64-bit hash chain.

Furthermore, in complex and dangerous environments, it is necessary to authenticate the fresh node supplemented in edge computing scenario. The absence of the trusted authentication interactions is easy for the vicious nodes to create fake data packets. In a severe case, the base station will suffer DoS attacks as an authentication centre. Similarly, the sensor nodes also need the ability to quickly distinguish valid messages from fake ones against DoS attacks. This means that these protocols cannot achieve a satisfactory security level.

In order to address the above problems in the existing protocols, further improve the security, and broaden application scenarios, this paper puts forward a reformed broadcast authentication protocol named DH-μTESLA. The main contributions of our proposal can be described as follows:

(i) We design a (t, n)-threshold-based self-reinitializable hash chain scheme (SRHC-TD), which constructs a reinitializable hash chain securely to improve the lifecycle of hash chains while keeping a desirable efficiency. In our scheme, hash chains will continue to generate without the need to predict and determine the lifetime of the network in advance. This scheme also has a strong tolerance to message loss.

(ii) A d-left-counting-Bloom-filter-based authentication scheme (AdlCBF) is proposed, which can handle the secure authentication of fresh sensor nodes. The AdlCBF scheme will effectively achieve the demands of authentication speed, memory space, and data security with the increasing number of sensor nodes joining the network. This scheme ensures that our protocol is well scalable for a large-scale network, and it can effectively resist DoS attacks on the base station caused by the request of massive illegal nodes.

The remainder of this paper is organized as follows. Section 2 gives the overview of related work. Some notations and concepts are defined in Section 3. The SRHC-TD scheme and AdlCBF scheme are described in Section 4 and Section 5, respectively. In Section 6, we describe the DH-μTESLA protocol in detail. Section 7 illustrates the security analysis. The evaluation and comparison of the proposed schemes are described in Section 8. Section 9 concludes this paper.

2. Related Work

2.1. Hash Chain. The hash chain was first proposed by Lamport in 1981 and has been widely used in various applications due to its high security and efficiency, including one-time password (OTP) system [23] and broadcast authentication [24]. However, after all hash values of a hash chain are consumed, a new hash chain must be generated, which is expensive in most applications. For example, in OTP system, the commitment of a new hash chain and corresponding parameters need to be reregistered to the server or the client, which will consume significant computation and communication overhead [25]. In μTESLA protocol, the problem of unicasting the initial parameters on a node-to-node basis is quite expensive [24]. In order to solve this problem, many schemes have been put forward.

Bicakci and Baykal [26] and Di Pietro et al. [27] employed public key cryptography to generate a hash chain. Although the problem of limited length was solved, the computational overhead was increased (one public key operation takes hundreds of times as long as one hash operation). Park [25] constructed an infinite length hash chain by employing multiple short hash chains for OTP system. Nevertheless, this scheme cannot be used in broadcast authentication because the hash chain will be broken if any authentication message is lost.

Goyal firstly proposed the reinitializable hash chain (RHC) scheme [28], whose main idea was that when an RHC is exhausted, a new RHC can be regenerated safely and undeniably. In 2006, Zhang and Zhu put forward the self-updating hash chain (SUHC) scheme based on Hard Core Predicate algorithm [29]. The main idea of SUHC is that it

distributes the first chain's every key value along with one bit of the second one's commitment. In this manner, the receiver would gain all bits of the second chain's commitment when the first one is exhausted. On the basis of [29], Zhang et al. designed the self-renewal hash chain (SRHC) scheme [30] as the improvement of SUHC, which has a different selection algorithm of random numbers. Xu et al. also proposed a self-updating one-time password (SUOTP) mutual authentication protocol in a similar way [31]. However, in all these schemes, the commitment can be reconstructed if and only if all the random numbers were received integrally.

2.2. Broadcasting Authentication Protocol in WSNs. μTESLA, originated from TESLA, was developed for source-constrained networks [13]. However, it did not overcome the problems that TESLA protocol experiences with mobile nodes losing the authentication packets caused by high velocities, requirement of loose time synchronization, limited length of hash chain, and reliability. Therefore, lots of solutions were proposed in order to address the above-mentioned problems.

Liu et al. [16] proposed a Scalable μTESLA that introduces the use of Merkel hash tree in μTESLA for the distribution of initial parameters and commitments and also enhances scalability by increasing the number of senders. Liu and Ning proposed Multilevel μTESLA to extend the capability of μTESLA in three aspects [14]. Firstly, it predetermines and broadcasts the initial parameters rather than unicasting them by point-to-point authentication used in μTESLA. Moreover, it adopts the multilevel hash chain to distribute broadcast messages, which prolongs the usage cycle while not increasing the length of hash chain compared with μTESLA. Finally, it uses redundant message transmission and random selection strategies to distribute key chain commitments, which improves the survivability against DoS attacks.

Recently, Al Dhaheri et al. [32] proposed a TLI-μTESLA based on Multilevel μTESLA, which reduced the delay between the sender and the receiver. Kwon and Hong presented an extendable broadcast authentication scheme called X-TESLA, which considers the problem arising from sleep modes, networks failures, and idle sessions [22]. Furthermore, a long-duration TESLA [33] was proposed to overcome the finite length of hash chain used in μTESLA by employing a hierarchical hash chain.

Apart from like-μTESLA protocols, there exist other types of broadcast authentication protocols that can be applied in WSNs. Groza et al. [34] designed a light-weight broadcast authentication for controller area network that achieves immediate authentication at small costs in bandwidth. Shim et al. [35] proposed an identity-based broadcast authentication scheme called EIBAS using ID-based signature. Similarly, a Chebyshev-map-based broadcast authentication is presented by Luo et al., which also uses ID-based signature [8]. However, the overhead of ID-based signature is higher than symmetric primitive. Besides, Bloom filter (BF) and counting Bloom filter (CBF) have been explicitly and widely used in the broadcast authentication schemes [36–39] to generate and verify authentication information, realize the public key management when combined with hash chain, and compare multiple MACs (message authentication codes) for reducing the message size. Kim and An employ BF-based source authentication (BFBSA) to achieve the security of packets with variable sizes in WSNs [38]. Bao et al. [39] proposed a light-weight authentication by combining BF and TESLA, which prevents active attacks and adds a privacy-preserving feature for vehicular ad hoc networks.

3. Preliminaries

3.1. Notation. The symbols and notations used in our protocol are listed in Table 1.

Additionally,
$M_{\langle \text{mtype} \rangle} = (\langle \text{mtype} \rangle, \langle \text{sadr} \rangle, \langle \text{dadr} \rangle, \langle * \rangle)$ is designed as a message format, where $\langle \text{mtype} \rangle$ is the message type, $\langle \text{sadr} \rangle$ is the source address, $\langle \text{dadr} \rangle$ is the destination address, and $\langle * \rangle$ represents some additional options.

3.2. Basic Definitions

Definition 1 (partition). Partition is the process of transforming a binary number of L bits into m 2^l-radix numbers. This process can be simply represented as $L = (m, l)$ or $m = \lceil L/l \rceil$. If l cannot divide L exactly, several zeros whose number is exactly the remainder should be filled on the front of the binary number.

Definition 2 (repetition vector, value vector, and repetition degree). Suppose that there are m $(m \geq 1)$ variables, and the value of each variable is in a set containing n $(n \geq 1)$ integers. For a group of these variables, there are q $(1 \leq q \leq \min(m, n))$ different values among these variables, which are denoted as v_i, $i = 1, 2, \ldots, q$. Then, $m - q$ is called repetition degree and $(v_1, v_2, \ldots, v_q)$ is called value vector. Suppose that there are p_i $(1 \leq p_i \leq m)$ variables whose value is v_i, such that $\sum_{i=1}^{q} p_i = m$. Then, $(p_1, p_2, \ldots, p_q)$ is called repetition vector.

Definition 3 (repetition rate). For m variables, the values are a set of n integers, such that repetition degree is $m - q$. Then the number of assignments of these variables is

$$S_{m,n}^{q} = C_m^{p_1}, C_{m-p_1}^{p_2}, \ldots, C_{p_q}^{p_q} \cdot \frac{n!}{(n-q)!}. \tag{1}$$

The repetition rate P_q is defined as

$$P_q = \frac{S_{m,n}^{q}}{\sum_{i=1}^{\min(m,n)} S_{m,n}^{i}}. \tag{2}$$

Definition 4 (difficulty degree). Suppose that there are m integers whose repetition degree is $m - q$. Then q is called difficulty degree.

Table 1: Notations used in the protocol.

Symbols	Description
n	Length of hash chain
L	Output length of hash function
S	The secret that needs to be shared
I_i	Secret shares
P_U, P'_U	Seed of hash chain
m_L	Number of secret shares
ξ_i, ζ_i	Key authentication code
T_c	Current time used to synchronize the time of the whole network
Δ	Maximum clock difference between the sender and the receiver
T_i	The start time of interval i
T_{int}	Duration of each time interval
δ	Disclosure delay
S_i	Sensor node
KS_i, KP_i	Private key and public key of sensor node, respectively
S_B	Base station
ID_i, ID_B	The addresses of sensor node and base station, respectively
N, N_i	Random nonce

Definition 5 (average difficulty degree). Suppose that there are m integers, whose difficulty degree is q and repetition rate is P_q. Then average difficulty degree D_q is defined as the weighted sum of difficulty degree:

$$D_q = \sum_{q=1}^{\min(m,n)} P_q \cdot q. \tag{3}$$

Definition 6 (Mignotte's sequence). Let n and t be integers; $n \geq 2$, and $2 \leq t \leq n$. A (t,n)-Mignotte sequence is a sequence of positive integers $m_1 < m_2 < \cdots < m_n$, such that, for all $1 \leq i < j \leq n$ and $(m_i, m_j) = 1$, $m_{n-t+2} \cdot m_{n-t+3} \cdot \cdots \cdot m_n < m_1 \cdot m_2 \cdot \cdots \cdot m_t$.

4. (t, n)-Threshold-Based Self-Reinitializable Hash Chain Scheme

4.1. (t,n)-Mignotte's Threshold Secret Sharing Scheme. Given a (t,n)-Mignotte sequence, the scheme works as follows [40]:

(1) The secret S is chosen as a random integer such that $\beta < S < \alpha$, wherein $\alpha = m_1 \cdot m_2 \ldots m_t$ and $\beta = m_{n-t+2} \cdot m_{n-t+3} \ldots m_n$.

(2) The secret share I_i is chosen by the formula $I_i = S \bmod m_i$, for all $1 \leq i \leq n$.

(3) Given t distinct shares $I_{i_1}, I_{i_2}, \ldots, I_{i_t}$, the secret S can be recovered using the Chinese Remainder Theorem, and any two such S are congruent moduli $m_{i_1} \cdot m_{i_2} \ldots m_{i_t}$:

$$\begin{cases} x \equiv I_{i_1} \bmod m_{i_1}, \\ x \equiv I_{i_2} \bmod m_{i_2}, \\ \quad \vdots \\ x \equiv I_{i_t} \bmod m_{i_t}. \end{cases} \tag{4}$$

4.2. (t,n)-Mignotte's Threshold Secret Sharing Scheme. The proposed self-reinitializable hash chain has multiple phases: initialization, publication, verification, recombination, and self-renewal. A new hash chain can be reinitialized without extra communication when the last hash chain is exhausted. Figure 1 shows the framework of the broadcast authentication process involving the construction of self-reinitializable hash chain.

4.2.1. Initialization. In the initialization phase, the sender and the receiver negotiate the length of hash chain n and a secure hash function h: $\{0,1\}^* \longrightarrow \{0,1\}^L$ with a security parameter L, which means that the output of h is an L-bits string, and L can be partitioned into (m_L, l_L). At first, the sender does the following steps.

(i) Initialize a seed. Choose an appropriate (t, m_L)-Mignotte sequence, denoted as $x_1, x_2, \ldots x_{m_L}$. Unite all the sequential values x_i into S_U and $h(x_i)$ into P_U, for all $1 \leq i \leq m_L$.

(ii) Initialize a hash chain. Consider P_U as the seed and generate a hash chain of length n, as shown in (I) of Figure 1:

$$P_U, h(P_U), h^2(P_U), \ldots, h^n(P_U), \tag{5}$$

where $n-1$ is an integral multiple of m_L.

(iii) Generate the next chain. Same as the first two steps, we choose a new $(t, m_L)'$-Mignotte sequence and compute S'_U and P'_U. Then, we get a new hash chain:

$$P'_U, h(P'_U), h^2(P'_U), \ldots, h^n(P'_U). \tag{6}$$

Partition $h^n(P'_U)$ into m_L 2^{l_L}-radix numbers, denoted as $c_1, c_2, \ldots, c_{m_L}$, whose repetition degree is $m_L - q_L$ and difficulty degree is q_L:

(i) Let $\alpha = x_1 \cdot x_2, \ldots, x_t$ and $\beta = x_{m_L-t+2} \cdot x_{m_L-t+3}, \ldots, x_{m_L}$. If $t = q_L$ and $\beta < h^N(P'_U) < \alpha$, then select $h^n(P'_U)$

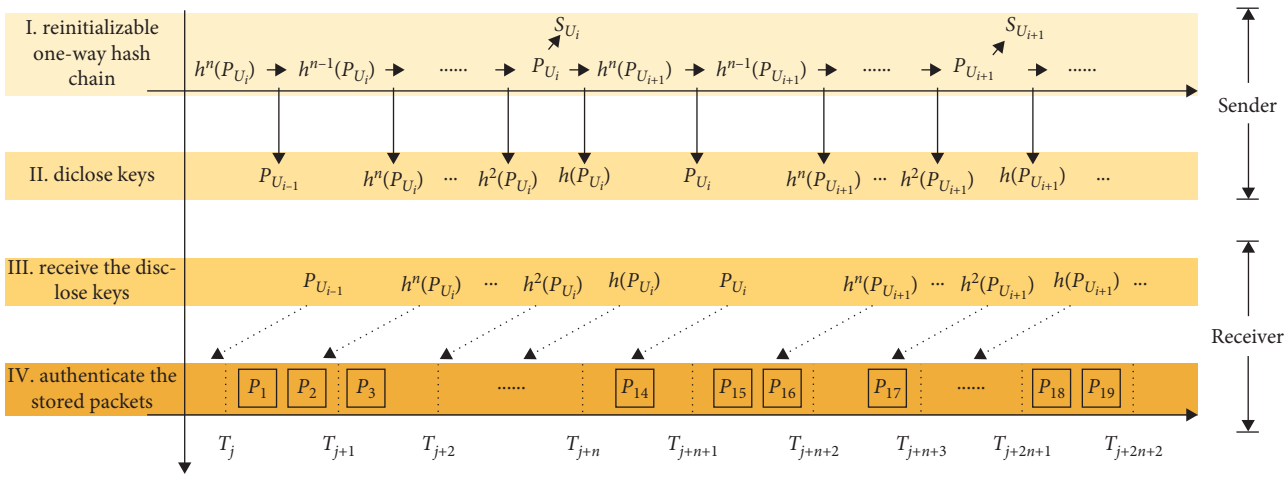


FIGURE 1: The broadcasting process uses a reinitializable one-way hash key chain. Each key is used in its corresponding time interval. After distributing the seed of the previous key chain, the next key chain's commitment $h^n(P_{U_{i+1}})$ will be calculated (I). After δ time intervals, a key will be disclosed (II). When the receiver receives the corresponding key (III), the stored packets can be authenticated (IV).

as the secret S. Otherwise, return to the previous step.

(ii) According to $I_i = S \bmod x_i$, calculate m_L secret shadows $I_1, I_2, \ldots, I_{m_L}$.

(iii) Compute the key authentication codes (KAC), denoted as ζ_i and ξ_{i-1}, where $\xi_{i-1} = h^{n-i}(P_U) \oplus \zeta_i$, $\zeta_i = (x_{c_k+1}, I_{c_k+1}) \ll r_i$, $k = i \bmod m_L$, and r_i is the number made up of the i-th bit, the $2i$-th bit, the $3i$-th bit, and so on of P_U. In particular, $\xi_0 = h^{n-1}(P_U) \oplus \zeta_1$, $\zeta_1 = (x_{c_1+1}, I_{c_1+1}) \ll r_1$, and $r_1 = S_U$.

(iv) Publish $(h^n(P_U), \xi_0)$ to the verifier securely. Actually, this pair of parameters will be sent cryptographically to the nodes in Algorithm 1 or 2 of Section 5.

4.2.2. Publication. In the phase of publication, the sender computes and distributes hash values and the corresponding certification proofs for verification. For the i-th $(i = 1, 2, \ldots, n-1)$ distribution, the sender does the following steps:

(i) Compute or retrieve the link values of hash chain $h^{n-i-1}(P_U)$ and $h^{n-i}(P_U) = h(h^{n-i-1}(P_U))$

(ii) Compute ζ_i and ξ_i

(iii) Construct and publish the certification frame $(h^{n-i}(P_U), \zeta_i, \xi_i)$, while in the n-th distribution, publish the seed P_U and S_U, as shown in (II) of Figure 1

4.2.3. Verification. For the i-th $(i = 1, 2, \ldots, n-1)$ verification, the receiver does the following steps, as shown in (III) and (IV) of Figure 1:

(1) If $\lfloor T_c + \Delta - T_i / T_{\text{int}} \rfloor < i + \delta - 1$, the receiver receives the certification frame $(h^{n-i}(P_U), \zeta_i, \xi_i)$ from the sender:

(i) Compute and verify whether $h(h^{n-i}(P_U))$ is equal to $h^{n-i+1}(P_U)$, where $h^{n-i+1}(P_U)$ is a link value sent and saved in the last valid session

(ii) Compute and verify whether $h^{n-i}(P_U) \oplus \zeta_i$ is equal to ξ_{i-1}

If all checks are passed, the receiver verifies the sender successfully and then stores ζ_i. The receiver also should store ξ_i in the buffer for the next verification.

(2) If $\lfloor T_c + \Delta - T_i / T_{\text{int}} \rfloor \geq i + \delta - 1$, the receiver drops $h^{n-i}(P_U)$ and ζ_i and saves ξ_i. Then, it will wait for the next valid certification frame $(h^{n-j}(P_U), \zeta_j, \xi_j)$, where $j > i$.

(i) Compute and verify whether $h^{j-i+1}(h^{n-j}(P_U))$ is equal to $h^{n-i+1}(P_U)$, where $h^{n-i+1}(P_U)$ is a link value sent and saved in the last valid session.

(ii) Compute and verify whether $h^{n-j}(P_U) \oplus \zeta_j$ is equal to ξ_{j-1}.

If all checks are passed, the receiver verifies the sender successfully and then stores ζ_j and ξ_j for the same reason.

4.2.4. Recombination. After all hash values have been published, the whole hash chain has been exhausted, and the receiver has stored the seed P_U and m_L or fewer ζ_is:

(i) Compute and check whether ζ_{i_1} and ζ_{i_2} share the same sequence value x_{c_k+1} and the same secret shadow I_{c_k+1}, where $i_1 - i_2 \equiv 0 \bmod m_L$, $k = i_1 \bmod m_L$, and $i_1, i_2 = 1, 2, \ldots, n-1$

(ii) After verifications for all KACs, the receiver obtains q_L distinct sequence values x_{c_k+1} and q_L distinct secret shadows I_{c_k+1}, and it also gets the ordering relation of permutation of all the m_L sequence values

Then, the receiver recovers $h^n(P'_U)$ in two ways:

(1) **for** fresh node S_i **do**
(2) $S_i \longrightarrow S_B$: $M_a = ('a', \mathrm{ID}_B, \mathrm{ID}_i, \mathrm{KP}_i, N_i)$
(3) S_B: computes and compares the fingerprint $\mathrm{ID}_i, \mathrm{KP}_i$
(4) **if** the fingerprint is true **then**
(5) $S_B \longrightarrow S_i$: $M_{ar} = ('\mathrm{ar}', \mathrm{ID}_i, \mathrm{ID}_B, A_1, \mathrm{Sign}(A_1, \mathrm{KS}_B))$, where $A_1 = T_c \,||\, (h^n(P_U)_c, \xi_{0c}, \ldots) \| T_{\mathrm{sc}} \| T_{i'} \| T_{\mathrm{int}} \| \delta \| N_B$
(6) **else**
(7) Authentication failure
(8) **end if**
(9) **end for**

Algorithm 1: Direct authentication of fresh nodes.

(1) **for** fresh node S_i **do**
(2) $S_i \longrightarrow S_B$: $M_a = ('a', \mathrm{ID}_B, \mathrm{ID}_i, \mathrm{KP}_i, N_i)$
(3) $S_{\mathrm{CH}} \longrightarrow S_B$: $M_{aa} = ('a', \mathrm{ID}_B, \mathrm{ID}_{\mathrm{CH}}, \mathrm{KP}_{\mathrm{CH}}, N_{\mathrm{CH}}, (\mathrm{ID}_B, \mathrm{ID}_i, \mathrm{KP}_i, N_i))$
(4) S_B: computes and compares the fingerprints $\mathrm{ID}_i, \mathrm{KP}_i$ and $\mathrm{ID}_{\mathrm{CH}}, \mathrm{KP}_{\mathrm{CH}}$
(5) **if** the fingerprint $\mathrm{ID}_i, \mathrm{KP}_i$ is true **then**
if the fingerprint $\mathrm{ID}_{\mathrm{CH}}, \mathrm{KP}_{\mathrm{CH}}$ is true **then**
(6) $S_B \longrightarrow S_i$: $M_{ar} = ('a', \mathrm{ID}_i, \mathrm{ID}_B, A_2, \mathrm{Sign}(A_2, \mathrm{KS}_B))$, where $A_2 = 'S_{\mathrm{CH}}$ is legal$' \| T_c \| (h^n(P_U)_c, \xi_{0c}, \ldots) \| T_{\mathrm{sc}} \| T_{i'} \| T_{\mathrm{int}} \| \delta \| N_B$
(7) $S_B \longrightarrow S_{\mathrm{CH}}$: $M_{arr} = ('\mathrm{arr}', \mathrm{ID}_{\mathrm{CH}}, \mathrm{ID}_B, A_3, \mathrm{Sign}(A_3, \mathrm{KS}_B))$, where $A_3 = 's_i$ is legal$' \| N_B$
(8) **else**
(9) $S_B \longrightarrow S_i$: $M_{ar} = ('\mathrm{arr}', \mathrm{ID}_i, \mathrm{ID}_B, A_4, \mathrm{Sign}(A_4, \mathrm{KS}_B))$, where $A_4 = 'S_{\mathrm{CH}}$ is illegal$' \| T_c \| (h^n(P_U)_c, \xi_{0c}, \ldots) \| T_{\mathrm{sc}} \| T_{i_r} \| T_{\mathrm{int}} \| \delta \| N_B$
(10) **end if**
(11) **else**
(12) **if** the fingerprint $\mathrm{ID}_{\mathrm{CH}}, \mathrm{KP}_{\mathrm{CH}}$ is true **then**
(13) $S_B \longrightarrow S_{\mathrm{CH}}$: $M_{arr} = ('\mathrm{arr}', \mathrm{ID}_{\mathrm{CH}}, \mathrm{ID}_B, A_5, \mathrm{Sign}(A_5, \mathrm{KS}_B))$, where $A_5 = 's_i$ is illegal$' \| N_B$
(14) **else**
(15) Authentication failure
(16) **end if**
(17) **end if**
(18) **end for**

Algorithm 2: Authentication of fresh nodes with cluster heads.

(i) Ordered verification: for all the m_L sequence values $x_{c_1}, x_{c_2}, \ldots, x_{c_{m_L}}$, the union of their subscripts is the commitment of the second chain, denoted by $h^n(P'_U)_1$. That is, $h^n(P'_U)_1 = c_1, c_2, \ldots, c_{m_L}$.

(ii) CRT verification: given t $(t = q_L)$ distinct pairs of (x_{c_k+1}, I_{c_k+1}), using CRT, the unique solution modulo $x_{c_1+1} \cdot x_{c_2+1}, \ldots, x_{c_t+1}$ of the set of equations is the commitment of the second chain, denoted by $h^n(P'_U)_2$.

If $h^n(P'_U)_1 = h^n(P'_U)_2$, then the recombination is successful and we can obtain the new hash chain $h^n(P'_U) = h^n(P'_U)_1 = h^n(P'_U)_2$.

4.2.5. Self-Renewal. After recombination, the next chain starts to work and another new chain is also generated including a pair of instances S''_U and P''_U and the corresponding commitment $h^n(P''_U)$. Iterations of the above processes have a result that hash chains work continuously and infinitely.

5. d-Left-Counting-Bloom-Filter-Based Authentication Scheme

In the combination of WSNs and edge computing, the base station is usually an edge server or is deployed as an unmanned aerial vehicle along with the sensor nodes, which requires the consideration of the computing and storage capabilities of the base station. These new sensor nodes need to be authenticated to make sure that they are not malicious nodes before joining the network. In the edge computing scenario, the storage space of the base station should be mainly used to cache and process the sensed data to provide data service for IoT applications. However, due to the participation of nodes, the base station needs to maintain a vast lookup table, which will occupy more storage space for authenticating the sensor nodes.

In the AdlCBF scheme, we aim to solve two problems: one is the reduction of storage overhead in the base station, and the other is the nodes' authentication and the parameters distribution of broadcast authentication when a new node joins the network. ECDSA is used to generate a signature $\mathrm{Sign}(A, \mathrm{KS}_A)$ with authentication information A

to be signed and the private key of the source KS_A. In order to reduce storage space, we introduce the d-left counting Bloom filter to construct a dlCBF to store the fingerprint for each node instead of allocating storage space directly.

5.1. Construction of dlCBF. In DH-μTESLA, every sensor node S_i holds a public key KP_i and a private key KS_i allocated by the base station S_B. The base station constructs a dlCBF to store the fingerprint of ID and key information instead of storing them directly. There are three steps to construct a dlCBF, as shown in Figure 2.

Step 1. The base station applies a hash function $H(\cdot)$ to map each node's pair of ID and public key $\langle ID_i, KP_i \rangle$ to the true fingerprint $f_{S_i} = H(ID_i \| KP)_i = (b_i, r_i)$. Each true fingerprint consists of two parts. The first part represents the bucket index b_i which corresponds to the storage location, while the second part r_i is the real practicable element to be stored in the corresponding array of b_i.

Step 2. The base station uses the additional pseudo-random permutations $P_1, P_2, \ldots, P_d$ to expand d locations, which are the d replaceable choices for each true fingerprint f_{S_i}. As shown in Figure 2, they are actually d subarrays $P_1(f_{S_i}) = (b_{i1}, r_{i1}), P_2(f_{S_i}) = (b_{i2}, r_{i2}), \ldots, P_d(f_{S_i}) = (b_{id}, r_{id})$, where $P_j(f_{S_i}) = (b_{ij}, r_{ij})$ represents the tuple of storage location and practicable element corresponding to the j-th choice of f_{S_i} $(j = 1, 2, \ldots, d)$. There exist two kinds of collisions in dlCBF. One is that two different fingerprint replacements map to the same location of subarray; that is, $P_1(f_{S_i}) = (b_{i1}, r_{i1}) \neq P_1(f_{S_j}) = (b_{j1}, r_{j1})$; however, $b_{i1} = b_{j1}$, which means that the bucket of each subarray needs more storage cells for different elements. The other is that two true but different fingerprints have the same replacement; that is, $f_{S_i} \neq f_{S_j}$; however, $P_d(f_{S_i}) = P_d(f_{S_j})$. In this case, a counter is needed to record the number of the same elements stored. So, the storage cost of bucket is the sum of all counting of its storage cells.

Step 3. The base station selects the leftmost one simultaneously with the minimum storage cost as the final storage location from the d choices.

5.2. Authentication Process of Fresh Nodes. When a fresh sensor node applies to join the network, it should be authenticated and only the legal one can acquire the key chain commitment and other configuration information. Because the participation of new nodes always occurs during the whole network lifetime, the authenticating process appears before and after the network initialization phase. In other words, there are two kinds of fresh nodes: one can communicate with the base station directly and the other one is not close to it. Thus, two different cases will be discussed, respectively. In the former case, the base station authenticates the fresh node directly by Algorithm 1. Meanwhile, in the latter case, the fresh node will be authenticated through cluster head by Algorithm 2.

Both Algorithms 1 and 2 only describe the authentication process of single fresh node. When multiple fresh sensor nodes participate in the network synchronously, the cluster heads will aggregate the authentication application messages to generate only one revised authentication message M_{aa} and then send it to the base station that can batch-process them. Thus, these two algorithms can also be used in the multiauthentication cases.

6. DH-μTESLA Protocol

Based on the above schemes, SRHC-TD and AdlCBF, there are five phases to describe the proposed protocol: setup, bootstrap, broadcast, authentication, and a new chain generation. We explain how the base station broadcasts message at the beginning of the network.

6.1. Setup. The base station first generates a sequence of secret messages using SRHC-TD scheme, which is described in Section 4.

6.2. Bootstrap. Any receiver in the network should have the commitment and relative parameters of the reinitializable one-way hash chain acquired by Algorithms 1 and 2 in Section 5, including the length of hash chain n, a secure hash function h: $\{0,1\}^* \longrightarrow \{0,1\}^L$, and the partition pair (m_L, l_L). Besides, the legal node also gains other initial parameters, such as the current time T_c, the maximum clock difference δ between the sender and the receiver, the start time T_i of interval i, and the duration of each time interval T_{int}.

6.3. Broadcast. The lifetime of a sensor node, which is much longer than that of one-way hash chain, is divided into fixed intervals of duration T_{int}. The sender uses the key value of hash chain to compute the message authentication code (MAC) of message packets in the current time interval. Then, the sender broadcasts the packet with MAC in the same time interval and discloses a key value of hash chain with corresponding key authentication code (KAC) after a certain delay $\delta \times T_{int}$.

6.4. Authentication. When a node receives a message packet with the MAC, it stores the packet and the MAC in the buffer. Once the node receives a key disclosure packet with KAC, the sensor node first verifies the key value with KAC, which is related to the message packet that has been stored, as described in Section 4.

6.5. Generate a New Chain. When the hash chain runs out, the base station should generate the next chain and the node needs to recover the commitment of the new chain.

7. Security Analysis

7.1. Resistance with Chosen Plaintext Attack. Except the initial seed and the commitment, each key of a hash chain is not only the ciphertext (as the output) but also the plaintext (as the

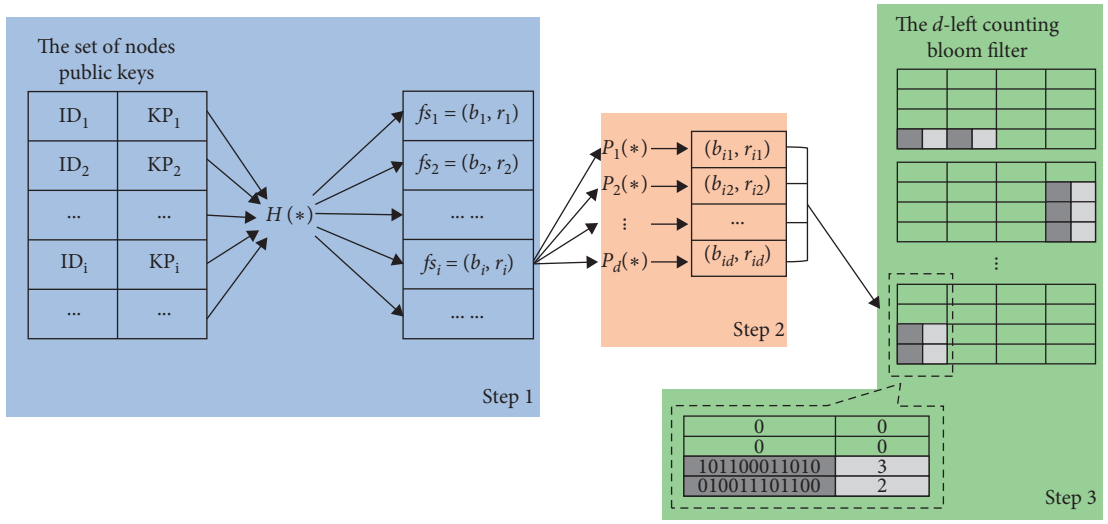


Figure 2: The construction of d-left counting Bloom filter.

input) of the hash function. Thus, each value shares the same length, which reduces the key space as well as the cracked time. Thus, it is easy to suffer from the chosen plaintext attack. In our scheme, the KAC consists of two parts, which can be used to check whether the shared secret has been changed. All ζ_i are different because of the left shift operation, which can prevent the chosen plaintext attacks effectively.

7.2. Fault Tolerance. The length of key chain $n-1$ is an integral multiple of m_L, so all (x_{c_i+1}, I_{c_i+1}) can be transmitted cyclically when the network is running. Thus, it can avoid the situation where one missing secret shadow would make the secret $h^n(P_U')$ unsolvable. Before a hash chain is running out, each receiver stores m_L (x_{c_i+1}, I_{c_i+1}) at most, while only q_L distinct sequence values $x_{(c_i+1)}$ and secret shadows $I_{(c_i+1)}$ are needed to recover the secret $h^n(P_U')$. In other words, the number of faults or missing message packets should be less than $m_L - q_L$. The highest packet loss rate that our solution can tolerate is $(m_L - q_L)/m_L$.

Assuming that the probability that a sensor node cannot receive a key disclosure packet is f and $n-1 = k \cdot m_L$ which means that the same secret shadow will be disclosed k times, the probability that a certain secret shadow I_{c_i+1} is not received by the sensor node is reduced to f^k. Suppose that the sensor node has a half chance to receive the packet (it will be lower in practice); that is, $f = 0.5$ and $k = 10$. The probability that the sensor node loses I_{c_i+1} is less than 0.1%. In our scheme, in order to recover the commitment of next hash chain, only q_L secret shadows are needed. Thus, the probability that the sensor nodes receives q_L different secret shadows is $C_{m_L}^{q_L}(1-f^k)^{q_L} \cdot (f^k)^{m_L - q_L}$. Then, the probability that the sensor node cannot obtain the commitment of next hash chain is

$$p_d = 1 - C_{m_L}^{q_L}\left(1-f^k\right)^{q_L} \cdot \left(f^k\right)^{m_L - q_L}. \tag{7}$$

If the length of hash chain is 321, which can cover about 5 minutes with time interval of 1 second, $q_L = 16$ and $m_L = 32$; the probability that the sensor node cannot recover the commitment of next hash chain when the hash chain runs out is $p_d \le 5.9 \times 10^{-40}$.

7.3. DoS Attack Tolerance. Both the base station and the sensor node are vulnerable to DoS attack that is hard to prevent. The attacker may forge the message, such as key distribution message and authentication message, to confuse both the sensor node and the base station.

In Multilevel μTESLA, when the lower level chain draws to an end, the upper level is used to authenticate the commitment for the next lower level chain by commitment distribution message (CDM),

$$\mathrm{CDM}_i = \left\|K_{i+2,0}\right\|\mathrm{MAC}_{K_i}\left(i\left\|K_{i+2,0}\right.\right)\left\|K_{i-1}\right., \tag{8}$$

in its i-th interval, where $\|$ denotes concatenation, $K_{i+2,0}$ is the commitment of the lower level chain, and K_{i-1} or K_i is a key of the upper level chain. The attacker can fabricate a CDM by replacing $K_{i+2,0}$ or $\mathrm{MAC}_{K_i}(i\|K_{i+2,0})$ and the sensor node can easily verify K_{i-1} with K_{i-2} which is released in CDM_{i-1}. Nevertheless, the sensor node has no way to authenticate other parameters in CDM_i because the key will be released later. In order to solve this problem, Multilevel μTESLA provides two variants, needing more buffers (hundreds of bytes) to store multiple CDMs in sensor node or requiring more storage space for additional precomputed chains in the base station and larger payload of CDMs. Meanwhile, in our protocol, the commitment of next chain is released with the distribution of keys. When a sensor node received a certification frame $(h^{n-i}(P_U), \zeta_i, \xi_i)$, it could be authenticated immediately by a single XOR operation. The sensor node can distinguish the validity of the message at a very fast speed without a little buffer (about 8 bytes). With the same condition in Section 7.2, assume that the bandwidth is 10 kbps and the hash function is 64 bits; the relative communication overhead is just $64 \cdot 2/10240 \cdot 1 = 0.0125$.

On the other hand, a fresh sensor node needs to be authenticated before joining the network. An attacker may launch the flood attack toward the base station with forged messages containing valid public keys, which can be obtained by eavesdropping. However, the attacker cannot alter the ID information of M_a or M_{aa}. Even if the ID can be forged, after receiving the same authentication message multiple times, the base station will find that it is being attacked. Meanwhile, a sensor node will be aware that an attack is going on and alert the base station when it fails to receive the authentication response message M_{ar} or receive the wrong responses multiple times in one round. Also, when a cluster head receives the authentication message M_a, the authentication response message M_{ar}, or added authentication response message M_{aar} multiple times from the same node, it will be aware that an attack is going on and alert the base station.

7.4. Provable Security. Because SRHC-TD is formed by many chains, we will discuss the provable security of SRHC-TD with two cases: one is the successive key in the same hash chain; and the other is the head-tail key connecting the current hash chain and the next one.

We choose the Random Oracle model to analyse the provable security of SRHC-TD. In the Random Oracle model, attackers have the polynomial calculation ability of any secret parameter k. The reliability of all algorithms of SRHC-TD is determined by the secret parameter k. Specifically, the probability of cracking hash functions is the reciprocal regarding the exponential function of k, the probability of cracking CRT is the reciprocal regarding the power function of k, and the probability of cracking partition algorithm is the reciprocal regarding the logarithmic function of k:

(a) After the ith time interval, we suppose that the receivers (including the attackers) have received $(h^{n-i}(P_U), \zeta_i, \xi_i)$. Attackers must forge h_{fake}^{n-i-1} and ζ_{i+1} within the $(i+1)$th time interval and make them meet the equations $h(h_{\text{fake}}^{n-i-1}) = h^{n-i}(P_U)$ and $h_{\text{fake}}^{n-i-1} \oplus \zeta_{i+1} = \xi_i$. Suppose that the calculation ability of attackers can be represented as the polynomial $T_{\text{adv}}(k)$ and they can query Oracle any number of times within each time interval. If the probability of cracking hash operation is e^{-k}, the probability of breaking SRHC-TD can be expressed as

$$\Pr[\text{Adv}(k) = 1] = \Pr\left[h_{\text{fake}}^{n-i-1} = h^{n-i-1}(P_U), h_{\text{fake}}^{n-i-1} \oplus \zeta_{i+1} = \xi_i\right] = \frac{T_{\text{adv}}(k)}{e^k \cdot e^k}, \tag{9}$$

where $T_{\text{adv}}(k)$ can be expressed as the general form of polynomial. So, we can deduce $\Pr[\text{Adv}(k) = 1] = a_n k^n + a_{n-1}k^{n-1} + \cdots + a_1 k + a_0/e^{2k}$. Evaluating the limit value of $\Pr[\text{Adv}(k) = 1]$, we can deduce

$$\lim_{k\longrightarrow\infty} \Pr[\text{Adv}(k) = 1] = \lim_{k\longrightarrow\infty} \frac{a_n k^n}{e^{2k}} = \lim_{k\longrightarrow\infty} \frac{a_n \cdot nk^{n-1}}{2 \cdot e^{2k}} = \cdots = \lim_{k\longrightarrow\infty} \frac{a_n \cdot n!}{2^n \cdot e^{2k}} = 0. \tag{10}$$

(b) At the ends of the $(n-1)$-th time interval, only P_U has not been published in the first hash chain. The receivers (including the attackers) have obtained m_L number of ζ_i. Attackers must forge P_U and $P_{U_{\text{fake}}}$ within the nth time interval and make them meet the following conditions: (1) $h(P_{U_{\text{fake}}}) = h(P_U)$; (2) when $i_1 - i_2 \equiv 0 \bmod m_L$, ζ_{i_1} and ζ_{i_2} contain the same sequence value x_{c_k+1} and the same shadow I_{c_k+1}, where $k = i_1 + 1 \bmod m_L$, $i_1, i_2 = 1, 2, \ldots, n-1$; (3) the solution $h^n(P'_U)_1$ derived from the proof method of ordering is the same as the solution $h^n(P'_U)_2$ derived from the proof method of CRT. The fake value $P_{U_{\text{fake}}}$ must meet all the three conditions. We suppose that the calculation ability of attackers can be represented as the polynomial $T_{\text{adv}}(k)$ and they can query Oracle any number of times within each time interval. If the probability of cracking hash operation is e^{-k}, the probability of cracking CRT is k^{-e}, and the probability of cracking partition algorithm is $1/\ln k$; then the probability of breaking SRHC-TD can be expressed as

$$\Pr[\text{Adv}(k) = 1] = \Pr\left[\begin{array}{c} h\left(P_{U_{\text{fake}}}\right) = h(P_U), h^n(P'_U)_1 = h^n(P'_U)_2 \\ x_{\left(c_{i_1}+1\right) \bmod m_L+1} = x_{\left(c_{i_2}+1\right) \bmod m_L+1}, I_{\left(c_{i_1}+1\right) \bmod m_L+1} = I_{\left(c_{i_1}+1\right) \bmod m_L+1} \end{array}\right] = \frac{T_{\text{adv}}(k)}{e^k \cdot k^e \cdot \ln k}, \tag{11}$$

where $T_{\text{adv}}(k)$ can be expressed as the general form of polynomial $T_{\text{adv}}(k) = a_n k^n + a_{n-1}k^{n-1} + \cdots + a_1 k + a_0$. Evaluating the limit value of $\Pr[\text{Adv}(k) = 1]$, we can deduce

$$\begin{aligned}\lim_{k\longrightarrow\infty} \Pr[\text{Adv}(k) = 1] &= \lim_{k\longrightarrow\infty} \frac{a_n k^n}{e^k \cdot k^e \cdot \ln k} \\ &= \lim_{k\longrightarrow\infty} \frac{a_n k^{n-e}}{e^k \cdot \ln k} = \lim_{k\longrightarrow\infty} \frac{a_n (n-e) k^{n-e-1}}{e^k \cdot (\ln k + 1/k)} = \cdots \\ &= \lim_{k\longrightarrow\infty} \frac{a_n (n-e)!}{e^k \cdot \left(\ln k + n - e/k - \cdots + (n-e-1)!/k^{n-e}\right)} \\ &= \lim_{k\longrightarrow\infty} \frac{a_n (n-e)!}{e^k \cdot \ln k} = 0.\end{aligned} \tag{12}$$

Considering the above two cases, there exists a positive integer N, when $k > N$; for any ε, there must be $\Pr[\text{Adv}(k) = 1] < \varepsilon$. So, the successful probability of attackers can be ignored and the SRHC-TD scheme has provable security.

8. Performance Evaluation

8.1. Evaluation of SRHC-TD. There are two possible ways to store a hash chain in base station. One is to generate a chain and allocate specialized space to store the whole chain. The other one is to store the seed of a chain and compute the link value when it needs to be used. Obviously, the former reduces the computation consumption but increases the memory consumption, while the latter is on the contrary.

Considering the above two situations, the consumptions of SRHC-TD will be compared with those of RHCs [28–30] from two aspects, computation and communication, as follows. Table 2 shows the variable symbols and the corresponding variable names. Table 3 shows the consumptions of SRHC-TD and RHCs in detail.

For ease of analysis, we assume that the length of the hash chain is greater than the output length of the hash function; that is $n > m$. In fact, the length of the hash chain is usually around 1000 because a short chain will lead to frequent regeneration or failure due to excessive packet loss rate. It is remarkable that if the base station only stores seeds, the calculation time increases exponentially. The length of hash chains n is a significant factor affecting the computation time.

From Table 3, we can see that the storage method of the hash chain has no effect on the communication overhead. In the initialization phase, the above schemes have the same communication overhead, while in the publication, verification, and recombination phases, our scheme has the same overhead as SRHC and is lower than RHC and SUHC, as shown in Figure 3. To evaluate the performance of our scheme in terms of computation overhead, we implement the operation used in the proposed scheme and other schemes on an Ubuntu 12.04 virtual machine with an Intel Core i5-4300 CPU @ 2.60 GHz. We use the 64-bit version of RC5 and generate a Mignotte's sequence whose length is 32. In this case, each hash operation takes 0.0037 ms. Besides, the times consumed by shifting, XOR, and modular operation, denoted as $T(M)$, $T(X)$, and $T(I)$, are 0.0002 ms, 0.0002 ms, and 0.0003 ms, respectively. The generation of a 32-bit random number and $T(B)$ takes 0.0004 ms, and it takes 0.0161 ms to recover the commitment. The time consumed in computing r_i is so little that it can be ignored.

Figure 4 shows that, in the initialization phase, the time taken by our scheme increases by approximately 0.26 ms compared to the other three schemes, due to the generation of sharing secrets. However, in publication, verification, and recombination phases, our scheme takes less time, especially when the base station stores the entire hash chain, as shown in Figure 5. If the hash chain contains 1000 keys and the base station stores the entire hash chain, compared to [28–30], the computation overhead of SRHC-TD is reduced by 39.07%, 77.08%, and 63.28%, respectively. If the base station only stores the seed of the hash chain, our scheme's computation overhead is reduced from 0.2% to 0.81%. It is worth noting that when the base station stores the entire chain, the computation time is 1000 times that of the case when it stores the seed. The main reason for this result is that when the base station releases the key which the sensor node used to authenticate the broadcast message, it needs to recover the key first. The base station needs to hash the seed value many times when the base station only stores the seed. However, if the base station stores the entire hash chain, which takes up more storage space, the key can be found easily. Thus, a more efficient way is that the base station stores some "checkpoints" for recovering the key. The greater the number of checkpoints stored by the base station is, the closer the time consumption is, according to Figure 5(a).

Thus, it is easy to compare the consumptions of SRHC-TD with the RHCs, as shown in Table 4. It shows that the SRHC-TD has a better performance in *publication*, *verification*, and *recombination* phase. The computation and communication overhead of SRHC-TD are much less than other schemes, especially when the node only stores the seed of hash chain.

Table 2: The variable symbols and the corresponding variable names.

Symbols	Description
L	The output length of hash function
n	The length of a hash chain
m	The length of Mignotte's sequence
H	The computational overhead of a hash function
R	The computational overhead of generating a random number
B	The computational overhead of mapping a random number to one single bit using Hard Core Predicate
I	The computational overhead of generating a secret shadow I_i in SRHC-TD
P_r	The computational overhead of computing r_i in SRHC-TD
M	The computational overhead of a left shift operation
X	The computational overhead of an XOR operation
C	The computational overhead of computing solutions to equations

Table 3: The consumptions in SRHC-TD and the RHCs.

Storage method			The whole chain	The seed
RHC [28]	Initialization	Computation	$(2n+3)\cdot H+2R$	$(3n+2)\cdot H+2R$
		Communication	$2L$	$2L$
	Publication and verification and recombination	Computation	$(2n+3)\cdot H+2R$	$0.5(n^2+3n-4)H+2(n-1)R$
		Communication	$4nL$	$4nL$
SUHC [29]	Initialization	Computation	$2(n+1)\cdot H+3R+B$	$3n\cdot H+3R+B$
		Communication	$2L$	$2L$
	Publication and verification and recombination	Computation	$5n\cdot H+n\cdot R+2nB$	$0.5(n^2+9n+2)\cdot H+n\cdot R+2nB$
		Communication	$4nL$	$4nL$
SRHC [30]	Initialization	Computation	$2(n+1)\cdot H+3R$	$3nH+3R$
		Communication	$2L$	$2L$
	Publication and verification and recombination	Computation	$3n\cdot H+n\cdot R+2nB$	$0.5(n^2+5n+2)\cdot H+n\cdot R+2nB$
		Communication	$3nL$	$3nL$
SRHC-TD	Initialization	Computation	$2(n+m)\cdot H+2m\cdot R+mI+X+M+P_r$	$(3n+2m-1)\cdot H+2m\cdot R+mI+X+M+P_r$
		Communication	$2L$	$2L$
	Publication and verification and recombination	Computation	$nH+2nM+2nX+2nP_r+C$	$0.5(n^2+n)\cdot H+2nM+2nX+2nP_r+C$
		Communication	$3nL$	$3nL$

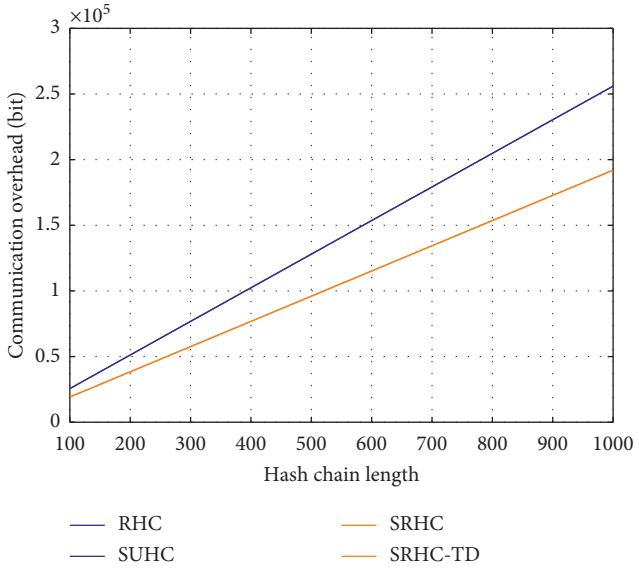


Figure 3: Comparison of communication overhead in publication, verification, and recombination phases.

As we know, the largest contributor to energy consumption in WSNs is communication complexity, and computation overhead is the second, where the energy cost of sending and receiving a packet is several times that of computing or processing a packet [41]. The *publication*, *verification*, and *recombination* phases last much longer than the *initialization* phase over the lifetime of the network, and the *initialization* phase runs in the base station. Although our scheme has a higher consumption overhead in *initialization* phase, our energy consumption is lower than other schemes on the whole.

8.2. Evaluation of AdlCBF. In AdlCBF scheme, the fresh node can be authenticated by the base station using the ECDSA signature and the fingerprint of the node. ECDSA is a light-weight algorithm and it is based on the elliptic curve. The sensor node (or cluster head node) only needs to generate and verify a signature once when it joins the network. Thus, the overhead of authenticating a fresh sensor node is acceptable. Besides, the d-left counting Bloom filter is

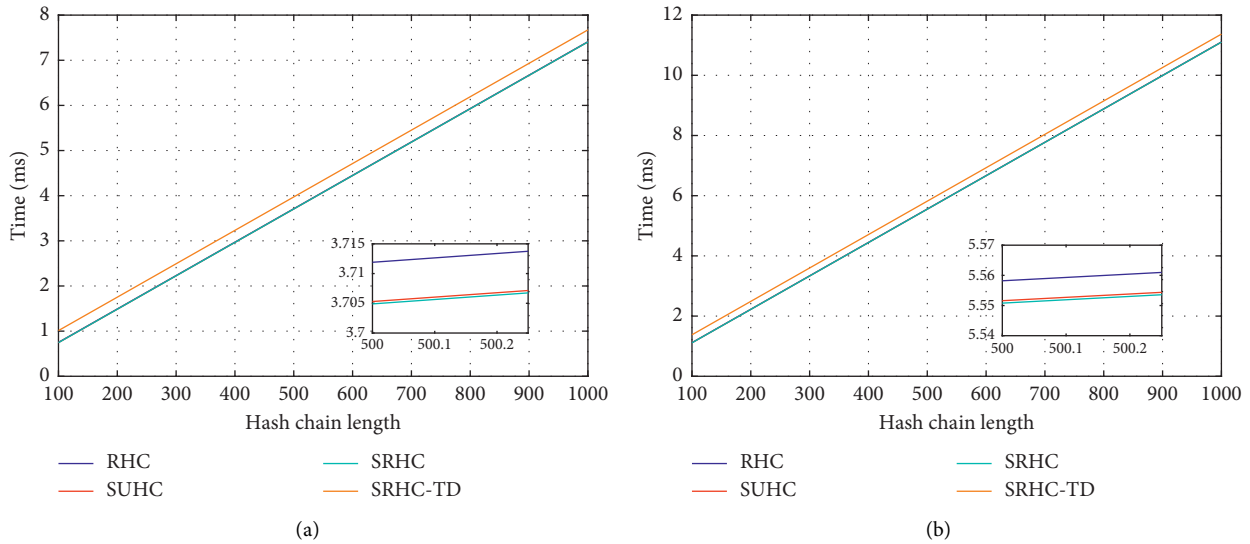


Figure 4: Comparison of time consumption in initialization phase. (a) The base station stores the whole hash chain. (b) The base station only stores the seed of hash chain. The lower right corner of the figure is a partial enlargement.

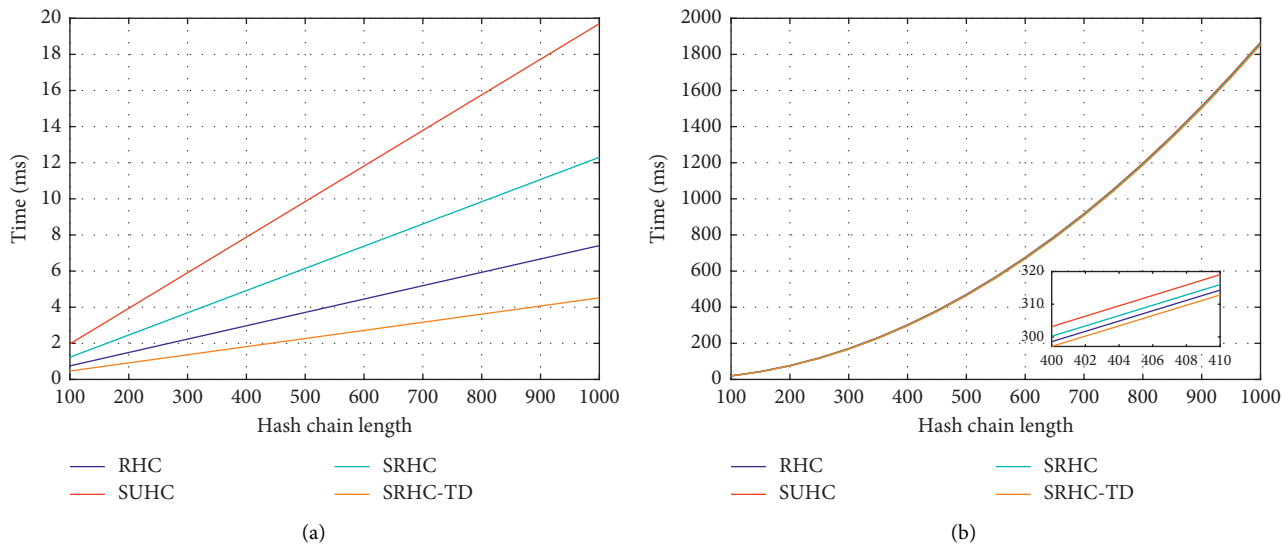


Figure 5: Comparison of time consumption in publication, verification, and recombination phases. (a) The base station stores the whole hash chain. (b) The base station only stores the seed of hash chain. The lower right corner of the figure is a partial enlargement.

Table 4: The comparisons of SRHC-TD and the RHCs.

Storage method		The whole chain	The seed
Initialization	Computation	SRHC < SUHC < RHC < SRHC-TD	SRHC < SUHC < RHC < SRHC-TD
	Communication	SRHC-TD = RHC = SRHC = SUHC	SRHC-TD = RHC = SRHC = SUHC
Publication and verification and recombination	Computation	SRHC-TD < RHC < SRHC < SUHC	SRHC-TD < RHC < SRHC < SUHC
	Communication	SRHC-TD = SRHC < RHC = SUHC	SRHC-TD = SRHC < RHC = SUHC

Table 5: The basic simulation configurations.

Parameters	Value
Elements number	$m = 500000$
Subsequences number	4
Bucket number per subsequence	$m/24 = 20833$
Average load per bucket	6
Cells number per bucket	8
Counter length	2 bits
Storage element length	$r = 14$ bits
Hash function	MD5
Pseudorandom permutation	$P_i(H(x)) = aH(x) \bmod 2^q$
Proportional parameter in permutation a	A random odd number below $[2^q]$

adopted to reduce the storage space. We will compare AdlCBF with Direct Access and CBF, respectively, regarding the storage overhead.

8.2.1. Comparison with Direct Access. The basic simulation configurations of constructing a dlCBF are shown in Table 5, where q stands for a large prime number.

Suppose that the length of public key KP_A in ECDSA is x; then, $2^x - 1 \geq n$, where n is the maximum on the elliptic curve. To guarantee the uniqueness of each node's public key, let $n \geq m$. Thus, we can obtain $x \geq \log_2(n+1) \geq \log_2(m+1) = 18.93$, and the minimum length of public key is 19 bits. In the same way, the minimum length of ID is 19 bits, too. So, if the 500000 elements are stored with the form of plaintext, the required memory space is $500000 \times (19 + 19) = 19000000$ bits. When the 500000 elements are stored in AdlCBF, the required memory space is $4m(r+2)/3 = 10666667$ bits. Thus, the former one is 1.78125 times more than the latter one. In conclusion, AdlCBF has better storage efficiency than the Direct Access.

On the constructed AdlCBF, we repeat operations of adding an element, querying an element, and deleting an element, respectively, 3000 times. We record the consumed time for each operation and calculate the average value of the recorded time for every 300 times. Thus, one set of tests is accomplished and then we keep on conducting the remaining 9 tests similarly. The result is shown in Table 6. The more intuitive representation of changes in consumption time is shown in Figure 6.

As shown in Table 6, the query time is actually the authenticating time in AdlCBF, whose changes are quite small, relative to addition and deletion. In AdlCBF, the main operations are done by CPU, while in the Direct Access, the main operations are done by RAM. As we know, the read-write speed of CPU is two or three orders of magnitude faster than RAM. According to the space and the query time of AdlCBF, we can obtain that the required time in the Direct Access is in the range of 1.78 ms to 17.81 ms. In conclusion, AdlCBF has more satisfactory query efficiency.

8.2.2. Comparison with CBF. The upper bound on false positive probability of AdlCBF is $(1/2)^r \times 2 \times 4 \times 6 = 24 \times 2^{(-r)}$, while the false positive probability of CBF is $(2^{-\ln 2})^c$, where the standard CBF uses cm counters for tracking m elements and each counter is 4 bits [42]. When $c = (r+2)/3$, the two approaches have the same amount of space. However, when considering the false positive probability ratio of CBF to AdlCBF, we obtain $(2^{-\ln 2})^{r+2/3}/(24 \times 2^{-r}) = 52.65$. Thus, with the same storage space, the false positive probability of the standard CBF is larger than that of AdlCBF.

The storage space of AdlCBF is $4\,\mathrm{m}(r+2)/3$, while the total bits of CBF is $4\,\mathrm{cm}$. When $(2^{-\ln 2})^c = 24 \times 2^{-r}$, the two approaches share the same false positive probability. However, when considering the space ratio of CBF to AdlCBF, we obtain $((4\,\mathrm{m} \cdot \ln(24 \times 2^{-r}))/(4\,\mathrm{m}(r+2)/3)) = 2.55$. Thus, with the same false positive probability, the storage cost of the standard CBF is larger than that of AdlCBF.

8.3. Comparison with Other Variants

8.3.1. Overhead. In this section, we discuss in detail the similarities and differences with Multilevel μTESLA [14] and Scalable μTESLA [22] in terms of the storage and computation requirement.

To concentrate on the comparison of storage and computation overhead, we fix the following parameters. First, we establish a two-level μTESLA, where the duration of each low-level time interval is 100 ms, and each low-level key chain consists of 600 keys. Thus, if the top-level hash chain has 300 keys, the two-level μTESLA can cover 300 minutes, where the hash function and MAC are both 64 bits. Second, we set 300 hash chains and each hash chain has 600 keys in Scalable μTESLA. Finally, in our protocol, we set the hash chain to be the same as the above hash chain, and the threshold is set to [16, 32] where each secret fragment of the new commitment will be distributed about 20 times. For clarity and ease of analysis, we assume that the packet rate is 100 per minute and these packets are sent continuously.

(1) Storage Overhead. To generate a new hash chain and obtain the commitment used in the next period, all these protocols have to buffer data in sensor nodes. In the two-level μTESLA, the sensor node uses 624-byte storage space to buffer 39 CDMs, which can achieve authenticating the commitment of next low-level hash chain with a nearly 100% probability, because the sensor node has no way to confirm whether the MAC value in CMD is modified before the

Table 6: The statistics of consumed time.

	Addition	Query	Deletion
Test 1	0.23	0.08	0.32
Test 2	0.21	0.07	0.35
Test 3	0.31	0.13	0.40
Test 4	0.25	0.12	0.42
Test 5	0.20	0.10	0.35
Test 6	0.27	0.10	0.39
Test 7	0.30	0.09	0.30
Test 8	0.24	0.08	0.38
Test 9	0.21	0.07	0.35
Test 10	0.23	0.09	0.36
Average time	0.25	0.10	0.36

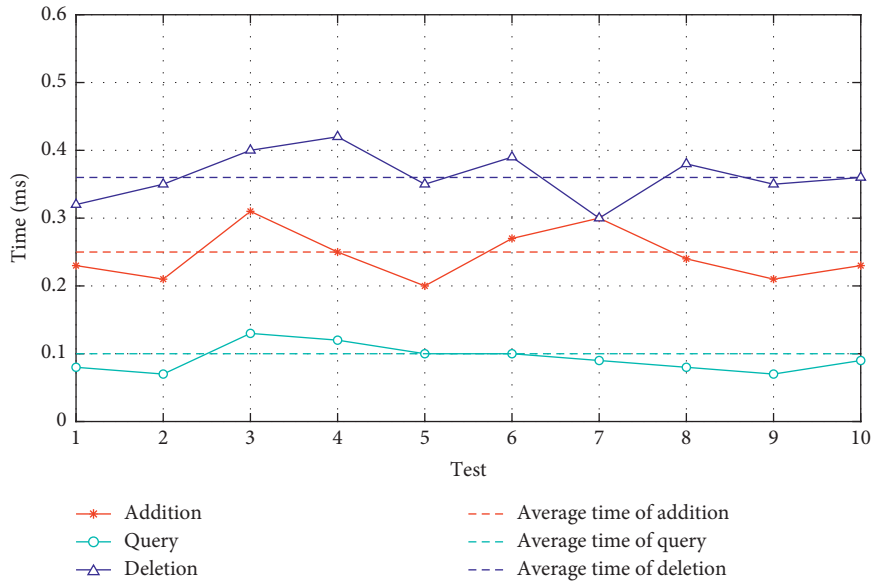


Figure 6: The statistics of consumed time.

corresponding key is released. In the Scalable μTESLA, each sensor node only needs to store a hash value, the root of Merkel hash tree, which occupies 8 bytes. If there are more than one potential sender in the network, the sensor node needs to store two roots of Merkel tree occupying 16 bytes. Meanwhile, in our scheme, the commitment of next hash chain is divided into 32 secret parts and each part is distributed along with keys used to authenticate the broadcast packets. We have to buffer 32 ζ_i's to recover the commitment of a new hash chain which only occupied 256 bytes.

(2) Computation Overhead. We focus on the computation overhead of authenticating or generating the commitment used in the next period of time and authenticate the disclosed key. In Multilevel μTESLA, when the sensor node receives a CDM, the sensor node will buffer it with a certain probability. For each buffered CDM, the sensor node needs to authenticate the top-level key first by a hash operation and the MAC will be authenticated when the corresponding key is released after a certain delay. Thus, during 300 minutes, the computation overhead is $300 \cdot 39 \cdot (T(H) + T(\text{MAC})) + 300 \cdot 100 \cdot T(H)$, where $T(\cdot)$ represents the consumed time of the operation. In the Scalable μTESLA, the sensor node needs $1 + [\log_2 300] = 10$ hash operations to verify the commitments according to the root of Merkel hash chain. Thus, the computation overhead is $300 \cdot (1 + [\log_2 300]) \cdot T(H) + 300 \cdot 100 \cdot T(H)$. However, when there are n potential senders in the network, the sensor node needs to verify the validity of the sender with additional $1 + [\log_2 n]$ hash operations first before verifying the commitment. In our proposal, part of the commitment is sent with the distribution of the corresponding key. For each distribution, the sensor node only needs a XOR operation to authenticate the commitment fragment and recover the commitment at the end of the hash chain. Therefore, the computation overhead is $300 \cdot 100 \cdot (T(H) + T(X)) + 299 \cdot T(C)$.

As shown in Table 7, in comparison with Multilevel μTESLA, our solution has less overhead due to the simple and immediate operation of authenticating the commitment of next hash chain. We have a similar computation overhead to that of Scalable μTESLA. Although Scalable μTESLA has smaller storage overhead, the sensor node needs more computation before verifying the commitment when there are

Table 7: Comparisons with other variants.

	Storage overhead (bytes)	Computation overhead (ms)
Multilevel μTESLA	624	197.5800
Scalable μTESLA*	8	122.1000
Ours	256	121.8139

*There is only one potential sender.

Table 8: Comparisons of various broadcast authentication protocols.

	Our scheme	μTESLA	Multilevel μTESLA	Long-duration TESLA	Scalable μTESLA
Structure	Single-chain	Single-chain	Multilevel	Multilevel	Merkel chain
Without using overlong hash chain	√	×	×	×	√
Reinitializable	√	×	×	√*	×
Backward security	√	√	√	×	√
Resistance with chosen plaintext attack	√	×	√	×	×
Resistance with DoS attack (sensor node)	√	×	√	×	√
Resistance with DoS attack (base station)	√	×	×	×	×
Fresh node authentication (by base station)	√	×	×	×	×
Fault tolerance	√	√	√	×	√

*The length of top-level hash chain is unlimited.

multiple potential senders, even if some of these senders only send the message once. More importantly, we eliminate the overlong or redundant hash chains that exist in Multilevel μTESLA and Scalable μTESLA in order to ensure that the hash chains are not exhausted before the whole network dies.

8.3.2. Features. Separately, we further compare DH-μTESLA with μTESLA [13], Multilevel μTESLA [14], Scalable μTESLA [22], and long-duration TESLA [33] in terms of some important aspects, and the details are shown in Table 8.

Our protocol seems more practical, since the structure of hash chain is simple while maintaining reinitialization and backward security. The long-duration TESLA is not secure once a sensor node is compromised. The attacker can forge the authentication message based on the information stored by the node due to the forward hash chain. In order to prolong the lifecycle of the network, long-duration TESLA uses an overlong hash chain which may weaken the security of the hash chain. Scalable μTESLA eliminates this threat using multiple short hash chains; however, the hash chain is not reinitializable.

In μTESLA and its variants, after receiving the broadcast message, the sensor node needs to buffer the message because it cannot be authenticated immediately. Attackers can cause huge damage to the network through DoS attacks. Therefore, resistance to DoS attacks is an important feature. In order to prevent nodes from DoS attacks, Multilevel μTESLA, Scalable μTESLA, and our proposal give a solution. However, none of these variants can handle the authentication of fresh node and DoS attack on the base station, which are important features of our protocol.

Furthermore, all protocols have a good fault tolerance except long-duration TESLA. In μTESLA, if a key disclosure packet of a broadcast message is lost, it also can be authenticated by the subsequent key disclosed. Multilevel μTESLA and Scalable μTESLA can handle the situation where the commitment of next hash chain is lost by broadcasting the same packet that contains the commitment periodically.

9. Conclusion

Our paper focuses on the key management and distribution of broadcast authentication in the combination of WSNs and edge computing with consideration of efficient storage and security issue caused by overlong hash chains, which are also important issues in IoT application scenarios. We propose an enhanced broadcast authentication protocol DH-μTESLA based on μTESLA, including SRHC-TD and AdlCBF schemes. Motivated by the fact that the disadvantages of μTESLA are the limited length of the hash chain and the security issue caused by an overlong hash chain, we design a (t, n)-threshold-based self-reinitializable hash chain scheme (SRHC-TD), which can withstand chosen plaintext attacks and is proven to be secure in Random Oracle model. Besides, in order to keep the scalability of the network, we put forward the d-left-counting-Bloom-filter-based authentication scheme (AdlCBF) and the base station is able to defend against DoS attack. The test simulation shows that, compared to other renewable hash chain schemes, our protocol has less energy consumption and the base station requires less storage space than Direct Access and CBF. Furthermore, compared to Multilevel μTESLA and Scalable μTESLA, our protocol takes less computation overhead in sensor node. In the end, we discussed the advantages of our protocol versus μTESLA, Multilevel μTESLA, long-duration TESLA, and Scalable μTESLA.

There are still some issues worthy of further study in the future work. First, in our paper, we suppose that there is only one base station in the entire network. However, in some very large-scale sensor networks, there may be multiple base stations and the fresh node needs to decide which base station to join. Second, broadcast authentication in IoT with high packet loss rate is still an open issue which would be addressed effectively.

Data Availability

Data are available upon request to the corresponding author.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant no. 61672297), the Key Research and Development Program of Jiangsu Province (Grant no. BE2017742), the Sixth Talent Peaks Project of Jiangsu Province (Grant no. DZXX-017), the National Key Research and Development Program (Grant no. 2018YFB0803403), the Postgraduate Research & Practice Innovation Program of Jiangsu Province (Grant no. KYCX19_0908), and the Key Project on Anhui Provincial Natural Science Study by Colleges and Universities (Grant nos. KJ2019A0579 and KJ2019A0554).

References

[1] M. S. Yousefpoor and H. Barati, "Dynamic key management algorithms in wireless sensor networks: a survey," *Computer Communications*, vol. 134, pp. 52–69, 2019.

[2] H. Gao, L. Kuang, Y. Yin, B. Guo, and K. Dou, "Mining consuming behaviors with temporal evolution for personalized recommendation in mobile marketing apps," *Mobile Networks and Applications*, vol. 25, no. 4, pp. 1233–1248, 2020.

[3] X. Ma, H. Gao, H. Xu, and M. Bian, "An IoT-based task scheduling optimization scheme considering the deadline and cost-aware scientific workflow for cloud computing," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, no. 1, 2019.

[4] H. Gao, W. Huang, and Y. Duan, "The cloud-edge based dynamic reconfiguration to service workflow for mobile ecommerce environments: a QoS prediction perspective," *ACM Transactions on Internet Technology*, 2020.

[5] C. Jiang, T. Fan, H. Gao et al., "Energy aware edge computing: a survey," *Computer Communications*, vol. 151, pp. 556–580, 2020.

[6] X. Sun and N. Ansari, "EdgeIoT: mobile edge computing for the internet of things," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 22–29, 2016.

[7] H. Gao, Y. Xu, Y. Yin, W. Zhang, R. Li, and X. Wang, "Context-aware QoS prediction with neural collaborative filtering for internet-of-things services," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4532–4542, 2020.

[8] Y. Luo, Y. Liu, J. Liu et al., "ECM-IBS: a Chebyshev map-based broadcast authentication for wireless sensor networks," *International Journal of Bifurcation and Chaos*, vol. 29, no. 9, Article ID 1950118, 2019.

[9] P. Soni, A. K. Pal, and S. H. Islam, "An improved three-factor authentication scheme for patient monitoring using WSN in remote health-care system," *Computer Methods and Programs in Biomedicine*, vol. 182, p. 105054, 2019.

[10] Y. Harbi, Z. Aliouat, A. Refoufi, S. Harous, and A. Bentaleb, "Enhanced authentication and key management scheme for securing data transmission in the internet of things," *Ad Hoc Networks*, vol. 94, Article ID 101948, 2019.

[11] H. Gao, C. Liu, Y. Li, and X. Yang, "V2VR: reliable hybrid-network-oriented V2V data transmission and routing considering RSUs and connectivity probability," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–14, 2020.

[12] M. A. Ferrag, L. A. Maglaras, H. Janicke, J. Jiang, and L. Shu, "Authentication protocols for internet of things: a comprehensive survey," *Security and Communication Networks*, vol. 2017, Article ID 6562953, 41 pages, 2017.

[13] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, "SPINS: security protocols for sensor networks," *Wireless Networks*, vol. 8, no. 5, pp. 521–534, 2002.

[14] D. Liu and P. Ning, "Multilevel μTESLA," *ACM Transactions on Embedded Computing Systems*, vol. 3, no. 4, pp. 800–836, 2004.

[15] W.-H. Chen and Y.-J. Chen, "A bootstrapping scheme for inter-sensor authentication within sensor networks," *IEEE Communications Letters*, vol. 9, no. 10, pp. 945–947, 2005.

[16] D. Liu, P. Ning, S. Zhu, and S. Jajodia, "Practical broadcast authentication in sensor networks," in *Proceedings of the Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services*, pp. 118–129, San Diego, CA, USA, July 2005.

[17] K. Ren, S. Yu, W. Lou, and Y. Zhang, "Multi-user broadcast authentication in wireless sensor networks," in *Proceedings of the 2007 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, vol. 58, no. 8, pp. 4554–4564, San Diego, CA, USA, June 2009.

[18] X. Li, N. Ruan, F. Wu, J. Li, and M. Li, "Efficient and enhanced broadcast authentication protocols based on multilevel μTESLA," in *Proceedings of the 2014 IEEE 33rd International Performance Computing and Communications Conference (IPCCC)*, pp. 1–8, Austin, TX, USA, December 2014.

[19] Y. Gao, P. Zeng, and K.-K. R. Choo, "Multi-sender broadcast authentication in wireless sensor networks," in *Proceedings of the 2014 Tenth International Conference on Computational Intelligence and Security*, pp. 633–637, Kunming, China, November 2014.

[20] B. Groza, "Broadcast authentication with practically unbounded one-way chains," *Journal of Software*, vol. 3, no. 3, pp. 11–20, 2008.

[21] J. Håstad and M. Näslund, "Practical construction and analysis of pseudo-randomness primitives," in *Proceedings of the Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Gold Coast, Australia, 2001.

[22] T. Kwon and J. Hong, "Secure and efficient broadcast authentication in wireless sensor networks," *IEEE Transactions on Computers*, vol. 59, no. 8, pp. 1120–1133, 2010.

[23] D. Kogan, N. Manohar, and D. Boneh, "T/Key: second factor authentication from secure hash chains," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security—CCS '17*, pp. 983–999, Dallas, TX, USA, 2017.

[24] K. Grover and A. Lim, "A survey of broadcast authentication schemes for wireless networks," *Ad Hoc Networks*, vol. 24, pp. 288–316, 2015.

[25] C.-S. Park, "One-time password based on hash chain without shared secret and re-registration," *Computers & Security*, vol. 75, pp. 138–146, 2018.

[26] K. Bicakci and N. Baykal, "Infinite length hash chains and their applications," in *Proceedings. Eleventh IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pp. 57–61, Pittsburgh, PA, USA, June 2002.

[27] R. Di Pietro, L. V. Mancini, A. Durante, and V. Patil, "Addressing the shortcomings of one-way chains," in *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security—ASIACCS '06*, p. 289, Taipei, Taiwan, 2006.

[28] V. Goyal, "How to re-initialize a hash chain," *IACR Cryptology ePrint Archive*, vol. 2004, p. 97, 2004.

[29] H. Zhang and Y. Zhu, "Self-updating hash chains and their implementations," in *Proceedings of the International Conference on Web Information Systems*, pp. 387–397, Springer, 2006.

[30] H. Zhang, X. Li, and R. Ren, "A novel self-renewal hash chain and its implementation," in *Proceedings of the 2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing*, vol. 2, pp. 144–149, Shanghai, China, December 2008.

[31] F. Xu, X. Lv, Q. Zhou, and X. Liu, "Self-updating one-time password mutual authentication protocol for ad hoc network," *KSII Transactions on Internet and Information Systems*, vol. 8, no. 5, pp. 1817–1827, 2014.

[32] A. Al Dhaheri, C. Y. Yeun, and E. Damiani, "New two-level μTESLA protocol for IoT environments," in *Proceedings of the 2019 IEEE World Congress on Services (SERVICES)*, pp. 84–91, Milan, Italy, July 2019.

[33] Y. Liu, J. Li, and M. Guo, "Long duration broadcast authentication for wireless sensor networks," in *Proceedings of the 2012 IEEE 75th Vehicular Technology Conference (VTC Spring)*, pp. 1–5, Yokohama, Japan, May 2012.

[34] B. Groza, S. Murvay, A. V. Herrewege, and I. Verbauwhede, "LiBrA-CAN," *ACM Transactions on Embedded Computing Systems*, vol. 16, no. 3, pp. 1–28, 2017.

[35] K.-A. Shim, Y.-R. Lee, and C.-M. Park, "EIBAS: an efficient identity-based broadcast authentication scheme in wireless sensor networks," *Ad Hoc Networks*, vol. 11, no. 1, pp. 182–189, Jan. 2013.

[36] J.-H. Son, H. Luo, and S.-W. Seo, "Denial of service attack-resistant flooding authentication in wireless sensor networks," *Computer Communications*, vol. 33, no. 13, pp. 1531–1542, 2010.

[37] E. Ayday and F. Fekri, "A secure broadcasting scheme to provide availability, reliability and authentication for wireless sensor networks," *Ad Hoc Networks*, vol. 10, no. 7, pp. 1278–1290, 2012.

[38] D. Kim and S. An, "Source authentication schemes for reprogramming with variable packet size in wireless sensor networks," in *Proceedings of the 2015 12th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, pp. 148–150, Seattle, WA, USA, June 2015.

[39] S. Bao, W. Hathal, H. Cruickshank, Z. Sun, P. Asuquo, and A. Lei, "A lightweight authentication and privacy-preserving scheme for VANETs using TESLA and Bloom Filters," *ICT Express*, vol. 4, no. 4, pp. 221–227, 2018.

[40] L. Harn and C. Lin, "Strong (n, t, n) Verifiable secret sharing scheme," *Information Sciences*, vol. 180, no. 16, pp. 3059–3064, 2010.

[41] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, Maui, HI, USA, January 2003.

[42] L. Fan, P. Cao, J. Almeida, and A. Z. Broder, "Summary cache: a scalable wide-area web cache sharing protocol," *IEEE/ACM Transactions on Networking*, vol. 8, no. 3, pp. 281–293, 2000.