Hindawi
Security and Communication Networks
Volume 2023, Article ID 9832148, 1 page
https://doi.org/10.1155/2023/9832148

Retraction

Retracted: Performance Test and Improvement of Computer Network Virtualization Software in Cloud Computing Environment

Security and Communication Networks

Received 5 December 2023; Accepted 5 December 2023; Published 6 December 2023

This article has been retracted by Hindawi, as publisher, following an investigation undertaken by the publisher [1]. This investigation has uncovered evidence of systematic manipulation of the publication and peer-review process. We cannot, therefore, vouch for the reliability or integrity of this article.

Please note that this notice is intended solely to alert readers that the peer-review process of this article has been compromised.

Wiley and Hindawi regret that the usual quality checks did not identify these issues before publication and have since put additional measures in place to safeguard research integrity.

We wish to credit our Research Integrity and Research Publishing teams and anonymous and named external researchers and research integrity experts for contributing to this investigation.

The corresponding author, as the representative of all authors, has been given the opportunity to register their agreement or disagreement to this retraction. We have kept a record of any response received.

References

[1] W. Liu, "Performance Test and Improvement of Computer Network Virtualization Software in Cloud Computing Environment," *Security and Communication Networks*, vol. 2022, Article ID 6965880, 7 pages, 2022.

Hindawi
Security and Communication Networks
Volume 2022, Article ID 6965880, 7 pages
https://doi.org/10.1155/2022/6965880

Research Article

Performance Test and Improvement of Computer Network Virtualization Software in Cloud Computing Environment

Wei Liu

Guangzhou Panyu Polytechnic, Guangzhou 511400, Guangdong, China

Correspondence should be addressed to Wei Liu; 20160676@ayit.edu.cn

Received 6 July 2022; Revised 9 August 2022; Accepted 16 August 2022; Published 30 August 2022

Academic Editor: C. Venkatesan

In order to make up for the problems of inconveniences, unavailability, differences, and exclusivity in the performance test of the computer virtual network software in the cloud computing environment, a method of building a distributed system performance test platform based on cloud computing is proposed. The distributed system performance test method based on the decision tree, the common performance index test method, and the resource scheduling method in the cloud environment is mainly adopted, and the cloud computing and the automatic performance test technology are combined so as to build a prototype system of the cloud performance test platform. Experimental data show that the CPU utilization and the memory utilization can be reduced to 32% and 30%, respectively, by using this method. It is about 30 times better than the result before application. The results show that it can improve the efficiency, availability, and convenience of software performance test greatly. Also, it also can improve the software performance to facilitate people's life and work in the cloud computing environment.

1. Introduction

With the rapid development of information technology, the era of cloud computing has come. And the way people apply information will undergo great changes, as well as IT-related industries and technologies such as software service mode, research and development mode, and testing mode. Enterprise application and business expand continuously, and the demand for software is growing, so the traditional software testing model is not very a good solution to the existing of the software development and application of cloud computing environment. The test resources deployment is changing and updating, including computer, software, and network equipment. It takes a lot of maintenance costs each time. The cloud computing application itself has strong security and stability. Through the cloud computing application, the storage and processing of data information can be realized. Cloud computing applications are relatively convenient. Through the cloud computing system, the application of various information resources is achieved, and resources can be quantified processing. Driven by the cloud computing technology, the data transmission and sharing are promoted to provide people with a safe and stable information sharing platform [1].

Due to the differences and exclusivity of the cloud computing architecture, each major cloud testing service provider has its own standards and specifications when studying the testing problems in the cloud environment, which directly leads to poor alignment of cloud testing standards and environments, and different test results will be obtained when tests are performed in different test environments [2].

To solve the problem in the environment, users only need to access the cloud test platform to obtain all kinds of resources and application services allocated by the cloud test platform, which brings great convenience and ease of use for the software testing. All operations are managed by the cloud test platform. After the user finishes the test project, the cloud testing platform is responsible for the recycling of resources and environment, which can be used again and meets the resources reusability principle. At the same time, it also can solve the difference and exclusiveness, which has a far-reaching influence on the traditional software testing industry and also provides convenience to people's life and work in the cloud computing environment.

2. Literature Review

According to the problems and defects existing in the current research, Guo et al. analyzed the characteristics, advantages, and challenges of the cloud testing in terms of the service categories, types, and contents. And the research direction and difficulties of TaaS were predicted, providing important references for TaaS-related researches [3]. Dehghan et al. analyzed the influence of the cloud computing on the software testing field and further elaborated the connection and similarities and differences between the local testing and the cloud testing, focusing on the new direction of cloud testing and key technologies in urgent need of breakthrough [4]. Pinkham et al. designed and implemented a software testing prototype system based on cloud computing by combining the cloud computing technology with the automated testing technology. At the 2014 Mobile World Congress, a Fusion-Sphere cloud platform based on an open-source OpenStack system was launched officially. With the accumulation of the hardware, software, and integration services in previous years, different software functions and virtualization schemes can be used to realize various network functions, achieve dynamic scheduling and allocation of network resources, improve network utilization and network expansibility, and achieve the agile business deployment [5]. Berezina et al. focused on the optimal selection of dynamic replica locations, which of course must be generated in a grid environment. The research first analyzed different grid types one by one and divided them into different categories to make preliminary preparations for the placement and selection of replica locations. If we could learn a little bit about replica technology, we would find that most of the replica technology was closely related to mathematics, and topology was a typical example of the mathematical principles followed. If the demand of users was a multidimensional network structure, the hierarchical arrangement and placement of classified copies will also be an optimal choice [6]. On June 14, 2012, Xue et al. launched an open network environment to make Cisco network more flexible and customizable, so as to adapt to the new network trend [7]. Recently, Cisco announced the launch of the Cisco Evolution Services Platform (ESP), further expanding its lineup of virtualization products and services for telecom carriers.

On the basis of the current research, a method was proposed in the research to build a distributed system performance test platform based on cloud computing. The method introduced dichotomy as the attribute value selection strategy in the process of decision tree expansion, which effectively reduced the number of performance tests of a distributed system in the cloud environment. In the cloud testing environment, the performance of the distributed system could be analyzed by simulating the load client configuration and the configuration changes of each server [8]. In the resource scheduling method of cloud environment, vector definition of test resources in a cloud environment was introduced, and the maximum and minimum thresholds of these four types of resources in the server were predefined to carry out effective load balancing control and improve the utilization efficiency of resources [9, 10]. By observing the performance indicators measured by the decision tree and the resource scheduling data of CPU before and after cloud computing, it was proved that this method could solve the problems of the reusability, difference, exclusivity, and effectiveness of the resource.

3. Methods

3.1. Decision Tree and Its Construction Method

3.1.1. The Overview of Decision Tree. Decision tree is an important data classification technique in the field of data mining. It is a tree structure similar to the flow chart. Given a set of data records, a decision tree is a graphical description of a set of classification rules, and the division based on tree structure is very clear and easier for people to understand as shown in Figure 1. Each node in the decision tree represents the set of elements in the data set to be classified. According to different values of test attributes, the decision tree expands from the root node to different child nodes. Each nonleaf node in the tree corresponds to the test of a noncategory attribute in the data set, each branch of the nonleaf node corresponds to a test result of the corresponding attribute value of the test attribute, and each leaf node represents a category in the data set [11].

3.1.2. The Implementation of Decision Tree Algorithm. Among the construction methods of decision tree, ID3, C4.5, and CART are the most commonly used algorithms to build decision tree. The ID3 algorithm selects the attribute with the highest information gain as the test attribute in the process of decision tree construction. ID3 algorithm is presented in the research, and on the basis of it, a set of pruning conditions is introduced in the process of decision tree construction. The pruning strategy is adopted to terminate the nodes that do not meet the reservation threshold in advance and further expand the test case set. The test case set is divided to reduce the size of the final generated decision tree, so as to effectively reduce the number of test case design in the performance test process [12].

Suppose the data set S is divided into S1, . . ., Sv with a total of v subsets according to v different attribute values of attribute A, then the information gain calculation formula of S divided by A is shown in the following formula:

$$\mathrm{Gain}(S, A) = \mathrm{Info}(S) - \mathrm{info}(S, A). \qquad (1)$$

In formula (1), Info(S) represents the information entropy of data set S. Suppose there are m categories in S, and the calculation formula is shown in the following formula:

$$\mathrm{Info}(S) = \sum_{i=1}^{m} p_i \times \log_2(p_i). \qquad (2)$$

Info(S, A) represents the information entropy required to divide data set S according to different values of attribute

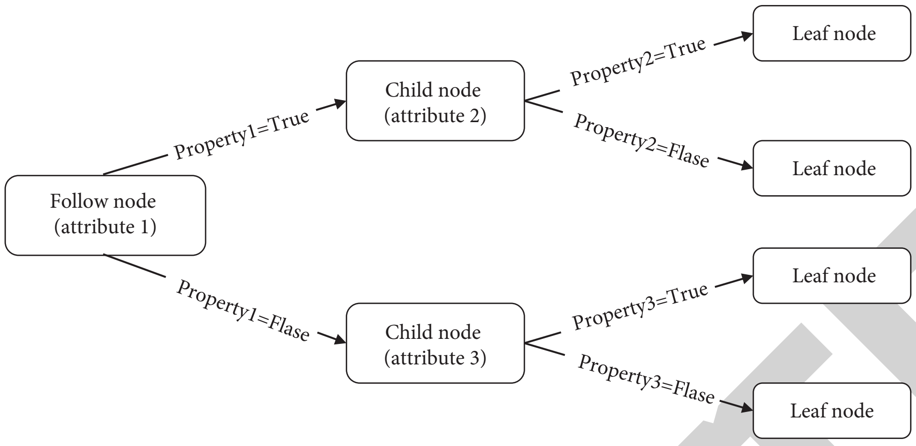

Figure 1: Decision tree structure diagram.

A. The smaller the information entropy is, the higher the purity of the partition is. The calculation formula is as follows:

$$\text{Info}\,(\text{S}, \text{A}) = \sum_{j=1}^{v} \frac{\left|S_j\right|}{|\text{S}|} \times \text{Info}\left(S_j\right), \tag{3}$$

where $|S_j|/|S|$ represents the weight in the jth partition of the test attribute A.

3.2. The Performance Test Indicators. At present, the main performance indicators used to measure the software system include response time, resource usage, system throughput, and user concurrency [13].

3.2.1. Response Time. Response time refers to the time required by the system to respond to users' requests, which can be divided into two parts: the presentation time and the system response time. The presentation time refers to the time required for the client to present the data after receiving it. System response time is the time consumed from the time a request is issued until the response data are received by the client. In order to better locate the system performance bottleneck, the system response time can be further decomposed into the network transmission time and the application delay time, which can be decomposed into the data delay time and the Web server delay time [14], for example, Web(B/S mode) user request time process decomposition diagram. The network transmission time is N1+N2+N3+N4, the application delay is A1+A2+A3, the Web server delay is A1+A3, and the database server delay is A2.

3.2.2. Resource Usage. Resource usage refers to the usage of different resources, such as server CPU usage, memory usage, disk usage, and network usage. It is the main basis for analyzing the system performance indicators and improving the system performance. The resource usage is directly proportional to the user load. If the resource usage remains constant at 100%, it indicates that the resource has become the bottleneck of the system. Increasing the capacity of this resource can significantly increase the throughput and response time of the system.

3.2.3. Throughput. Throughput refers to the number of user requests processed by the system per unit time, which directly reflects the carrying capacity of the system performance. Throughput is typically expressed in user requests per second (RPS). For systems that focus on user interaction, throughput performance indicator reflects the pressure that the system can bear [15].

3.2.4. Number of Concurrent Users. The number of concurrent users refers to the number of users who open sessions to the same application or module in the system at the same time in a given time [16]. As the number of concurrent users increases, the resource usage of the system increases. The purpose of the performance test is to verify whether the current system can support access requests of the current scale of users. The best method is to analyze the operation behavior of historical users using the system and evaluate how many users will access the system under test in the same time period [17]. Automated performance test tools are used to simulate the same number of users and their real behavior. Then, the obtained performance test results can truly and objectively reflect the performance of actual users when accessing the system.

3.3. The Case Selection Strategy of the Test. The essence of the performance test is that a collection of the typical test cases are selected from the total test cases to test the system under test and analyze the changes of the performance indicators under different load scenarios to find the main factors that may affect the performance of the system and then to make the corresponding adjustment, modification, and improvement. However, due to the large scale and high complexity of the input field in the distributed system, it is very difficult to select test cases conveniently and efficiently. Therefore, the key to distributed system performance test is how to select as few test cases as possible to find as many

Table 1: Comparative analysis of selection methods.

Methods	Advantages	Disadvantages
Manual selection method	Common or familiar test errors are avoided	It has greater blindness and uncertainty
Equivalence class partition method	Comprehensive selection of test cases	The partition requirement is very high, and the equivalence relation is difficult to determine
Boundary value method	Test cases are easy to select	There are too many redundant test cases, and the test cannot be complete
Causality diagram method	Fewer test cases are selected	The dependence between conditions and outcomes is difficult to determine

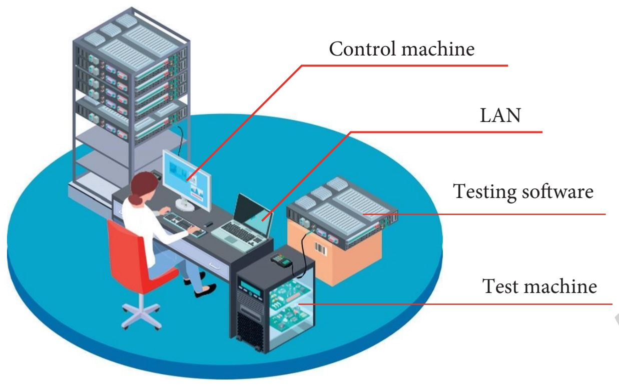

Figure 2: LAN-LAN (high-end).

performance defects of the system under test as possible. In the traditional test environment, the commonly used test case selection methods mainly include the manual selection method, the equivalence class division method, the boundary value method, and the causality graph method [18]. The manual selection method is mainly based on the experience and intuition of relevant testers to divide and select test cases. The equivalence class partition method divides the input set into several subsets according to the predetermined equivalence relation, and the attribute values of test cases in each subset are the same. Then, one test case can be selected from each subset. The boundary value method generates and selects test cases near the boundaries of the possible input domain space of the system under test, usually as a supplement to equivalence class partitioning. Cause-and-effect diagrams generate test cases by focusing on examining combinations of input conditions to form cause-and-effect diagrams. The comparative analysis of various selection methods of traditional test cases is shown in Table 1.

For some difficulties of the distributed system performance test, the traditional test case selection method can not solve the problems well. It can be seen from Table 1 that the manual selection method and the boundary value method cannot conduct a complete performance test on the system under test due to their incomplete selection of test cases, thus failing to comprehensively detect possible performance problems of the system under test. Due to the large scale of the distributed system and the high complexity of business logic, there are many factors that affect system performance, so it is difficult to determine the equivalence relationship between test cases in the input domain by the equivalence class partition method and the dependence relationship between conditions and results in the input domain by the causality graph method, which cannot be well applied [19]. Therefore, it is necessary to adopt new performance test methods to evaluate the performance of distributed systems effectively and discover the potential performance problems of the distributed system.

3.3.1. Task Scheduling Requirements. For the cloud test platform, the scheduling of test tasks should meet the following requirements:

Minimum execution time:

For users of the cloud testing service, task execution time refers to the period during which the cloud testing system selects and executes the test task through the task scheduling algorithm after submitting the test task to the cloud testing platform and returns the test result to users after the test task is completed. From the point of view of users, their biggest wish is that after submitting the test task, the cloud test system can complete their tasks with quality and quantity as soon as possible, so that they can see the test results in the shortest time and save their time cost to the greatest extent. The shorter the execution time of the test task and the shorter the waiting time for the user, the better the task scheduling strategy will be.

Load balancing:

In the cloud computing environment, load balancing between hosts is a key indicator to measure the quality of the scheduling algorithm. Due to the heterogeneity of physical resources in the cloud environment, the scale of virtual machine resources is extremely large. Therefore, real-time monitoring and management of virtual machine instance resources in the host become extremely important. A good task scheduling algorithm should ensure that the resource utilization rate of each virtual machine in the host always fluctuates within a certain range [20].

Service quality:

The performance of the cloud test platform is reflected by the quality of service (QoS) provided to users. To select the task scheduling algorithm, it is very important to ensure the quality of service of the cloud test system. The minimum execution time and load balancing between VMS belong to the performance QoS category.

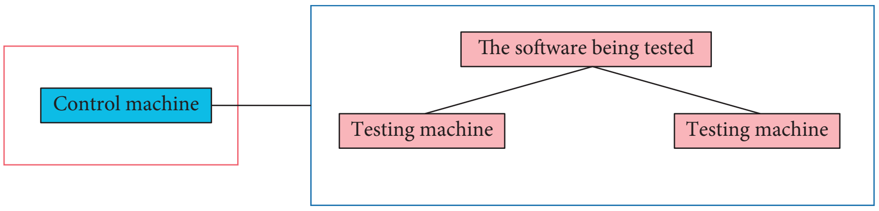

FIGURE 3: LAN-cloud environment (the software under test is deployed in the cloud environment).

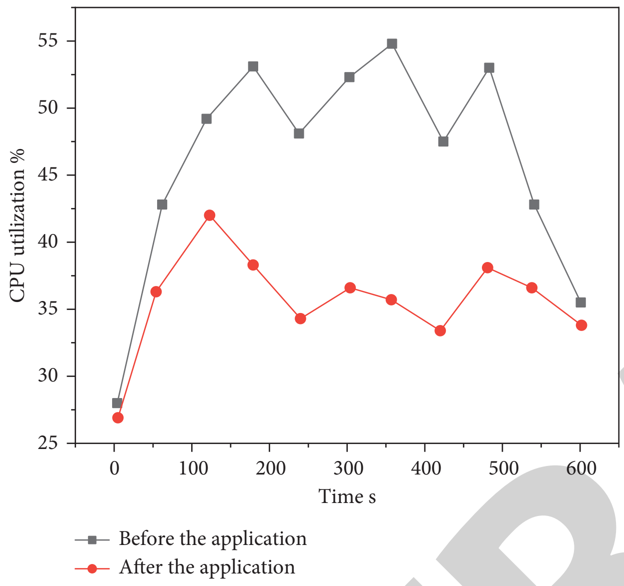

FIGURE 4: Comparison of the CPU utilization.

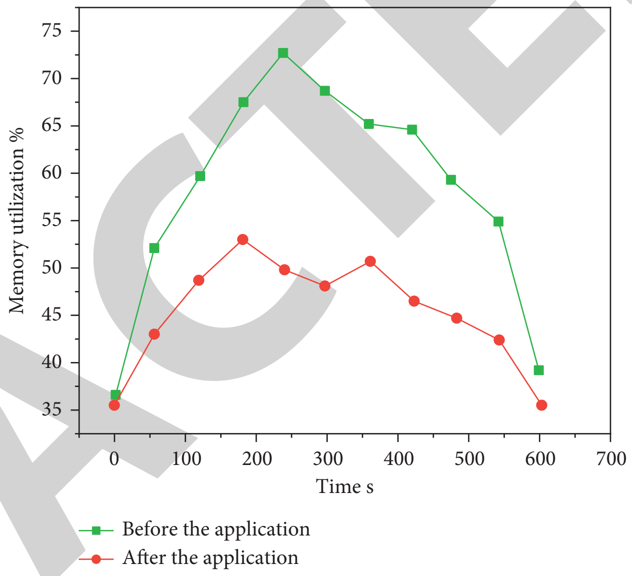

FIGURE 5: Comparison of the memory utilization.

3.3.2. The Deployment of the Test Environment. It is a new application of the cloud computing technology in the field of software testing to transfer the process of a software performance test to the cloud environment and use the massive resources provided by cloud computing such as computing, storage, and network to automatically configure and build a testing environment to carry out various performance test activities. As shown in Figures 2 and 3, there are three deployment modes for the establishment of a performance test environment at present [21].

Figure 2 shows the topology of performance test deployment on a traditional test platform. Both the software under test and the test agent are deployed on the same LAN. Figure 3 shows a deployment topology for a performance test in a cloud computing environment. In the cloud environment, cloud computing, as an effective way to quickly acquire test resources, has been involved in all stages of performance test activities [22].

In Figure 3, the software under test is deployed in the cloud, which is a performance test activity for the software deployed in the cloud. This deployment mode supports the demand-oriented software testing service market. Cloud computing service providers can provide online software testing services, namely test as a service (TaaS), to individuals or enterprises who need testing services through the network.

The research is based on the deployment mode of the performance test environment as shown in Figure 3; that is, the research of performance test-related technologies for software deployed in a cloud environment. This method can improve the efficiency of the research object in the research.

4. Results and Discussion

Through the test experiment, the CPU utilization and memory utilization of load virtual machines before and after applying the performance test method based on the decision tree model to the cloud performance test platform are compared and analyzed. Based on the system under test proposed in the above section, the scale of the initial test case design is expanded, and a total of 500 test case data are designed as the initial input field of the system under test. On the cloud test platform, the resource usage of virtual machine instances is recorded every minute. In order to ensure the correctness and reliability of the experimental results, a total of 20 experiments under the same test conditions are conducted. After averaging them, the variation trend of the corresponding CPU utilization and memory utilization is shown in Figures 4 and 5.

The results show that the application of the performance test method based on the decision tree model to the cloud test platform can reduce the CPU utilization and the memory utilization of virtual machine instances in the cloud test platform significantly, and the CPU utilization and the memory utilization can be reduced to 32% and 30%, respectively. Compared with before the application, the optimization is about 30%, and the utilization efficiency of corresponding resources is improved.

5. Conclusions

In the research, a method is proposed to build a distributed system performance test platform based on cloud computing. The specific content of this method mainly includes the distributed system performance test method based on the decision tree, the common performance index test method, and the resource scheduling method in a cloud environment. The CPU utilization before and after the application is as high as 55% and as low as 32%, which reduces the number of performance tests and efficiency in software testing performance effectively. Furthermore, combined with the cloud computing and the automated performance testing technology, the prototype system of the cloud performance test platform is designed and implemented, which solves the problems of the difficulty of the performance test environment construction under the traditional testing platform, lack of the testing resources, and the reusability, difference, exclusivity, and effectiveness of resources.

Data Availability

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

Conflicts of Interest

The author declares that there are no conflicts of interest.

Acknowledgments

This work was supported by the Guangdong Provincial Educational Science Planning Project (Higher Education Special Project): Exploration and Practice of Full-time Teaching Mode and Resource Construction Based on VR + Cloud Platform in Post-epidemic Period (No: 2021GXJK617).

References

[1] V. K. Et al, "Least afflicted load balancing algorithm (lalba) for performance improvement of multi-scale applications in cloud environment," *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, vol. 12, no. 2, pp. 1709–1720, 2021.

[2] S. Rani, P. Agarwal, M. Jain, and R. Solanki, "A software reliability growth model considering testing coverage subject to field environment," *International Journal of Mathematics in Operational Research*, vol. 18, no. 2, p. 145, 2021.

[3] H. Guo, J. Zhang, Y. Zhao, H. Zhang, and J. Zhao, "Delay Aware Rsa Algorithm Based on Scheduling of Differentiated Services with Dynamic Virtual Topology Construction," no. 8, pp. 44559–44575, IEEE Access, 2020.

[4] S. Dehghan-Dehnavi, M. Fotuhi-Firuzabad, M. Moeini-Aghtaie, P. Dehghanian, and F. Wang, "Decision-making tree analysis for industrial load classification in demand response programs," *IEEE Transactions on Industry Applications*, vol. 37, no. 1, pp. 26–35, 2020.

[5] S. Pinkham and S. Sansiribhan, "A new hybrid algorithm of bisection and modified Newton's method for the nth root-finding of a real number," *Journal of Physics: Conference Series*, vol. 1593, no. 1, Article ID 012020, 2020.

[6] N. Berezina and A. Buzhilova, "Rare cases of rare diseases: re-examining early 20th century cases of anencephaly from the collection of the moscow state university, Russia," *International Journal of Paleopathology*, vol. 34, no. 6, pp. 12–19, 2021.

[7] J. Xue, Z. Xiang, and G. Ou, "Predicting single freestanding transmission tower time history response during complex wind input through a convolutional neural network based surrogate model," *Engineering Structures*, vol. 233, no. 3, Article ID 111859, 2021.

[8] J. Feng, S. Huang, and C. Chen, "Modeling user interaction with app-based reward system: a graphical model approach integrated with max-margin learning," *Transportation Research Part C: Emerging Technologies*, vol. 120, Article ID 102814, 2020.

[9] S. Vassilyev, A. Galyaev, E. Yakushenko, and V. Zaletin, "Capabilities for monitoring and controlling dynamic parameters of extended structures based on distributed system of integrating sensors," *Journal of Physics: Conference Series*, vol. 1864, no. 1, Article ID 012146, 2021.

[10] M. L. Chiang and W. L. Su, "Thread-aware mechanism to enhance inter-node load balancing for multithreaded applications on numa systems," *Applied Sciences*, vol. 11, no. 14, p. 6486, 2021.

[11] G. Li, F. Liu, A. Sharma et al., "Research on the natural language recognition method based on cluster analysis using neural network," *Mathematical Problems in Engineering*, vol. 2021, pp. 1–13, 2021.

[12] D. Selva, B. Nagaraj, D. Pelusi, R. Arunkumar, and A. Nair, "Intelligent network intrusion prevention feature collection and classification algorithms," *Algorithms*, vol. 14, no. 8, p. 224, 2021.

[13] J. Chen, J. Liu, X. Liu, X. Xu, and F. Zhong, "Decomposition of toluene with a combined plasma photolysis (cpp) reactor: influence of uv irradiation and byproduct analysis," *Plasma Chemistry and Plasma Processing*, vol. 41, no. 1, pp. 409–420, 2020.

[14] R. Huang, S. Zhang, W. Zhang, and X. Yang, "Progress of zinc oxide-based nanocomposites in the textile industry," *IET Collaborative Intelligent Manufacturing*, vol. 3, no. 3, pp. 281–289, 2021.

[15] M. K. A. Kaabar, V. Kalvandi, N. Eghbali, M. E. Samei, Z. Siri, and F. Martínez, "A generalized ML-hyers-ulam stability of quadratic fractional integral equation," *Nonlinear Engineering*, vol. 10, no. 1, pp. 414–427, 2021.

[16] S. Bin and W. Yongjie, "Computer network security under the cloud computing technology environment," *Journal of Physics: Conference Series*, vol. 1982, no. 1, Article ID 012204, 2021.

[17] Y. Li, Q. Cheng, and X. Li, "Analysis and improvement of a key exchange and authentication protocol in client-server environment," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 9, pp. 3787–3799, 2020.

[18] G. Yuan, Z. Xu, B. Yang et al., "Fault tolerant placement of stateful vnfs and dynamic fault recovery in cloud networks," *Computer Networks*, vol. 166, pp. 106953–106953.18, 2020.

[19] S. Choi and H. T. Lee, "Attack and improvement of the recent identity-based encryption with authorized equivalence test in cluster computing," *Cluster Computing*, vol. 25, no. 1, pp. 633–646, 2021.

[20] Y. Zhang, Y. Yao, W. Du, and H. Zhou, "Experimental study on improvement design of loess curing in engineering environment," *Bulletin of Engineering Geology and the Environment*, vol. 80, no. 4, pp. 3151–3162, 2021.

[21] P. Ajay, B. Nagaraj, R. A. Kumar, R. Huang, and P. Ananthi, "Unsupervised hyperspectral microscopic image segmentation using deep embedded clustering algorithm," *Scanning*, vol. 2022, p. 1, Article ID 1200860, 2022.

[22] G. Veselov, A. Tselykh, A. Sharma, and R. Huang, "Special issue on applications of artificial intelligence in evolution of smart cities and societies," *Informatica*, vol. 45, no. 5, p. 603, 2022.