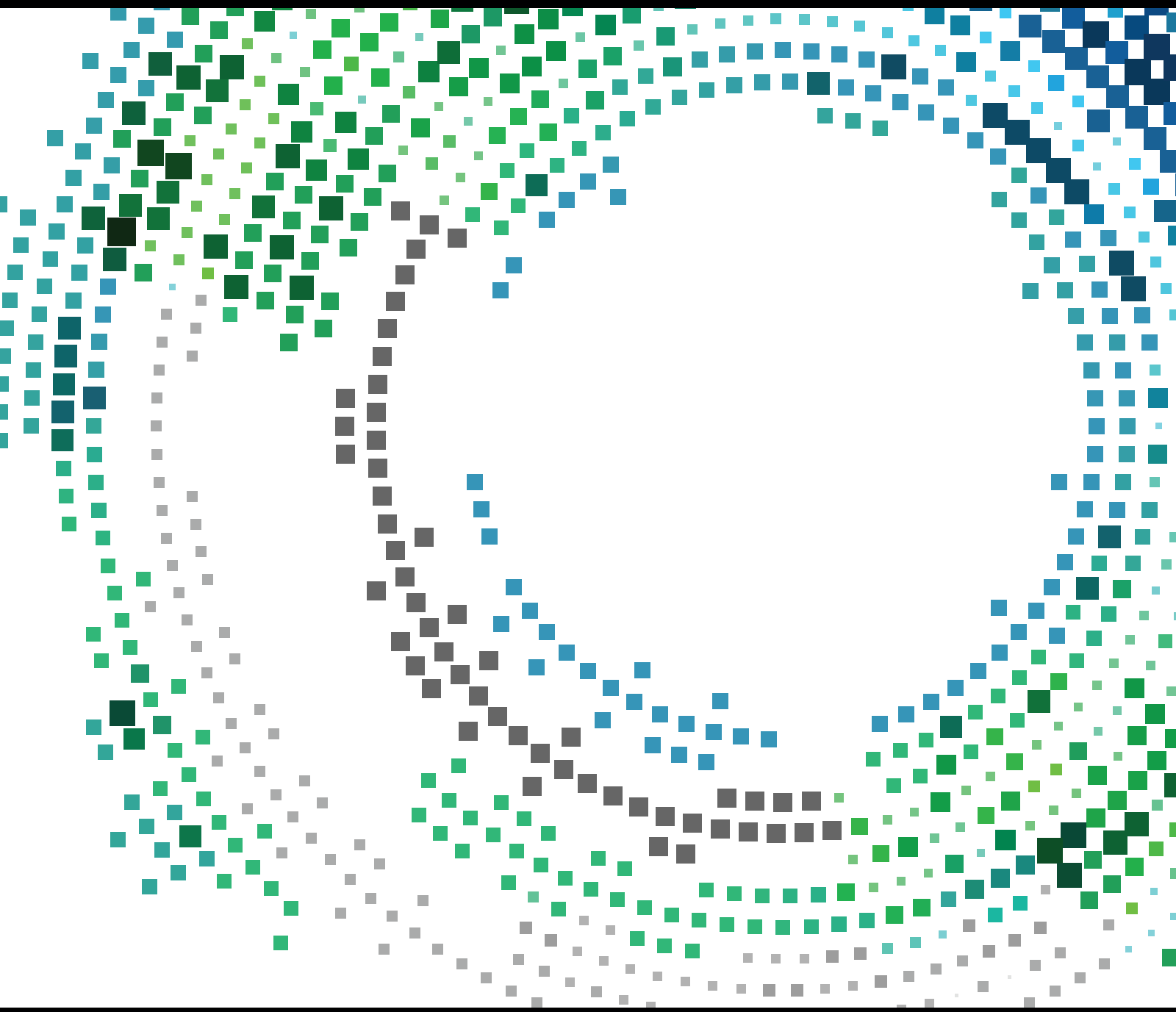
Mobile Information Systems

Advanced Technologies for Mobile IoT and Cyber-Physical Systems

Guest Editors: Kyungtae Kang, Kyung-Joon Park, Qixin Wang, Sibin Mohan, and Wenyao Xu



Advanced Technologies for Mobile IoT and Cyber-Physical Systems

Mobile Information Systems

Advanced Technologies for Mobile IoT and Cyber-Physical Systems

Guest Editors: Kyungtae Kang, Kyung-Joon Park, Qixin Wang, Sibin Mohan, and Wenyao Xu

Contents

Hindawi Publishing Corporation
Mobile Information Systems
Volume 2016, Article ID 7968707, 3 pages
http://dx.doi.org/10.1155/2016/7968707

Editorial

Advanced Technologies for Mobile IoT and Cyber-Physical Systems

Kyungtae Kang,[1] Kyung-Joon Park,[2] Qixin Wang,[3] and Wenyao Xu[4]

[1]*Department of Computer Science & Engineering, Hanyang University, Ansan 15588, Republic of Korea*
[2]*Department of Information and Communication Engineering, DGIST, Daegu 42988, Republic of Korea*
[3]*Department of Computing, The Hong Kong Polytechnic University, Hong Kong*
[4]*Department of Computer Science & Engineering, State University of New York, Buffalo, NY 14260, USA*

Correspondence should be addressed to Kyungtae Kang; ktkang@hanyang.ac.kr

Received 18 August 2016; Accepted 18 August 2016

Recently, cyber-physical systems (CPS) have emerged as a promising research paradigm. This development is the convergence of control, communication, and computation. As computing and communication capabilities have become both faster and cheaper, we can expect to find them embedded in diverse objects and structures in the physical environment. This provides the basis for applications that link the "cyberworld" of computing and communications with the physical world, thereby promising great potential for significant social and economic impact. Unlike traditional embedded systems that primarily focus on computing elements, CPS also takes into account the physical, real-world components that interact with computing elements in cyber space. Examples of CPS encompass a broad range of complex man-made systems such as avionics, vehicles, transportation, healthcare, smart grid systems, and other fields.

In the US, the President's Council of Advisors on Science and Technology has recommended CPS as a top priority for federal research investments. As a result, the CPS program was initiated at the National Science Foundation (NSF) with a funding level of around 30 million USD per year in 2009. In addition, during the past few years, the NSF has sponsored Workshops on CPS and related fields. Most recently, in 2010, the ACM and the IEEE have jointly launched the First International Conference on Cyber-Physical Systems (ICCPS), which was very successful with an impressive number of submitted papers.

However, so far, most of these initial research efforts on CPS have been made by the real-time and embedded systems community. For example, ICCPS was cosponsored by the ACM Special Interest Group on Embedded Systems (SIGBED) and the IEEE Technical Committee on Real-Time Systems (TCRTS). As mentioned above, since the salient feature of CPS is tight integration between cyber and physical components, it is obvious that networking (or mobile computing) plays a key role in the coordination between the cyber and physical elements of CPS. In fact, CPS applications are necessarily promoted by the "Internet of Things" (IoT) architectures and protocols. This combination of IoT and CPS facilitates the collection, management, and processing of large datasets, as well as the support of complex processes to manage and control physical systems at different scales.

This special issue aims to identify emerging research topics in CPS, especially from the mobile communication and networking perspective, and to define the future of networking research in CPS. In particular, by properly addressing complex interactions between the cyber and physical elements of CPS, we focus on challenging issues in IoT research that need to be resolved for deriving eventual working solutions for CPS. This facilitates timely dissemination of state-of-the-art research on CPS to the networking and mobile computing research communities.

Mobile IoT/CPS is concerned with the provision of IoT-enabled CPS for mobile objects and devices. Mobile internet devices, such as the iPhone and Android phones, with their increasing processing power, range of sensors, and pervasive cellular connections, already provide ubiquitous platforms for building robust, reliable, and secure mobile IoT/CPS applications. Therefore, the objective of this special issue is also to contribute to the direction of research on

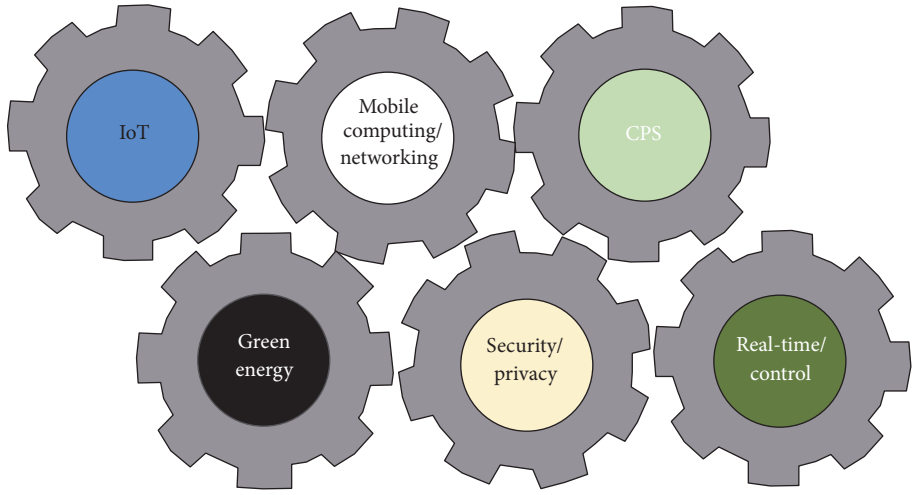


Figure 1: Major areas of research focus in this special issue.

mobile IoT/CPS by addressing issues critical to mobility, from advances in the underlying science to the challenges of development and implementation. The primary areas of research on which this special issue focuses are depicted in Figure 1.

We received 25 papers, of which 6 papers were accepted after a rigorous review process. Among these, the first paper entitled "A Dynamic Programming Solution for Energy-Optimal Video Playback on Mobile Devices" by M. Song and J. Park proposes an algorithm that determines the CPU frequency needed to decode each frame in a video. The aim is to minimize power consumption while meeting buffer size and deadline constraints using a dynamic programming technique. The proposed scheme finds the sequence of frequencies by taking into account the frame decoding time, decoding deadlines, and the measured active and idle power consumption of a system. Experimental results based on measurements made on a physical device show that it saves an appreciable amount of energy, which provides useful guidelines for a low-power video service by providing the minimum bound on the power consumption required for video playback.

Next, the paper entitled "A Remote Medical Monitoring System for Heart Failure Prognosis" by L. Zhang et al. presents the design and implementation of a remote medical monitoring system for heart failure (HF) prediction. The system realized early prediction (or prognosis) of future HF occurrence by estimating future NT-proBNP level based on a patient's historical data (body weight and blood pressure) where the data were obtained remotely using noninvasive devices (i.e., a Bluetooth-based weight scale and sphygmomanometer). A Heart Failure Risk Score (HFRS) was proposed to evaluate the risk of future occurrence of HF based on the prediction results. The HFRS scoring was designed so that it could be easily understood and perceived by patients. This system optimizes early-stage delivery of multiple suggestions/interventions to patients at different risk levels. This end-to-end system can also be used to manage patients and their data by doctors or by the system's data center. To validate this system, a set of real-life data from 34 patients was collected in a pilot clinical trial. The proposed HF prediction algorithms achieved an overall accuracy of 79.4% and 67.6% when using data collected over a 30-day and 7-day period, respectively. Therefore, from a clinical perspective, this system is promising in reducing the morbidity and mortality caused by HF and therefore improves clinical outcomes.

The third paper entitled "Dynamic Vehicular Route Guidance Using Traffic Prediction Information" by K. Kim et al. proposes a dynamic vehicular routing algorithm with traffic prediction for improved routing performance. The primary idea of the proposed algorithm is to use both real-time and predictive traffic information provided by a central routing controller. In order to evaluate the performance, the authors develop a microtraffic simulator that provides road networks created from real maps, routing algorithms, and vehicles that travel from origins to destinations depending on traffic conditions. The performance is evaluated by a newly defined metric that reveals the travel time distributions more accurately than the commonly used metric of mean travel time. Simulation results show that the proposed dynamic routing algorithm with prediction outperforms both the static and dynamic types without prediction routing algorithms under various traffic conditions and road configurations. Traffic scenarios are included where not all vehicles comply with the proposed dynamic routing with a prediction strategy. Even under these conditions, the results suggest that more than half the benefit of the new routing algorithm is realized even when only 30% of vehicles comply.

The fourth paper entitled "Energy-Efficient Real-Time Human Activity Recognition on Smart Mobile Devices" by J. Lee and J. Kim presents a novel approach for the human activity recognition (HAR) process that dynamically controls the activity recognition duration for energy-efficient HAR. Conventional HARs using the built-in accelerator in smart mobile devices are known to be the most energy-efficient in

that field, but they still incur high power consumption due to the sensor operation itself as well as the accompanying CPU computation overhead. To further enhance the energy efficiency, the proposed approach first classifies a user's activities as static and dynamic and controls the classification duration and sleep time for the HAR process accordingly based on two factors, the acceleration-sampling frequency and the window size. The experimental results showed that the proposed approach reduced energy consumption by a minimum of about 44.23% and a maximum of about 78.85% compared to conventional HAR (such as the support vector machine) without sacrificing accuracy. Moreover, this paper reports on how the acceleration-sampling frequency, window size, and feature vector dimensionality alter the battery power consumption behavior with HAR.

The fifth paper entitled "Control-Scheduling Codesign Exploiting Trade-Off between Task Periods and Deadlines" by H.-J. Cha et al. proposes a novel task set synthesis algorithm that exploits the trade-off relation between a control task's period and the deadline for maximizing the overall system control performance. This paper first shows how the control task period and the deadline affect the control performance; that is, the shorter period and shorter deadline both enhance the control task's control performance. From this observation, the authors conduct a measurement study using a Lane Keeping Assist System control application, which gives the control performance profile of the task for each possible period/deadline pair. Next, assuming multiple similar tasks with their control profiles, the paper defines a period and deadline selection problem that optimizes the overall control performance, that is, the sum of the control performance of the control tasks in the system. For this optimization problem, the authors propose a heuristic algorithm that finds a high-quality suboptimal solution with very low complexity, which makes the proposed solution practically applicable to large task sets.

The last paper entitled "Efficient Attribute-Based Secure Data Sharing with Hidden Policies and Traceability in Mobile Health Networks" by C. Hahn et al. presents a novel technique to efficiently and securely share data in mobile healthcare systems. The proposed solution enables the data owner to encrypt data and to attach an attribute-based policy of interest to the cipher text. The proposed solution hides a description of the policy so that not only the data but also the policy is available only to authorized users. The proposed solution enables the size of any cipher text to remain constant, irrespective of the number of attributes. The proposed solution embeds a user-specific identifier into each decryption key so that the identity of a user can be traced in case of malicious activities, for example, illegal key sharing.

Acknowledgments

We sincerely thank the editorial board for their approval of this concept and continuous help in successful publication of this special issue. We would also like to thank the contributors to this special issue for their innovative work. We extend our thanks to the reviewers for critical assessment of each paper, their constructive criticism, and timely responses that made this special issue possible. This work was supported by the research fund of Hanyang University (HY-2014-P).

Kyungtae Kang
Kyung-Joon Park
Qixin Wang
Wenyao Xu

Hindawi Publishing Corporation
Mobile Information Systems
Volume 2016, Article ID 6545873, 13 pages
http://dx.doi.org/10.1155/2016/6545873

Research Article

Efficient Attribute-Based Secure Data Sharing with Hidden Policies and Traceability in Mobile Health Networks

Changhee Hahn, Hyunsoo Kwon, and Junbeom Hur

Department of Computer Science and Engineering, Korea University, Seoul 136-701, Republic of Korea

Correspondence should be addressed to Junbeom Hur; jbhur@korea.ac.kr

Received 26 November 2015; Accepted 12 June 2016

Academic Editor: Wenyao Xu

Mobile health (also written as mHealth) provisions the practice of public health supported by mobile devices. mHealth systems let patients and healthcare providers collect and share sensitive information, such as electronic and personal health records (EHRs) at any time, allowing more rapid convergence to optimal treatment. Key to achieving this is securely sharing data by providing enhanced access control and reliability. Typically, such sharing follows policies that depend on patient and physician preferences defined by a set of attributes. In mHealth systems, not only the data but also the policies for sharing it may be sensitive since they directly contain sensitive information which can reveal the underlying data protected by the policy. Also, since the policies usually incur linearly increasing communication costs, mHealth is inapplicable to resource-constrained environments. Lastly, access privileges may be publicly known to users, so a malicious user could illegally share his access privileges without the risk of being traced. In this paper, we propose an efficient attribute-based secure data sharing scheme in mHealth. The proposed scheme guarantees a hidden policy, constant-sized ciphertexts, and traces, with security analyses. The computation cost to the user is reduced by delegating approximately 50% of the decryption operations to the more powerful storage systems.

1. Introduction

mHealth is an abbreviation for mobile health, which can encompass a wide range of healthcare technologies such as mobile computing, medical sensors, and communication technologies [1]. Rapid growth in wireless communications, availability and miniaturization of mobile devices, and computing resources in parallel with mobile and wearable systems can boost the wide adoption of mHealth. Such developments can greatly impact on and reshape the processes of existing healthcare services. For instance, semiconductor-implanted smart intelligent sensors will allow drugs to be delivered in real time to a personal server when they sense a patient who needs a dose of drugs. Personal servers, such as mobile devices, supply global connectivity to the storage center, which can thereby serve clinical healthcare from a distance [2]. The storage center holds the information that forms the electronic health record (EHR), a digital version of a patient's paper chart. Physicians intermittently upload diagnostic reports based on their observations of the EHRs stored in the storage center. Figure 1 shows an example of an mHealth monitoring and data transfer system. Reportedly, a growing number of healthcare-specific mobile applications are available, and it has been estimated that about 500 million patients around the globe will be in the reach of such apps as of 2015 [3].

EHRs contain sensitive information such as patients' medical history, diagnoses, immunization dates, allergies, and medications, which are bound to the real identities of patients. That is, whoever can freely access the storage center is able to learn both the identity and clinical information of a specific patient, which clearly threatens the patient's privacy. Thus, privacy concerns are arguably a major issue, and related requirements are enacted nationwide. For example, in the United States compliance to HIPPA (Health Information Technology for Economic and Clinical Health Act) encourages healthcare providers to not only adopt EHRs but also keep them confidential [4]. This clearly indicates that EHRs must be kept under strict conditions and be accessible only by the authorized user.

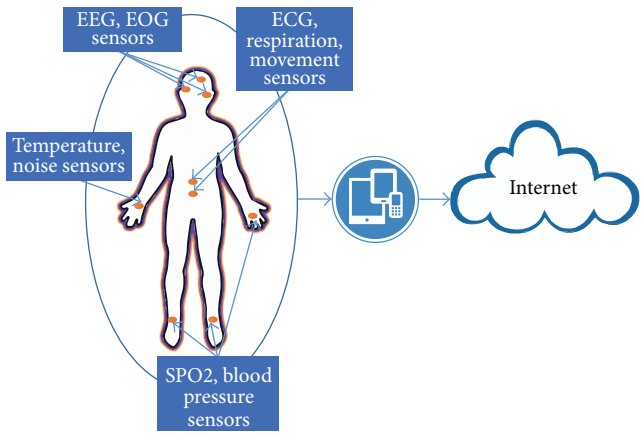


Figure 1: Typical system architecture of mHealth monitoring systems.

Unfortunately, standard encryption schemes are not suitable for mHealth systems for the following reasons [5].

(i) *Absence of Proper Access Control*. Well-known encryption schemes, such as AES, guarantee the confidentiality of data if security parameters are well-chosen. However, such schemes are not designed to support fine-grained access control.

(ii) *Expensive Key Management*. Public key encryption schemes do not support one-to-many relationships between the ciphertext and decryption key, necessitating the burdensome distribution and management of public keys.

Since healthcare delivery is a decentralized process taking place across many institutional boundaries, standard approaches to securing health records include role-based access control because the flexible assignment of permissions to a wide range of user is possible only with fine-grained access control. At the same time, the confidentiality of EHRs must be maintained without hindering clinical care by denying legitimate access requests of authorized users, such as doctors, nurses, lab technicians, researchers, and receptionists [6, 7]. Thus, a variety of policy-based encryption schemes have been proposed to share data securely and provide reliable access control [8–11]. These schemes are promising in that the accessibility of shared data is dependent on the user's capacity to satisfy a given policy. Furthermore, encryptors do not require a priori knowledge of the recipients, such as identities or certificates. Specifically, ciphertext policy attribute-based encryption (CP-ABE) allows the construction of policies by utilizing attributes as public keys, thereby protecting shared data against unauthorized users [12–16]. As access to EHRs varies across the space of uneven distributions of healthcare providers and consumers and among population groups with different socioeconomic and demographic characteristics [17], CP-ABE is a convincing alternative to the conventional cryptographic primitive for mHealth. CP-ABE can provide fine-grained and flexible access control to the shared data in mHealth systems.

It is notable that not only the data, but also the policies for sharing that data are sensitive. Typically, the access policies may reveal sensitive information, such as the underlying data, the identity of a patient, or symptoms indicating what diseases a patient is suffering from. To some extent, patients are reluctant to expose such private information, preferring instead to keep their privacy intact through securing both the EHRs and their access policies. Although CP-ABE provides a desirable access policy, it has one drawback: the access policies attached to ciphertexts are public. From these access policies, unauthorized users can learn information about the underlying data itself. This weakness is known as the policy privacy problem.

To overcome the policy privacy problem, several CP-ABE schemes with hidden access policies were proposed [9, 18]. In these schemes, the encryptor-chosen access policies are associated with each ciphertext in a way hidden such that even an authorized user learns no information about the underlying policy other than that he is authorized to decrypt. Although these schemes feature hidden policies, they suffer from being inefficient; that is, the ciphertext size is linear with respect to the number of attributes in the access policy.

To limit ciphertext size, Zhou et al. introduced a CP-ABE scheme which provides both a hidden access policy and a constant-sized ciphertext [19]. However, their scheme lacks user traceability. In general, most CP-ABE schemes supporting constant-sized ciphertext or hidden access policies cannot trace malicious users who illegally share their decryption keys. Specifically, the secret keys of policy-based encryption consist of sharable attributes so that the decryption keys have no uniquely identifiable information. Thus, if a malicious user leaks his decryption key to others, then there is no clear evidence indicating that the key belongs to him. Although Li et al. proposed a CP-ABE scheme featuring a hidden access policy and traceability [20], it lacks constant-sized ciphertext, resulting in increased communication and storage costs.

Contribution. In this paper, we propose an efficient attribute-based secure data sharing scheme for mHealth with hidden policies and traceability. The proposed scheme enforces hidden access policies with wildcards and supports constant-sized ciphertext, regardless of the number of attributes. Also, we embed a uniquely identifiable point into each decryption key in order to prevent the user from intentionally distributing the decryption key to others, thereby achieving traceability. Additionally, the proposed scheme allows users to outsource part of the decryption process to the more powerful storage center to minimize computation cost at the user side. Our performance results show that the storage center computes almost 50% of the decryption process on behalf of users. To the best of our knowledge, this is the first construction that achieves all these functionalities simultaneously.

Organization. The rest of this paper is organized as follows. We begin with a discussion of related work in Section 2. In Section 3, we describe the cryptographic background and define a general CP-ABE with a hidden policy, constant-sized ciphertext, and traceability. Section 4 describes the mHealth

architecture and security model. In Section 5, we present the construction of the proposed scheme in detail, followed by a performance analysis in Section 6. We analyze its security in Section 7 and conclude the paper in Section 8.

2. Related Work

The idea of Identity-Based Encryption (IBE) was first introduced by Shamir [21]. In IBE, the encryptor makes an access policy based on an identity, and only a user with the matching identity obtains the decryption privilege. Encryption by identity, however, leads to the following limitations: lack of one-to-many relationship between the ciphertext and decryption key and the need for the encryptor to know each user's identity in advance. Later, Sahai and Waters introduced Fuzzy Identity-Based Encryption, which is the first prototype of attribute-based encryption (ABE) [22]. While the IBE scheme views an identity as a string of characters, in ABE, an identity is viewed as a set of descriptive attributes (a.k.a., identity set) such as name and affiliation. The ABE scheme allows the encryption of a message based on some identity set ω', and the decryption ability is given if and only if a user's set ω is close enough to ω' to satisfy a system-defined threshold. This property enables fine-grained access control and a one-to-many relationship between a ciphertext and its receivers since anyone whose identity set satisfies a given threshold can obtain the decryption privilege. However, the threshold semantics are not very expressive and cannot support fine-grained access control. This drawback means that the threshold-based ABE scheme cannot be applied to more general systems.

In CP-ABE [12–16], a ciphertext is associated with an access policy and decryption keys are labeled with an arbitrary number of attributes. The encryptor specifies an access policy over encryptor-chosen attributes. The access right is given if and only if the attributes in the decryption key satisfy the access policy in the ciphertext. In these schemes, however, the size of a ciphertext has a linear relationship with the number of attributes in the access policies, resulting in inapplicability for resource-constrained environments.

To limit the size of ciphertexts, Zhou and Huang proposed constant-sized CP-ABE (C-CP-ABE) with a logical *AND* access policy with wildcards [23]. This scheme limits the size of each ciphertext to up to 300 bytes in total, where a ciphertext consists of encrypted data, an access policy, and 2 bilinear group elements. Chen et al. further improved the C-CP-ABE scheme in terms of security [24] making it CPA-secure under a well-established assumption in the standard model without loss of efficiency. Overall, these schemes successfully make the size of ciphertexts constant. However, they reveal the underlying access policy publicly.

While previous works feature open access policies, Hur introduced a CP-ABE scheme with hidden access policy in smart grid [9]. To preserve policy privacy, a one-way anonymous key agreement scheme is used as a building block in order to replace identity hashes with user-generated pseudonyms. However, this scheme does not support constant-sized ciphertext. Interestingly, an efficient CP-ABE scheme with a hidden policy was proposed [19]. In this scheme, AND-gate access policies with wildcards are used and each ciphertext header requires 2 bilinear group elements, each of which is limited to 100 bytes in total. Also, access policies are obfuscated by computing the intersection between a given access policy and an all-wildcard attribute set. This technique, however, partially leaks the access policy, because unauthorized users can guess at a minimum which attributes are treated as *do not care*. In addition, the user must run the decryption algorithm at least once, to determine whether he satisfies the access policy, since only decryption failure notifies whether the decryption key satisfies the underlying access policy.

The ability to resist illegal key sharing is a highly desirable characteristic for ABE. To achieve this, Li et al. introduced a user-accountable CP-ABE scheme that binds user identity in the private key, thereby allowing illegally-shared keys to be traced [25]. Although this methodology has also been adopted by other traceable CP-ABE schemes [26, 27], none of them fully support either constant-sized ciphertext or hidden access policies. In addition to supporting these features, in this paper, we also insert a unique identifier into each private key such that any key can be traced in constant time, regardless of the number of attributes.

3. Preliminaries

3.1. Bilinear Map. Let $\mathbb{G}_0$ be a multiplicative cyclic group of large prime order p. The bilinear map e is defined as follows: $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_1$, where $\mathbb{G}_1$ is the codomain of e. The bilinear map e has the following properties:

(i) *Bilinearty.* $e(P^a, Q^b) = e(P, Q)^{ab}$, where $\forall P, Q \in \mathbb{G}_0$, $\forall a, b \in \mathbb{Z}_p^*$.

(ii) *Symmetry.* One has $\forall P, Q \in \mathbb{G}_0$, $e(P, Q) = e(Q, P)$.

(iii) *Nondegeneracy.* $e(g, g) \neq 1$, where g is the generator of $\mathbb{G}_0$.

(iv) *Computability.* There exists an efficient algorithm to compute the bilinear map e.

3.2. Security Assumption. The security of the proposed scheme is based on the Bilinear Diffie-Hellman Exponent assumption (BDHE) [28]. Let $\mathbb{G}_0$ be a bilinear group of large prime order p and let g be a generator of $\mathbb{G}_0$. The K-BDHE problem in $\mathbb{G}_0$ is defined as follows. Given the vector of $2K+1$ elements

$$\left(h, g, g^{\alpha}, g^{\alpha^2}, \ldots, g^{\alpha^K}, g^{\alpha^{K+2}}, \ldots, g^{\alpha^{2K}}\right) \in \mathbb{G}_0^{2K+1} \quad (1)$$

as the input where $g^{\alpha^{K+1}}$ is not in the vector, the goal of the computational K-BDHE problem is to compute $e(g, h)^{\alpha^{K+1}}$. Define the set $Y_{g,\alpha,K}$ as

$$Y_{g,\alpha,K} = \left\{g^{\alpha}, g^{\alpha^2}, \ldots, g^{\alpha^K}, g^{\alpha^{K+2}}, \ldots, g^{\alpha^{2K}}\right\}. \quad (2)$$

Then, we have the following definition.

Definition 1 (Decisional K-BDHE). The decisional K-BDHE assumption is said to be hold in $\mathbb{G}_0$ if there is no probabilistic polynomial time adversary who is able to distinguish

$$\left\langle h, g, Y_{g,\alpha,K}, e(g,h)^{\alpha^{(K+1)}} \right\rangle, \quad \left\langle h, g, Y_{g,\alpha,K}, e(g,h)^{R} \right\rangle \tag{3}$$

with nonnegligible advantage, where $\alpha, R \in \mathbb{Z}_p$ and $g, h \in \mathbb{G}_0$ are chosen independently and uniformly at random.

We exploit Boneh et al.'s l-Strong Diffie-Hellman assumption (l-SDH) to prove traceability [29]. Given a $(l+1)$-tuple $(g, g^x, g^{x^2}, \ldots, g^{x^l})$ as input where $x \in \mathbb{Z}_p^*$ is chosen uniformly at random, the l-SDH assumption is stated as follows: there is no probabilistic polynomial time adversary who is able to output $(c, g^{1/(x+c)}) \in \mathbb{Z}_p^* \times \mathbb{G}_0$ with nonnegligible probability, where c is not allowed to be zero.

Formally, we have the following l-SDH assumption.

Assumption 2 (l-SDH). The l-Strong Diffie-Hellman problem in $\mathbb{G}_0$ is defined as follows: given a $(l+1)$-tuple $(g, g^x, g^{x^2}, \ldots, g^{x^l})$ as input, output $(c, g^{1/(x+c)}) \in \mathbb{Z}_p^* \times \mathbb{G}_0$. An algorithm $\mathcal{A}$ has advantage ϵ in solving l-SDH in $\mathbb{G}_0$ if the following holds:

$$\Pr\left[\mathcal{A}\left(g, g^x, g^{x^2}, \ldots, g^{x^l}\right) = \left(c, g^{1/(x+c)}\right)\right] \geq \epsilon, \tag{4}$$

where the probability is over the random choice of x in $\mathbb{Z}_p^*$.

Definition 3. The l-SDH assumption is (t, ϵ)-secure if no t-time algorithm has advantage at least ϵ in solving the l-SDH problem in $\mathbb{G}_0$.

3.3. Access Policy. Given an attribute universe $U = \{A_1, A_2, \ldots, A_k\}$, each A_i has one of three values $\{A_i^+, A_i^-, A_i^*\}$, where A_i^+ denotes that the user has A_i, A_i^- denotes that the user does not have A_i or A_i is not a proper attribute of this user, and A_i^* denotes a wildcard specifying *do not care*. We define the user's attribute set as follows.

Definition 4. Let $L = \{A_1^{+,-}, A_2^{+,-}, \ldots, A_k^{+,-}\}$ be a user's attribute set, where $A_i^{+,-} \in \{A_i^+, A_i^-\}$ and k is the order of the attribute universe. Then, $L = L^+ \cup L^-$, where $L^+ = \{A_i^+ \mid \forall i \in \{1, \ldots, k\}\}$ and $L^- = \{A_i^- \mid \forall i \in \{1, \ldots, k\}\}$. One has $L^+ \cap L^- = \emptyset$.

Next we define the *AND*-gate access policy as follows.

Definition 5. Let $W = \{A_1, \ldots, A_k\}$ be an *AND*-gate access policy where $A_i \in \{A_i^+, A_i^-, A_i^*\}$. Denote $L \vDash W$ that the user's attribute set L satisfies W. Then,

$$L \vDash W \Longleftrightarrow W \subset L \cup \{A_1^*, \ldots, A_k^*\}. \tag{5}$$

3.4. One-Way Anonymous Key Agreement. In this paper, the key idea used to obfuscate attributes in the policy starts from Boneh-Franklin Identity-Based Encryption [30]. In their scheme, a private key generator (PKG) takes the role of issuing private keys. It generates a private key $d_i = H(\mathrm{ID}_i)^s \in \mathbb{G}_0$ for each user ID_i using a master secret s, where $H : \{0,1\}^* \to \mathbb{G}_0$ is a cryptographic hash function.

Based on [30], Kate et al. proposed a one-way anonymous key agreement scheme by replacing $H(\mathrm{ID}_i)$ with a pseudonym chosen by each user [31]. This scheme guarantees anonymity for just one receiver when two users engage in it. We give a specific example as follows. Suppose Alice and Bob hold identity ID_A and identity ID_B, respectively, and they are clients of the same key authority which holds a master secret s. Given the private key $d_A = Q_A^s = H(\mathrm{ID}_A)^s$, Alice wants to communicate with Bob, without disclosing her identity.

To achieve this, the key agreement protocol runs as follows:

(1) Alice computes $Q_B = H(\mathrm{ID}_B)$, chooses a random $r_A \in \mathbb{Z}_p^*$, sets a pseudonym $P_A = Q_A^{r_A}$, and computes the session key $K_{A,B} = e(d_A, Q_B)^{r_A} = e(Q_A, Q_B)^{sr_A}$. She sends the pseudonym P_A to Bob.

(2) Given his private key d_B, Bob computes the session key $K_{A,B} = e(P_A, d_B) = e(Q_A, Q_B)^{sr_A}$.

In this noninteractive manner, the session key is implicitly authenticated such that Alice is assured that the no one can derive the key other than Bob. Based on the BDH assumption, this protocol is proved to be secure in the random oracle model satisfying unconditional anonymity, no impersonation, and session key secrecy. To hide the policy we exploit the technique used in [9] as a building block instead of building a new method for policy obfuscation from scratch.

3.5. Definitions. In this section, we define a general CP-ABE with hidden policy, constant-sized ciphertexts, and traceability capabilities for secure data sharing. The scheme consists of the following seven algorithms:

(i) $Setup(k) \to (MK, PK)$. The Setup algorithm takes as input the number of attributes k. It outputs a public key PK and a master key MK and initializes an identity table $T = \emptyset$.

(ii) $KeyGen(MK, PK, L, id) \to (SK)$. The key generation algorithm takes as input the master key MK, the public key PK, and the user's attribute set L with identity id. It outputs a decryption key SK and inserts id into T.

(iii) $Encrypt(PK, W, M) \to (CT)$. The encryption algorithm takes as input the public key PK, an access policy W, and a message M. It outputs a ciphertext CT such that only the users whose decryption keys satisfying W should be able to extract M. CT is associated with the obfuscated policy W.

(iv) $GenToken(SK_u, \Lambda) \to (TK_{\Lambda,u})$. The token generation algorithm takes as input the user u's secret key SK_u and a set of attributes $\Lambda \vDash W$. It outputs a token $\mathrm{TK}_{\Lambda,u}$.

(v) $PDecrypt(TK_{\Lambda,u}, CT) \to (CT')$. The partial decryption algorithm takes as input the token and outputs a partially decrypted ciphertext CT' for a user u.

(vi) *Decrypt* $(PK, SK, CT', CT) \rightarrow M$ *or* $\perp$. The decryption algorithm takes as input the public key PK, a decryption key SK, and ciphertexts CT′, CT. If $L \vDash W$, then it outputs a message M, where L is the user's attribute set and W is the access policy. Otherwise, it outputs $\perp$ which indicates the failure of decryption.

(vii) *Trace* $(PK, SK, T) \rightarrow id$ *or* $\top$. The tracing algorithm takes as input the public key PK, a decryption key SK, and the table T. It determines whether SK is *well-formed* indicating that SK is the real output of KeyGen. If SK is well-formed, the algorithm outputs an identity id which corresponds to SK. Otherwise it outputs $\top$ implying that SK is not well-formed. The *well-formed* decryption key is guaranteed to work correctly in the well-formed decryption process.

In the proposed scheme, each public key component is mapped to an attribute value A_i. When encrypting data, the encryptor specifies an access policy W, where $A_i \in \{+, -, *\}$. The decryption succeeds only when the user's attribute set L satisfies the (obfuscated) policy W.

4. mHealth Architecture

4.1. System Model. In mHealth systems, intelligent wireless sensors perform data acquisition and processing [32]. Individual sensors monitor certain physiological signals and communicate with each other and the personal server such as a tablet PC as shown in Figure 1. Then, the personal server integrates the data received from the different sensors and plays the role of a gateway by sending data to the upper layer of the mHealth system. From a security point of view, the mHealth system components are categorized as follows:

(1) *Trust Authority*. This is a key entity that issues the public and secret parameters for the mHealth system. It publishes diverse access privileges to individual entities based on their attributes. The trust authority is assumed to be fully trusted in the mHealth system [10].

(2) *Storage Center*. This is a data repository center that stores EHRs. In mHealth systems, hospitals or clinics with certain qualifications certified by the trust authority can be employed as a storage center. It is assumed to be honest-but-curious [10]. Thus, it will honestly execute the assigned tasks and like to learn as much information from the encrypted data as possible.

(3) *Encryptor*. This is a patient who generates data and sends it to the storage center. It uses mobile devices to interact with the storage center. Encryptors are responsible for defining access policy based on attributes, obfuscating the policy, associating it with the data, and encrypting the data according to the policy. Hereafter, we will use "encryptor" and "patient" interchangeably.

(4) *User*. This includes entities such as the patient, physicians, nurses, lab technicians, researchers, or receptionists who want to access EHRs contained in the storage center. A user will be authorized to decrypt a ciphertext given by the storage center if and only if his key satisfies the access policy of that ciphertext.

4.2. Security Model

CPA Security. The security model of the proposed scheme is similar to that of the CP-ABE scheme with constant-sized ciphertexts [23] except that each key query is labeled with an explicit identity and attributes are obfuscated. We first introduce the semantic security game. A CP-ABE scheme is considered to be CPA-secure if no probabilistic polynomial time adversaries have nonnegligible advantages in the following CPA security game.

(i) *Init*. The adversary chooses a challenge access policy W and gives it to the challenger.

(ii) *Setup*. The challenger runs the Setup algorithm and gives the adversary the public parameter PK.

(iii) *Phase 1*. The adversary queries the challenger for decryption keys corresponding to (id, L), where $L \nvDash W$. The challenger answers with a decryption key SK for L. The adversary repeats this phase adaptively.

(iv) *Challenge*. The challenger obtains $\{\langle C_0, C_1\rangle, \mathrm{Key}\}$ by running the Encrypt algorithm. The challenger sets $\mathrm{Key}_0 = \mathrm{Key}$ and picks a random Key_1 of the same length as Key_0. It then flips a random coin $\beta \in \{0, 1\}$ and gives $\{\langle C_0, C_1\rangle, \mathrm{Key}_\beta\}$ to the adversary.

(v) *Phase 2*. It is the same as Phase 1.

(vi) *Guess*. The adversary outputs a guess $\beta' \in \{0, 1\}$.

The adversary wins the game if $\beta' = \beta$ under the restriction that L cannot satisfy the access policy W. The adversary may run Phase 2 to make multiple key queries in the midst of the challenge. Note that the adversary declares the access policy at the start of the game.

The advantage of an adversary in this game is defined as

$$\left|\Pr\left[\beta' = \beta\right] - \frac{1}{2}\right|. \tag{6}$$

Traceability. The traceability definition for the proposed scheme is described by the following security game:

(i) *Setup*. The challenger runs the Setup algorithm to obtain the public parameter PK. Then, the challenger gives PK to the adversary.

(ii) *KeyQuery*. The adversary makes decryption key queries q-times to the challenger, where sets of attributes $(\mathrm{id}_1, L_1), \ldots, (\mathrm{id}_q, L_q)$ correspond to decryption keys.

(iii) *KeyForgery*. The adversary outputs a decryption key SK_*.

The adversary wins the game if the following holds:

(1) $\mathrm{Trace}(\mathrm{PK}, \mathrm{SK}_*, T) \neq \perp$.

(2) $\mathrm{Trace}(\mathrm{PK}, \mathrm{SK}_*, T) \notin \{\mathrm{id}_1, \ldots, \mathrm{id}_q\}$.

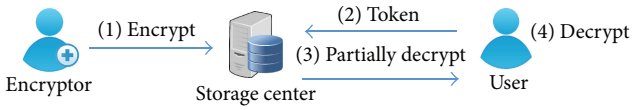


FIGURE 2: Overview of the proposed data sharing process.

Then, the advantage of the adversary in this game is

$$\Pr\left[\text{Trace}\left(\text{PK}, \text{SK}_*, T\right) \notin \{\perp\} \cup \left\{\text{id}_1, \ldots, \text{id}_q\right\}\right]. \quad (7)$$

Definition 6. A traceable ciphertext policy attribute-based encryption scheme is fully traceable if all polynomial time adversaries have at most negligible advantage in this game.

Policy Privacy. While sharing data in the mHealth system, the storage center or unauthorized users must learn no information about the attributes associated with the access policy of the encrypted data. Also, even authorized users should not obtain any information about these attributes other than the fact that they are authorized to access the data.

5. Proposed Scheme

5.1. System Architecture. The proposed data sharing process in the mHealth system runs as follows. An encryptor defines the access policy with a set of attributes, encrypts the EHRs associated with clinical reports under the policy, and uploads the ciphertext and the obfuscated policy to the storage center. When a user wants to access the uploaded data, he first generates a token using his attributes and sends it to the storage center. If the attributes in the token satisfy the access policy, then the storage center partially decrypts the ciphertext and sends the result to the user. Then, the user finishes the decryption of the ciphertext using his secret key and the partially decrypted ciphertext as inputs. The outline of data sharing process is depicted in Figure 2.

5.2. Scheme Construction. The proposed scheme is constructed on the basis of the following seven algorithms as follows.

Setup $(k) \rightarrow (MK, PK)$. Given k attributes $\{A_1, A_2, \ldots, A_k\}$ as the attribute universe, the proposed scheme has $K = 3k$ attribute values such that $A_i \in \{A_i^+, A_i^-, A_i^*\}$. Specifically, we map $\{A_1^+, A_2^+, \ldots, A_k^+\}$ to $\{1, 2, \ldots, k\}$, $\{A_1^-, A_2^-, \ldots, A_k^-\}$ to $\{k+1, k+2, \ldots, 2k\}$, and $\{A_1^*, A_2^*, \ldots, A_k^*\}$ to $\{2k+1, 2k+2, \ldots, 3k\}$.

Let $\mathbb{G}_0$ be a bilinear group of prime order p. The Setup algorithm chooses a random generator $g \in \mathbb{G}_0$ and random $\alpha, \beta, \gamma \in \mathbb{Z}_p$. For $i = 1, 2, \ldots, K, K+2, \ldots, 2K$, it computes $g_i = g^{(\alpha^i)}$. Then, it computes $v = g^\gamma$ and $h = g^\beta$. The master and public keys are set to $\text{MK} = (\alpha, \beta, \gamma)$; $\text{PK} = (g, g_1, \ldots, g_K, g_{K+2}, \ldots, g_{2K}, v, h) \in \mathbb{G}_0^{2K+1}$. The algorithm initializes an identity table $T = \emptyset$.

KeyGen $(MK, PK, L_{u_t}, id_{u_t}) \rightarrow (SK_{u_t})$. Assume that each user u_t is tagged with an attribute set $L_{u_t} = L_{u_t}^+ \cup L_{u_t}^-$, where $L_{u_t}^+ \subset \{1, 2, \ldots, k\}$ and $L_{u_t}^- \subset \{k+1, k+2, \ldots, 2k\}$. The KeyGen algorithm randomly chooses $a, c \in \mathbb{Z}_p^*$, $\{r_1, r_2, \ldots, r_k\} \in \mathbb{Z}_p$. Then, it computes $r = \sum_{i=1}^k r_i$, $D'' = g^r$ and $D = g^{r\gamma/(a+c)}$. For all $j \in L_{u_t}$, it computes $D''' = H(j)^\beta$, where H is a hash function $H : \{0,1\}^* \rightarrow \mathbb{G}_0$.

Next, the algorithm computes the following:

(i) For every $i \in L_{u_t}^+$, compute $D_i = g^{\gamma(c\alpha^i + r_{i'})/(a+c))}$, where $i' = i$.

(ii) For every $i \in L_{u_t}^-$, compute $D_i = g^{\gamma(c\alpha^i + r_{i'})/(a+c))}$, where $i' = i - k$.

(iii) For every $i \in L_{u_t}^*$, compute $D_i = g^{\gamma(c\alpha^i + r_{i'})/(a+c))}$, where $i' = i - 2k$.

The decryption key for user u_t is set to

$$\text{SK}_{u_t} = \left(D = g^{r\gamma/(a+c)}, \left\{D_i \mid i \in \left\{L_{u_t}^+, L_{u_t}^-, L_{u_t}^*\right\}\right\}, D' = c, D'' = g^r, D''' = H(j)^\beta, D_a = g^a\right). \quad (8)$$

Note that $1/(a+c)$ is computed modulo p. If $\gcd(a+c, p) \neq 1$ or c is already in T, the algorithm is run repeatedly with another random $c \in \mathbb{Z}_p^*$. Then, it puts a tuple (c, id_{u_t}) into T and uploads $(\text{id}_{u_t}, \{g_i^{D'} \mid \forall i \in L_{u_t}\})$ to the storage center.

Encrypt $(PK, W, M) \rightarrow (CT)$. W is an *AND*-gate access policy with k attributes specified by an encryptor u_b, where each attribute is either positive/negative or wildcard. The algorithm chooses a random $b \in \mathbb{Z}_p^*$ and computes $s_j = e(h^b, H(j))$, $H_1(s_j)$ for all $j \in W$, where H_1 is a hash function $H : \mathbb{G}_1 \rightarrow \{0,1\}^{\log p}$. Then, the access policy W is obfuscated by replacing each attribute with $H_1(s_j)$.

Next, the algorithm picks a random $t \in \mathbb{Z}_p$ and computes a one-time symmetric key $\text{Key} = e(g_K, g_1)^{kt}$. It encrypts the message M as $\{M\}_{\text{Key}}$ and computes g^t. Then, it computes $(v\prod_{j \in W} g_{K+1-j})^t$. The ciphertext CT is set to

$$\text{CT} = \left(W, \{M\}_{\text{Key}}, C_0 = g^t, C_1 = \left(v\prod_{j \in W} g_{K+1-j}\right)^t, \text{id}_{u_t}, g^b\right). \quad (9)$$

The encryptor uploads CT to the storage center.

GenToken $(SK_{u_t}, \Lambda) \rightarrow (TK_{\Lambda, u_t})$. When a user u_t needs to access the ciphertext of u_b in the storage center with a set of attributes $\Lambda \vDash W$, u_t receives g^b from the storage center and generates the token for Λ as follows. For all $j \in \Lambda$, the algorithm computes $s_j = e(g^b, D_j''') = e(g^b, H(j)^\beta)$. Then, it constructs the token $\text{TK}_{\Lambda, u_t} = \{I_j \mid \forall j \in \Lambda,\ I_j = H_1(s_j)\}$. Each I_j will be used as an index for the obfuscated attribute j. The user u_t sends TK_{Λ, u_t} to the storage center.

PDecrypt $(TK_{\Lambda, u_t}, CT) \rightarrow (CT')$. Given TK_{Λ, u_t} from the user u_t, the storage center checks if each I_j in the token satisfies

Table 1: Comparison of different schemes.

	Enc.	Dec.	Ciphertext length	Assumption
Constant-sized ciphertexts [23]	2ex	$2tp$ + ex	$2\lvert\mathbb{G}_0\rvert + \lvert\mathbb{G}_1\rvert$	n-DBDH
Hidden policy [25]	$(t+2)$ex	$(2t+1)p + t$ex	$(t+1)\lvert\mathbb{G}_0\rvert + \lvert\mathbb{G}_1\rvert$	DBDH
Traceability [26]	$(2t+3)$ex	$(2t+1)p + t$ex	$2(t+1)\lvert\mathbb{G}_0\rvert + \lvert\mathbb{G}_1\rvert$	l-BDHI
Proposed	2ex + tp	$(2t+1)(p$ + ex$)$	$3\lvert\mathbb{G}_0\rvert + \lvert\mathbb{G}_1\rvert$	n-BDHE

the access policy associated with CT. If satisfied, the storage center partially decrypts CT using $(\mathrm{id}_{u_t}, \{g_i^{D'} \mid \forall i \in L_{u_t}\})$ as

$$\begin{aligned} A_i &= e\left(g_i^{D'}, C_1\right) = e\left(g, v\prod_{j\in W} g_{K+1-j}\right)^{\alpha^i t D'} \\ &= e\left(g, g^{\gamma+\sum_{j\in W}\alpha^{K+1-j}}\right)^{\alpha^i t D'} \\ &= e\left(g,g\right)^{\alpha^i t D'\gamma + t D'\sum_{j\in W}\alpha^{K+1-j+i}} \end{aligned} \tag{10}$$

for all $i \in W$. Then, it computes a production of all A_i as $\mathrm{CT}' = \prod_{i\in W} A_i$. The storage center sends CT' to u_t.

Decrypt $(PK, SK_{u_t}, CT', CT) \rightarrow M$ *or* $\perp$. On receipt of the partially decrypted ciphertext CT' from the storage center, the user u_t computes B_i for all $i \in W$ as

$$\begin{aligned} B_i &= e\left(C_0, \left(\prod_{j\in W, j\neq i} g_{K+1-j+i}\right)^{D'} \cdot D_i\right) \\ &= e\left(g,g\right)^{tD'\sum_{j\in W, j\neq i}\alpha^{K+1-j+i} + t\gamma(D'\alpha^i + r_i/(a+c))}. \end{aligned} \tag{11}$$

Then, it computes $B = \prod_{i\in W} B_i$ and divides CT' by B. Using the quotient term CT'/B, the user concludes decryption as follows:

$$\begin{aligned} \frac{\mathrm{CT}'}{B}\cdot e\left(D, C_0\right) &= \frac{\mathrm{CT}'}{B}\cdot e\left(g^{\gamma r/(a+c)}, g^t\right) \\ &= e\left(g,g\right)^{D' kt\alpha^{K+1}}. \end{aligned} \tag{12}$$

Then,

$$\begin{aligned} \left(\frac{\mathrm{CT}'}{B}\cdot e\left(D, C_0\right)\right)^{1/D'} &= e\left(g,g\right)^{kt\alpha^{K+1}} \\ = e\left(g^{\alpha^K}, g^{\alpha}\right)^{kt} &= e\left(g_K, g_1\right)^{kt} = \mathrm{Key}. \end{aligned} \tag{13}$$

The user decrypts $\{M\}_{\mathrm{Key}}$.

Trace $(PK, SK_u, T) \rightarrow id_u$ *or* $\top$. SK_u is called well-formed if it passes the following conditions hold:

$$\begin{aligned} &D' \in \mathbb{Z}_p^*, \\ &D, D_i, D'' \in \mathbb{G}_0^{2K+1}, \\ &e\left(D_a \cdot g^{D'}, D\right) = e\left(v, D''\right) \neq 1. \end{aligned} \tag{14}$$

If SK_u is well-formed, the algorithm searches D' in T. If D' is in T, the algorithm outputs the corresponding id_u, and if not, the algorithm outputs the corresponding $\mathrm{id}_\emptyset$ indicating that the corresponding identity never appears in T. If SK_u is not well-formed, the algorithm outputs $\top$.

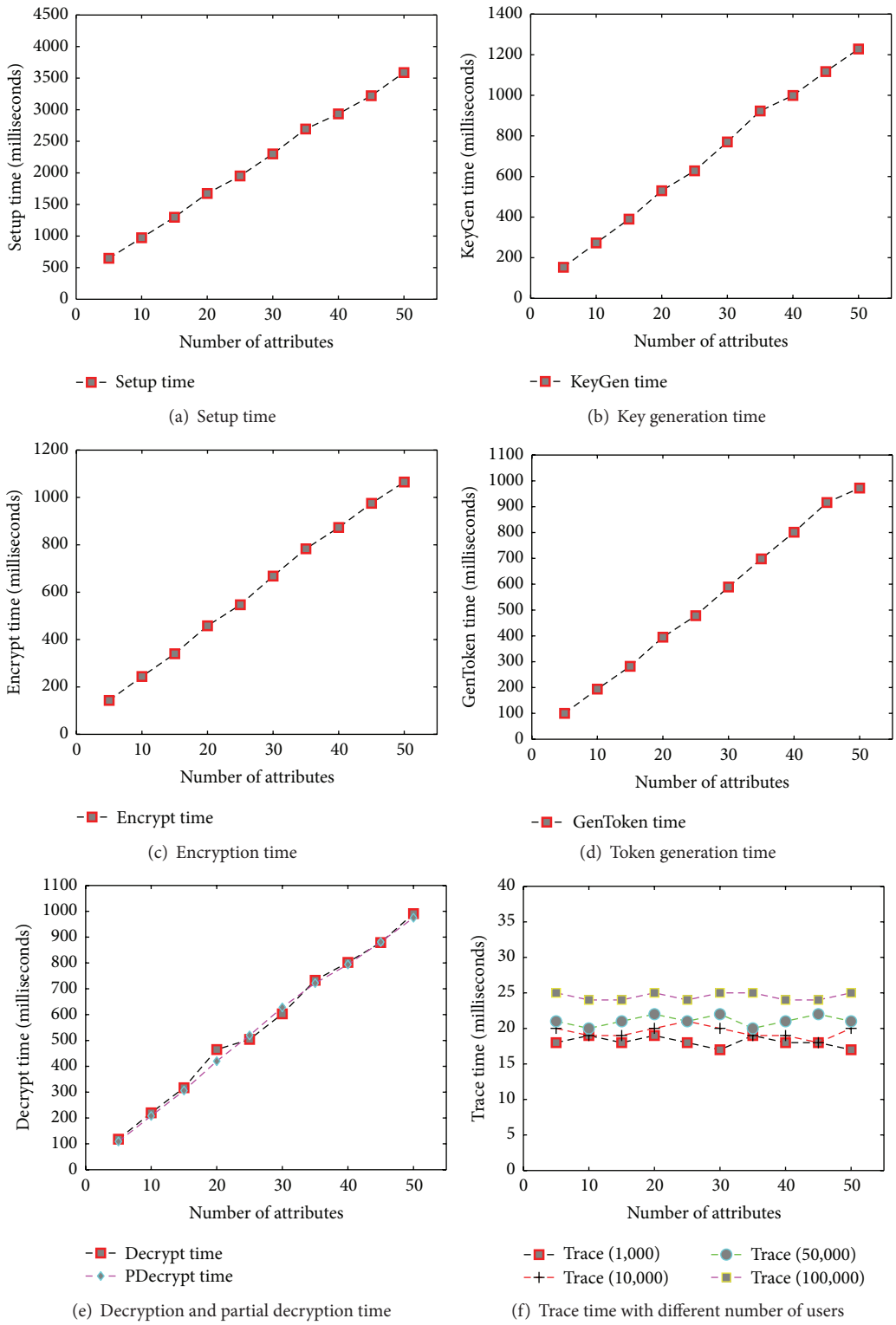
6. Performance Analysis

In this section, we analyze the performance of the proposed scheme compared with the previous schemes including a constant-sized ciphertexts scheme [23], a hidden policy scheme [25], and a traceability scheme [26]. We compare each scheme in several ways such as the computational cost of encryption and decryption and the ciphertext length and in terms of the complexity assumption. Also, we implemented the proposed scheme to evaluate its actual performance. We programmed our system using the Java-based pairing based cryptography (jPBC) library [33] on a GIGABYTE desktop with 4 Intel Core i5-3570 3.40 GHz CPUs, 4 GB RAM, and running Windows 7 Ultimate K.

Table 1 shows the results of comparing the different schemes. The notations we use in the table are as follows: t denotes the number of attributes involved in the access policy, n denotes the number of attributes in the attribute universe, ex denotes the exponentiation operation, and denotes p the paring operation. Note that, following convention, the bit-length of the expression of the access policy and its computational costs over $\mathbb{Z}_p$ are ignored.

In terms of computational cost, the constant-sized ciphertext scheme [23] shows the best encryption phase efficiency, requiring a constant number of exponentiations. The proposed scheme also needs two exponentiations in data encryption, but an additional tp operations are required to obfuscate the access policy. In the decryption phase, the proposed scheme requires more computations than [23] since the user identity is exponentiated to every attribute value to support traceability. In contrast to [25, 26], the proposed scheme requires approximately t number of exponentiations. With regard to the ciphertext length, the proposed scheme and [23] guarantee constant-sized ciphertext. On the other hand, the hidden policy scheme [25] and the traceability scheme [26] incur linearly increasing ciphertexts as the attribute number t increases. Overall, the proposed scheme is efficient in terms of the ciphertext size and provides hidden policy traceability at the cost of more exponentiation operations.

Figure 3 shows the computation overhead incurred in the core algorithms, Setup, KeyGen, Encrypt, GenToken, Decrypt, PDecrypt, and Trace, under various conditions. Figure 3(a) shows how system-wide setup time varies according



(a) Setup time (b) Key generation time

(c) Encryption time (d) Token generation time

(e) Decryption and partial decryption time (f) Trace time with different number of users

Figure 3: Time costs of different algorithms.

to the number of attributes. Figure 3(b) shows the total key generation time against different numbers of attributes. The setup occurs only once at the start of the system, and key generation occurs every time a new user joins. Figure 3(c) shows encryption time against different numbers of attributes. It increases linearly due to the time taken to obfuscate the policy attached to the data. Figure 3(d) shows the token generation time against the number of attributes. The token

Table 2: jPBC and PBC benchmark comparison results [33].

Operation	jPBC	PBC
Pairing	14.654	2.688
Exponentiation in $\mathbb{G}_1$	18.592	4.122
Exponentiation in $\mathbb{G}_T$	2.112	0.529
Exponentiation in $\mathbb{Z}_r$	0.068	0.087

generation process requires a pairing operation time linear to the number of attributes. Figure 3(e) shows the partial decryption time at the storage center and decryption time at the user against the number of attributes. Interestingly, the storage center can undertake nearly 50% of whole decryption process on behalf of users. This property can be most useful for relatively resource-constrained user side devices. Lastly, Figure 3(f) shows the trace time with different numbers of attributes and users. The trace time depends only but not strongly on the number of users.

Further Efficiency Improvement. jPBC is a complete Java port of the PBC library which was originally written in C [34]. Java is widely considered to be slower than C because Java programs run on the Java Virtual Machine rather than directly on the computer's processor. Based on this, we additionally provide benchmark comparison results between jPBC and PBC in order to demonstrate how fast the proposed scheme can be when it is implemented in C language [33]. Table 2 shows the performance comparison between Java and C with respect to pairing and exponentiation operations conducted on the same machine. The two libraries were applied to the curve $y^2 = x^3 + x$ over the field $\mathbb{F}_q$ for some prime $q = 3 \bmod 4$. The order of $\mathbb{F}_q$ is some prime factor of $q + 1$ [33]. Since the cost of the pairing operation in PBC is approximately 12 seconds less than in jPBC, PBC is expected to improve the performance of pairing-dependent algorithms, such as GenToken and policy obfuscation process in Encrypt, by up to 81%. Similarly, the cost of the exponentiation operations in $\mathbb{G}_1$ and $\mathbb{G}_T$ are reduced by 14.47 and 1.583 seconds, respectively. Such a difference between the two libraries implies that moving from Java to C implementation of the proposed scheme can speed up the Setup and KeyGen algorithms by approximately 77.8% and the PDecrypt and Decrypt algorithms by approximately 74.9%.

7. Security Analysis

7.1. Data Confidentiality. In this section, we reduce the chosen plaintext attack (CPA) security of the proposed scheme to a decisional K-BDHE problem. Given an access policy W, a user with an attribute set $L \not\models W$ colludes with $x \leq k$ decryption proxies. Intuitively, this attack works successfully if $L \cup \{i_1, \ldots, i_x\} \models W$. Based on the CPA security game in Section 4.2, we have the following.

Theorem 7. *If a probabilistic polynomial time adversary wins the CPA security game with a nonnegligible advantage, then one can construct a simulator that distinguishes a K-DBHE tuple with a nonnegligible advantage.*

Proof. Suppose that an adversary $\mathcal{A}$'s advantage for winning the game is ϵ. Then, we can construct a simulator $\mathfrak{B}$ which solves the decisional K-BDHE problem with the advantage $\epsilon/2$. The simulator $\mathfrak{B}$ takes an input vector $(h, g, Y_{g,\alpha,K}, Z)$, where Z is either $e(g,h)^{\alpha^{K+1}}$ or a random element in $\mathbb{G}_0$. Then, $\mathfrak{B}$ breaks the decisional K-BDHE problem with the advantage $\epsilon/2$. Specifically, $\mathfrak{B}$ takes a random decisional K-BDHE challenge $\langle h, g, Y_{g,\alpha,K}, Z\rangle$ as input, where Z is either $Z = e(g,h)^{\alpha^{K+1}}$ or a random value.

Next, $\mathfrak{B}$ runs the following CPA game with the role of challenger.

(i) *Init*. $\mathcal{A}$ sends an access policy W to $\mathfrak{B}$.

(ii) *Setup*. $\mathfrak{B}$ runs the Setup algorithm to obtain PK and chooses a random $d \in \mathbb{Z}_p$. Then, $\mathfrak{B}$ computes

$$v = g^d \left(\prod_{j \in W} g_{K+1-j} \right)^{-1} = g^{\gamma}. \tag{15}$$

$\mathfrak{B}$ outputs the public key $\mathrm{PK} = (g, Y_{g,\alpha,K}, v) \in \mathbb{G}_0^{2K+1}$.

Phase 1. The adversary $\mathcal{A}$ submits L, where $L \not\models W$. Then, there exists $j \in L$ such that $j + k \in W$, where $j \in \{1, \ldots, k\}$, or $j - k \in W$, where $j \in \{k+1, \ldots, 2k\}$.

For $i = 1, \ldots, k$, $\mathfrak{B}$ picks k random $r_i \in \mathbb{Z}_p$ and sets $r = r_1 + \cdots + r_k$. Next, $\mathfrak{B}$ randomly chooses $a, c \in \mathbb{Z}_p^*$ and computes

$$D = \left(g^d \prod_{j \in W} \left(g_{K+1-j} \right)^{-1} \right)^{r/(a+c)} = g^{\gamma r/(a+c)}. \tag{16}$$

Next, $\mathfrak{B}$ computes

$$D_i = g_i^d \prod_{j \in W} \left(g_{K+1-j+i} \right)^{-c} \cdot \prod_{j \in W} \left(g_{K+1-j} \right)^{-r_{i'}/(a+c)}, \tag{17}$$

where i falls into one of the following conditions: (1) $i + k \in W$ for all $i \in L^+$, (2) $i - k \in W$ for all $i \in L^-$, and (3) $i \notin W$ for all $i \in L^*$.

Then, each D_i is valid such that

$$D_i = \left(g^d \left(\prod_{j \in W} g_{K+1-j} \right)^{-1} \right)^{c\alpha^i + r_{i'}/(a+c)} = g^{\gamma(c\alpha^i + r_{i'}/(a+c))}. \tag{18}$$

Challenge. $\mathfrak{B}$ sets $C_0 = h$ and $C_1 = h^d$ and gives the challenge $\langle C_0, C_1, Z^k\rangle$ to $\mathcal{A}$. Note that $C_0 = h = g^t$ for some t such that

$$h^d = \left(g^d\right)^t = \left(g^d \prod_{j \in W} \left(g_{K+1-j} \right)^{-1} \cdot \prod_{j \in W} \left(g_{K+1-j} \right) \right)^t = \left(v \prod_{j \in W} \left(g_{K+1-j} \right) \right)^t, \tag{19}$$

and $Z^k = \mathrm{Key}$ if $Z = e(g,h)^{\alpha^{(K+1)}}$.

Phase 2. Repeat Phase 1.

Guess. The adversary $\mathcal{A}$ outputs a guess b', where $b' = 0$ implies that $Z = e(g,h)^{\alpha^{K+1}}$. If $b' = 1$, then Z is a random element which indicates that $\Pr[\mathfrak{B}(h, g, Y_{g,\alpha,K}, Z) = 0] = 1/2$. Note that each decryption proxy $p_i(r)$ simulates a legal decryption key component with a random r. Specifically, the adversary $\mathcal{A}$ passes r as a guess of $r_{i'}$ which is embedded in D_i, where $i \in W$. We further define a decryption proxy to model collusion attacks. □

Definition 8. Given $2k$ decryption proxies in the security game, each decryption proxy $p_i(r) = g^{\gamma(c\alpha^i + r/(a+c))}$, where $r \in \mathbb{Z}_p$ and $i \in \{1, \ldots, 2k\}$.

Lemma 9 (collision with 1 decryption proxy). *Suppose that $\mathcal{A}$ has issued q queries and there is only 1 attribute $i \notin W$, where $\mathcal{A}$ makes l queries to $p_i(r)$. The probability that none of the queries returns a legal decryption key component of any q is $(1 - q/p)^l$.*

Proof. The probability that at least one query returns an illegal decryption key component of any q is $1 - q/p$. Thus, if none of the l queries succeeds, then $\Pr[r \neq r_{i'}] = (1 - q/p)^l$, where r is a random number in the decryption proxy and $r_{i'}$ is a random number in the decryption key. □

Lemma 10 (collision with multiple decryption proxies). *Suppose $\mathcal{A}$ has issued q queries and there are m attributes dissatisfying W, where $\mathcal{A}$ makes l queries to each decryption proxy $p_{i_1}(r_1)$, $p_{i_2}(r_2), \ldots, p_{i_m}(r_m)$. The probability that none of the queries returns a legal decryption key component of any q is $(1 - (1 - q/p)^l)^m$.*

Proof. The probability that one decryption proxy fails is $\Pr[r \neq r_{i'}] = (1 - q/p)^l$. Thus, the probability that all m decryption proxies succeed is $(1 - (1 - q/p)^l)^m$. □

In case of $Z = e(g,h)^{\alpha^{(K+1)}}$, we have 3 collusion scenarios as follows.

0-Collusion. If no decryption proxy is used, then $\mathcal{A}$ has at least $\epsilon/2$ advantage in breaking the proposed scheme. Thus, $\mathfrak{B}$ has at least the following advantage in breaking K-BDHE problem:

$$\left| \Pr\left[\mathfrak{B}\left(h, g, Y_{g,\alpha,K}, Z\right) = 0\right] - \frac{1}{2} \right| \geq \frac{\epsilon}{2}. \tag{20}$$

1-Collusion. If one decryption proxy $p_i(r)$ is used, then we have $\Pr[r \neq r_{i'}] = (1 - q/p)^l$. Thus, if $\mathcal{A}$ has at least ϵ advantage in breaking the proposed scheme, then $\mathfrak{B}$ has at least $(1 - q/p)^l \epsilon/2$ advantage in breaking the K-BDHE problem.

m-Collusion. If m decryption proxies $p_{i_1}(r_1), \ldots, p_{i_m}(r_m)$ are used, then we have

$$\Pr\left[r_{i_j} \neq r_{i'_j},\ \exists j \leq m\right] = \left(1 - \left(1 - \frac{q}{p}\right)^l\right)^m. \tag{21}$$

Thus, if $\mathcal{A}$ has at least ϵ advantage in breaking the proposed scheme, then $\mathfrak{B}$ has at least the following advantage in breaking the K-BDHE problem:

$$\left(1 - \left(1 - \left(1 - \frac{q}{p}\right)^l\right)^m\right) \cdot \frac{\epsilon}{2}. \tag{22}$$

7.2. Traceability. In this section, we prove the traceability of the proposed scheme based on the l-SDH assumption.

Theorem 11. *If l-SDH assumption holds, then the proposed scheme is fully traceable provided that $q < l$.*

Proof. Suppose that there is a PPT adversary $\mathcal{A}$ who wins the traceability game with nonnegligible advantage ϵ after q key queries. Without loss of generality, assume that $l = q + 1$. Then, we can construct a PPT simulator $\mathfrak{B}$ that breaks l-SDH assumption with nonnegligible advantage.

$\mathfrak{B}$ is given an instance of the l-SDH problem as follows. Let $\mathbb{G}_0$ be a bilinear group of prime order p, let $\overline{g} \in \mathbb{G}_0$, let $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_1$ be a bilinear map, and let $a \in \mathbb{Z}_p$. $\mathfrak{B}$ is given $\mathrm{IN}_{\mathrm{SDH}} = (p, \mathbb{G}_0, \mathbb{G}_1, e, \overline{g}, \overline{g}^a, \ldots, \overline{g}^{a^l})$ as in instance of the l-SDH problem. $\mathfrak{B}$'s goal is to output a pair $(c_r, w_r) \in \mathbb{Z}_p^* \times \mathbb{G}_0$ satisfying $w_r = \overline{g}^{1/(a+c_r)}$ for solving the l-SDH problem. $\mathfrak{B}$ sets $A_i = \overline{g}^{a^i}$ for $i = 0, 1, \ldots, l$ and interacts with $\mathcal{A}$ in the traceability game as follows.

Setup. $\mathfrak{B}$ randomly picks q distinct values $c_1, \ldots, c_q \in \mathbb{Z}_p^*$. Let $f(y)$ be the polynomial $f(y) = \prod_{i=1}^{q}(y + c_i)$. Expand $f(y)$ and write $f(y) = \sum_{i=0}^{q} \alpha_i y^i$, where $\alpha_0, \alpha_1, \ldots, \alpha_q \in \mathbb{Z}_p$ are the coefficients of the polynomial $f(y)$. $\mathfrak{B}$ computes

$$\begin{aligned} g &\longleftarrow \prod_{i=0}^{q} (A_i)^{\alpha_i} = \overline{g}^{f(a)}, \\ g^a &\longleftarrow \prod_{i=1}^{q+1} (A_i)^{\alpha_{i-1}} = \overline{g}^{f(a)\cdot a}. \end{aligned} \tag{23}$$

$\mathfrak{B}$ randomly chooses $\alpha, \gamma \in \mathbb{Z}_p$ and computes $v = g^\gamma$. For $i = 1, 2, \ldots, K, K+2, \ldots, 2K$, $\mathfrak{B}$ sets $g_i = g^{(\alpha^i)}$, where $K = 3k = l$. $\mathfrak{B}$ then gives $\mathcal{A}$ the public parameter

$$\mathrm{PK} = (g, g_1, \ldots, g_K, g_{K+2}, \ldots, g_{2K}, v) \in \mathbb{G}_0^{2K+1}. \tag{24}$$

KeyQuery. $\mathcal{A}$ submits (id_x, L) to $\mathfrak{B}$ to request a decryption key. Assume that it is the xth query. For $x \leq q$, let $f_x(y)$ be the polynomial $f_x(y) = f(y)/(y + c_x) = \prod_{j=1, j\neq x}^{q}(y + c_j)$. Expand $f_x(y)$ and write $f_x(y) = \sum_{j=0}^{q-1} \beta_j y^j$, where $\beta_0, \beta_1, \ldots, \beta_{q-1} \in \mathbb{Z}_p$. $\mathfrak{B}$ computes

$$\sigma_x \longleftarrow \prod_{j=0}^{q-1} \left(A_j\right)^{\beta_j} = \overline{g}^{f_x(a)} = \overline{g}^{f(a)/(a+c_x)} = g^{1/(a+c_x)}. \tag{25}$$

$\mathfrak{B}$ randomly chooses $\{r_1, r_2, \ldots, r_k\} \in \mathbb{Z}_p$. Then, it computes $r = \sum_{i=1}^{k} r_i$, $D'' = g^r$, and $D = \sigma_x^{r\gamma}$.

Finally, $\mathfrak{B}$ computes the following:

(i) For every $i \in L_u^+$, compute $D_i = g^{\gamma c_x \alpha^i} \sigma_x^{\gamma r_{i'}}$, where $i' = i$.

(ii) For every $i \in L_u^-$, compute $D_i = g^{\gamma c_x \alpha^i} \sigma_x^{\gamma r_{i'}}$, where $i' = i - k$.

(iii) For every $i \in L_u^*$, compute $D_i = g^{\gamma c_x \alpha^i} \sigma_x^{\gamma r_{i'}}$, where $i' = i - 2k$.

$\mathfrak{B}$ responds to $\mathscr{A}$ with $\mathrm{SK}_{\mathrm{id}_x, L_x}$ as

$$\mathrm{SK}_{\mathrm{id}_x, L_x} = \left(D = \sigma_x^{r\gamma}, \{D_i \mid i \in \{L_u^+, L_u^-, L_u^*\}\}, D' = c_x, D'' = g^r\right). \tag{26}$$

$\mathfrak{B}$ puts tuple (c_x, id_x) into T.

KeyForgery. $\mathscr{A}$ submits to $\mathfrak{B}$ a decryption key SK_*.

Note that the distributions of PK and SK in the above game are the same as in the real game. Let $\Upsilon_{\mathscr{A}}$ denote the event that $\mathscr{A}$ wins the game; that is, SK_* is well-formed, and $c_r \notin \{c_1, c_2, \ldots, c_q\}$. The adversary's advantage over the game is $\epsilon/2$ since there is no decryption proxy used. If $\Upsilon_{\mathscr{A}}$ does not happen, $\mathfrak{B}$ chooses a random a random pair $(c_r, w_r) \in \mathbb{Z}_p^* \times \mathbb{G}_0$ as its solution for l-SDH problem. If $\Upsilon_{\mathscr{A}}$ happens, $\mathfrak{B}$ writes the polynomial $f(y) = \gamma(y)(y + D') + \gamma_{-1}$ for some polynomial $\gamma(y) = \sum_{i=0}^{q-1} (\gamma_i y^i)$ and some $\gamma_{-1} \in \mathbb{Z}_p$. Then, $\gamma_{-1} \neq 0$ since $f(y) = \prod_{i=1}^{q} (y + c_i)$, where $c_i \in \mathbb{Z}_p^*$ and $D' \notin \{c_1, c_2, \ldots, c_q\}$. Thus $y + D'$ does not divide $f(y)$. $\mathfrak{B}$ computes the value of $\gcd(\gamma_{-1}, p)$.

Next, let $\Omega_{\mathrm{SDH}}(c_r, w_r)$ denote the event that (c_r, w_r) is a solution to the l-SDH problem. Note that when $\mathfrak{B}$ chooses (c_r, w_r) randomly, $\Omega_{\mathrm{SDH}}(c_r, w_r)$ happens with negligible probability, say zero. $\mathfrak{B}$ solves the l-SDH problem with probability

$$\begin{aligned}
&\Pr\left[\Omega_{\mathrm{SDH}}(c_r, w_r)\right] \\
&\quad = \Pr\left[\Omega_{\mathrm{SDH}}(c_r, w_r) \mid \overline{\Upsilon_{\mathscr{A}}}\right] \cdot \Pr\left[\overline{\Upsilon_{\mathscr{A}}}\right] \\
&\qquad + \Pr\left[\Omega_{\mathrm{SDH}}(c_r, w_r) \mid \Upsilon_{\mathscr{A}} \wedge \gcd(\gamma_{-1}, p) \neq 1\right] \\
&\qquad \cdot \Pr\left[\Upsilon_{\mathscr{A}} \wedge \gcd(\gamma_{-1}, p) \neq 1\right] \\
&\qquad + \Pr\left[\Omega_{\mathrm{SDH}}(c_r, w_r) \mid \Upsilon_{\mathscr{A}} \wedge \gcd(\gamma_{-1}, p) = 1\right] \\
&\qquad \cdot \Pr\left[\Upsilon_{\mathscr{A}} \wedge \gcd(\gamma_{-1}, p) = 1\right] \\
&\quad = 0 + 0 + 1 \cdot \Pr\left[\Upsilon_{\mathscr{A}} \wedge \gcd(\gamma_{-1}, p) = 1\right] \leq \epsilon.
\end{aligned} \tag{27}$$

Thus, $\mathfrak{B}$ can break the l-SDH assumption with advantage $\leq \epsilon$. □

7.3. Policy Privacy. When an encryptor uploads its ciphertext to the storage center, every attribute j in the access policy is obfuscated as $H_1(e(h^b, H(j)))$ with a random b using the one-way anonymous key agreement protocol [31] such that only users in possession of valid corresponding attributes are able to compute the same value. It is infeasible to guess j from $H_1(e(h^b, H(j)))$ without having the corresponding attributes due to b which is chosen uniformly at random by the encryptor. Specifically, the storage center does not have $D''' = H(j)^\beta$ which is a secret key component owned by users whose attribute sets satisfy the access policy. Due to the secrecy property of the key agreement protocol [31], the storage center cannot compute $e(g^b, H(j)^\beta)$.

In token generation phase, a user computes indices $I_j = H_1(e(g^b, H(j)^\beta))$ for each (obfuscated) attribute j. Due to the secrecy property of the key agreement protocol, only the authorized users are able to construct indices corresponding to j. Thus, the storage center cannot generate correct indices for the attributes in the access policy. Also, even though the storage center conducts partial decryptions, the user learns nothing about the underlying access policy except that he can decrypt the ciphertext since he receives only the partially decrypted value and no more. Therefore, the proposed scheme guarantees the policy privacy against the storage center and authorized users.

8. Conclusion

In this paper, we proposed an efficient attribute-based secure mHealth data sharing scheme with hidden policies and traceability. The proposed scheme significantly reduces storage and communication costs. The access policies are obfuscated such that not only data privacy but also policy privacy is preserved. The computational costs of users are reduced by delegating approximately 50% of the decryption operation to the more powerful storage systems. Lastly, the proposed scheme is able to trace malicious users who illegally leak their keys. Our security analysis shows that the proposed scheme is secure against chosen-ciphertext and key forgery attacks under the decisional K-BDHE and l-SDE assumptions. We also prove that the policy privacy of the proposed scheme is preserved against the storage center and authorized users.

Competing Interests

The authors declare that they have no competing interests.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (no. 2016R1A2A2A05005402). This work was also supported by Institute for Information & Communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (no. B0190-15-2028). This work was also supported by the research fund of Signal Intelligence Research Center supervised by the Defense Acquisition Program Administration and Agency for Defense Development of Korea.

References

[1] P. Germanakos, C. Mourlas, and G. Samaras, "A mobile agent approach for ubiquitous and personalized eHealth information systems," in *Proceedings of the Workshop on 'Personalization for*

e-Health' of the 10th International Conference on User Modeling, pp. 67–70, Edinburgh, UK, July 2005.

[2] E. Jovanov, A. O'Donnell Lords, D. Raskovic, P. G. Cox, R. Adhami, and F. Andrasik, "Stress monitoring using a distributed wireless intelligent sensor system," *IEEE Engineering in Medicine and Biology Magazine*, vol. 22, no. 3, pp. 49–55, 2003.

[3] J. A. Wolf, J. F. Moreau, O. Akilov et al., "Diagnostic inaccuracy of smartphone applications for melanoma detection," *JAMA Dermatology*, vol. 149, no. 4, pp. 422–426, 2013.

[4] United States Department of Health & Human Services, *Health Information Privacy*, 2011, http://www.hhs.gov/ocr/privacy/index.html.

[5] S. Alshehri, S. P. Radziszowski, and R. K. Raj, "Secure access for healthcare data in the cloud using Ciphertext-policy attribute-based encryption," in *Proceedings of the IEEE 28th International Conference on Data Engineering Workshops (ICDEW '12)*, pp. 143–146, IEEE, Arlington, Va, USA, April 2012.

[6] M. Poulymenopoulou, F. Malamateniou, and G. Vassilacopoulos, "E-EPR: a cloud-based architecture of an electronic emergency patient record," in *Proceedings of the 4th ACM International Conference on PErvasive Technologies Related to Assistive Environments (PETRA '11)*, article 35, Crete, Greece, May 2011.

[7] H. A. J. Narayanan and M. H. Gunes, "Ensuring access control in cloud provisioned healthcare systems," in *Proceedings of the IEEE Consumer Communications and Networking Conference (CCNC '11)*, pp. 247–251, Las Vegas, Nev, USA, January 2011.

[8] R. Bobba, H. Khurana, M. Alturki, and F. Ashraf, "PBES: a policy based encryption system with application to data sharing in the power grid," in *Proceedings of the 4th International Symposium on ACM Symposium on Information, Computer and Communications Security (ASIACCS '09)*, pp. 262–275, March 2009.

[9] J. Hur, "Attribute-based secure data sharing with hidden policies in smart grid," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 11, pp. 2171–2180, 2013.

[10] L. Guo, C. Zhang, J. Sun, and Y. Fang, "PAAS: a privacy-preserving attribute-based authentication system for eHealth networks," in *Proceedings of the 32nd IEEE International Conference on Distributed Computing Systems (ICDCS '12)*, pp. 224–233, IEEE, Macau, June 2012.

[11] A. Kapadia, P. P. Tsang, and S. W. Smith, "Attribute-based publishing with hidden credentials and hidden policies," *Proceedings of the 14th Annual Network and Distributed System Security Symposium (NDSS '07)*, vol. 7, pp. 179–192, 2007.

[12] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attributebased encryption," in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 321–334, Berkeley, Calif, USA, May 2007.

[13] B. Waters, "Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization," in *Public Key Cryptography-PKC*, pp. 53–70, 2011.

[14] V. Goyal, A. Jain, O. Pandey, and A. Sahai, "Bounded ciphertext policy attribute based encryption," in *Automata, Languages and Programming: 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7–11, 2008, Proceedings, Part II*, vol. 5126 of *Lecture Notes in Computer Science*, pp. 579–591, Springer, Berlin, Germany, 2008.

[15] L. Ibraimi, M. Petkovic, S. Nikova, P. Hartel, and W. Jonker, "Mediated ciphertext-policy attribute-based encryption and its application," in *Information Security Applications*, H. Y. Youm and M. Yung, Eds., vol. 5932 of *Lecture Notes in Computer Science*, pp. 309–323, 2009.

[16] T. Jung, X.-Y. Li, Z. Wan, and M. Wan, "Privacy preserving cloud data access with multi-authorities," in *Proceedings of the IEEE INFOCOM*, pp. 2625–2633, Turin, Italy, April 2013.

[17] F. Wang and W. Luo, "Assessing spatial and nonspatial factors for healthcare access: towards an integrated approach to defining health professional shortage areas," *Health & Place*, vol. 11, no. 2, pp. 131–146, 2005.

[18] R. W. Bradshaw, J. E. Holt, and K. E. Seamons, "Concealing complex policies with hidden credentials," in *Proceedings of the 11th ACM Conference on Computer and Communications Security (CCS '04)*, pp. 146–157, October 2004.

[19] Z. Zhou, D. Huang, and Z. Wang, "Efficient privacy-preserving ciphertext-policy attribute based-encryption and broadcast encryption," *IEEE Transactions on Computers*, vol. 64, no. 1, pp. 126–138, 2015.

[20] J. Li, K. Ren, B. Zhu, and Z. Wan, "Privacy-aware attribute-based encryption with user accountability," in *Information Security*, pp. 347–362, Springer, Berlin, Germany, 2009.

[21] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Advances in Cryptology*, G. R. Blakley and D. Chaum, Eds., vol. 196 of *Lecture Notes in Computer Science*, pp. 47–53, 1985.

[22] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Advances in Cryptology—EUROCRYPT 2005*, vol. 3494 of *Lecture Notes in Computer Science*, pp. 457–473, Springer, Berlin, Germany, 2005.

[23] Z. Zhou and D. Huang, "On efficient ciphertext-policy attribute based encryption and broadcast encryption," in *Proceedings of the 17th ACM Conference on Computer and Communications Security (CCS '10)*, pp. 753–755, ACM, Chicago, Ill, USA, October 2010.

[24] C. Chen, Z. Zhang, and D. Feng, "Efficient ciphertext policy attributebased encryption with constant-size ciphertext and constant computationcost," in *Provable Security: 5th International Conference, ProvSec 2011, Xi'an, China, October 16–18, 2011. Proceedings*, vol. 6980 of *Lecture Notes in Computer Science*, pp. 84–101, Springer, Berlin, Germany, 2011.

[25] J. Li, K. Ren, and Z. Wan, "Privacy-aware attribute-based encryption with user accountability," in *Information Security*, pp. 347–362, Springer, Berlin, Germany, 2009.

[26] Z. Liu, Z. Cao, and D. S. Wong, "White-box traceable ciphertext-policy attribute-based encryption supporting any monotone access structures," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 1, pp. 76–88, 2013.

[27] Z. Liu, Z. Cao, and D. S. Wong, "Blackbox traceable CP-ABE: How to catch people leaking their keys by selling decryption devices on eBay," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security (CCS '13)*, pp. 475–486, ACM, November 2013.

[28] D. Boneh, X. Boyen, and E. J. Goh, "Hierarchical identity based encryption with constant size ciphertext," in *Advances in Cryptology—EUROCRYPT 2005*, pp. 440–456, Springer, Berlin, Germany, 2005.

[29] D. Boneh and X. Boyen, "Short signatures without random oracles," in *Advances in Cryptology-EUROCRYPT 2004*, vol. 3027 of *Lecture Notes in Computer Science*, pp. 56–73, Springer, Berlin, Germany, 2004.

[30] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Advances in Cryptology—CRYPTO 2001*, pp. 213–229, Springer, Berlin, Germany, 2001.

[31] A. Kate, G. Zaverucha, and I. Goldberg, "Pairing-based onion routing," in *Privacy Enhancing Technologies*, N. Borisov and P.

Golle, Eds., vol. 4776 of *Lecture Notes in Computer Science*, pp. 95–112, Springer, Berlin, Germany, 2007.

[32] E. Jovanov and D. Raskovic, "Wireless intelligent sensors," in *M-Health*, pp. 33–49, Springer, New York, NY, USA, 2006.

[33] A. De Caro and V. Iovino, "jPBC: java pairing based cryptography," in *Proceedings of the IEEE Symposium on Computers and Communications (ISCC '11)*, pp. 850–855, June-July 2011.

[34] B. Lynn, *The Pairing-Based Cryptography (PBC) Library*, 2010, http://crypto.stanford.edu/pbc.

Hindawi Publishing Corporation
Mobile Information Systems
Volume 2016, Article ID 2316757, 12 pages
http://dx.doi.org/10.1155/2016/2316757

Research Article

Energy-Efficient Real-Time Human Activity Recognition on Smart Mobile Devices

Jin Lee and Jungsun Kim

Department of Computer Science and Engineering, Hanyang University, Ansan, Gyeonggi-Do 15588, Republic of Korea

Correspondence should be addressed to Jungsun Kim; kimjs@hanyang.ac.kr

Received 31 December 2015; Accepted 30 May 2016

Academic Editor: Wenyao Xu

Nowadays, human activity recognition (HAR) plays an important role in wellness-care and context-aware systems. Human activities can be recognized in real-time by using sensory data collected from various sensors built in smart mobile devices. Recent studies have focused on HAR that is solely based on triaxial accelerometers, which is the most energy-efficient approach. However, such HAR approaches are still energy-inefficient because the accelerometer is required to run without stopping so that the physical activity of a user can be recognized in real-time. In this paper, we propose a novel approach for HAR process that controls the activity recognition duration for energy-efficient HAR. We investigated the impact of varying the acceleration-sampling frequency and window size for HAR by using the variable activity recognition duration (VARD) strategy. We implemented our approach by using an Android platform and evaluated its performance in terms of energy efficiency and accuracy. The experimental results showed that our approach reduced energy consumption by a minimum of about 44.23% and maximum of about 78.85% compared to conventional HAR without sacrificing accuracy.

1. Introduction

Interest in u-health and wellness-care has recently been growing [1–3]. Various technologies that recognize the physical activities of users using various embedded sensors in smart mobile devices are actively studied. Recognized physical human activities can be used to develop applications that predict a falling accident or measure calorie consumption [4–7]. Such applications mainly use the triaxial accelerometer because it consumes the least power compared to other available sensors [8, 9]. Therefore, the use of "sensor" hereafter in this paper refers to a triaxial accelerometer.

These applications need the accelerometer to operate continuously without stopping in order to recognize different physical human activities in real-time. Unfortunately, this incurs unnecessary power consumption by the sensor and computational overhead; it is regarded as a big problem considering the limited power resources of smart mobile devices [10–12]. For example, while the battery life of LG Optimus Pro reaches up to over 60 hours when all applications and sensors are turned off, it decreases to 22 hours when a human activity recognition (HAR) application is activated with a sensor (100 Hz).

One facile solution is to blindly limit the usage of the accelerometer, but this may cause another problem of sacrificing the accuracy of human activity recognition. Another solution is to adopt a lower acceleration-sampling frequency (SF) for the sensor, but this may result in the loss of important sampling data. For this reason, previous studies have mostly focused on achieving a rather suboptimal balance between energy efficiency and HAR accuracy, instead of seeking optimal power consumption without sacrificing the HAR accuracy [13–16]. An analysis of the previous studies showed that they required the accelerometer to be operating at all times; as a result, the power consumption due to the continuous operation of the sensor itself and the accompanying data processing by the CPU remain unaddressed. In this paper, we argue that it is possible to save energy to great extent without continuous sensor operation.

In order to further improve the energy efficiency, we propose an approach that dynamically controls the variable activity recognition duration (VARD) for HAR. Our approach

classifies a user's activities as dynamic or static and controls the classification duration and sleep time for the HAR process based on two factors: the acceleration-sampling frequency and window size (WS). We performed experiments and conducted a thorough analysis of the result to show that the proposed VARD strategy performs well in terms of both energy efficiency and HAR accuracy.

The remainder of the paper is organized as follows. Section 2 presents an analysis of previous HAR approaches for efficient power consumption. Section 3 describes our initial motivations and a basic HAR system. Section 4 presents the impact of varying the SF, WS, and feature vector dimensionality (FVD) on the classification accuracy and the power consumption. Section 5 explains the VARD strategy. Section 6 reports on the evaluation results for our approach. Finally, Section 7 concludes with a summary and future directions.

2. Related Works

In this section, we first present a variety of accelerometer-based HAR technologies and then discuss relevant previous studies.

2.1. Human Activity Recognition Using Accelerometer. Early-stage researchers investigated the wearable sensor-based HAR; they demonstrated that the usage of wearable sensors can provide elevated accuracy in the area of HAR [17–20]. Recent wearable sensor-based HAR has been enhanced by some previous work. Hong et al. [21] presented a personalized HAR system using Bayesian network and support vector machine (SVM).

Due to the rapid advancement of smart mobile devices technology, many researchers focused on the mobile device-based HAR. Their work [4, 22–24] also turned out to be successful in providing high recognition rate. Torres-Huitzil and Nuno-Maganda [25] showed a position-independent HAR system using time-domain features and neural network. Vo et al. [13] presented a personalized HAR system through SVM, along with a k-medoids clustering method. Albert et al. [2] studied a HAR system for Parkinson's patients.

Smart mobile devices are promising platform for HAR because they not only are equipped with embedded built-in sensors but also are a natural part of everyday human daily life [26]. However, smart mobile device needs energy management due to its limited resources.

2.2. Human Activity Recognition with the Energy-Saving. A naive solution to reducing the power consumption of mobile devices is to limit the usage of the accelerometer. However, such an approach may negatively affect the HAR accuracy and therefore should be applied with caution.

Vo et al. [13] aimed to reduce the power consumption of the accelerometer and CPU by improving the HAR algorithm. Their approach relied on a SVM and time-domain features and reduced the power consumption by about 6.7% when compared to a conventional approach adopting SVM and fast Fourier transform (FFT). However, they focused more on the HAR accuracy than on reducing the power consumption.

Vo et al. [14] and Yan et al. [15] improved the power consumption efficiency by changing the SFs of the accelerometer and classification features. The key concept was identifying the best combination of SF and classification feature for a specific activity. Their approach reduced the power consumption by about 20%–25% compared to previous approaches. However, their approach also requires continuous operation of the triaxial accelerometer when the application is running.

Liang et al. [16] reduced the power consumption of HAR by using lower SFs. They proposed a hierarchical recognition algorithm that uses time-domain features, frequency-domain features, and similarity measurements. Their algorithm applies a decision tree instead of SVM. In their results, the battery life was extended by 3.2 h. However, because this algorithm tried to use a lower SF, the HAR accuracy was at best over 85%, which is less than that of other studies [13–15].

In this paper, we propose a new approach for HAR process that reflects the physical states of the mobile user. Our approach can secure a similar or higher HAR accuracy compared to previous approaches while providing better energy efficiency.

3. Human Activity Recognition on Smart Mobile Devices

In this study, our aim was to develop a lightweight HAR approach that uses the embedded accelerometer in smart mobile devices. To build a mobile HAR system on smart mobile devices, methods for sensor monitoring and real-time detection of user activity need to be considered, as depicted in Figure 1.

Typical HAR can simply be defined as the process of interpreting raw sensor data to classify a set of physical human activities [27]. Statistical machine learning techniques are used to infer information about the activities from raw sensor readings; this process usually includes a training phase and predicting phase. The training phase requires collecting labeled data to learn the model parameters and build a training model from the collection. The predicting phase uses the training model to classify physical activities of users in the following sequence: preprocessing, segmentation, feature extraction, and classification. The following subsections explain the details of the proposed HAR process.

3.1. Collecting Acceleration. Physical human activities consist of basic movements such as walking, sitting, standing, and running. We selected the six most common activities as target activities, which have been recognized in previous works [8, 13–16, 19, 24]. Table 1 presents the target activities for our study.

We collected data from the triaxial accelerometer (MPU-6050; maximum range: 39.227 m/s^2; resolution: 0.001 m/s^2) on the LG Optimus Pro (Android Kitkat 4.4.2 OS) of two male subjects who are 28 and 32 years old, respectively. A smart mobile device was placed inside the back pocket of the pants of a subject.

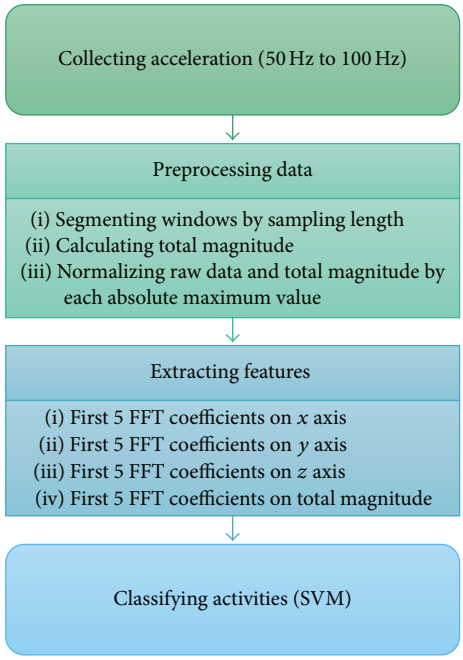


Figure 1: Human activity recognition using a single accelerometer.

Table 1: Classification of target activities.

Activity type	Activity
Static	Standing and sitting
Dynamic	Walking, running, ascending stairs, and descending stairs

With the Android operating system, four different SFs (NORMAL: 5 Hz, UI: 16 Hz, GAME: 50 Hz, and FASTEST) can be selected for the accelerometer. The FASTEST SF depends on the computational workload of each specific mobile device and thus can differ from device to device. For our device, the FASTEST frequency was 100 Hz. In this study, we collected training data for six activities from two subjects. For each activity, 30 samples were collected at four different SFs; thus, we collected 1440 samples in total. A sample was a unit with a single activity classification and corresponded to a window that contained the preset number of contiguous accelerometer data, which we called the WS. Section 4 discusses the experiments performed with the above samples. Figure 2 illustrates an example of the acceleration signals of human activities on each axis. This example was obtained at an SF of 100 Hz and WS of 128.

3.2. Preprocessing Data. The preprocessing step consists of segmentation, the total magnitude (TM), and normalization. In the segmentation phase, the raw accelerometer data are segmented into windows with size n, where $n/2$ accelerometer samples overlap between two consecutive windows. Feature extraction has been successfully performed on windows with 50% overlap in previous work [17]. The TM is the intensity (vibration) of a user activity and is a significant metric for discriminating between activities [8, 16, 24]. The TM is

Figure 2: An example of acceleration signals of target activities on three axes.

calculated according to $\mathrm{TM} = \sqrt{A_x + A_y + A_z}$, where A_i is the magnitude of the sampled data on i-axis. Figure 3 plots the acceleration on each axis, and the TM data are a sample of the "walking" activity.

Finally, the raw data and TM data are normalized to have values in range of (−1, 1) for later feature extraction and classification [28].

3.3. Extracting Features. The selection of proper features from raw data plays an important role in the HAR performance. In general, the relevant features extracted for HAR are grouped into three categories: (i) time-domain features such as the mean, standard deviation, energy, and correlation between axes [13–17, 23]; (ii) frequency-domain features such as the FFT coefficient, zero crossing rate, and autocorrelation of the magnitude [13–17, 29, 30]; and (iii) other features such as wavelet features [16, 29], the autoregressive coefficient [31], and discrete cosine transform coefficients [32].

The FFT coefficient demonstrates a higher average accuracy than the rest of the features [16, 31]. Thus, the first 20 FFT coefficients (first five for each of the three axes and five from TM; see Sections 4.1 and 4.2) are selected for each window, as illustrated in Figure 4. The FFT coefficients on each axis reflect the amplitude of basic waves which can be combined to reconstruct the original signal. For FFT, we utilized the decimation-in-time (DIT) Radix-2 FFT [33], which recursively partitions a discrete Fourier transform (DFT) into two half-length DFTs of the even- and odd-indexed time samples.

3.4. Classifying Activities. The extracted feature vectors can be classified by using the SVM classifier, which is widely used for HAR [13–15, 23]. LibSVM [34] was adopted to classify the dataset. SVM is a learning algorithm that separates training samples into their corresponding classes by maximizing the margin of a separating hyperplane between classes in order to solve the classification problem. SVM efficiently finds the complex hyperplane in nonlinear data by using the kernel trick. We used the radial basis function (RBF) kernel in order to map support vectors to multiple dimensions because there were 20 FFT attributes [23].

Human activities were classified into two activity types, as given in Table 1: (i) the static activity type (SAT) includes "sitting" and "standing" and (ii) the dynamic activity type

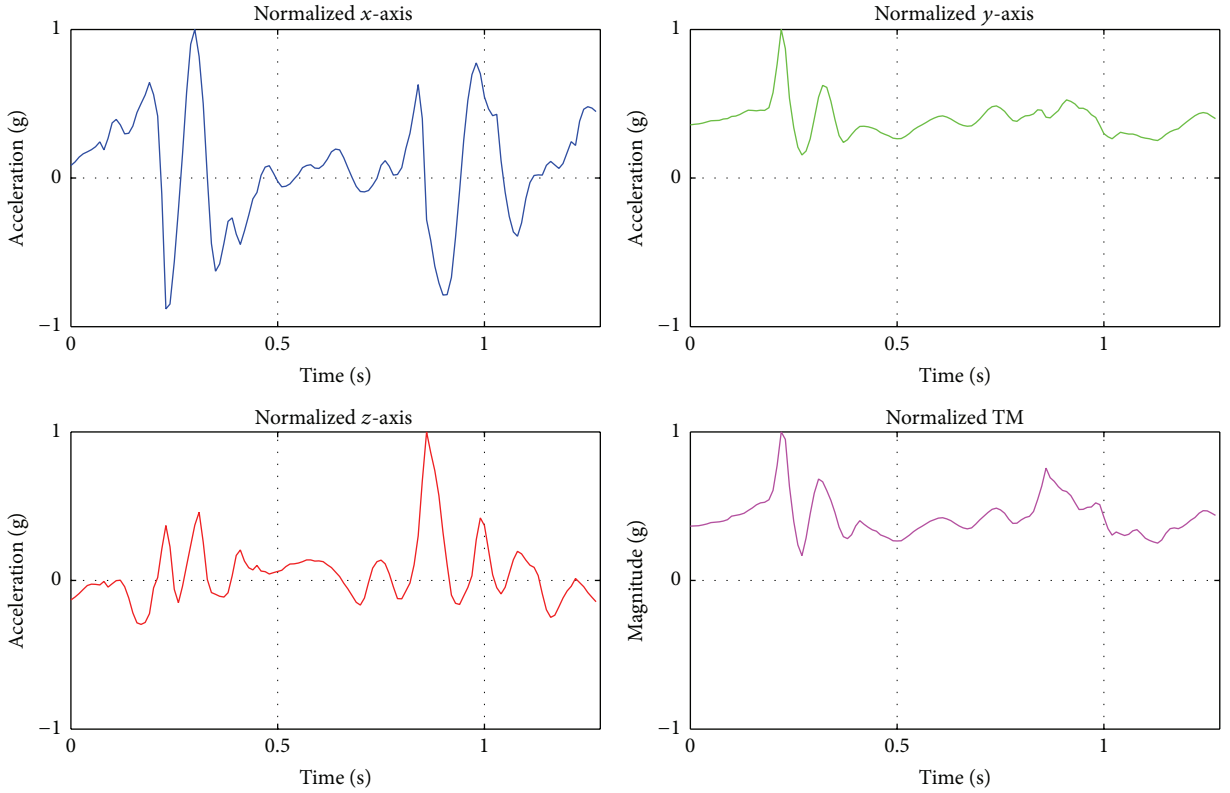


Figure 3: Normalized amplitude of the "walking" activity on x-, y-, and z-axes and the total magnitude.

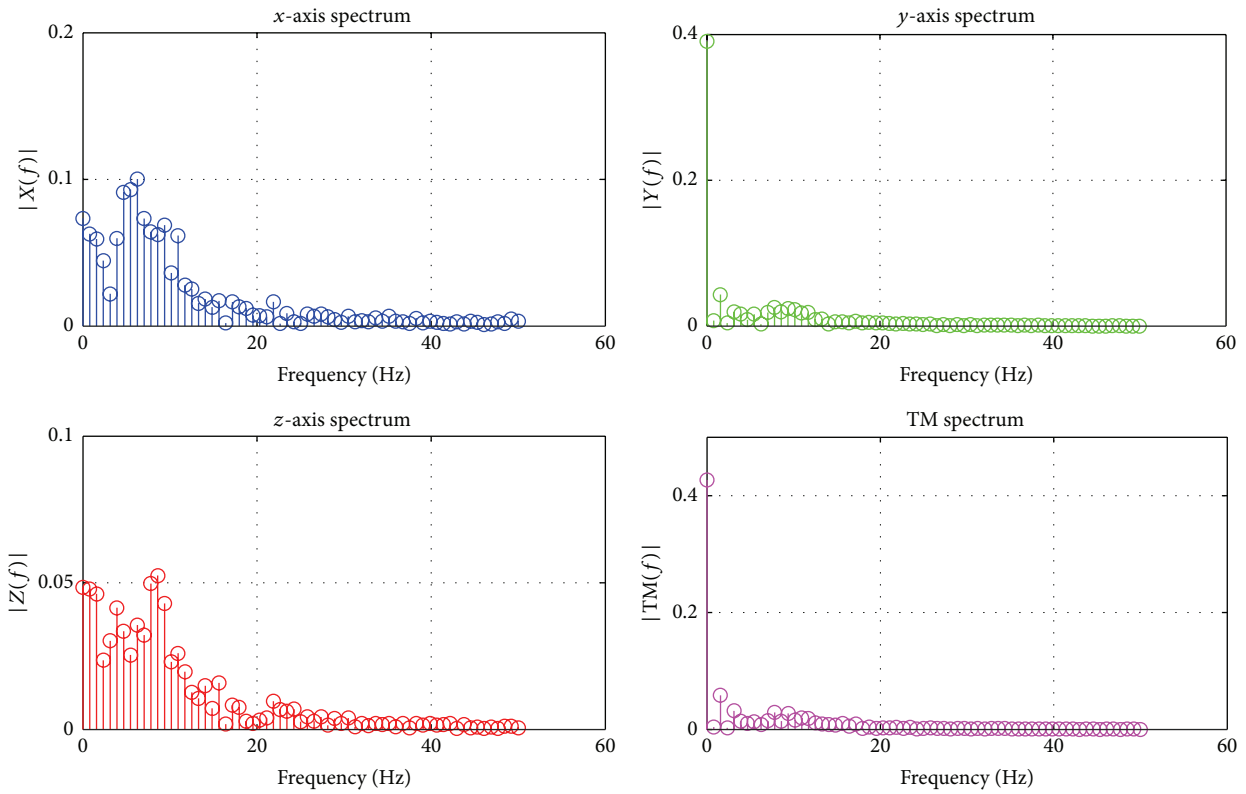


Figure 4: 3D acceleration and total magnitude after fast Fourier transform.

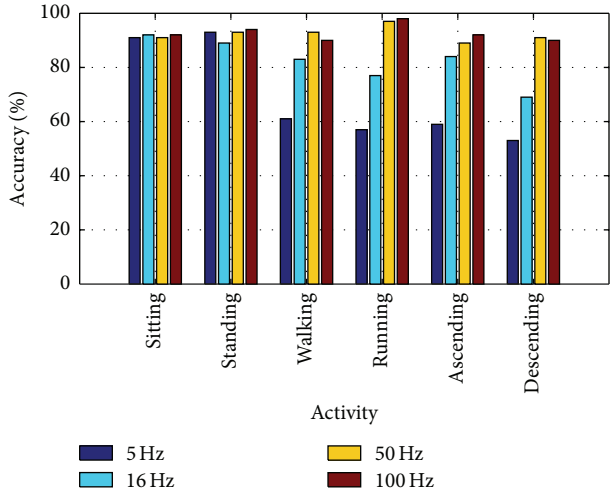


Figure 5: Accuracy across six activities with different acceleration-sampling frequencies.

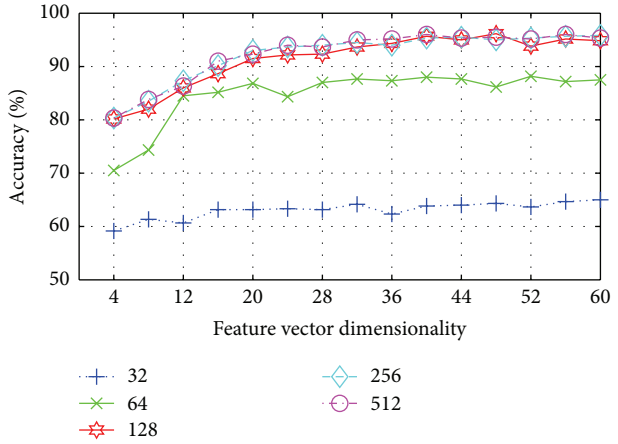


Figure 6: Accuracy versus feature vector dimensionality with different window sizes.

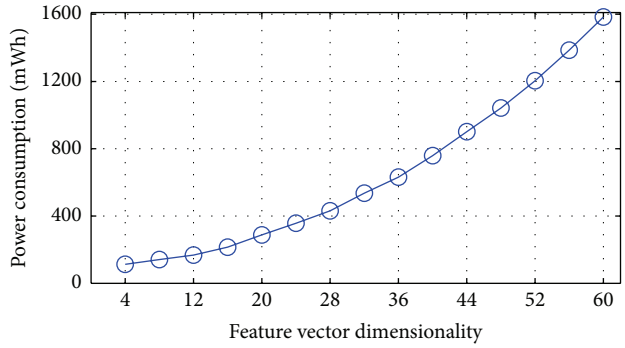


Figure 7: Power consumption of feature vector dimensionalities at an acceleration-sampling frequency of 100 Hz and window size of 128 over 30 min.

(DAT) includes "walking," "running," "ascending stairs," and "descending stairs." The SAT is equivalent to a nonmoving relaxed state, and the DAT denotes active movement. Our strategy exploits the fact that humans are likely to maintain the same activity type for some time, especially for the SAT.

4. Tradeoff between Energy and Accuracy

The effects of the SF, WS, and FVD on the classification accuracy and power consumption were evaluated, and the FVD and combination of SF and WS were identified for application to our method. To obtain the readings, we turned off the network interfaces and display of our mobile device during the experiment. We used PowerTutor [35] utility to measure the power consumption.

4.1. Classification Accuracy and Acceleration-Sampling Frequency. We investigated the impact of different SFs on the classification accuracy with a WS of 128 and FVD of 20. Here, 2400 test samples were used (six activities × four SFs × 100 samples).

As shown in Figure 5, high SFs normally produced better predictions, especially for the DAT cases. The SFs of 50 and 100 Hz recorded an average accuracy of 90% or more in six activities and were sufficiently higher than the minimum SF of 20 Hz that is required to assess daily activities [36].

4.2. Classification Accuracy and Feature Vector Dimensionality with Differing Window Sizes. Figure 6 illustrates how the classification accuracy changed with the number of coefficients for each WS. Using the first 20 FFT coefficients (first five for each of the three axes and five from TM) produced an accuracy of more than 90% for a WS of 128 or more. Our experiments showed a slightly different result compared to Preece et al. [29], who analyzed the discriminative ability of individual FFT coefficients. They found that applying the first 18 coefficients (first six on each of the three axes) produced the maximal accuracy. This discrepancy may be due to our incorporation of TM coefficients in our feature vectors.

4.3. Power Consumption and Feature Vector Dimensionality. For this experiment, we set the SF and WS to 100 Hz and 128, respectively. The SF of 100 Hz had the best classification accuracy, as shown in Figure 5, and the WS of 128 had a prediction accuracy of over 90%, as shown in Figure 6. Figure 7 plots the power consumption over 30 min against different numbers of FFT coefficients. The power consumption showed a quadratic increase with the dimensionality. Based on the results shown in Figures 3 and 4, we selected an FVD of 20 in our study. This had the least power consumption among FVDs with an accuracy of more than 90%.

4.4. Power Consumption and Acceleration-Sampling Frequency with Differing Window Sizes. Figure 8 illustrates the power consumption for different SFs and WSs with an FVD of 20 over 2 h. The results can be summarized as follows:

(i) The power consumption clearly increases with the SF. A high frequency mandates more frequent raw data collection.

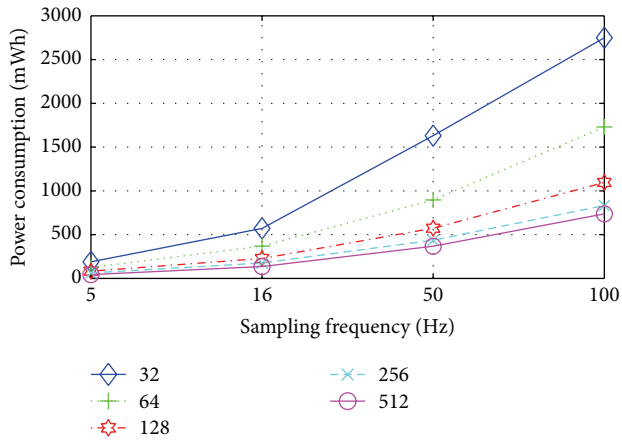


Figure 8: Power consumption at different acceleration-sampling frequencies and window sizes over 2 h.

Table 2: Summary of tradeoff between energy and accuracy.

Observation	Accuracy	Power consumption
Higher SF	Higher	Increase
Bigger WS	Higher	Decrease
More FVD	Higher	Rapid increase

(ii) Larger WSs normally consume less power because they decrease the number of classifications, which take up a large proportion of the power consumption.

Table 2 summarizes our investigations. We adopted SFs (50 Hz and 100 Hz), WSs (128, 256, and 512), and an FVD (20) which yielded an accuracy of 90% or more with low power consumption.

5. Experiments on the Variable Activity Recognition Duration Strategy

To monitor user activities on smart mobile devices in an energy-efficient manner, our study focused on two key ideas.

First, humans more often tend to maintain the same activity than change from one activity to another (e.g., walk-to-run and sit-to-stand). When one activity is recognized in succession, we assumed that the activity will be lasted for a while. Therefore, we focused on developing an energy-saving scheme that increases the classification duration this situation. If we increase the period in which an activity is recognized in a given time, the frequency of activity recognition will decrease. Consequently, this reduces the power consumption necessary for activity recognition. To increase the classification duration, we adopted a method that lowers the SF and/or increases the WS. We verified that a low SF and large WS consume less power, as shown in Figure 8.

Second, dynamic activity (e.g., walking and running) is more meaningful than static activity (e.g., sitting and standing) equivalent to a nonmoving relaxed state because it can be used as data for dynamic health information such as calorie consumption. Thus, we first classified a user's activities as a DAT and SAT, as indicated in Table 1. And then, when an SAT is recognized, we gave a break to the HAR process in order to save more energy.

Table 3: Variable activity recognition duration configuration for dynamic activity type.

Activity type	Acceleration-sampling frequency	Window size
Dynamic	100 Hz	128
	50 Hz	128
	50 Hz	256
	50 Hz	512

Based on these ideas, we applied different strategies for each type with regard to the classification duration. To control the duration, the SF and WS were used for the DAT, and a sleep time was additionally used for the SAT. We call this energy-saving scheme the variable activity recognition duration (VARD) strategy.

5.1. Variable Activity Recognition Duration Strategy for the Dynamic Activity Type. To increase the classification duration, we can lower the SF and/or increase the WS. However, a low SF and large WS are insensitive to rapidly changing activities because they yield fewer samples than a high SF and small WS. Therefore, our strategy is to start with a high SF and small WS to quickly identify changing activities. If the same dynamic activity is maintained for a long time, we assume that the same activity will continue and adopt a method to lower the SF and increase the WS.

To guarantee the energy efficiency and high accuracy of HAR, we can choose SFs of 50 and 100 Hz, as shown in Figure 5, and WSs of 128, 256, and 512, as shown in Figure 6. Each SF and WS can be combined for a total of six combinations. The classification durations of ⟨100 Hz, 256⟩ and ⟨50 Hz, 128⟩ are the same at 2.56 s.

However, the power consumption of ⟨50 Hz, 128⟩ (573 mWh) is less than that of ⟨100 Hz, 256⟩ (832 mWh), as shown in Figure 8. Another difference between the two combinations is that the larger WS provides better HAR accuracy because it extracts more precise features in the raw data with noise comprising the latter part of previous acceleration from the changing activity, as shown in Figure 9. These two differences have conflicting tendencies for the energy efficiency and HAR accuracy. If the classification durations overlap, we can choose the energy-efficient combination to focus on saving energy.

Accordingly, we adopted four combinations for the strategy with the DAT, as listed in Table 3: ⟨100 Hz, 128⟩, ⟨50 Hz, 128⟩, ⟨50 Hz, 256⟩, and ⟨50 Hz, 512⟩. We used the repeating count of the same activity in order to check that the same activity is continuous. A threshold for this count was set, and we implemented a strategy of changing from the current combination to the next combination with a low frequency and large WS if the count carries over the threshold. The progression to each configuration away from the first combination causes the improvement in energy efficiency and marginal weakening of the HAR accuracy.

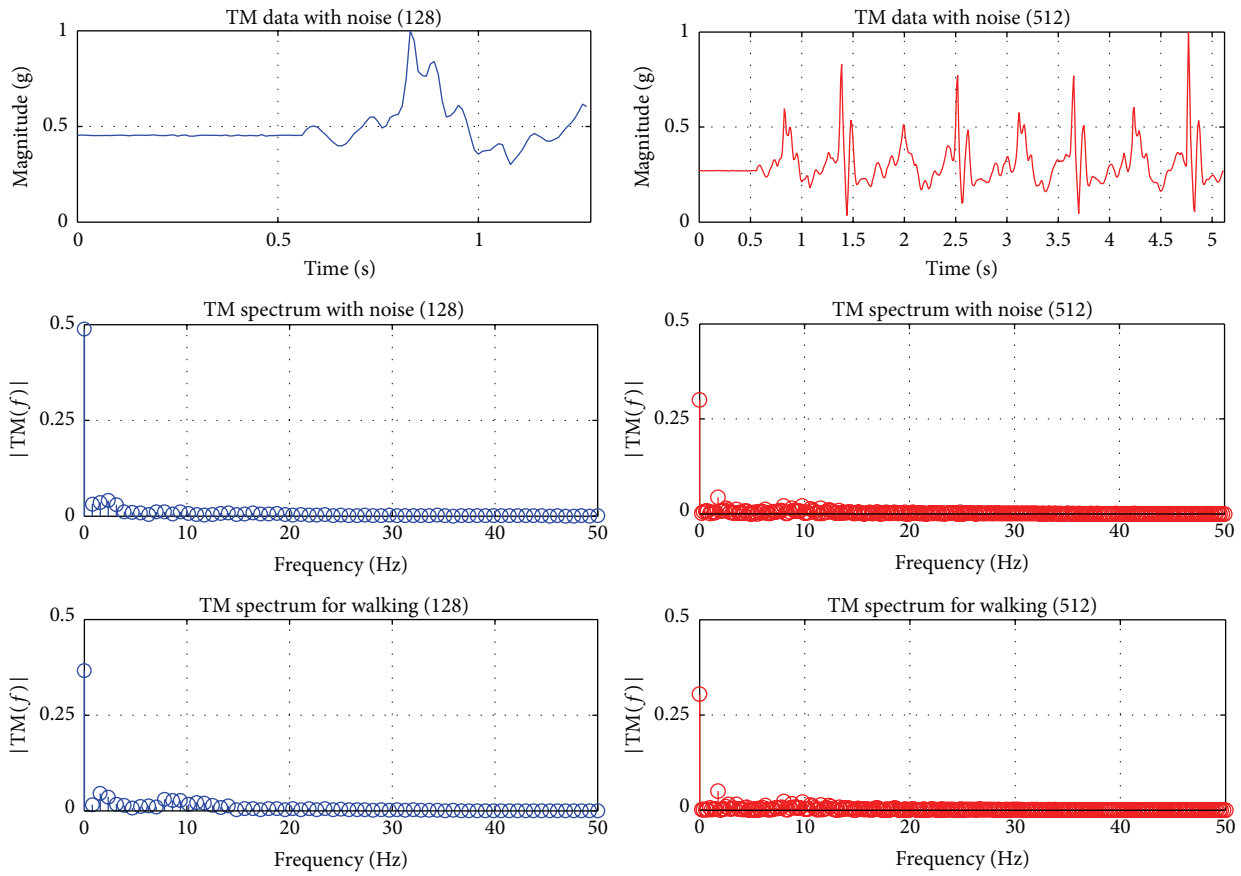


Figure 9: Comparison of feature extraction precision with noise.

5.2. Variable Activity Recognition Duration Strategy for the Static Activity Type. Our strategy for the SAT is based on a similar concept for the DAT strategy. However, there is no need to recognize SAT often because there is less movement compared with DAT. Our SAT strategy, therefore, uses the sleep time during the HAR process along with the SF and WS for better energy efficiency compared to the DAT strategy. In addition, a DAT should be stably recognized in the SAT state because it is more important than the SAT for extracting processed information.

In our strategy, when an SAT is recognized during the classification of human activity, the process takes a break. After the break, the human activity is reclassified. As a result, the classification duration increases within a given time because this strategy incorporates a sleep time.

To ensure stable HAR accuracy while reducing energy consumption, this strategy involves Sleeping 0 s when an SAT is initially recognized and gradually increasing the sleep time in increments of 1 s whenever an SAT is continuously recognized.

The power consumption can be reduced with a break. Nevertheless, the extent to which the break can be increased while ensuring stable HAR accuracy needed to be evaluated. Therefore, we investigated the HAR accuracy with six combinations: ⟨100 Hz, 128⟩, ⟨50 Hz, 128⟩, ⟨100 Hz, 256⟩, ⟨50 Hz, 256⟩, ⟨100 Hz, 512⟩, and ⟨50 Hz, 512⟩. This was done in order to calculate the preferred maximum sleep time. In this experiment, the HAR accuracy was observed as the break was increased from 0 s to 60 s for each combination. The observation times for each break were 5 and 10 min. We made a total of 732 observations ($6 \times 61 \times 2$) of the HAR accuracy. Figure 10 plots the observed HAR accuracy.

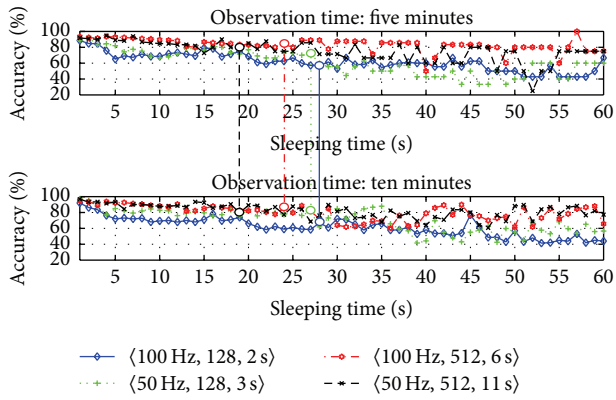


Figure 10: Recognition accuracy of human activities with variable activity recognition duration. The circle of each combination represents the break where the human activity recognition accuracy violently fluctuates.

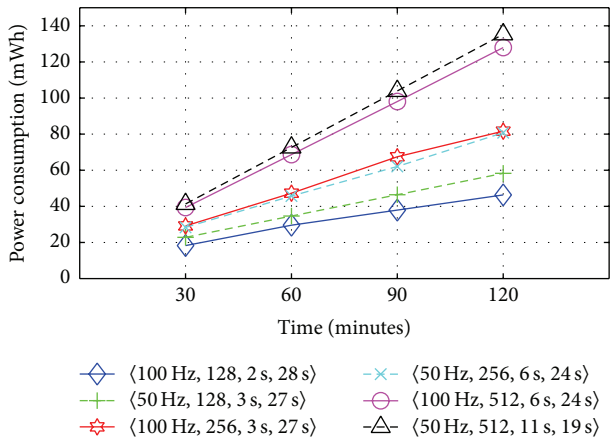


Figure 11: Power consumption of each of six combinations with a maximum sleep time of 2 h.

This experiment showed that the HAR accuracy became unstable every time the break was over a certain amount. The circular symbols in Figure 10 show the break after which the HAR accuracy badly fluctuated. This point was the limit to the break for each combination. The limit can be calculated by

$$t_{\text{lim}}(f_s, n) = T - \left\lceil \frac{n}{f_s} \right\rceil, \quad (1)$$

where f_s is an SF, n is a WS, and T is a constant of 30 s as determined in this experiment. Based on this limit, we can guarantee efficient power consumption and stable accuracy during HAR.

Figure 11 plots the measured power consumption of six combinations in 2 h with a preset maximum sleep time: ⟨100 Hz, 128⟩, ⟨50 Hz, 128⟩, ⟨100 Hz, 256⟩, ⟨50 Hz, 256⟩, ⟨100 Hz, 512⟩, and ⟨50 Hz, 512⟩. The power consumption increased with a larger WS relative to a small WS, and changes to the SF had less effect on the power consumption than changes to the WS. This is because the numbers of activity recognition processes for every combination within a given time are equal if the HAR process has a sleep time, and a large WS increases the computational cost of HAR. As a result, the samples with a large WS consumed more power. Therefore, using a small WS can ensure high energy efficiency.

As shown in Figure 10, however, the average accuracy is higher for large WSs than small WSs. Thus, we adopted three combinations for the SAT strategy: ⟨100 Hz, 512⟩, ⟨100 Hz, 256⟩, and ⟨100 Hz, 128⟩. As indicated in Table 4, we defined the VARD combination configuration for SAT strategy. We used the repeating count of SAT in order to check that the type is continuous and employed a strategy of changing from the current combination to the next combination with a smaller WS if the count carried over a threshold based on the sleep time limit. Progressing to further configurations away from the first combination increases the energy efficiency and destabilizes the HAR accuracy.

Table 4: Variable activity recognition duration configuration for static activity type.

Activity type	Acceleration-sampling frequency	Window size
Static	100 Hz	512
	100 Hz	256
	100 Hz	128

5.3. Real-Time Human Activity Recognition with the Variable Activity Recognition Duration Strategy. The VARD strategy can effectively guarantee not only classification accuracy but also energy efficiency because it does not need to constantly keep a specific SF and WS for HAR. Figure 12 represents our approach as a state machine diagram, and the strategy is described in Algorithm 1. In order to obtain a break, our HAR process is divided into a Sensing State and Sleeping State, as shown in Figure 12.

The Sensing State repeats the following cycles: collecting, preprocessing, feature extraction, and classification. Our classifier in the HAR process uses a variety of training models for VARD configuration, as indicated in Tables 3 and 4. These models are built by an offline SVM using the training samples discussed in Section 3.1.

By classifying a recognized activity as a DAT or SAT, the HAR process transfers from the Sensing State to the state for each type. For a DAT, the HAR process goes into the Dynamic State to perform the DAT strategy. Otherwise, the SAT strategy is performed for the Static State. When the SAT strategy is performed, the HAR process transfers to the Sleeping State unconditionally and takes a break. This break time is set by the repeating count of SAT. After the break, the process returns to the Sensing State in order to reclassify the human activity.

When an event listener for the triaxial accelerometer is registered in the initial Idle State, the HAR process transfers to the Active State. The Active State comprises two substate machines: the Sensing State and Sleeping State. In the Active State, the process initializes the SF and WS and loads the classification model for this combination. It also sets a threshold for the repeating count of the same activity. The HAR process transfers to the Sensing State after the accelerometer is started. While this transition is performed, the repeating count of the SAT and repeating count of the same activity are initialized with zero. When all of the initializations are completed, the Sensing State begins so that a human activity can be recognized. This portion is equivalent to lines (1)–(10) in Algorithm 1.

When a recognized activity is a DAT, the HAR process is transferred to the Dynamic State. In this state, the repeating count of the same activity and maximum sleep time are initialized. In the Dynamic State, the current activity is checked to see if it is equivalent to the previous activity. If they are the same, the repeating count of the same activity is increased. If this count exceeds the threshold, then the current VARD configuration is changed to the next combination, and the count is initialized. If the current and previous activities are not the same, the repeating count of the same activity is initialized, and the VARD configuration is changed to the first

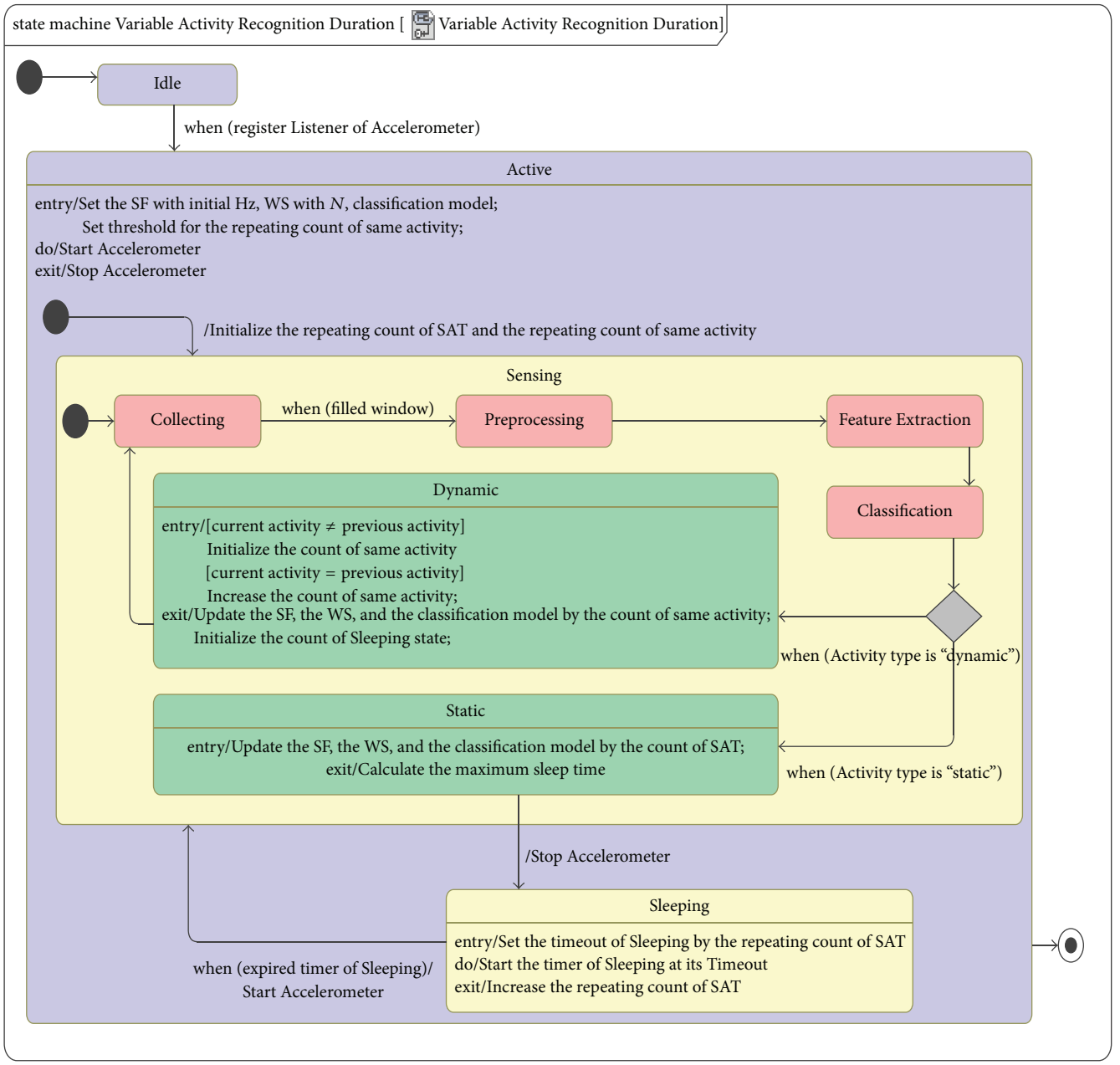


Figure 12: State machine for the variable activity recognition duration strategy.

DAT combination. In the Dynamic State, SF and WS are updated by the current configuration, and a classification model is loaded for the configuration. At the end of the Dynamic State, HAR is started. This portion is equivalent to lines (11)–(22) in Algorithm 1.

When a recognized activity is an SAT, the HAR process is transferred to a Static State. If the previous activity is a DAT, the VARD configuration is first changed to an SAT. In the Static State, SF and WS are updated by the current configuration, and a classification model is loaded for the configuration. If the repeating count of the SAT exceeds the maximum sleep time, the current VARD configuration is changed to the next SAT combination, and the count is initialized. Then, the HAR process is transferred to the Sleeping State, and the accelerometer stops. In this state, the sleep time is set by using the repeating count of the SAT, and the HAR process takes a break during this time. After the break, the repeating count of the SAT is increased, and the HAR process is transferred to the Active State. This portion is equivalent to lines (23)–(38) in Algorithm 1.

6. Performance Evaluation and Discussion

To evaluate the performance of the proposed algorithm, we performed independent experiments with regard to the recognition accuracy and power consumption. An application

```
(1) Set the SF f with the initial Hz and the WS N with the initial size;
(2) Load the classification model M for f and N;
(3) Load the dynamic configuration table as [D1: (100 Hz, 128), D2: (50 Hz, 128), D3: (50 Hz, 256), D4: (50 Hz, 512)];
(4) Load the static configuration table as [S1: (100 Hz, 512), S2: (100 Hz, 256), S3: (100 Hz, 128)],
(5) the repeating count of SAT c ← 0, and the repeating count of the same activity j ← 0; i ← 1;
(6) Set the threshold th for j
(7) and the maximum sleep time t_lim ← 0;
(8) While App_Running Do
(9)     Start the accelerometer; fill window with N;
(10)    Classify the current activity from the window with M;
(11)    If the current activity is DAT Then
(12)        c ← 0; t_lim ← 0;
(13)        If the current activity is equivalent to the previous activity Then
(14)            Increase j;
(15)            If j exceeds th Then
(16)                Increase i up to the size of the dynamic configuration table; j ← 0;
(17)            End If
(18)        Else
(19)            j ← 0; i ← 1;
(20)        End If
(21)        Update f and N with the control table D_i;
(22)        Load the classification model M for f and N;
(23)    Else
(24)        If t_lim is equivalent to 0 Then
(25)            i ← 1;
(26)        End If
(27)        Update f and N with the control table S_i;
(28)        Load the classification model M for f and N;
(29)        Calculate t_lim based on (1);
(30)        If c exceeds t_lim Then
(31)            Increase i up to the size of the static configuration table; c ← 0;
(32)        End If
(33)        Stop the accelerometer;
(34)        Set the timeout of Sleeping with the repeating count of SAT c;
(35)        Delay for the timeout of Sleeping;
(36)        Increase c;
(37)    End If
(38) End While
```

Algorithm 1: Variable activity recognition duration algorithm. The elements (D_i, S_i) of the dynamic and static configuration tables contain the acceleration-sampling frequency and window size where th is the threshold to maintain a combination in the dynamic configuration table, c is the repeating count of the static activity type, j is the count continuously kept of any activity, and i is the index of an element in the configuration table. There are seven classification models for each element in the control tables, and t_{lim} is the maximum sleep time.

employing our approach was installed as an Android service that can operate in the background. The initial WS and SF were set to 128 and 100 Hz, respectively. The threshold th for a DAT was set to 10. The experimental results were as follows.

6.1. Energy Efficiency. Five cases were considered, each for a span of 12 h:

(i) No HAR: there is no HAR application running on the phone.

(ii) Typical SVM: the SF is fixed at 100 Hz, and the WS is fixed at 128.

(iii) VARD with DAT only: all activities are assumed to be DAT.

(iv) VARD with SAT only: all activities are assumed to be SAT.

(v) VARD with daily activities: daily activities include walking to the lab, moving on stairs, studying at a desk, and jogging.

We measured the battery level by using BatteryManager Android API and powered off the network interfaces and display of our mobile device during the experiment. Figure 13 compares the battery drainage time series in our experiment. HAR with VARD showed slow and stable power consumption of the smart mobile device over time. VARD with DAT represented only the maximum power consumption of our approach. This case clearly reduced the energy consumption by 23% compared to the typical

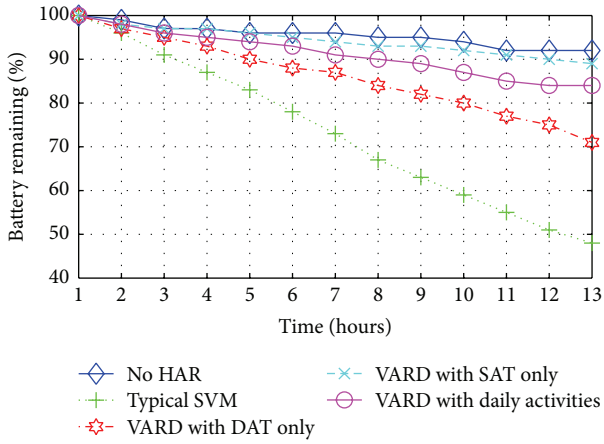


Figure 13: Comparing power consumption of different types.

Table 5: Confusion matrix of human activity recognition with variable activity recognition duration.

Classified as	a	b	c	d	e	f
a	**92**	5				
b	8	**94**				
c		1	**91**		6	4
d			2	**97**	2	2
e			5	1	**89**	4
f			2	2	3	**90**

Key: a, sitting; b, standing; c, walking; d, running; e, ascending stairs; f, descending stairs.

SVM case. VARD with SAT represented the minimum power consumption and consumed 3% more power than with no HAR. Finally, VARD with daily activities showed a reduction of 36% in energy consumption compared to typical SVM. The increase in energy efficiency compared to typical SVM was computed by (power (Typical SVM) − power (type))/power (Typical SVM). The increase in efficiency was about 44.23% for VARD with dynamic activity only, about 78.85% for VARD with static activity only, and about 69.23% for VARD with daily activities.

6.2. Human Activity Recognition Accuracy. The confusion matrix in Table 5 represents HAR errors for a real dataset (six activities × 100 samples). The confusion matrix shows that 5% of "walking" was misclassified as "ascending stairs" and 6% for opposite misclassification. Also, 8% of "sitting" was misclassified as "standing" and 5% for the opposite misclassification. The experimental results showed that the average HAR accuracy was 92.17%. If the activities "sitting" and "standing" are unified into a relaxation activity, the HAR accuracy for an SAT would be 99.5%.

7. Conclusions

Conventional HAR using the built-in accelerometer in smart mobile devices still has high power consumption due to not only the sensor itself but also the accompanying CPU computation overhead. Inspired by such challenge, we presented a new approach for energy-efficient real-time HAR on smart mobile devices. The experimental results showed that our method can achieve greater than 64% average energy-saving as compared to conventional HAR (SVM). We also showed that the average HAR accuracy was about 92% with six different activities. Moreover, we reported on how the SF, WS, and FVD alter the battery power consumption behavior with HAR. This report may be helpful to the field of HAR. However, if the Sleeping State persists for a long time, sudden human activities such as a fall cannot be recognized properly. In order to solve this problem, future work on improving the accuracy for recognizing sudden activity changes is needed.

Competing Interests

The authors declare that they have no competing interests.

References

[1] H. Yan, H. Huo, Y. Xu, and M. Gidlund, "Wireless sensor network based E-health system-implementation and experimental results," *IEEE Transactions on Consumer Electronics*, vol. 56, no. 4, pp. 2288–2295, 2010.

[2] M. V. Albert, S. Toledo, M. Shapiro, and K. Kording, "Using mobile phones for activity recognition in Parkinson's patients," *Frontiers in Neurology*, vol. 3, article 158, 2012.

[3] L. Tang, X. Zhou, Z. Yu, Y. Liang, D. Zhang, and H. Ni, "MHS: a multimedia system for improving medication adherence in elderly care," *IEEE Systems Journal*, vol. 5, no. 4, pp. 506–517, 2011.

[4] A. Anjum and M. U. Ilyas, "Activity recognition using smartphone sensors," in *Proceedings of the IEEE 10th Consumer Communications and Networking Conference (CCNC '13)*, pp. 914–919, January 2013.

[5] J. Wang, Z. Zhang, B. Li, S. Lee, and R. S. Sherratt, "An enhanced fall detection system for elderly person monitoring using consumer home networks," *IEEE Transactions on Consumer Electronics*, vol. 60, no. 1, pp. 23–29, 2014.

[6] M.-W. Lee, A. M. Khan, and T.-S. Kim, "A single tri-axial accelerometer-based real-time personal life log system capable of human activity recognition and exercise information generation," *Personal and Ubiquitous Computing*, vol. 15, no. 8, pp. 887–898, 2011.

[7] S. Abbate, M. Avvenuti, F. Bonatesta, G. Cola, P. Corsini, and A. Vecchio, "A smartphone-based fall detection system," *Pervasive and Mobile Computing*, vol. 8, no. 6, pp. 883–899, 2012.

[8] Y. He and Y. Li, "Physical activity recognition utilizing the built-in Kinematic sensors of a smartphone," *International Journal of Distributed Sensor Networks*, vol. 2013, Article ID 481580, 10 pages, 2013.

[9] J. W. Lockhart, T. Pulickal, and G. M. Weiss, "Applications of mobile activity recognition," in *Proceedings of the 14th International Conference on Ubiquitous Computing (UbiComp '12)*, pp. 1054–1058, Pittsburgh, Pa, USA, September 2012.

[10] H. Lu, J. Yang, Z. Liu, N. D. Lane, T. Choudhury, and A. T. Campbell, "The Jigsaw continuous sensing engine for mobile phone applications," in *Proceedings of the 8th ACM International Conference on Embedded Networked Sensor Systems (SenSys '10)*, pp. 71–84, Zurich, Switzerland, November 2010.

[11] G. Raffa, J. Lee, L. Nachman, and J. Song, "Don't slow me down: bringing energy efficiency to continuous gesture recognition," in *Proceedings of the 14th IEEE International Symposium on Wearable Computers (ISWC '10)*, pp. 1–8, Seoul, Republic of Korea, October 2010.

[12] Y. Wang, J. Lin, M. Annavaram et al., "A framework of energy efficient mobile sensing for automatic user state recognition," in *Proceedings of the 7th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys '09)*, pp. 179–192, Kraków, Poland, June 2009.

[13] Q. V. Vo, M. T. Hoang, and D. Choi, "Personalization in mobile activity recognition system using K-medoids clustering algorithm," *International Journal of Distributed Sensor Networks*, vol. 2013, Article ID 315841, 12 pages, 2013.

[14] Q. V. Vo, M. T. Hoang, and D. Choi, "Adaptive energy-saving strategy for activity recognition on mobile phone," in *Proceedings of the 12th IEEE International Symposium on Signal Processing and Information Technology (ISSPIT '12)*, pp. 95–100, Ho Chi Minh City, Vietnam, December 2012.

[15] Z. Yan, V. Subbaraju, D. Chakraborty, A. Misra, and K. Aberer, "Energy-efficient continuous activity recognition on mobile phones: an activity-adaptive approach," in *Proceedings of the 16th International Symposium on Wearable Computers (ISWC '12)*, pp. 17–24, Newcastle, UK, June 2012.

[16] Y. Liang, X. Zhou, Z. Yu, and B. Guo, "Energy-efficient motion related activity recognition on mobile devices for pervasive healthcare," *Mobile Networks and Applications*, vol. 19, no. 3, pp. 303–317, 2014.

[17] L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data," in *Pervasive Computing: Second International Conference, PERVASIVE 2004, Linz/Vienna, Austria, April 21–23, 2004. Proceedings*, vol. 3001 of *Lecture Notes in Computer Science*, pp. 1–17, Springer, Berlin, Germany, 2004.

[18] N. Kern, B. Schiele, and A. Schmidt, "Recognizing context for annotating a live life recording," *Personal and Ubiquitous Computing*, vol. 11, no. 4, pp. 251–263, 2007.

[19] A. M. Khan, Y.-K. Lee, S. Y. Lee, and T.-S. Kim, "A triaxial accelerometer-based physical-activity recognition via augmented-signal features and a hierarchical recognizer," *IEEE Transactions on Information Technology in Biomedicine*, vol. 14, no. 5, pp. 1166–1172, 2010.

[20] I. C. Gyllensten and A. G. Bonomi, "Identifying types of physical activity with a single accelerometer: evaluating laboratory-trained algorithms in daily life," *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 9, pp. 2656–2663, 2011.

[21] J.-H. Hong, J. Ramos, and A. K. Dey, "Toward personalized activity recognition systems with a semipopulation approach," *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 1, pp. 101–112, 2016.

[22] V. Könönen, J. Mäntyjärvi, H. Similä, J. Pärkkä, and M. Ermes, "Automatic feature selection for context recognition in mobile devices," *Pervasive and Mobile Computing*, vol. 6, no. 2, pp. 181–197, 2010.

[23] M. Khan, S. I. Ahamed, M. Rahman, and R. O. Smith, "A feature extraction method for real time human activity recognition on cell phones," in *Proceedings of the 3rd International Symposium on Quality of Life Technology (isQoLT '11)*, Toronto, Canada, 2011.

[24] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," *ACM SIGKDD Explorations Newsletter*, vol. 12, no. 2, pp. 74–82, 2011.

[25] C. Torres-Huitzil and M. Nuno-Maganda, "Robust smartphone-based human activity recognition using a tri-axial accelerometer," in *Proceedings of the 6th IEEE Latin American Symposium on Circuits and Systems (LASCAS '15)*, pp. 1–4, February 2015.

[26] M. F. A. bin Abdullah, A. F. P. Negara, M. S. Sayeed, D. J. Choi, and K. S. Muthu, "Classification algorithms in human activity recognition using smartphones," *International Journal of Computer and Information Engineering*, vol. 6, pp. 77–84, 2012.

[27] O. D. Incel, M. Kose, and C. Ersoy, "A review and taxonomy of activity recognition on mobile phones," *BioNanoScience*, vol. 3, no. 2, pp. 145–171, 2013.

[28] S. Wang, J. Yang, N. Chen, X. Chen, and Q. Zhang, "Human activity recognition with user-free accelerometers in the sensor networks," in *Proceedings of the International Conference on Neural Networks and Brain (ICNNB '05)*, pp. 1212–1217, Beijing, China, October 2005.

[29] S. J. Preece, J. Y. Goulermas, L. P. J. Kenney, and D. Howard, "A comparison of feature extraction methods for the classification of dynamic activities from accelerometer data," *IEEE Transactions on Biomedical Engineering*, vol. 56, no. 3, pp. 871–879, 2009.

[30] Y. E. Ustev, O. D. Incel, and C. Ersoy, "User, device and orientation independent human activity recognition on mobile phones: challenges and a proposal," in *Proceedings of the 2013 ACM Conference on Ubiquitous Computing (UbiComp '13)*, pp. 1427–1435, Zurich, Switzerland, September 2013.

[31] Z.-Y. He and L.-W. Jin, "Activity recognition from acceleration data using AR model representation and SVM," in *Proceedings of the 7th International Conference on Machine Learning and Cybernetics (ICMLC '08)*, pp. 2245–2250, Kunming, China, July 2008.

[32] Y. Xue and L. Jin, "A naturalistic 3D acceleration-based activity dataset & benchmark evaluations," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SMC '10)*, pp. 4081–4085, Istanbul, Turkey, October 2010.

[33] D. Jones, "Decimation-in-time (DIT) radix-2 FFT," *Connexions*, vol. 15, p. 2006, 2006.

[34] C. Chang and C. Lin, "LIBSVM: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 1–27, 2011.

[35] PowerTutor, "A Power Monitor for Android-Based Mobile Platforms," http://ziyang.eecs.umich.edu/projects/powertutor/.

[36] C. V. C. Bouten, K. T. M. Koekkoek, M. Verduin, R. Kodde, and J. D. Janssen, "A triaxial accelerometer and portable data processing unit for the assessment of daily physical activity," *IEEE Transactions on Biomedical Engineering*, vol. 44, no. 3, pp. 136–147, 1997.

Hindawi Publishing Corporation
Mobile Information Systems
Volume 2016, Article ID 3727865, 11 pages
http://dx.doi.org/10.1155/2016/3727865

Research Article

Dynamic Vehicular Route Guidance Using Traffic Prediction Information

Kwangsoo Kim,[1] Minseok Kwon,[2] Jaegeun Park,[3] and Yongsoon Eun[3]

[1]*Hanbat National University, Daejeon 34158, Republic of Korea*
[2]*The Rochester Institute of Technology, Rochester, NY, USA*
[3]*DGIST, Daegu 42988, Republic of Korea*

Correspondence should be addressed to Yongsoon Eun; yeun@dgist.ac.kr

Received 10 December 2015; Revised 8 April 2016; Accepted 31 May 2016

Academic Editor: Qixin Wang

We propose a dynamic vehicular routing algorithm with traffic prediction for improved routing performance. The primary idea of our algorithm is to use real-time as well as predictive traffic information provided by a central routing controller. In order to evaluate the performance, we develop a microtraffic simulator that provides road networks created from real maps, routing algorithms, and vehicles that travel from origins to destinations depending on traffic conditions. The performance is evaluated by newly defined metric that reveals travel time distributions more accurately than a commonly used metric of mean travel time. Our simulation results show that our dynamic routing algorithm with prediction outperforms both Static and Dynamic without prediction routing algorithms under various traffic conditions and road configurations. We also include traffic scenarios where not all vehicles comply with our dynamic routing with prediction strategy, and the results suggest that more than half the benefit of the new routing algorithm is realized even when only 30% of the vehicles comply.

1. Introduction

Recent data show that traffic conditions in metropolitan areas continue to worsen with increased wasted hours, extra fuel cost, and travel unreliability, for example, the extra time needed to arrive at destinations [1]. Intelligent Transportation Systems (ITS) attempt to solve this problem by exploiting the advances in information technology, for example, dynamically controlling traffic lights based on traffic conditions and routing vehicles using current and historical traffic information.

Intelligent Transportation System is one of the major applications of the Internet of Things (IoT) technology, which measures traffic conditions using various sensors including cameras, loop detectors, and mobile devices, and collects traffic-related data through communication channels and then shares the data among vehicles on the road and road facilities, for example, traffic lights and ramp meterings. As several commercial platforms supporting the IoT architecture have come into the market [2, 3] and wireless communication technologies have been advanced, a car has become a sensor platform and an essential element of Internet of Vehicles, a representative instantiation of the IoT [4]. The technology development and large-scale employment of the IoT introduce new services and also make the existing services more reliable and more sophisticated. One example is intelligent vehicle navigation service which computes the least travel time path or the most economical path based on real-time traffic information rather than the traditional shortest path.

Specifically, we consider in this paper the scenario where current traffic information is collected by the central vehicular routing system using in-car navigation systems, traffic sensors deployed in the road network, and other applications. The routing system then computes the shortest path for a given origin-destination (OD) pair at the request of an individual vehicle and sends the route to each vehicle. Such a system may reroute the path as updates of traffic conditions continue to become available [5, 6].

For such a central vehicular routing system to operate efficiently, it is critical to incorporate predictive traffic

information. Otherwise, traffic routing would suffer from the instability problem; that is, a road segment with little traffic can quickly turn into a heavily congested segment as many vehicles are routed or rerouted through this road segment at the same time. This is because the system computes for each vehicle its shortest path independently using the current traffic condition without taking into account other vehicles routed concurrently. In other words, instability arises primarily because every vehicle responds to the same traffic conditions with lack of knowledge about other vehicles.

In this work, we show that predictive traffic information is important to improve road efficiency. We develop a routing algorithm called Dynamic with prediction that periodically reroutes all vehicles based on current and predictive traffic conditions. The central routing system collects all the routing requests from the vehicles arriving at any nodes (intersections) and randomizes the order of priority of the requests. From the highest priority order to the lowest, the central routing system computes new routes considering the real-time traffic information and anticipatory traffic changes on the links which will be occupied by the vehicles with the higher priorities. We compare Dynamic with prediction with two other routing strategies, namely, Static and Dynamic. In the Static routing, vehicles follow the routes computed initially without changes, and in Dynamic, vehicles are rerouted periodically like Dynamic with prediction, but only using the current traffic conditions.

We develop a microlevel traffic simulator and use it to evaluate the performance of the Dynamic with prediction routing. The simulator consists of a map creator, a traffic generator, a routing controller, a vehicle simulator, and a performance evaluator. Although several microtraffic simulators are available commercially or for research purposes (e.g., VISSIM [7], VISSUM, and CORSIM [8]), they are not suitable to test our routing algorithms mainly due to their rigid routing policy. We use OpenStreetMap [9] to create a realistic map and parse the map to extract the information that we need for simulation.

Mean travel time, which is the average time taken for all vehicles to move from the origin to the destination, is popularly used for performance evaluation [10, 11]. This metric, however, does not reveal travel time variations effectively that may be more important for individual driver experience. For example, mean travel time can be similar in two vastly different cases with respect to travel time distributions. We develop a new metric that captures the travel time distribution of entire vehicles and evaluate the performance based on this metric. Our results indicate that Dynamic with prediction effectively reduces travel time in comparison to the Static and Dynamic routings under a variety of traffic conditions.

The rest of this paper is organized as follows. Section 2 formulates our problem and discusses related work. Section 3 discusses details of our simulator, and Section 4 presents the three routing algorithms that we develop and compare. Section 5 defines metrics for performance evaluation. Section 6 presents results and their analysis followed by conclusions and future work in Section 7.

2. Related Work

Vehicular route guidance deals with the problem of assigning an optimal path to each vehicle from its origin to the destination. The optimality in this problem is defined on several criteria, for example, the shortest path, the shortest time, and the least usage of local paths. The traditional routing algorithms embedded in vehicle navigation systems used only road network features and have evolved to consider real-time traffic information in part thanks to broadcasting networks such as DAB in Europe and DMB in Korea [12, 13]. Recent navigation software runs on mobile platforms and receives route updates periodically from a central telecommunication center over the cellular network [14, 15]. While both real-time traffic information and historical data are used to compute those route updates, the effects of individual routing decisions on overall system stability are not considered and well studied.

In the literature, there have been attempts to design route guidance strategies that effectively find shortest paths for given OD pairs while achieving stability when road networks are large and dynamically change. Claes et al. [10] develop a decentralized routing guidance system for anticipatory vehicle routing in which vehicles are routed based on current as well as forecast traffic information. Their system is modeled after the food foraging mechanism in ant colonies using pheromones. Multiagents are deployed in vehicles, infrastructure, and the central server, and they collect and communicate traffic information to compute the best path for a given OD pair. Most literature on dynamic routing systems did not show the details of how traffic prediction is obtained or how routing algorithm deals with traffic predictions. There are some dynamic routing algorithms receiving a traffic prediction as an input, but the prediction is based on a traffic flow model or traffic history [16, 17].

Most commonly used routing algorithms are Dijkstra's and A^* algorithms. One fundamental problem with them is their prohibitively high time complexity when the number of nodes (intersections) in the network increases. Jagadeesh et al. [18] use hierarchical routing to reduce this time complexity and propose a simple heuristic method that compensates the loss of accuracy in route quality, which is inevitable in hierarchical routing. Motivated by the same problem, Song and Wang [19] also use hierarchical routing but rather focus on scalability by reducing heavy precomputation, storage, and querying costs. They use recently discovered aspects of network topology, specifically hierarchical communities, to decompose the network and to design a heuristic for fast search. While lowering the computational complexity of routing algorithms is important, rerouting as the traffic conditions dynamically change is another critical challenge. There have been several attempts to tackle such a problem using different approaches including dynamic programming, genetic algorithms, and hierarchical routing [20–22]. Kim et al. show that dynamic route determination can be modeled as a Markov deception process and propose procedures for identifying unnecessary traffic data that can be removed for route decision making. Using their approach, we can selectively use only a subset of vast real-time traffic data in

route selection; otherwise, dealing with all of the incoming data will be computationally challenging.

Forecasting traffic conditions has been investigated as an important problem in ITS research. Lv et al. and Huang et al. adopt a deep learning approach and architecture [23, 24] to predict traffic flows. Abadi et al. [25] propose a traffic flow prediction algorithm with current demands, historical data, and limited real-time data based on an estimated dynamic origin-destination matrix and simulations. Recent work on traffic flow prediction or forecasting can be found in [23, 26]. To the best of our knowledge, however, the traffic prediction methods used by the work in the literature do provide system-wide optimal route guidance.

3. Traffic Simulator

In this section, we discuss our approach for creating a map, running an experiment, and modeling roads and vehicle movement in the simulator.

3.1. Map Creation. The first step in a simulation is to create a map where a vehicle moves from the source to the destination passing through intersections. We consider the map as a graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is a set of nodes and $\mathcal{E}$ is a set of edges. A node n_i denotes an intersection where traffic light systems are in operation, and an edge e_{ij} denotes a road segment from n_i to n_j in the road network. Our simulator reads the map as an image and converts it into an adjacency linked list, which is a data structure popularly used to represent a graph. A major challenge is to automate this map creation process, so that the user does not need to manually create a complex map, which is tedious and time consuming; for example, the user needs to enter map information manually with CORSIM [8] and VISSIM [7]. We use OpenStreetMap [9] for this purpose. With the help of OpenStreetMap, we can grab an area of interest of the map image and convert that into an XML file. We then parse the XML file filtering unnecessary information (e.g., stores and gas stations) leaving only the information that we need such as intersections and road segments between intersections. We can extract the properties of each link and node including the number of lanes, the distance of a link, and even speed limits. After parsing, our preprocessor converts the simplified XML file into a target adjacency linked list. The preprocessing steps for map creation are summarized in Figure 1.

In our simulations, we use a real map topology instead of artificially made ones, namely, the Greater Rochester area in New York, the United States, as shown in Figure 2.

3.2. Simulator Structure. Our simulator runs on a discrete time basis, which is modeled as time tick. Events occur at each time tick such as vehicles movement, turning at intersections, recomputing their routes, changing their next move, departure from the origins, and arrival at the destinations. Of course, the time tick value that represents the wall clock time is parametrized to be changed according to the time scale in a simulation, for example, 1 tick as 1 second in our experiments. The overall simulator structure is illustrated in Algorithm 1, where $R(t)$ is the set of vehicles on the road at time t and $\mathbb{N}(R(t))$ is the number of elements in $R(t)$.

3.3. Modeling Roads and Vehicle Movement. An intersection and the street between two intersections are modeled as a node and a link in the graph, respectively. A vehicle arriving at the intersection can proceed in three directions; namely, it can turn left, go straight, and turn right. In the real world, the vehicle goes straight or turns left/right following the traffic lights, while no traffic light control exists in the simulator. Each direction maintains a queue where incoming vehicles await their turn. Since no traffic light control is provided, the vehicles in the queues can move to their next road segment at every tick. However, if the next road segment experiences traffic congestion, that is, there is no room for a vehicle to enter, the vehicle needs to wait leading to delay at the intersection. The delay in this case is the time elapsed from when the vehicle arrives at the queue to when the vehicle departs the intersection.

A road segment has three primary properties: distance, the number of lanes, and capacity. The capacity of the road segment is defined as the number of vehicles in transit and can be computed as the product of distance and the number of lanes. Different road segments have different capacities. For example, highways have high capacity mainly due to long distance, road segments in downtown in a metropolitan area have high capacity due to the high number of lanes, and local roads have low capacity due to short distance and the small number of lanes.

Several car-following models exist in the literature [27], and these models can be used to keep track of the position of each vehicle on the road. For simplicity, however, we use a vehicle moving rule instead based on traffic conditions. The velocity of a vehicle v on a road segment is determined by

$$v = \begin{cases} v_{\max} & \rho \leq \rho_{\text{th}} \\ \dfrac{v_{\min} - v_{\max}}{1 - \rho_{\text{th}}} \left(\rho - \rho_{\text{th}}\right) + v_{\max} & \text{otherwise,} \end{cases} \tag{1}$$

where $v_{\max}$ and $v_{\min}$ are the speed limit and the minimum speed of the segment, respectively, ρ_{th} is a constant between 0 and 1, and ρ is the ratio between the number of current vehicles and the maximum number of vehicles that the road segment can accommodate denoted by $N_{\max}$, which is in turn computed as

$$N_{\max} = \frac{dL}{l_{\text{car}} + h} \tag{2}$$

in which d is the length of the road segment, L is the number of lanes, l_{car} is the average length of a vehicle, and h is the average headway. An example of a vehicle's velocity is shown in Figure 3.

4. Routing Controller

Using the map, roads, and vehicle movement defined earlier, we design a routing controller in this section. We first discuss how travel time is estimated and then provide details of our route guidance algorithms.

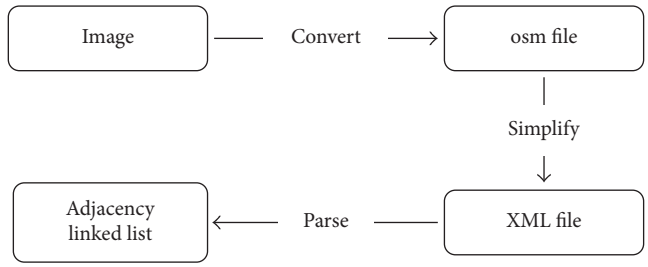


Figure 1: Preprocessing steps for map creation.

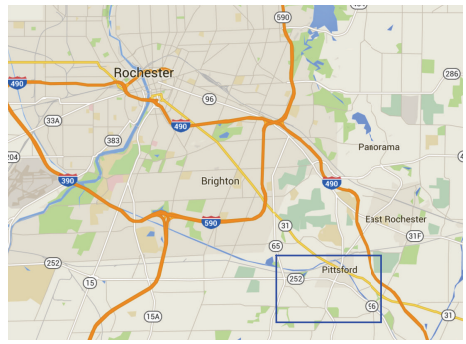


Figure 2: A map of the Great Rochester area in New York, the United States.

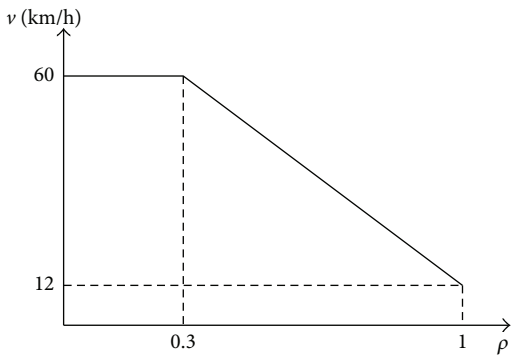


Figure 3: Example of a vehicle's velocity curve ($v_{max} = 60$ km/h, $v_{min} = 12$ km/h, and $\rho_{th} = 0.3$).

4.1. Travel Time Estimation. Two types of delay arise as a vehicle travels: (1) delay to travel from one intersection to another and (2) delay waiting at an intersection. The latter is hard to measure because it depends on traffic signals with probabilistic characteristics. In this research, we focus only on delay on road segments (the former, also known as link delay) for route guidance, and the latter will be dealt with in the future work. Specifically, we estimate delay d_{ij} on link e_{ij} as follows:

$$d_{ij} = \frac{l_{ij}}{v_{ij}}, \tag{3}$$

where l_{ij} is the length of e_{ij}, that is, the distance from n_i to n_j, and v_{ij} is the velocity of a vehicle on the link calculated by (1).

4.2. Route Guidance. With this estimated link delay, our route guidance mechanism directs each vehicle to its destination along a high-quality path. We assume that a central server possesses the entire map information including vehicle positions, delays at both intersections, and road segments and computes paths for all vehicles. Before a vehicle leaves from the origin, the vehicle sends its routing request to the server, which in turn computes the shortest path from the origin to the destination. The vehicle may be rerouted in the middle of travel. We use a well-known single-source shortest path algorithm like the Dijkstra or Bellman-Ford algorithms [28] for routing.

We formally define network state $X(t)$ as a column vector of all link delays. Let us consider a road network with three vertices, n_1, n_2, and n_3, and the corresponding edges e_{12}, e_{23}, and e_{13} as shown in Figure 4. Then the state at time t is $X(t) = [d_{12}(t)\ \ d_{21}(t)\ \ d_{23}(t)\ \ d_{32}(t)\ \ d_{13}(t)\ \ d_{31}(t)]^T$, where $d_{ij}(t)$ is the delay from n_i to n_j as a function of time representing the dynamic nature of the traffic network. The three routing strategies that we use and compare for route guidance are described in the following in detail.

(i) Static Routing. For a vehicle with an OD pair generated at time t, we find the shortest path from the origin to the destination using $X(t)$. This vehicle does not change the route until it reaches the destination.

(ii) Dynamic Routing. Unlike static routing, we reroute all vehicles periodically. When a vehicle arrives at any intersections at time t, it is rerouted based on the current traffic conditions which is represented by $X(t)$. This rerouting

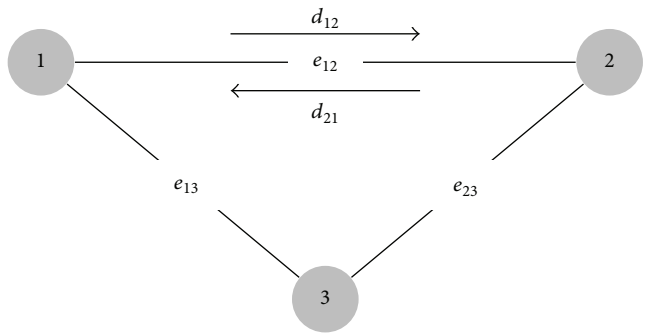


Figure 4: An example of simple road network and notations.

(1) $t \leftarrow 0$ & Continue $\leftarrow$ TRUE
(2) empty $R(t)$
(3) $N \leftarrow$ Number of vehicles generated at each tick
(4) **while** Continue = TRUE **do**
(5) **if** Vehicles to be generated remain **then**
(6) Generate N vehicles
(7) Update $R(t)$ to include generated vehicles
(8) Assign OD pairs randomly for N vehicles
(9) Get a route guidance for N vehicles
(10) **end if**
(11) **for** $i \leftarrow 1, \mathbb{N}(R(t))$ **do**
(12) Update position
(13) **if** Arrive at the destination **then**
(14) Store time t & Remove from $R(t)$
(15) **else**
(16) **if** Dynamic routing algorithm & Arrive at a node **then**
(17) Get a new direction for the next link
(18) **end if**
(19) **end if**
(20) **end for**
(21) **if** $R(t)$ is empty **then** Continue = FALSE
(22) **end if**
(23) $t \leftarrow t + \Delta t$
(24) **end while**

Algorithm 1

repeats at every intersection through the journey until the vehicle reaches its destination.

(iii) Dynamic Routing with Prediction. One potential drawback of dynamic routing is its limited knowledge on future network states since it only utilizes current traffic information when rerouting vehicles. So, all vehicles arriving at the same intersection with the same destination will be guided into same links, which may lead to traffic jam on those links. One way to mitigate this problem is to consider the network states at time $t + k$, $k > 0$, when vehicles are routed at time t. In other words, dynamic routing is extended to incorporate predictive information. Unlike many other researches on dynamic routing algorithm in which traffic conditions are predicted statistically with a long time horizon, for example, 30 minutes or 1 hour, our proposed algorithm corresponds to the case when $k = 1$. Prediction over multiple ticks (which is proportional to the prediction time horizon) is not necessary because we reroute all vehicles at every tick. There is no guarantee that vehicles will travel as predicted after multiple ticks elapse. They will receive new routes calculated using updated traffic condition.

At time t, a priority order is randomly generated for each vehicle arriving at intersections. Let k be the number of vehicles to be rerouted at time t. We define an order function $I(\cdot)$ that maps integers 1 to k to randomly arranged k vehicle IDs. The vehicle with ID $I(1)$ is rerouted first, the vehicle with $I(2)$ is rerouted second, and then the vehicle with $I(3)$ is rerouted. The random function $I(\cdot)$ is different at every time t.

When the first vehicle is rerouted to one of the links connected to the current node, this link will be occupied by the vehicle at time $t + 1$. This prediction is taken into account when the second vehicle is rerouted. After the second

(1) Update $X(t)$ & $C(t)$
(2) $k \leftarrow$ the number of vehicles arriving at some nodes
(3) $X_p(1) \leftarrow X(t)$
(4) Create an order function I with integers 1 to n randomly arranged
(5) **for** $i \leftarrow 1, k$ **do**
(6) Get a routing guidance for vehicle with $I(i)$ based on $X_p(i)$
(7) Update $X_p(i+1)$ including vehicle with $I(i)$'s route
(8) **end for**

Pseudocode 1

rerouting is completed, another change is predicted on the link into which the second vehicle is guided. The change caused by the second vehicle as well as the first vehicle is taken into consideration when the third vehicle is rerouted, and this process continues until all k vehicles are rerouted. This means that the road segments which will be occupied by the already rerouted vehicles are penalized when the next vehicles are rerouted, so the penalized roads have lower chances to be selected by the vehicles rerouted later.

The pseudocode for dynamic routing with prediction is described in Pseudocode 1, where $C(t)$ is the set of all the vehicles arriving at some nodes, $X(t)$ is the state vector representing the current traffic condition, and $X_p(i)$ is a temporary state vector that takes into account the predictive future traffic condition caused by previous rerouted vehicles with $I(n), n = 1, \ldots, i-1$.

5. Performance Metrics

Most intelligent traffic control systems use the average time elapsed from departure to arrival for all vehicles. Specifically, the metric widely used is given by

$$\frac{1}{N}\sum_{i=1}^{N} T_m(i), \tag{4}$$

where N is the number of vehicles and $T_m(i)$ is the measured travel time of the ith vehicle. This measure, however, does not provide traffic conditions that individual vehicles experience accurately. For an example, assume that three vehicles traveled as shown in Table 1. The average travel time for three vehicles is 1 hour. However, Vehicle #1 experienced heavier traffic, compared with Vehicle #2 and Vehicle #3. It can be told that Vehicle #2 took a journey without any traffic interrupts. However, the metric like (4) does not inform such individual experiences.

Motivated by this drawback, we develop a new performance metric that evaluates traffic conditions more accurately. Our metric is defined as

$$M(\alpha) = \text{Percentage of the vehicles that satisfy } T_m \le (1+\alpha)\, T_e, \tag{5}$$

where T_m is the measured travel time of a vehicle, α is a parameter, and T_e is the expected shortest travel time of the vehicle, that is, the time taken by the vehicle to travel from the origin to its destination along the shortest route without any traffic interruptions.

Table 1: A simple example of vehicle travel time.

Vehicle ID	Estimated shortest travel time	Measured travel time
1	1/6 of an hour	1 hour
2	1 hour	1 hour
3	1/2 hour	1 hour

One interpretation of this metric is that the higher $M(\alpha)$ is, the better the performance is, for a fixed α. For example, if routing algorithm A gives a higher percentage of vehicles whose measured travel time is less than or equal to 120% of the expected shortest travel time than routing algorithm B does, then more vehicles by algorithm A than algorithm B finish their travels in the 20% extended time. Note that the conventional average travel time metric like (4) does not tell about such a distribution of travel time.

Of course, instead of the percentage value of the vehicles satisfying (5), we can compute α as the percentage values change. In such a case, for a fixed percentage value, smaller α values imply better performance. For instance, algorithm A performs better than algorithm B if algorithm A gives 50 vehicles satisfying (5) with $\alpha = 0.1$ and algorithm B gives 50 vehicles with $\alpha = 0.2$ for a total of 100 vehicles. In this case, the vehicles following algorithm A travel in less time than those following algorithm B for the same number of vehicles.

For ease of computation, we define $\beta(i)$ for a vehicle i as

$$\beta(i) = \frac{T_m(i)}{T_e(i)} - 1. \tag{6}$$

With $\beta(i)$, $M(\alpha)$ is now computed as

$$M(\alpha) = \frac{\mathbf{N}(\{i \mid \beta(i) \le \alpha\})}{\#\text{ of vehicles routed}} \times 100. \tag{7}$$

In addition, we define

$$\beta_m = \frac{1}{N}\sum_{i=1}^{N} \beta(i). \tag{8}$$

Note that β_m is similar to the average travel time in (4) as large (small) β_m corresponds to large (small) average travel

Table 2: Comparison of $M(\alpha)$ values for the routing strategies when α = 0.1, 0.2, 5, and 10 and vehicles are generated at rate = 1, 2, 5, 7, and 10 per sampling time.

Gen. rate	Algorithms	$M(0.1)$	$M(0.2)$	$M(5)$	$M(10)$
1	Static	74.32	98.85	100.00	100.00
	Dynamic	74.32	98.85	100.00	100.00
	Dynamic/prediction	73.01	97.20	99.99	100.00
2	Static	69.76	98.68	100.00	100.00
	Dynamic	69.76	98.68	100.00	100.00
	Dynamic/prediction	68.37	93.83	99.98	100.00
5	Static	52.14	67.59	84.10	95.51
	Dynamic	52.17	67.59	84.10	95.51
	Dynamic/prediction	53.75	74.54	99.19	99.95
7	Static	48.53	66.83	78.19	88.66
	Dynamic	48.60	67.01	78.18	88.64
	Dynamic/prediction	47.48	72.57	93.32	98.60
10	Static	38.52	55.90	75.325	84.65
	Dynamic	35.76	52.94	77.71	86.57
	Dynamic/prediction	34.26	51.25	91.20	96.68

time. Moreover, a large β_m suggests heavy traffic conditions during simulation, about which a simple average travel time does not give any information.

6. Simulation Results

We compare the performance of the routing strategies discussed in Section 4 using the developed traffic simulator. The results are analyzed based on the metrics proposed in Section 5.

6.1. Vehicle Generation Rates. We first examine which routing strategy minimizes $M(\alpha)$ for different vehicle generation rates. For this experiment, we use a part of the Rochester map with 337 intersections and a total of 20,000 vehicles generated. The vehicle generation rate varies from one to 10 every sampling time. The simulation begins when the first vehicle departs the origin and ends when all the vehicles generated arrive at their destinations. The results are summarized in Table 2. As an example, $M(0.1)$ when Dynamic with prediction is used is 53.75% when the vehicle generation rate is 5. This means that 53.75% of the vehicles arrive at the destinations with $\beta(i) \leq 0.1$, that is, less than 110% of the shortest expected travel time. The results show that the values of $M(\alpha)$ decrease in general as the vehicle generation rate becomes high (from 1 to 10), which indicates that more vehicles arrive at their destinations late. This is because more vehicles are added to traffic over a fixed duration as the rate increases.

These results are also graphically depicted in Figure 5. Note that more vehicles arrive at their destination close to the shortest travel time with higher $M(\alpha)$ values. In the figure, all the algorithms perform similarly under light traffic conditions regardless of α, whereas Dynamic with prediction outperforms the other two under heavy traffic conditions, for example, when the vehicle generation rate is 7 or 10.

Table 3: β_m for Static, Dynamic, and Dynamic with prediction with different vehicle generation rates.

Generation rate	Algorithms	β_m
1	Static	0.087
	Dynamic	0.087
	Dynamic/prediction	0.102
2	Static	0.090
	Dynamic	0.090
	Dynamic/prediction	0.121
5	Static	1.929
	Dynamic	1.929
	Dynamic/prediction	0.455
7	Static	2.883
	Dynamic	2.888
	Dynamic/prediction	0.913
10	Static	3.707
	Dynamic	3.283
	Dynamic/prediction	1.450

In addition, we compute β_m (similar to average travel time) for the routing strategies and compare them under the same traffic conditions as Table 3. Like $M(\alpha)$, Dynamic with prediction shows the lowest β_m values under heavy traffic conditions, while all the three algorithms behave similarly when not many vehicles are on the road. With higher vehicle generation rates, β_m for the Static and Dynamic algorithms increases more abruptly than that of Dynamic with prediction, as shown in Figure 6. It can be interpreted as Dynamic with prediction results in low average travel time and low $M(\alpha)$ while being more robust to dynamically changing traffic conditions.

6.2. Map Topology. Our routing strategies are also tested with five different maps for more reliable performance evaluation.

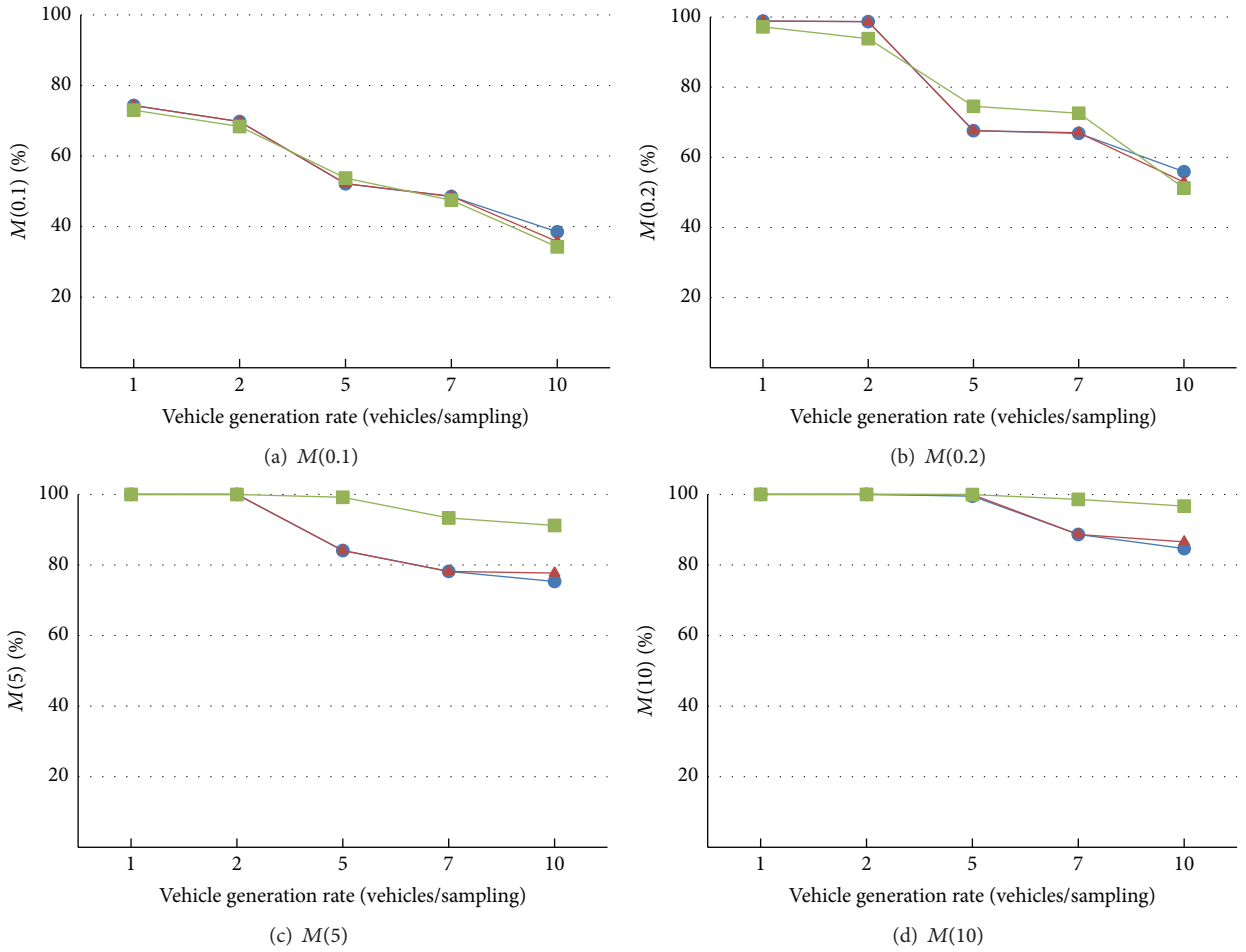


FIGURE 5: $M(\alpha)$ under various vehicle generation rates: blue = Static, red = Dynamic, and green = Dynamic with prediction.

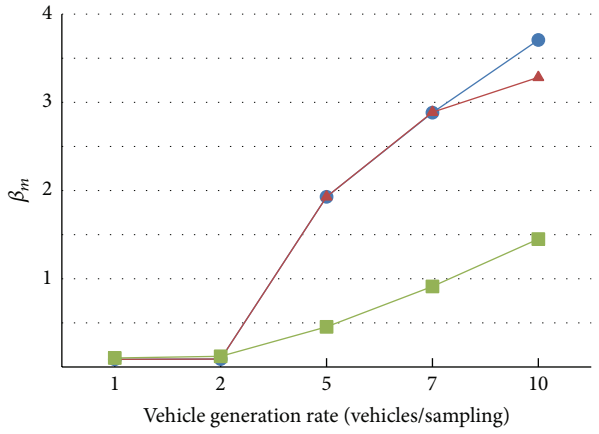
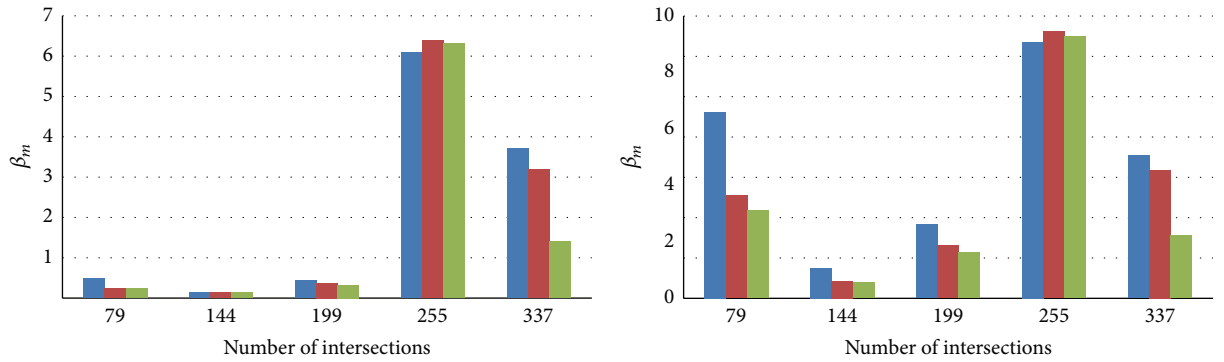


FIGURE 6: β_m under various vehicle generation rates (blue = Static, red = Dynamic, and green = Dynamic with prediction).

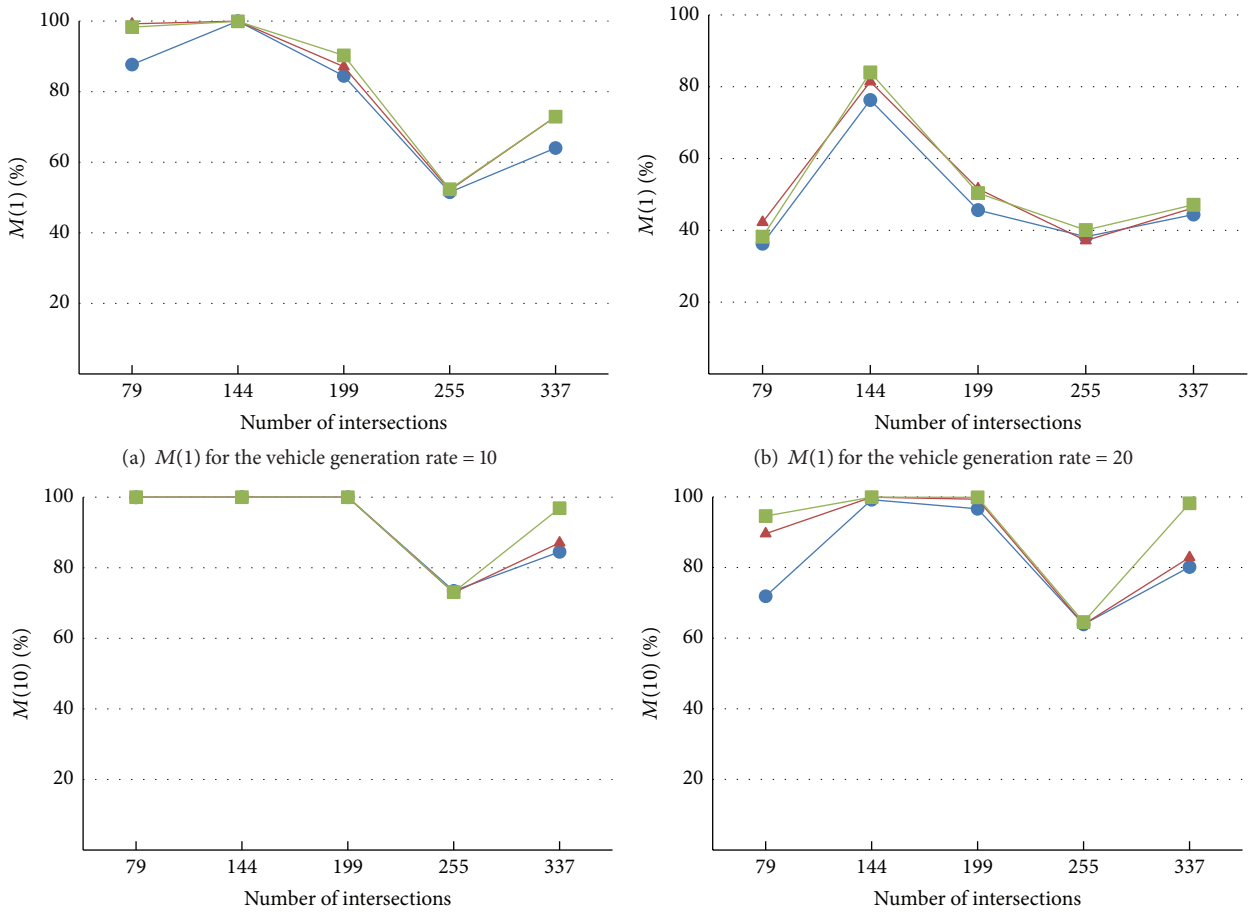
Specifically, the number of intersections in the maps is 79, 144, 199, 255, and 337, in which different areas of the Rochester map are included with various numbers of intersections. In Figure 7, the results (β_m) are plotted in both relatively light and heavy traffic cases, when the vehicle generation rate is 10 in Figure 7(a) and the rate is 20 in Figure 7(b).

In Figure 7(a), the Dynamic with prediction algorithm exhibits the lowest β_m for all the maps but the number of intersections equals 255. This trend is more pronounced in Figure 7(b) where heavy traffic occurs. We observe that all the algorithms perform poorly with high β_m and Static yields slightly lower β_m than the other two when the number of



(a) Vehicle generation rate = 10 (b) Vehicle generation rate = 20

FIGURE 7: Use of different routing strategies on various road maps (blue = Static, red = Dynamic, and green = Dynamic with prediction).



(a) $M(1)$ for the vehicle generation rate = 10 (b) $M(1)$ for the vehicle generation rate = 20

(c) $M(10)$ for the vehicle generation rate = 10 (d) $M(10)$ for the vehicle generation rate = 20

FIGURE 8: $M(\alpha)$ with different numbers of intersections (blue = Static, red = Dynamic, and green = Dynamic with prediction).

intersections equals 255. Although the reason behind this outlier is not clearly understood, we surmise that certain properties of map topology affect the performance.

The results of $M(1)$ and $M(10)$ are also plotted for the same data set in Figure 8. We see that $M(1)$ and $M(10)$ of the Dynamic with prediction algorithm are higher than the others for all the maps including the number of intersection equal to 255, and this observation is more pronounced in $M(10)$, which is under heavy traffic.

6.3. Coexisting Routing Strategies. In practice, it is unlikely that all the vehicles in a particular region adopt a new route guidance system altogether at the same time, not to mention to convince all the drivers to comply with the

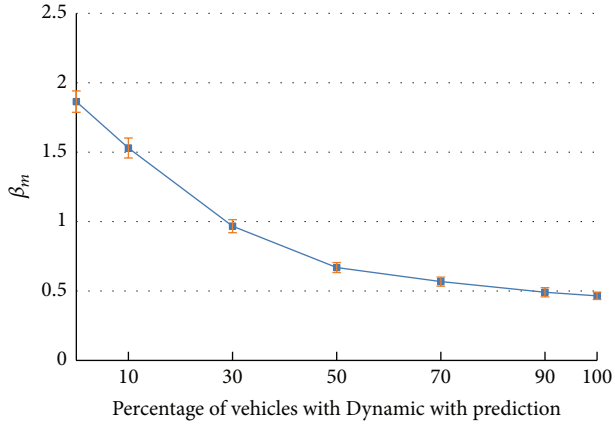


FIGURE 9: β_m as vehicles following Dynamic with prediction coexist with the ones using the Static routing and their percentage increases from 0% to 100%.

routing guidance. Hence, it is important for our system to be incrementally deployable, not to disrupt the existing system, but to gradually improve the overall performance. To get a glimpse on how well our routing strategies coexist with others, we measure β_m at different compliance levels. We define a compliance level as the percentage of the vehicles that follow a dictated routing guidance. We assume that the rest of the vehicles use the Static algorithm for their routing; that is, these vehicles drive through the best route given at departure and do not comply with rerouting decisions provided by the central routing controller. For this experiment, we use a map of 337 intersections with 20,000 vehicles generated at a rate of 5 vehicles per sampling. Note that other cases show a similar trend.

In Figure 9, the β_m results of vehicles using the Dynamic with prediction routing are illustrated as their compliance level changes from 0% to 100%. The results were obtained by 10 simulations with different random seeds. The case with 0% indicates that all vehicles follow the Static routing algorithm, whereas the case with 100% means that all vehicles comply with Dynamic with prediction. In the figure, β_m is high (a little less than 2) when the compliance level is 0% and monotonically decreases as the level increases to 100% (less than 0.5). More interestingly, β_m drops sharply at around the compliance level of 30% implying that more than half of the travel time reduction already occurs then. Nearly all travel time reduction occurs at the compliance level of 50%. This result demonstrates high potentials of the Dynamic with prediction routing algorithm, as the results imply that the algorithm can be deployed for higher road efficiency even when not all the vehicles are equipped with the new routing guidance and even if not all the drivers comply with the guidance.

7. Conclusions and Future Work

We have demonstrated that traffic routing can benefit from using predictive information as it helps reduce travel time and improve road efficiency based on simulation studies. We propose a traffic routing algorithm that utilizes both current and near-future traffic conditions using already routed vehicles. The performance of this routing algorithm is evaluated via simulations under various traffic conditions including light and heavy traffic and small to large areas of the road network. Our results show that the algorithm outperforms other more conventional ones and also successfully reduces travel time even when not all vehicles comply with the guidance of the algorithm.

This study has some limitations including the assumptions made in the simulator, for example, traffic light control at intersections. A more realistic simulation may be possible with traffic light control, car-following dynamics, lane change rules, and routing algorithms that take into account intersection delays. Such simulations will certainly help assess the proposed algorithm more accurately.

Competing Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors would like to thank Kuangwei Zhang for his valuable contribution that helped develop the simulator initially. This research was supported in part by GRL Program (2013K1A1A2A02078326) and by the BSR Program (NRF-2013R1A1A2058304) both through the NRF of Korea, in part by the IITP grant funded by the Korea government (B0101-15-0557), and in part by Hanbat National University in 2011.

References

[1] D. Schrank, B. Eisele, and T. Lomax, "TTI's 2012 urban mobility report," Tech. Rep., Texas A&M Transportation Institute, 2012.

[2] Intel, "Intel® In-Vehicle Solutions: Brief," http://www.intel.com/content/www/us/en/embedded/automotive/in-vehicle-solutions-platform-brief.html.

[3] IBM, "Internet of Things for Automotive," http://www.ibm.com/analytics/us/en/industry/automotive/iot-for-automotive/.

[4] M. Gerla, E.-K. Lee, G. Pau, and U. Lee, "Internet of vehicles: from intelligent grid to autonomous cars and vehicular clouds," in *Proceedings of the IEEE World Forum on Internet of Things (WF-IoT '14)*, pp. 241–246, Seoul, Republic of Korea, March 2014.

[5] Google, Google Maps, http://maps.google.com.

[6] LOCNALL Inc, Kimgisa Navigation, http://www.kakao.com/services/67.

[7] PTV Planung Transport, PTV VISSIM, http://vision-traffic.ptvgroup.com/en-uk/products/ptv-vissim/.

[8] McTrans Center, University of Florida, TSIS-CORSIM, http://mctrans.ce.ufl.edu/featured/tsis/.

[9] OpenStreetMap, OpenStreetMap: The Free Wiki World Map, http://www.openstreetmap.org.

[10] R. Claes, T. Holvoet, and D. Weyns, "A decentralized approach for anticipatory vehicle routing using delegate multiagent systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 2, pp. 364–373, 2011.

[11] I. Leontiadis, G. Marfia, D. MacK, G. Pau, C. Mascolo, and M. Gerla, "On the effectiveness of an opportunistic traffic management system for vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1537–1548, 2011.

[12] WorldDAB, Digital Radio in Car, http://www.worlddab.org/technology-rollout/digital-radio-in-car.

[13] Y. H. Jeong and W. W. Kim, "A novel TPEG application for location based service using terrestrial-DMB," *IEEE Transactions on Consumer Electronics*, vol. 52, no. 1, pp. 281–286, 2006.

[14] SK Planet, "T-map," http://www.tmap.co.kr.

[15] Verizon Wireless, VZ Navigator, http://www.verizonwireless.com/wcms/consumer/products/navigator.html.

[16] Z. Liang and Y. Wakahara, "Real-time urban traffic amount prediction models for dynamic route guidance systems," *EURASIP Journal on Wireless Communications and Networking*, vol. 2014, article 85, 2014.

[17] D. Park, H. Kim, C. Lee, and K. Lee, "Location-based dynamic route guidance system of Korea: system design, algorithms and initial results," *KSCE Journal of Civil Engineering*, vol. 14, no. 1, pp. 51–59, 2009.

[18] G. R. Jagadeesh, T. Srikanthan, and K. H. Quek, "Heuristic techniques for accelerating hierarchical routing on road networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 3, no. 4, pp. 301–308, 2002.

[19] Q. Song and X. Wang, "Efficient routing on large road networks using hierarchical communities," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 1, pp. 132–140, 2011.

[20] M. K. Mainali, K. Shimada, S. Mabu, and K. Hirasawa, "Optimal route of road networks by dynamic programming," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '08)*, pp. 3416–3420, June 2008.

[21] Y. Wang, S. Mabu, Q. Meng, M. K. Mainali, and K. Hirasawa, "Multiple ODs routing algorithm for traffic systems using GA," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '10)*, pp. 1–8, Barcelona, Spain, July 2010.

[22] M. K. Mainali, S. Mabu, and K. Hirasawa, "Hierarchical efficient route planning in road networks," in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC '11)*, pp. 2779–2784, Anchorage, Alaska, USA, October 2011.

[23] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: a deep learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2015.

[24] W. Huang, G. Song, H. Hong, and K. Xie, "Deep architecture for traffic flow prediction: deep belief networks with multitask learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 5, pp. 2191–2201, 2014.

[25] A. Abadi, T. Rajabioun, and P. A. Ioannou, "Traffic flow prediction for road transportation networks with limited traffic data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 653–662, 2015.

[26] E. I. Vlahogianni, M. G. Karlaftis, and J. C. Golias, "Short-term traffic forecasting: where we are and where we're going," *Transportation Research Part C: Emerging Technologies*, vol. 43, pp. 3–19, 2014.

[27] M. Treiber and A. Kesting, *Traffic Flow Dynamics: Data, Models and Simulation*, Springer, Berlin, Germany, 2013.

[28] T. H. Cormen, C. E. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*, MIT Press, Cambridge, Mass, USA, 3rd edition, 2009.

Hindawi Publishing Corporation
Mobile Information Systems
Volume 2016, Article ID 1042525, 10 pages
http://dx.doi.org/10.1155/2016/1042525

Research Article

A Dynamic Programming Solution for Energy-Optimal Video Playback on Mobile Devices

Minseok Song and Jinhan Park

School of Computer and Information Engineering, Inha University, Incheon 22212, Republic of Korea

Correspondence should be addressed to Minseok Song; mssong@inha.ac.kr

Received 28 December 2015; Revised 8 April 2016; Accepted 11 April 2016

Academic Editor: Wenyao Xu

Due to the development of mobile technology and wide availability of smartphones, the Internet of Things (IoT) starts to handle high volumes of video data to facilitate multimedia-based services, which requires energy-efficient video playback. In video playback, frames have to be decoded and rendered at high playback rate, increasing the computation cost on the CPU. To save the CPU power, dynamic voltage and frequency scaling (DVFS) dynamically adjusts the operating voltage of the processor along with frequency, in which appropriate selection of frequency on power could achieve a balance between performance and power. We present a decoding model that allows buffering frames to let the CPU run at low frequency and then propose an algorithm that determines the CPU frequency needed to decode each frame in a video, with the aim of minimizing power consumption while meeting buffer size and deadline constraints, using a dynamic programming technique. We finally extend this algorithm to optimize CPU frequencies over a short sequence of frames, producing a practical method of reducing the energy required for video decoding. Experimental results show a system-wide reduction in energy of 27%, compared with a processor running at full speed.

1. Introduction

The Internet of Things (IoT) allows physical objects to interact and cooperate with one another by exchanging data, and multimedia-related services based on the IoT are now gaining popularity in various applications areas [1]. For example, users of home security systems now see images from cameras on a smartphone, and telemedicine systems allow doctors to monitor a patient's health using video communication.

To support multimedia applications within the IoT, the characteristics of video need to be considered carefully. For example, the amount of data involved requires the use of compression techniques for codecs, but encoding and decoding processes are computationally intensive. Video transmission is a real-time process, which requires continuously periodic decoding to avoid distorted playback. Most importantly, mobile IoT devices have a limited energy budget, making the energy requirements of video transmission an important issue.

An effective way of reducing CPU power consumption is to use a dynamic voltage and frequency scaling (DVFS) technique, which adjusts the operating voltage and frequency of the processor [2–4]. Because the energy dissipated by the CPU scales quadratically with the supply voltage, reducing the voltage saves a lot of energy but also slows program execution, so that an appropriate compromise is always required.

In video playback, frames have to be decoded and rendered at playback rate to avoid a loss of quality. For example, to play a video at 25 frames per second, a frame must be decoded every 40 ms. This decoding process needs to finish within this period, but workload imposed by each frame varies significantly with video content [5–10].

In most previous work on the application of DVFS to videos, the lowest frequency that satisfies the deadline of the decoding time is chosen to reduce power consumption [11], but more energy can be saved by introducing flexibility in timing, by means of buffering techniques: if several frames are decoded in advance, the CPU can operate at lower frequencies on average, but buffering comes with its own costs [12]. Therefore, power saving is only effective with an appropriate frequency selection method subject to buffer constraints, but previous work took no account of this issue.

We propose a new scheme that determines the CPU frequency needed to decode each frame, which minimizes energy consumption while avoiding buffer overrun. We start by developing a video playback and energy model, formulate the energy optimization problem, and go on to use a dynamic programming technique to determine a sequence of frequencies. We finally present experimental results based on measurement of smartphone energy consumption and decoding times.

The rest of this paper is organized as follows. We present related work in Section 2 and the system model in Section 3. We formulate an optimization problem in Section 4, propose a new frequency selection algorithm in Section 5, and extend it in Section 6. We assess our scheme in Section 7 and finally conclude the paper in Section 8.

2. Related Work

CPU power management has been the subject of a lot of research, and most of the resulting techniques involve either dynamic power management (DPM) or DVFS. DPM puts an idle CPU into sleep mode [13], whereas DVFS reduces the voltage and frequency of an active CPU [2, 4]. DPM is not generally suitable for real-time applications that run continuously, because the idle intervals are too short to allow the CPU to enter sleep mode [8]. Therefore, we only review previous works about DVFS only in this section.

DVFS techniques can be classified into interval-based and task-based algorithms [7, 14]. Interval-based schemes monitor the CPU load at intervals and respond by changing the CPU frequency and voltage. A representative scheme is the Linux Ondemand governor, which adjusts frequency periodically based on CPU utilization in the preceding interval [15]. Another scheme is LongRun [16] which varies the frequency to suit the measured utilization. These methods are typically easy to implement but can make inaccurate predictions based on the assumption that loads are similar to recent loads [14].

Task-based schemes can overcome this problem to some extent by classifying tasks into several types to which different frequency selection policies are applied. Ayoub et al. [17] manage frequency and voltage to meet a performance target, expressed as a fraction of maximum system performance. Flautner and Mudge [18] propose a method that chooses a CPU frequency for each task based on its recent computational requirements. Seo et al. [14] present a frequency allocation method to reduce the average response time of tasks. However, all of these methods have been developed for general workloads and therefore may not be suitable for multimedia applications with real-time constraints.

DVFS techniques for real-time systems are generally integrated with real-time scheduling [2–4]. Based on the analysis of worst-case execution times, they select CPU frequencies that satisfy the real-time constraints; but tasks are often complete before their worst-case execution times, so several algorithms incorporate methods of reclaiming the unused time [2, 4]. The CPU starts each period running at a frequency which will meet the worst-case demands and the frequency is then reduced in response to the actual computation requirement.

Several groups have investigated DVFS techniques for video applications [6, 7], in which the key issue is to estimate the computational requirements of successive frames. Most of these techniques predict the workload required to decode a frame from the workloads incurred in decoding previous frames and adjust the CPU frequency. The accuracy of these schemes has been improved by feedback mechanisms, which take previous prediction errors into account [19].

It has been widely observed [5–10] that frame decoding times vary significantly. For example, some of the frames in an MPEG video can take ten times as long to decode as an average frame [20]. That makes it difficult to estimate the computational requirements of successive frames to meet their deadlines [5–7]. Several workload estimation techniques have been proposed for video applications [5–7, 11, 19], and they can be categorized [5] into methods which make use of the relationship between the amount of data in a frame and decoding time and methods which predict decoding times based on recent times and aim to correct prediction errors using a feedback mechanism.

A close relationship between frame size and decoding time has been widely observed [5, 6, 11], especially in videos encoded with MPEG-style compression, and this relationship allows decoding times to be predicted with reasonable confidence. For example, Liu et al. [6] established a linear relationship between frame size and decoding time and used it to predict decoding times, while Yang and Song [11] improved the accuracy of this approach by introducing a logarithmic relationship, and Bavier et al. [21] used it to predict decoding times. Lee et al. [5] introduced particle-filter techniques to further improve the accuracy of this approach for H.264 codecs.

Yuan et al. [8–10] proposed several DVFS techniques in which the CPU speed is adjusted on the basis of a statistical analysis of past workloads. Urunuela et al. [7] developed a history-based DVFS technique, but it is tailored to video kiosks rather than general video players. Choi et al. [22] adopted a hybrid approach in which different DVFS policies are applied depending on the characteristics of each frame. Im and Ha [12] presented DVFS techniques in which buffers were used to reclaim unused CPU time, and Huang et al. [20] introduced a method of predicting decoding times from offline analysis of frame characteristics.

Most of these techniques do not consider the characteristics of video playback, in which some deadline misses and frame skipping are acceptable. Kim et al. [23] presented a DVFS scheme specifically for scalable video coding (SVC) codecs, which makes use of temporal scalability. The scheme put forward by Yang and Song [11] acknowledges the effect of the ratio of deadline miss on energy consumption, but this paper does not provide a satisfactory solution that selects the appropriate frequency while minimizing energy consumption, nor does it examine how buffering affects power consumption.

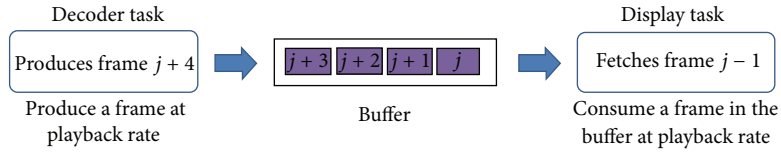


Figure 1: Video playback model.

3. Model

3.1. System Model. To support periodic nature of video playback, a video player decodes r frames per second, and so the decoding period of a frame, T^d, is $1/r$. The Notations explain important symbols used in this paper. Suppose that a CPU supports N^f frequency levels and that level k is the frequency f_k ($k = 1, \ldots, N^f$). If $k < m$, then $f_k < f_m$, so that f_{N^f} is the highest possible frequency. Let N^d be the number of frames decoded in a video. Let P_k^{active} and P_k^{idle}, respectively, be the active and idle power consumption of the system at frequency level k.

We will assume that the decoding time of each frame is known in advance: decoding times can be predicted by an offline analysis of the bitstream of a video [20] or by formulating a relationship between frame size and decoding time [5, 6, 11]. This decoding time information can be inserted into the header of a video [20], and we assume that these frames are available to our frequency selection algorithm. Specifically, $d_{j,k}$ is the decoding time of frame j at frequency level k.

3.2. Video Playback Model. Frame-level DVFS is appropriate for a media player [5–7, 11], which then selects the frequency which best matches the CPU workload imposed by the current frame, before that frame is decoded. The CPU does not change its frequency until the frame has been decoded.

Figure 1 shows our video playback model. The decoding task produces frames at playback rate and passes them to a buffer which stores frames for consumption by a display task, which fetches frames at playback rate. If there was no buffer, then only one frame can be handled by the display task, so the decoder enters sleep state until the frame is consumed by the display task. However, if a number of frames can be stored in the buffer, then decoding can run late, allowing lower frequencies to be selected, but this flexibility is limited by the size of buffer. For example, suppose that the buffer can accommodate N^b frames. If there are already N^b frames in the buffer, then decoding of a new frame must be delayed until the next decoded frame has gone to the display task. For example, consider Figure 1, where $N^b = 4$. If the buffer already contains 4 frames, then the decoder enters sleep state until frame j has been consumed by the display task.

To explain how this buffering technique can decrease CPU power consumption, consider a CPU with 4 frequency levels of 0.8 GHz, 1.2 GHz 1.6 GHz, and 1.8 GHz. We assume that $T^d = 40$ ms and that the process of a frame requires 36 ms at level 4, 40.5 ms at level 3, 54 ms at level 2, and 81 ms at level 1. If there was no buffer, then frequency level 4 must be chosen for every frame to keep the decoding time within 40 ms, as shown in Figure 2(a). However, if there is a buffer, which contains frames decoded when playback starts, then frequency level 1 can be selected for the first three frames, level 2 for the next two frames, and level 3 for the final frame as shown in Figure 2(b), without violating deadlines.

4. Problem Formulation

We formulate an optimization problem with a solution which will minimize energy consumption subject to the constraints of buffer size and decoding deadlines. Frame j ($j = 1, \ldots, N^d$) must be decoded before its deadline T_j^{end}, which is jT^d. At T_j^{end}, frame j leaves the buffer to be displayed on the screen. Let S_j be the frequency level selected for decoding frame j; and let T_j^{start} be the earliest possible time at which the decoding of frame j can start. Because the buffer can contain N^b frames, the decoding of frame j can start at $T_{j-N^b}^{\text{end}}$ (i.e., $(j-N^b)T^d$), at which frame $j-N^b$ can be removed from the buffer and displayed, allowing a new frame to be decoded and stored in the buffer. Thus, T_j^{start} can be expressed as follows:

$$T_j^{\text{start}} = \begin{cases} 0 & j = 1, \ldots, N^b \\ \left(j - N^b\right) T^d & j = N^b + 1, \ldots, N^d. \end{cases} \quad (1)$$

Let $T_{j,k}^{\text{over}}$ ($j = 1, \ldots, N^d - 1$) be the length of time by which the decoding of frame j would overrun if frequency level k is chosen, relative to the start time of the next frame T_{j+1}^{start}. The value of $T_{0,k}^{\text{over}}$ $\forall k$ is initialized to 0. This overrun can be expressed as follows:

$$T_{j,k}^{\text{over}} = \max\left(0, d_{j,k} + T_{j-1,S_{j-1}}^{\text{over}} + T_j^{\text{start}} - T_{j+1}^{\text{start}}\right). \quad (2)$$

If the decoding of frame j indeed finishes after T_{j+1}^{start}, then $T_{j,k}^{\text{over}}$ is the time difference between the actual time at which the decoding of frame j finishes (i.e., $d_{j,k} + T_{j-1,S_{j-1}}^{\text{over}} + T_j^{\text{start}}$) and T_{j+1}^{start}, when frequency level k is chosen for frame j. Conversely, if time remains after the decoding of frame j, the CPU enters its idle state and remains in this state until T_{j+1}^{start}, when $T_{j,k}^{\text{over}}$ is set to 0.

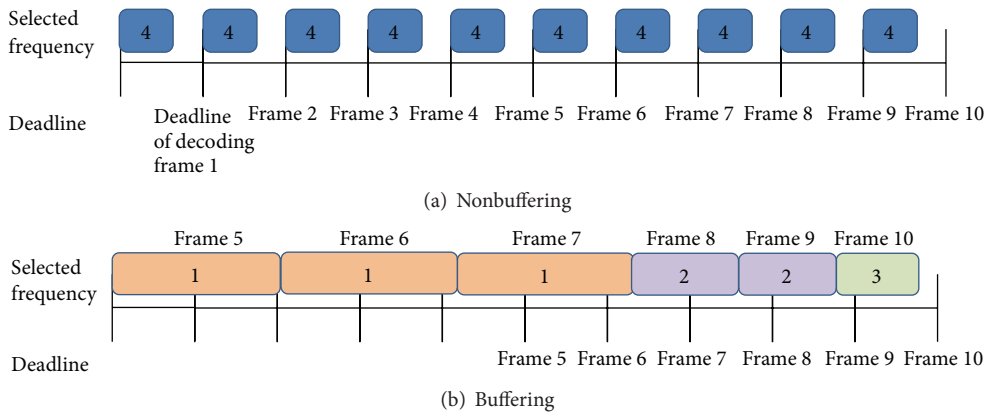


Figure 2: Comparison between buffering and nonbuffering.

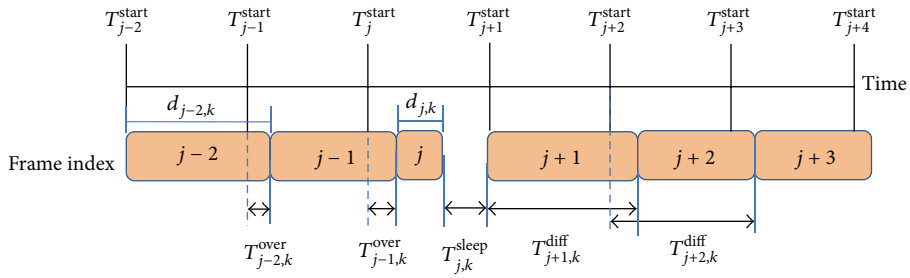


Figure 3: Sequence of frames showing the relationship between $T^{over}_{j,k}$, $T^{sleep}_{j,k}$, and $T^{diff}_{j,k}$.

If $T^{over}_{j,k} = 0$, then the CPU stays in its idle state for the length of $T^{sleep}_{j,k}$, which can be expressed as $T^{start}_{j+1} - (T^{start}_{j} + d_{j,k} + T^{over}_{j-1,S_j})$; otherwise, $T^{sleep}_{j,k}$ is set to 0. The determination of $T^{sleep}_{j,k}$ can thus be summarized as follows:

$$T^{sleep}_{j,k} = \begin{cases} T^{start}_{j+1} - T^{start}_{j} - d_{j,k} - T^{over}_{j-1,S_{j-1}} & \text{if } T^{over}_{j,k} = 0 \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

The energy consumed during the period of $d_{j,k} + T^{sleep}_{j,k}$ while frame j is decoded at frequency level k is written as $E_{j,k}$, which can be expressed as follows:

$$E_{j,k} = d_{j,k} P^{active}_{k} + T^{sleep}_{j,k} P^{idle}_{k}. \quad (4)$$

At this point, we must introduce a further variable $T^{diff}_{j,k}$, which is the difference between the actual time at which the decoding of frame j finishes ($d_{j,k} + T^{over}_{j-1,S_{j-1}} + T^{start}_{j}$) and T^{start}_{j}, and this difference can be expressed as follows:

$$T^{diff}_{j,k} = d_{j,k} + T^{over}_{j-1,S_{j-1}}. \quad (5)$$

Figure 3 shows the relationship between $T^{over}_{j,k}$, $T^{sleep}_{j,k}$, and $T^{diff}_{j,k}$ in a short sequence of frames.

The decoding of frame j must start after T^{start}_{j} and finish before T^{end}_{j}. We can express this period for each frame j ($j = 1, \ldots, N^d$) as T^{round}_{j}, so that $T^{round}_{j} = T^{end}_{j} - T^{start}_{j}$. Each frame j must be decoded before its deadline T^{end}_{j}, so $T^{diff}_{j,k} \leq T^{round}_{j}$. Our frequency selection policy has to minimize the total energy consumption $\sum_{j=1}^{N^d} E_{j,S_j}$. We can now formulate this frequency selection problem that determines S_j ($j = 1, \ldots, N^d$) as follows:

$$\begin{aligned} &\text{Minimize} \quad \sum_{j=1}^{N^d} E_{j,S_j} \\ &\text{Subject to} \quad T^{diff}_{j,S_j} \leq T^{round}_{j}, \quad \forall j,\ j = 1, \ldots, N^d. \end{aligned} \quad (6)$$

5. Frequency Allocation Algorithm

5.1. Algorithm Concept. We now propose an algorithm to solve the problem using a dynamic programming technique. We will use a resolution of 1 ms for time values such as $T^{diff}_{j,k}$. Let $V^{energy}_{j,u}$ be the minimum amount of energy when T^{diff}_{j,S_j} is u milliseconds and frame j is decoded ($j = 1, \ldots, N^d$ and $u = 1, \ldots, T^{round}_{j}$). Let $F_{j,u}$ be the frequency level required to achieve the energy consumption of $V^{energy}_{j,u}$; further, let $V^{over}_{j,u}$ be the corresponding value of T^{over}_{j,S_j} and $V^{diff}_{j,u}$ the value of T^{diff}_{j-1,S_j}.

Table 1: Table for the value of $V_{j,u}^{\text{energy}}$ against $T_{j,k}^{\text{diff}}$.

Frame number	$T_{j,k}^{\text{diff}}$ values				
	1	2	⋯	$T_j^{\text{round}} - 1$	T_j^{round}
1	$V_{1,1}^{\text{energy}}$	$V_{1,2}^{\text{energy}}$	⋯	$V_{1,T_1^{\text{round}}}^{\text{energy}}$	$V_{1,T_1^{\text{round}}}^{\text{energy}}$
2	$V_{2,1}^{\text{energy}}$	$V_{2,2}^{\text{energy}}$	⋯	$V_{2,T_2^{\text{round}}}^{\text{energy}}$	$V_{2,T_2^{\text{round}}}^{\text{energy}}$
⋮	⋮	⋮	⋯	⋮	⋮
N^d	$V_{N^d,1}^{\text{energy}}$	$V_{N^d,2}^{\text{energy}}$	⋯	$V_{N^d,T_{N^d-1}^{\text{round}}}^{\text{energy}}$	$V_{N^d,T_{N^d}^{\text{round}}}^{\text{energy}}$

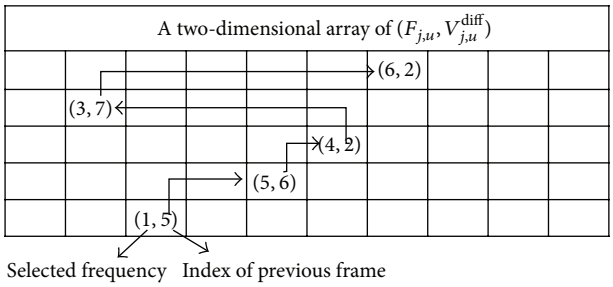


Figure 4: Process of backtracking.

The main idea of the dynamic programming is to construct a table of the optimal energy $V_{j,u}^{\text{energy}}$ for each frame j when $T_{j,k}^{\text{diff}} = u$ ($u = 1, \ldots, T_j^{\text{round}}$), as described in Table 1, where the minimum value in the final row, $\min_{u=1,\ldots,T_{N^d}^{\text{round}}} V_{N^d,u}^{\text{energy}}$, represents the amount of energy consumed by the optimal frequency allocation. For this purpose, we first initialize the values of $V_{1,u}^{\text{energy}}$ and then develop the recurrence relationship between consecutive frames so as to find all the values of $V_{j,u}^{\text{energy}}$ in the table.

We also maintain a two-dimensional array of tuples $(F_{j,u}, V_{j,u}^{\text{diff}})$ which leads to the minimum energy of $V_{j,u}^{\text{energy}}$ as illustrated in Figure 4. Using this array, a backtracking phase starts from frame N^d to frame 1 to select frequency of every frame. For example, Figure 4 shows an array of these tuples when $N^d = 5$ and $T_j^{\text{round}} = 10$. Suppose that the third column in the last row has the minimum energy value. Because $V_{j,u}^{\text{diff}}$ points to the column index of the previous frame, a sequence of frequencies can be selected as follows: (6, 3, 4, 5, 1). Likewise, our dynamic programming algorithm has three phases: (1) initialization, (2) establishment of recurrence relation, and (3) backtracking.

5.2. Initialization. For initialization, consider the following:

(1) $V_{j,u}^{\text{energy}}$, $V_{j,u}^{\text{over}}$, $V_{j,u}^{\text{diff}}$, and $F_{j,u}$ are all initialized to ∞ ($\forall j, u$, $j = 1, \ldots, N^d$ and $u = 1, \ldots, T_j^{\text{round}}$).

(2) $\forall k$, where $d_{1,k} \le T_1^{\text{end}}$, the values of $T_{1,k}^{\text{over}}$, $T_{1,k}^{\text{sleep}}$, and $E_{1,k}$ are calculated from (2), (3), and (4), respectively. Next, $V_{1,d_{1,k}}^{\text{over}}$ is replaced with $T_{1,k}^{\text{over}}$; then, $V_{1,d_{1,k}}^{\text{energy}}$ is updated to $E_{1,k}$, and $F_{1,d_{1,k}}$ is replaced with frequency level k.

5.3. Establishment of Recurrence Relation. During the recurrence establishment phase, $V_{j,u}^{\text{energy}}$, $F_{j,u}$, $V_{j,u}^{\text{diff}}$, and $V_{j,u}^{\text{over}}$ ($j = 2, \ldots, N^d$ and $u = 1, \ldots, T_j^{\text{round}}$) are updated as follows:

(1) For each value of j and u ($j = 2, \ldots, N^d$ and $u = 1, \ldots, T_j^{\text{round}}$), we maintain a two-dimensional array, $A_{j,u}(i,k)$ ($i = 1, \ldots, T_{j-1}^{\text{round}}$ and $k = 1, \ldots, N^f$). The following steps are repeated to find the value of $A_{j,u}(i,k)$, $\forall i, k$ if $V_{j-1,i}^{\text{energy}} \neq \infty$:

(a) Calculate the value of $T_{j,k}^{\text{over}}$ using (2), after replacing $T_{j-1,S_{j-1}}^{\text{over}}$ with $V_{j-1,i}^{\text{over}}$.

(b) Using the resulting value of $T_{j,k}^{\text{over}}$, calculate $T_{j,k}^{\text{sleep}}$ from (3).

(c) Using the resulting value of $T_{j,k}^{\text{sleep}}$, calculate $E_{j,k}$ from (4) and use this value of $E_{j,k}$ to update $A_{\{j,d_{j,k}+V_{j-1,i}^{\text{over}}\}}(i,k)$.

(2) $V_{j,u}^{\text{energy}}$, $F_{j,u}$, $V_{j,u}^{\text{diff}}$, and $V_{j,u}^{\text{over}}$ are updated as follows:

$$
\begin{aligned}
V_{j,u}^{\text{energy}} &= \min_{i=1,\ldots,T_{j-1}^{\text{round}},\ k=1,\ldots,N^f} \left(V_{j-1,i}^{\text{energy}} + A_{j,u}(i,k)\right),\\
F_{j,u} &= \arg \min_{k=1,\ldots,N^f} V_{j,u}^{\text{energy}},\\
V_{j,u}^{\text{diff}} &= \arg \min_{i=1,\ldots,T_{j-1}^{\text{round}}} V_{j,u}^{\text{energy}},\\
V_{j,u}^{\text{over}} &= \max\left(0, V_{V_{j,u}^{\text{diff}}} + d_{j,F_{j,u}} - T_{j+1}^{\text{start}}\right).
\end{aligned}
\tag{7}
$$

5.4. Backtracking. We find values of S_j using a backtracking technique as follows:

(1) j is initialized to N^d, and u is set to $\arg\min_{\{u=1,\ldots,T_{N^d}^{\text{round}}\}} V_{N^d,u}^{\text{energy}}$, so that $V_{N^d,u}^{\text{energy}}$ represents the amount of minimum energy consumption.

(2) While $j > 0$, the following procedures are repeated: (1) S_j is set to $F_{j,u}$, (2) $V_{j,u}^{\text{diff}}$ is substituted for u, and (3) j is decremented by 1.

Pseudocode for this frequency selection algorithm (FSA) is presented as Algorithm 1. If $T^{\max}$ is the maximum round length so that $T^{\max} = \max_{j=2,\ldots,N^d} T_j^{\text{round}}$, we can easily see from Algorithm 1 that the complexity of FSA is $O(N^d N^f T^{\max})$.

6. Algorithm Execution

If the frame decoding time is known in advance, then FSA can run without modification. For example, before playback, frequency allocation table during the entire playback can be obtained as a result of algorithm execution. However, since the algorithm complexity depends on the number of frames to be decoded, we divide the algorithm into iterations and limit the number of frames taken by the algorithm to

(1) Temporary variables: j, u, i and k;
(2) **for** $j = 1$ to N^d **do**
(3) **for** $u = 1$ to T_j^{round} **do**
(4) $V_{j,u}^{\text{energy}}$, $V_{j,u}^{\text{over}}$, $V_{j,u}^{\text{diff}}$ and $F_{j,u} \leftarrow \infty$;
(5) **end for**
(6) **end for**
(7) **for** $u = 1$ to T_j^{round} **do**
(8) **for** $k = 1$ to N^f **do**
(9) **if** $d_{1,k} \leq T_1^{\text{end}}$ and $u = d_{1,k}$ **then**
(10) Calculate $T_{1,k}^{\text{over}}$, $T_{1,k}^{\text{sleep}}$ and $E_{1,k}$ using (2), (3), and (4), respectively;
(11) $V_{1,u}^{\text{energy}} \leftarrow E_{1,k}$;
(12) $F_{1,u} \leftarrow k$;
(13) $V_{1,u}^{\text{over}} \leftarrow T_{1,k}^{\text{over}}$;
(14) **end if**
(15) **end for**
(16) **end for**
(17) **for** $j = 2$ to N^d **do**
(18) **for** $i = 1$ to T_{j-1}^{round} **do**
(19) **for** $k = 1$ to N^f **do**
(20) **if** $V_{j-1,i}^{\text{energy}} \neq \infty$ **then**
(21) Calculate the value of $T_{j,k}^{\text{over}}$ from (2) by replacing $T_{j-1,S_{j-1}}^{\text{over}}$ with $V_{j-1,i}^{\text{over}}$;
(22) $T_{j,k}^{\text{sleep}}$ is calculated from (3) by replacing $T_{j-1,S_{j-1}}^{\text{over}}$ with $V_{j-1,i}^{\text{over}}$;
(23) $E_{j,k}$ is calculated from (4), and $A_{\{j,d_{j,k}+V_{j-1,i}^{\text{over}}\}}(i,k)$ is updated using this value of $E_{j,k}$;
(24) **end if**
(25) **end for**
(26) **end for**
(27) $V_{j,u}^{\text{energy}} \leftarrow \min_{\{i=1,\ldots,T_{j-1}^{\text{round}} \text{ and } k=1,\ldots,N^f\}}(V_{j-1,i}^{\text{energy}} + A_{j,u}(i,k))$;
(28) $F_{j,u} \leftarrow \arg\min_{\{k=1,\ldots,N^f\}} V_{j,u}^{\text{energy}}$;
(29) $V_{j,u}^{\text{diff}} \leftarrow \arg\min_{\{i=1,\ldots,T_{j-1}^{\text{round}}\}} V_{j,u}^{\text{energy}}$;
(30) $V_{j,u}^{\text{over}} \leftarrow \max(0, V_{V_{j,u}^{\text{diff}}} + d_{j,F_{j,u}} - T_{j+1}^{\text{start}})$;
(31) **end for**
(32) $j \leftarrow N^d$;
(33) $u \leftarrow \arg\min_{i=1,\ldots,T_{N^d}^{\text{round}}} V_{N^d,i}^{\text{energy}}$;
(34) **while** $j > 0$ **do**
(35) $S_j \leftarrow F_{j,u}$;
(36) $u \leftarrow V_{j,u}^{\text{diff}}$;
(37) $j \leftarrow j - 1$;
(38) **end while**

Algorithm 1: FSA (frequency selection algorithm).

N^l ($N^l < N^d$). Therefore, at the beginning of the mth iteration, the algorithm chooses the frequency for frames between $(m-1)N^l + 1$ and mN^l, which we call FSA-split, as shown in Algorithm 2.

FSA-split has the following characteristics in comparison with FSA:

(i) FSA-split determines the frequencies of frames between $(m-1)N^l + 1$ and mN^l.

(ii) An initialization part (lines between (9) and (22) in Algorithm 2) takes the length of overrun in the previous iteration ($V_{(m-1)N^l,I^{\text{prev}}}^{\text{over}}$) for the calculation of the parameter values.

Several methods were developed for decoding time estimation, most of which predict future decoding times based on recent measured times [5–7, 11, 19]. The decoding times of the frames in a certain GOP do not change a lot compared with those of its neighboring GOPs [11]. We can therefore predict the decoding times of the next GOP on the basis of those of the current GOP. For example, if N^l is set to the number of frames of a GOP, then the frequency allocation table can be established for the next GOP by passing predicted decoding times of the next GOP to the input parameters of the FSA-split.

7. Experimental Results

7.1. Setup. We performed simulations to evaluate our schemes using power data and timings obtained experimentally. The power consumption of a Samsung Nexus S smartphone (not just the CPU) was measured, and Table 2 shows its

(1) Temporary variables: j, u, i and k;
(2) Input parameter from the previous iteration ($m > 1$): I^{prev}
(3) $I^{\text{prev}} \leftarrow \arg\min_{i=1,\ldots,T^{\text{round}}_{(m-1)N^l}} V^{\text{energy}}_{(m-1)N^l,i}$;
(4) **for** $j = (m-1)N^l + 1$ to mN^l **do**
(5) **for** $u = 1$ to T^{round}_j **do**
(6) $V^{\text{energy}}_{j,u}$, $V^{\text{over}}_{j,u}$, $V^{\text{diff}}_{j,u}$ and $F_{j,u} \leftarrow \infty$;
(7) **end for**
(8) **end for**
(9) **for** $u = 1$ to T^{round}_j **do**
(10) **for** $k = 1$ to N^f **do**
(11) **if** $d_{(m-1)N^l+1,k} \le T^{\text{end}}_{(m-1)N^l+1}$ and $u = d_{(m-1)N^l+1,k}$ **then**
(12) **if** $m > 1$ **then**
(13) Calculate $T^{\text{over}}_{(m-1)N^l+1,k}$, $T^{\text{sleep}}_{(m-1)N^l+1,k}$ and $E_{(m-1)N^l+1,k}$ using (2), (3), and (4), respectively, by replacing $T^{\text{over}}_{j-1,S_{j-1}}$ with $V^{\text{over}}_{(m-1)N^l,I^{\text{prev}}}$;
(14) **else**
(15) Calculate $T^{\text{over}}_{(m-1)N^l+1,k}$, $T^{\text{sleep}}_{(m-1)N^l+1,k}$ and $E_{(m-1)N^l+1,k}$ using (2), (3), and (4), respectively, by replacing $T^{\text{over}}_{j-1,S_{j-1}}$ with 0;
(16) **end if**
(17) $V^{\text{energy}}_{(m-1)N^l+1,u} \leftarrow E_{(m-1)N^l+1,k}$;
(18) $F_{(m-1)N^l+1,u} \leftarrow k$;
(19) $V^{\text{over}}_{(m-1)N^l+1,u} \leftarrow T^{\text{over}}_{(m-1)N^l+1,k}$;
(20) **end if**
(21) **end for**
(22) **end for**
(23) **for** $j = (m-1)N^l + 2$ to mN^l **do**
(24) **for** $i = 1$ to T^{round}_{j-1} **do**
(25) **for** $k = 1$ to N^f **do**
(26) **if** $V^{\text{energy}}_{j-1,i} \neq \infty$ **then**
(27) Calculate the value of $T^{\text{over}}_{j,k}$ from (2) by replacing $T^{\text{over}}_{j-1,S_{j-1}}$ with $V^{\text{over}}_{j-1,i}$;
(28) $T^{\text{sleep}}_{j,k}$ is calculated from (3) by replacing $T^{\text{over}}_{j-1,S_{j-1}}$ with $V^{\text{over}}_{j-1,i}$;
(29) $E_{j,k}$ is calculated from (4), and $A_{\{j,d_{j,k}+V^{\text{over}}_{j-1,i}\}}(i,k)$ is updated using this value of $E_{j,k}$;
(30) **end if**
(31) **end for**
(32) **end for**
(33) $V^{\text{energy}}_{j,u} \leftarrow \min_{\{i=1,\ldots,T^{\text{round}}_{j-1} \text{ and } k=1,\ldots,N^f\}}(V^{\text{energy}}_{j-1,i} + A_{j,u}(i,k))$;
(34) $F_{j,u} \leftarrow \arg\min_{\{k=1,\ldots,N^f\}} V^{\text{energy}}_{j,u}$;
(35) $V^{\text{diff}}_{j,u} \leftarrow \arg\min_{\{i=1,\ldots,T^{\text{round}}_{j-1}\}} V^{\text{energy}}_{j,u}$;
(36) $V^{\text{over}}_{j,u} \leftarrow \max(0, V_{V^{\text{diff}}_{j,u}} + d_{j,F_{j,u}} - T^{\text{start}}_{j+1})$;
(37) **end for**
(38) $j \leftarrow mN^l$;
(39) $u \leftarrow \arg\min_{i=1,\ldots,T^{\text{round}}_{mN^l}} V^{\text{energy}}_{mN^l,i}$;
(40) **while** $j > (m-1)N^l$ **do**
(41) $S_j \leftarrow F_{j,u}$;
(42) $u \leftarrow V^{\text{diff}}_{(m-1)N^l+j,u}$;
(43) $j \leftarrow j - 1$;
(44) **end while**

Algorithm 2: FSA-split (frequency selection algorithm for the mth iteration).

active and idle power values. The time taken to decode video frames was also measured for the two videos in Table 3. We compared our scheme with two other algorithms as follows:

(1) HF always selects the highest frequency, which is equivalent to no DVFS.

Table 2: Measured power consumption against frequency of Samsung Nexus S phone.

Frequency (MHz)	1000	800	400	200	100
Active power (mW)	1324	1082	741	557	444
Idle power (mW)	545	527	503	471	420

Table 3: Video characteristics.

Type	Title	Resolution	Average bitrate	Playback time	GOP length
Animation	Ice Age 4	352 × 288	736 kb/s	5 minutes	12
Sports	Soccer	352 × 288	199 kb/s	5 minutes	12

Table 4: Relative energy consumption of FSA against a number of buffers.

Video type	Animation				Sports			
Number of buffers (N^b)	1	2	3	4	1	2	3	4
Energy used relative to HF	68.6%	67.1%	66.9%	66.9%	80%	77.8%	77.3%	77.2%
Energy used relative to LF	84.2%	82.3%	82.1%	82%	93.2%	90.6%	90.1%	89.9%

Table 5: Percentage of frames decoded at each frequency.

Video type	Animation					Sports				
Frequency (MHz)	100	200	400	800	1000	100	200	400	800	1000
LF	28.5%	3.7%	63.5%	4.3%	0%	0%	9.3%	90.4%	0.2%	0%
FSA ($N^b = 1$)	72%	0.8%	9.8%	14.8%	2.6%	4.8%	52.7%	31.1%	11.1%	0.3%
FSA ($N^b = 2$)	75.5%	0%	0.8%	19.9%	3.8%	24.7%	8.6%	27.2%	38.2%	1.3%
FSA ($N^b = 3$)	75.2%	0%	0.3%	19.7%	4.8%	28.1%	2.1%	21.8%	46.7%	1.3%
FSA ($N^b = 4$)	75.3%	0%	0.3%	20.0%	4.4%	28.9%	1.1%	18.4%	50.6%	1.1%

Table 6: Percentage energy difference between FSA and FSA-split against different values of N^l.

Video type	Animation					Sports				
N^l	12	24	48	96	192	12	24	48	96	192
$N^b = 1$	0.88%	0.48%	0.26%	0.14%	0.07%	0.22%	0.11%	0.05%	0.03%	0.02%
$N^b = 2$	1.26%	0.66%	0.33%	0.17%	0.09%	0.42%	0.21%	0.11%	0.06%	0.03%
$N^b = 3$	1.40%	0.70%	0.35%	0.18%	0.09%	0.49%	0.24%	0.12%	0.07%	0.04%
$N^b = 4$	1.47%	0.76%	0.38%	0.20%	0.10%	0.52%	0.24%	0.12%	0.06%	0.04%

(2) LF selects the lowest frequency level which will get each frame decoded in time. This method is a good heuristic, because CPU frequency can be expected to have a monotonic relationship with energy consumption [4–7, 11].

7.2. Efficacy of FSA. Table 4 shows how energy consumption depends on the number of frames that are buffered. We see that (1) FSA always shows the best performance, using 13% less energy than LF and 27% less energy than HF on average, and (2) increasing the size of the buffer saves more energy, but this amount of energy saved gradually tails off. In particular, even when $N^b = 1$ so that only one additional buffer is used, FSA uses 11% less energy than LF on average, suggesting that the buffer overhead of FSA is not high.

The results in Table 4 can be attributed to FSA's effective use of the slack times generated by storing decoded frames in the buffer, allowing the CPU to operate at lower frequencies. For example, Table 5 shows the average percentage of the frames in both video clips that are decoded at each frequency; FSA chooses lower frequencies than LF, decreasing energy consumption. FSA chooses the highest frequency (1000 MHz) more often than LF, which increases the idle time, allowing relatively lower frequencies to be chosen than LF. These results suggest that frequency selection has a great effect on energy consumption.

7.3. Efficacy of FSA-Split. To evaluate the efficacy of FSA-split, we examined how the values of N^l affect the energy consumption against different values of N^b as tabulated in Table 6. We see that (1) their energy difference is marginal, exhibiting 1.47% difference at maximum, even when N^l is set to 12 which is the GOP size; (2) increasing the value of N^l decreases the energy gap; and (3) increasing the buffer size increases the energy gap even though the difference is negligible. Although FSA exhibits slightly better performance than FSA-split, it takes all frame parameters for algorithm execution, requiring a lot of computation. These results suggest that FSA-split is a practical method of reducing the energy required for video decoding.

8. Conclusions

We have proposed a new frequency allocation scheme which minimizes energy consumption while avoiding buffer overrun, using a dynamic programming technique. This

scheme establishes recurrence relationship between consecutive frames to construct a table of the minimum energy values required to decode each frame and determines a sequence of frequencies required to decode every frame using a backtracking technique. It was extended to optimize CPU frequencies over a short sequence of frames, which gives a basis for energy-saving video decoding in practice.

Experimental results show that it uses 27% less energy than a processor at the highest frequency on average. In particular, it uses 13% less energy, compared to the widely used heuristic which chooses the lowest frequency to get each frame decoded in time. We believe that these results give a useful guideline for low-power video service by providing the minimum bound on power consumption required for video playback.

Notations

r:	Playback rate of a video (fps)
N^f:	Number of frequency levels supported by a CPU
T^d:	Decoding period of a video
f_k:	Frequency corresponding to the frequency level k
P_k^{active}:	Active power at frequency level k
P_k^{idle}:	Idle power at frequency level k
N^b:	Number of frames that a display buffer can accommodate
N^d:	Number of frames decoded in a video
$d_{j,k}$:	Decoding time of frame j at frequency level k
T_j^{end}:	Decoding deadline for frame j
T_j^{start}:	Earliest possible start time for decoding frame j
T_j^{round}:	$T_j^{\text{end}} - T_j^{\text{start}}$
$T_{j,k}^{\text{over}}$:	$\max(0, d_{j,k} + T_{j-1,S_{j-1}}^{\text{over}} + T_j^{\text{start}} - T_{j+1}^{\text{start}})$
$T_{j,k}^{\text{sleep}}$:	CPU sleep time when frequency level k is chosen for frame j
$E_{j,k}$:	Energy consumed during the decoding period for frame j at frequency k
$A_{j,u}(i,k)$:	Two-dimensional array for $E_{j,k}$ when $i = T_{j-1,S_j}^{\text{diff}}$, $u = T_{j,k}^{\text{diff}}$, and k is the frequency level chosen for decoding frame j
$T_{j,k}^{\text{diff}}$:	Time between completion of decoding frame j and T_j^{start}
S_j:	Frequency level selected for decoding frame j
$V_{j,u}^{\text{energy}}$:	Minimum energy consumption in decoding frames 1 to j, when $T_{j,k}^{\text{diff}} = u$ ms
$F_{j,u}$:	Frequency level selected for decoding frame j to achieve an energy of $V_{j,u}^{\text{energy}}$
$V_{j,u}^{\text{over}}$:	Value of T_{j,S_j}^{over} to achieve an energy of $V_{j,u}^{\text{energy}}$
$V_{j,u}^{\text{diff}}$:	Value of $T_{j-1,S_j}^{\text{diff}}$ to achieve an energy of $V_{j,u}^{\text{energy}}$
N^l:	Number of frames for which frequencies are determined by FSA-split.

Competing Interests

The authors declare that they have no competing interests.

Acknowledgments

This research is supported by Inha University Research Grant.

References

[1] S. A. Alvi, B. Afzal, G. A. Shah, L. Atzor, and W. Mahmood, "Internet of multimedia things: vision and challenges," *Ad Hoc Networks*, vol. 33, pp. 87–111, 2015.

[2] H. Aydin, P. Mejia-Alvarez, D. Mosse, and R. Melhem, "Dynamic and aggressive scheduling techniques for power-aware real-time systems," in *Proceedings of the IEEE Real-Time Systems Symposium*, p. 95, London, UK, December 2001.

[3] M. Marinoni and G. Buttazzo, "Elastic DVS management in processors with discrete voltage/frequency modes," *IEEE Transactions on Industrial Informatics*, vol. 3, no. 1, pp. 51–62, 2007.

[4] P. Pillai and K. G. Shin, "Real-time dynamic voltage scaling for low-power embedded operating systems," in *Proceedings of the ACM Symposium on Operating Systems Principles*, pp. 89–102, October 2001.

[5] J.-B. Lee, M.-J. Kim, S. Yoon, and E.-Y. Chung, "Application-support particle filter for dynamic voltage scaling of multimedia applications," *IEEE Transactions on Computers*, vol. 61, no. 9, pp. 1256–1269, 2012.

[6] X. Liu, P. Shenoy, and M. D. Corner, "Chameleon: application-level power management," *IEEE Transactions on Mobile Computing*, vol. 7, no. 8, pp. 995–1010, 2008.

[7] R. Urunuela, G. Muller, and J. L. Lawall, "Energy adaptation for multimedia information kiosks," in *Proceedings of the 6th ACM and IEEE International Conference on Embedded Software (EMSOFT '06)*, pp. 223–232, October 2006.

[8] W. Yuan and K. Nahrstedt, "Practical voltage scaling for mobile multimedia devices," in *Proceedings of the 12th Annual ACM International Conference on Multimedia (MULTIMEDIA '04)*, pp. 924–931, New York, NY, USA, October 2004.

[9] W. Yuan and K. Nahrstedt, "Energy-efficient CPU scheduling for multimedia applications," *ACM Transactions on Computer Systems*, vol. 24, no. 3, pp. 292–331, 2006.

[10] W. Yuan, K. Nahrstedt, S. V. Adve, D. L. Jones, and R. H. Kravets, "GRACE-1: cross-layer adaptation for multimedia quality and battery energy," *IEEE Transactions on Mobile Computing*, vol. 5, no. 7, pp. 799–815, 2006.

[11] A. Yang and M. Song, "Aggressive dynamic voltage scaling for energyaware video playback based on decoding time estimation," in *Proceedings of the ACM International Conference on Embedded Software*, pp. 1–9, Grenoble, France, October 2009.

[12] C. Im and S. Ha, "Dynamic voltage scaling for real-time multi-task scheduling using buffers," in *Proceedings of the ACM Conference on Languages, Compilers and Tools for Embedded Systems*, pp. 88–94, Washington, DC, USA, June 2004.

[13] M. Weiser, B. Welch, A. Demers, and S. Shenker, "Scheduling for reduced CPU energy," in *Proceedings of the 1st USENIX Conference on Operating Systems Design and Implementation (OSDI '94)*, article 2, USENIX Association, 1994.

[14] E. Seo, S. Park, J. Kim, and J. Lee, "TSB: a DVS algorithm with quick response for general purpose operating systems," *Journal of Systems Architecture*, vol. 54, no. 1-2, pp. 1–14, 2008.

[15] V. Pallipadi and A. Starikovskiy, "The ondemand governor: past, present, and future," in *Proceedings of the Linux Symposium*, pp. 223–238, Ottawa, Canada, July 2006.

[16] M. Fleischmann, "Longrun power management—dynamic power management for crusoe processors," Tech. Rep., Transmeta, 2001.

[17] R. Ayoub, U. Ogras, E. Gorbatov et al., "OS-level power minimization under tight performance constraints in general purpose systems," in *Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED '11)*, pp. 321–326, IEEE, Fukuoka, Japan, August 2011.

[18] K. Flautner and T. Mudge, "Vertigo: automatic performance-setting for linux," in *Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI '02)*, pp. 105–116, December 2002.

[19] Y. Gu and S. Chakraborty, "A hybrid DVS scheme for interactive 3D games," in *Proceedings of the 14th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS '08)*, pp. 3–12, St. Louis, Mo, USA, April 2008.

[20] Y. Huang, S. Chakraborty, and Y. Wang, "Using offline bitstream analysis for power-aware video decoding in portable devices," in *Proceedings of the 13th ACM International Conference on Multimedia (MM '05)*, pp. 299–302, Singapore, November 2005.

[21] A. Bavier, A. Montz, and L. Peterson, "Prediction MPEG decoding time," in *Proceedings of the ACM SIGMETRICS Conference*, pp. 131–140, Madison, Wis, USA, June 1998.

[22] K. Choi, K. Dantu, W.-C. Cheng, and M. Pedram, "Frame-based dynamic voltage and frequency scaling for a MPEG decoder," in *Proceedings of the IEEE/ACM International Conference on Computer Aided Design (ICCAD '02)*, pp. 732–737, ACM, November 2002.

[23] E. Kim, H. Jeong, J. Yang, and M. Song, "Balancing energy use against video quality in mobile devices," *IEEE Transactions on Consumer Electronics*, vol. 60, no. 3, pp. 517–524, 2014.

Hindawi Publishing Corporation
Mobile Information Systems
Volume 2016, Article ID 3414816, 11 pages
http://dx.doi.org/10.1155/2016/3414816

Research Article

Control-Scheduling Codesign Exploiting Trade-Off between Task Periods and Deadlines

Hyun-Jun Cha,[1] Woo-Hyuk Jeong,[2] and Jong-Chan Kim[1]

[1]*Graduate School of Automotive Engineering, Kookmin University, 77 Jeongneung-ro, Seongbuk-gu, Seoul 02707, Republic of Korea*
[2]*Department of Computer Science, Kookmin University, 77 Jeongneung-ro, Seongbuk-gu, Seoul 02707, Republic of Korea*

Correspondence should be addressed to Jong-Chan Kim; jongchank@kookmin.ac.kr

Received 1 January 2016; Revised 22 March 2016; Accepted 27 March 2016

Academic Editor: Qixin Wang

A control task's performance heavily depends on its sampling frequency and sensing-to-actuation delay. More frequent sampling, that is, shorter period, improves the control performance. Similarly, shorter delay also has a positive effect. Moreover, schedulability is also a function of periods and deadlines. By taking into account the control performance and schedulability at the same time, this paper defines a period and deadline selection problem for fixed-priority systems. Our problem is to find the optimal periods and deadlines for given tasks that maximize the overall system performance. As our solution, this paper presents a novel heuristic algorithm that finds a high-quality suboptimal solution with very low complexity, which makes the algorithm practically applicable to large size task sets.

1. Introduction

In a cyberphysical system (CPS), real-time control systems monitor and control physical systems (i.e., target plants) with precise control performance requirements and tight resource constraints. When designing such a system, two different approaches can be applied. First, the traditional approach separates the *control design phase* and *implementation phase* such that scheduling parameters such as sampling rates are determined in the control design phase without considering the scheduling issue. The second approach, known as *control-scheduling codesign*, takes into account both the control performance and task scheduling simultaneously for the purpose of enhancing control performance with limited resources [1–4]. This paper advocates the second approach when designing a CPS.

Performance of a real-time control system in a CPS depends on not only its functional correctness but also its scheduling parameters such as *sampling frequency*. With more frequent sensing and actuation, more accurate control results can be obtained [5]. Certainly, this performance enhancement is at the cost of increased computing demands. Another important but often ignored timing property is the delay between sensing and actuation, that is, *input-output delay*. Since a shorter delay means more recent sensing data has been used to produce the actuation value, it can provide higher control performance [5]. Then, one interesting observation is that a task's period, that is, inverse of sampling frequency, can be lengthened without hurting the control performance if we can somehow reduce the delay. To maintain a task's delay within a certain range, the desired maximum delay should be used as the relative deadline in the schedulability check. If the schedulability check passes, the task's input-output delay is guaranteed less than the relative deadline. One more timing attribute we have to discuss is *jitter*, which is the amount of uncertain variation of sampling time or input-output delay, called *sampling jitter* and *input-output jitter*, respectively. Generally, large jitter has negative effects on the control performance even though the effect is not that significant as the input-output delay [6]. Even regarding jitters, a shorter relative deadline also gives tighter upper bounds of the sampling jitter and input-output jitter such that a better control result can be produced [6]. As a result, the control performance can be enhanced by either way of shorter periods or shorter deadlines, and periods and deadlines have a *trade-off relation* in terms of control performance.

Besides control performance, schedulability is also heavily affected by periods and deadlines of the tasks. Generally speaking, longer periods and longer deadlines both make the system more schedulable, however, at the cost of a reduced control performance. In other words, a better control performance can be obtained with a lower chance of being schedulable. Moreover, similar to the control performance case, periods and deadlines are mutually tradable to maintain schedulability. Therefore, it is important to select proper periods and deadlines which satisfy both the control performance and the schedulability. For this control-scheduling codesign issue, an optimization problem can be formulated which finds the best feasible periods and deadlines for given tasks that maximize the overall system performance.

In the literature, with a similar motivation, *period selection problem* has been extensively studied [7–10] for both dynamic and fixed-priority scheduling algorithms. However, *period and deadline selection problem*, which this paper is dealing with, has gathered relatively little attention and only the dynamic-priority case has been studied [6, 11]. With this motivation, targeting fixed-priority systems, this paper proposes a novel task set synthesis algorithm that finds the proper periods and deadlines which maximize the overall system performance while guaranteeing the system schedulability. Since, even with a small number of tasks, finding the optimal solution is intractable due to the huge solution space to be searched, our algorithm is basically structured as a search-based heuristic algorithm. As will be shown in Section 6, our algorithm has a linear complexity and finds a high-quality suboptimal solution even with a large task set.

For the quantitative analysis of control performance with varying periods and deadlines, we also conduct a measurement study with an automotive control application. The measured control performance of the task is defined as a nonlinear and nonconvex function of period and deadline. This function is used as an input to our heuristic algorithm.

This paper's contribution can be summarized as follows:

(i) We identify and demonstrate the trade-off relation between period and deadline in terms of control performance through actual experimental studies with an automotive control application.

(ii) Exploiting the above trade-off relation, a novel task set synthesis algorithm is proposed, which heuristically finds near-optimal feasible (period and delay) combinations maximizing the overall control performance.

The rest of our paper is organized as follows. The next section briefly explains related work. Section 3 presents brief background knowledge and formally describes our problem. In Section 4, the trade-off relation of control performance is formally described. Section 5 presents our heuristic algorithm for the period and deadline selection problem. The experimental results are presented in Section 6. Finally, Section 7 concludes this paper.

2. Related Work

Control-scheduling codesign problem has been extensively studied in the literature. Seto et al. [7] first defined the period selection problem assuming that the control performance can be expressed as an exponential decay function of the sampling period and the tasks are scheduled using dynamic-priority methods. The problem is extended to fixed-priority systems by Seto et al. [8] by finding the finite set of feasible period ranges using a branch and bound-based integer programming method. In their work, the cost function is assumed to be a monotonically increasing function of task period. Later, Bini and Di Natale [9] proposed a faster algorithm that finds a suboptimal period assignment, which can be used for a task set of practical size that was intractable by previous methods due to its high computing demands. Recently, Du et al. [12] proposed an analytical solution using the method of Lagrange multipliers and an online algorithm for the overloaded situation.

The common assumption of the above researches regarding the period selection problem is that the control performance is only affected by the sampling rate, that is, task period, of the controller. However, the delay between sensing and actuation also has a significant effect on the control performance. With this motivation, Bini and Cervin [10] incorporated each task's sensing to actuation delay into the cost function. In their work, in order to find the optimal period assignment, cost functions are approximated as linear functions of period and delay, and the delay is also approximated assuming the fluid model scheduler. Through these approximations, they proposed an analytical solution.

Wu et al. [6] further enhanced the algorithm by finding task periods and deadlines altogether for EDF scheduled systems. As a result, the problem had become a period and deadline selection problem. They showed that, by regulating relative deadlines of tasks, we can upper-limit the amount of delays and jitter each task can experience. In their work, the cost function is assumed to be a nonlinear function which increases in both period and deadline of tasks. A two-step approach was presented which first fixes periods and tries to minimize deadlines using unused resources. Recently, Tan et al. [11] proposed a new algorithm which simultaneously adjusts periods and deadlines assuming EDF scheduling and LGQ controller tasks. They showed that the new algorithm is more robust with different workloads than the previous method.

Despite the above researches, however, compared to the period selection problem, the period and deadline selection problem has gained less attention even though it has more flexibility to enhance control performance with scarce resources. Moreover, only EDF scheduling is considered in the period and deadline selection problem due to its ease of schedulability analysis, though the fixed-priority scheduling is more commonly used in the practice. On the contrary, this paper is dealing with the period and deadline selection problem under the fixed-priority scheduling.

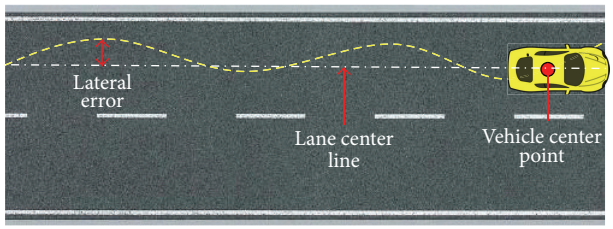


Figure 1: Lane keeping assist system. Its system error is defined as the lateral distance between the vehicle center point and the lane center line.

3. Background and Problem Description

3.1. System Model. This paper considers a system with n independent periodic real-time tasks $\{\tau_1, \tau_2, \ldots, \tau_n\}$, which control n different plants, respectively. The tasks are scheduled by the fixed-priority scheduler and the priorities are assigned according to the Deadline Monotonic (DM) order. Each τ_i is characterized by the following scheduling parameters:

(i) C_i: the worst-case execution time (WCET).

(ii) T_i: the sampling period.

(iii) D_i: the relative deadline.

From the above, C_i is a known parameter decided by the control code, whereas T_i and D_i are operational parameters which can be controlled by system designers. Regarding deadlines, this paper considers only *constrained deadlines* where D_i is always less than or equal to T_i. During system execution, each τ_i generates infinite sequence of periodic jobs $\tau_{i,1}, \tau_{i,2}, \ldots$ with its period T_i, which controls its corresponding target plant by (i) sensing the state of the plant, (ii) calculating the actuation values, and (iii) actuating the plant.

3.2. Control Performance as a Function of Period and Deadline. When defining the performance of a controller, various metrics can be used, such as transient response time and steady-state accuracy [6, 13]. In some cases, even the energy consumption can be a control performance metric [7]. Among the various control performance metrics, this paper chooses the system error as our optimization target. System error is defined as the difference between the desired state and the actual state of the plant [6]. This can be thought of as how well the plant is acting following the controller's intention.

More specifically, let us take the lane keeping assist system (LKAS) as an example. In a modern vehicle, LKAS controls the steering angle such that the vehicle is able to follow the center of its lane. Then, the system error can be defined as the lateral error between the center of the vehicle and the center of the lane, which is illustrated in Figure 1. Since this system error also varies along with time t, we further define the worst-case system error as the largest lateral error the vehicle can experience during its driving. More interested readers are referred to [14].

Following the notation in [6], each task's control performance is defined as a function of its period and deadline, which is denoted by

$$J_i\left(T_i, D_i\right). \tag{1}$$

Generally, $J_i(T_i, D_i)$ is defined as a nonlinear cost function which increases in both T_i and D_i; that is, if period or deadline increases, system error always increases. The intuition behind this assumption will be discussed in Section 4. We assume that $J_i(T_i, D_i)$ is not continuous but discrete in both T_i and D_i, which are also constrained within $[T_i^{\min}, T_i^{\max}]$ and $[D_i^{\min}, D_i^{\max}]$, respectively.

Besides each task's system error, the overall system error is denoted by

$$J\left(T, D\right), \tag{2}$$

where T is the vector of T_i's and D is the vector of D_i's. For the notational simplicity, $J(T, D)$ is shortened to J as in the following:

$$J = \sum_{1 \le i \le n} w_i J_i\left(T_i, D_i\right), \tag{3}$$

where J is defined as a weighted sum of $J_i(T_i, D_i)$'s and w_i is a user-defined weight constant for the purpose of normalizing each $J_i(T_i, D_i)$ to a desired range. Now, the system's overall system error can be obtained by giving every task's period and deadline.

3.3. Problem Description. Assuming the above concepts and notations, this subsection describes our period and deadline selection problem, which can be formally defined as follows.

Problem Description. For a given task set $\{\tau_1, \tau_2, \ldots, \tau_n\}$, each τ_i's C_i and $J_i(T_i, D_i)$ are given a priority. Then, our problem is to find the optimal $T = (T_1, T_2, \ldots, T_n)$ and $D = (D_1, D_2, \ldots, D_n)$ such that the overall system error $J(T, D)$ is minimized while guaranteeing every τ_i's schedulability.

Since it is assumed that $J_i(T_i, D_i)$ is not continuous, we do not try to make an analytical solution for our optimization problem. Instead, we formulate our problem as a combinatorial optimization problem. Figure 2 shows a graphical representation of an example $J_i(T_i, D_i)$ with 10 discrete (T_i, D_i) combinations. Note that since we only consider constrained deadlines, the left upper triangular matrix is not considered at all.

4. How Scheduling Affects Control Performance

This section deals with the rationale behind the definition of $J_i(T_i, D_i)$, which is introduced in Section 3.2. It is claimed in many literatures that the control performance of a task can be defined as a function of the task period and deadline [6, 10, 11]. If the control system is composed of only a single periodic task, it should be strictly periodic and the input-output delay is simply bounded by C_i. Thus, the control performance should be defined as a function of T_i and C_i. However, when

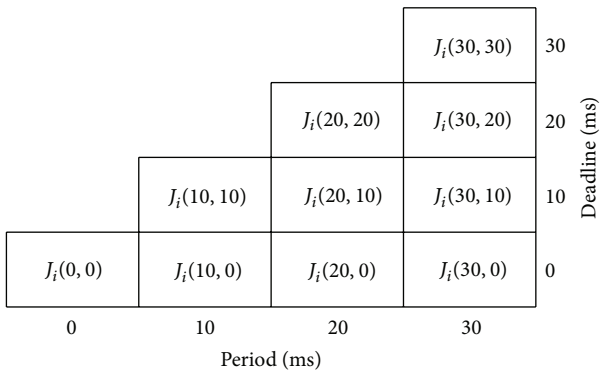


Figure 2: A graphical representation of an example $J_i(T_i, D_i)$ function with its period varying from 0 ms to 30 ms and its deadline also varying from 0 ms to 30 ms. The granularity of periods and deadlines is 10 ms.

multiple control tasks are implemented and scheduled on a shared processor, their execution is also subject to the following scheduling effects [6]:

(i) *Sampling Delay*. Although a job is released at a certain time point, the actual beginning of the job can be delayed due to the execution of higher-priority jobs. The time distance between the release time and the actual beginning of a job is defined as a sampling delay.

(ii) *Input-Output Delay*. The time distance between the beginning and the completion of a job is defined as an input-output delay, which is composed of the actual execution time of the job which is bounded by C_i and the blocking time by higher-priority jobs.

(iii) *Sampling Jitter*. As a task produces consecutive jobs, its sampling delay is not constant but varies according to the scheduling pattern each job experiences. The difference between the minimum and maximum sampling delay is defined as the sampling jitter of the task.

(iv) *Input-Output Jitter*. The difference between the minimum and maximum input-output delay of a task is defined as the input-output jitter of the task.

By the above scheduling effects, the actual execution of a control task is not strictly periodic and the input-output delay significantly varies due to different scheduling scenarios. Thus, the above scheduling effects as well as the control period collectively affect the control performance. Note that, for each task, the only tunable scheduling parameters are T_i and D_i in our system model. The control period T_i itself has a significant effect on the control performance since it determines the control frequency. Besides, for the above four scheduling effects, Wu et al. [6] analyzed the worst-case scenarios as in the following under the condition that every task is schedulable:

(i) Sampling delay is bounded by $D_i - C_i$, which happens when the beginning of a job is delayed as long as possible without hurting the schedulability, and the job executes without any interference by higher-priority jobs once it begins.

(ii) Input-output delay is bounded by D_i, which happens when a job begins right after it is released and finishes right before the job's deadline.

(iii) Sampling jitter is bounded by $D_i - C_i$. The minimum possible sampling delay is 0 and the maximum possible sampling delay is $D_i - C_i$. Thus, the difference between the maximum and minimum sampling delay is $D_i - C_i$.

(iv) Input-output jitter is bounded by $D_i - C_i^b$, where C_i^b is the best-case execution time which can be simply assumed to be zero when not available. The minimum possible input-output delay is C_i^b and the maximum possible input-output delay is D_i. Thus, the difference between the maximum and minimum input-output delay is $D_i - C_i^b$.

The analysis above shows that a shorter deadline makes shorter sampling delay and shorter input-output delay. Even the jitters can be controlled by a shorter deadline. The implication of the above analysis is that a better control performance can be obtained by shortening D_i as well as T_i [6]. From the above observation, we can conclude that the control performance generally increases as T_i or D_i decreases. In other words, $J_i(T_i, D_i)$, the system error, is a monotonically increasing function in both T_i and D_i. Note that $J_i(T_i, D_i)$ also reflects the effect of jitters as well as delays and sampling frequency. In order to account for the above scheduling effects when estimating the control performance, Jitterbug [15], which is a MATLAB toolbox, can be used to analyze the cost of delay and jitter in terms of control performance. Besides, TrueTime [16] provides a simulation environment which facilitates cosimulation of controller task execution and real-time scheduling.

Although analysis and simulation methods are useful when estimating control performance, this paper proposes a different approach when finding $J_i(T_i, D_i)$. As will be further explained in Section 6.1, a measurement environment is developed for an automotive control application. In practice, T_i and D_i cannot be an arbitrary number but should be chosen from a number of predefined candidate parameters the software platform provides. Thus, it is practically feasible to measure the system errors for each T_i and D_i combination for each control task by arbitrarily making the worst-case scenarios. Using these discrete functions $J_i(T_i, D_i)$'s, the optimization algorithm in Section 5 can be used to find optimal T and D that minimize $J(T, D)$.

5. Task Set Synthesis Algorithm

In order to find T and D with the minimum $J(T, D)$, the simplest solution is to explore the entire solution space for every (T_i, D_i) combination checking the schedulability and calculating the overall system error. This exhaustive search method, however, has a too high complexity such that it is not applicable to a task set even with a small number of

tasks like five or six tasks. The computational feasibility of the exhaustive search algorithm will be further discussed in Section 6.2. Instead, this section proposes an alternative heuristic algorithm that finds a suboptimal result, however, with a very low computational complexity. Even with this low complexity, our solution can find a very-high-quality solution for very large task sets, which will be shown later in Section 6.2.

For the ease of explanation, let us define the following function:

$$\text{Schedulability}\,(T, D, C)\,, \tag{4}$$

where T and D are vectors of chosen periods and deadlines. C is a vector of each task's C_i's. It is assumed that C is a constant vector. Inside this function, tasks are sorted according to the DM order where the task with the shortest D_i gets index 1 and the task with the longest D_i gets index n. Then, following Audsley et al. [17], the exact schedulability check is performed. For that, the following recursive equation computes the worst-case response time R_i of each τ_i:

$$R_i^{k+1} = C_i + \sum_{1 \le m < i} \left\lceil \frac{R_i^k}{T_m} \right\rceil \cdot C_m, \tag{5}$$

where $R_i^0 = C_i$. The recursive equation continues until $R_i^k = R_i^{k+1}$ and the converged value is taken as the final R_i. Then, since we know every R_i, we can simply check each τ_i's schedulability by comparing R_i with D_i. If every R_i is less than D_i for $1 \le i \le n$, the system is schedulable and every τ_i's input-output delay is guaranteed under D_i. From the result of the schedulability check, Schedulability(T, D, C) returns either of the following values:

(i) *True*: if the system is schedulable.

(ii) *False*: if the system is not schedulable.

This schedulability check function is used inside the outer loop of our heuristic search algorithm to check the feasibility of the chosen T and D.

In the beginning of our heuristic algorithm, the initial solution is set to

$$\{\tau_1\,(0,0)\,, \tau_2\,(0,0)\,, \ldots, \tau_n\,(0,0)\}\,; \tag{6}$$

that is, all the periods and deadlines are equal to zero. Certainly, this initial solution has the lowest possible $J(T, D)$ but is definitely not schedulable. Then, our heuristic algorithm iteratively selects (i) the task and (ii) the direction to move the chosen task until Schedulability(T, D, C) = True. Our rule of thumb for selecting the proper task is to choose the task with the lowest $J_i(T_i, D_i)$ for the purpose of preventing a certain task from moving too quickly to the higher $J_i(T_i, D_i)$. For choosing the moving direction for each iteration, the basic idea is to choose the direction with the lower slope in order to minimize the resulting $J_i(T_i, D_i)$ after the move.

When applying the above basic idea, however, it can suffer from the following worst-case scenario. Starting a new iteration, the algorithm chooses τ_i as the task to be moved. By looking at τ_i's current position in $J(T_i, D_i)$, the right direction has the lower slope compared to the upper direction. Naturally, our algorithm moves τ_i to the right direction. However, imagine that even though the upper direction requires higher slope, the task set can be schedulable immediately after moving τ_i to the upper direction. If this case happens repeatedly, τ_i will move to the right direction too many times, but still making the system unschedulable.

To prevent such scenarios, we slightly tune the algorithm by looking at the system schedulability as well as $J(T_i, D_i)$'s slope when determining the moving direction. Figure 3 shows the four cases our algorithm should consider when the current location of the task is $\tau_i(10, 0)$:

(i) Figure 3(a) shows a case where both directions make the system schedulable. In that case, it is desirable to choose the direction with the lower slope.

(ii) Figure 3(b) shows another case where both directions are not schedulable. Then, our choice is also to choose the direction with the lower slope.

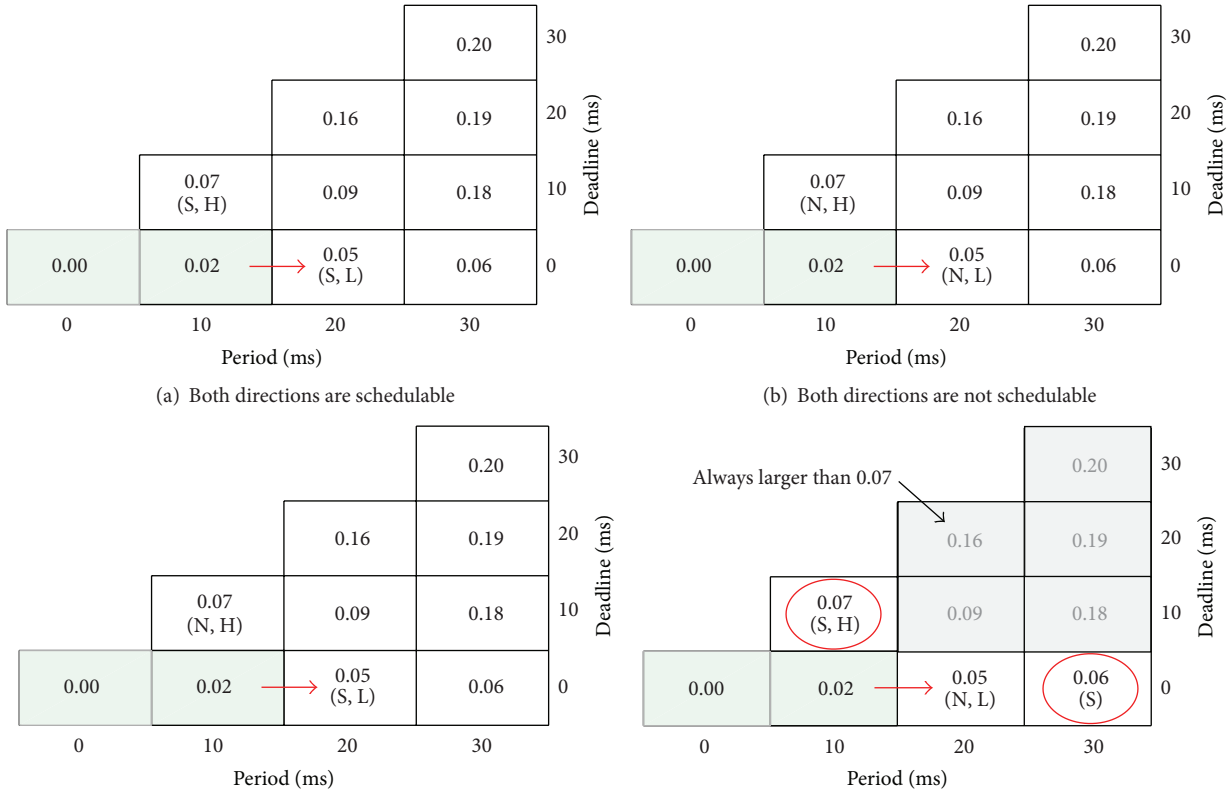
(iii) Figure 3(c) shows a case where the lower slope direction is schedulable, but the higher slope direction is not schedulable. Then, the choice is to take the lower slope direction, which makes the system schedulable immediately.

(iv) Figure 3(d) shows a case where the lower slope is not schedulable, but the higher slope is schedulable. In this case, it is not possible to decide the correct direction from the current information available. If we take the upper direction, the system will be immediately schedulable with $J_i(T_i, D_i) = 0.07$. However, if we move to the right direction two times, it will also make the system schedulable with even lower $J_i(T_i, D_i) = 0.06$. One interesting observation is that the cells in the right-hand side of $\tau_i(10, 10)$ make $J_i(T_i, D_i)$ always larger than 0.07. Thus, if there is a better solution compared to $\tau_i(10, 10)$, it must be among the cells in the right-hand side of the current location, that is, $\tau(10, 0)$. Then, our quick fix is that, upon meeting this condition, every cell in the lower slope direction is quickly visited to compare the resulting $J_i(T_i, D_i)$ with $J_i(T_i, D_i)$ when taking the high slope direction to choose the better direction.

Procedure 1 shows the pseudocode of our heuristic iterative search algorithm. After positioning each τ_i at the initial solution, the while loop iteratively chooses the next moving τ_i and the direction to move considering the four cases in Figure 3 until the system becomes schedulable. Then, using break, the while loop is terminated and the output is finally decided.

6. Experiments

6.1. Control Performance Measurement Study. In this subsection, the experimental results for the control performance measurement study are presented. First, we explain how the measurement environment is designed and implemented. Then, the actual measurement data is presented for an



(a) Both directions are schedulable

(b) Both directions are not schedulable

(c) Lower slope is schedulable, but higher slope is not schedulable

(d) Lower slope is not schedulable, but higher slope is schedulable

Figure 3: The four cases to be considered by our heuristic algorithm. S and N mean schedulable and not schedulable, respectively. H and L mean higher slope and lower slope, respectively.

example control application. For the measurement study, as the target control application, we use LKAS, in which the system error is defined as the maximum lateral error the vehicle experiences during the measurement.

Figure 4 shows the measurement environment, which consists of a vehicle dynamics simulator and two automotive electronic control units (ECUs) connected by a controller area network (CAN) bus. Inside the ECUs, control application codes are deployed upon our specially tuned real-time operating system (RTOS), which is a modified version of Erika Enterprise [18]. We specifically tuned the RTOS such that the worst-case job release and delay pattern always happens to simulate the worst-case scenario the vehicle can experience. Among the two ECUs, the first one contains the LKAS code and the second one has the cruise control (CC) code. The CC algorithm controls the throttle and brake to make the vehicle run at a predefined constant speed. In the following, each component is explained in more detail:

(i) *Vehicle Simulator*. For simulating the real-time dynamics of a vehicle, we use a modified version of the open source TORCS [19] simulator on a PC with Ubuntu-14.04. TORCS has a precise vehicle dynamics engine and 3D visualization features.

(ii) *ECU (LKAS)*. This ECU contains the LKAS code, which receives sensing data (e.g., vehicle speed, steering angle, yaw, and lateral distance) from the vehicle simulator and sends out the steering actuation values. Infineon TC1797 MCU [20] is used with 180 MHz CPU, 4 MB Flash, and 1 MB RAM.

(iii) *ECU (Cruise Control)*. This ECU actuates the throttle and brake of the vehicle simulator to keep the vehicle at a constant speed. Since we are only interested in the LKAS performance, we just set this ECU to maintain 100 km/h speed throughout the experiment with 1 ms period. Infineon TC1796 MCU [21] with 150 MHz CPU, 2 MB Flash, and 512 KB RAM is used.

(iv) *Operation and Measurement Console*. We made a control panel using LabVIEW [22], which can control the period and deadline of the ECUs by the operator person. Also, it can gather the resulting system error and visualize it using a real-time plotting screen.

(v) *CAN Bus Interfaces*. For the real-time communication between the vehicle simulator, ECUs, and the operation and measurement console, a 500 kbps CAN bus is used. For PCs, UBS-CAN interfaces [23] are used. For ECUs, its onboard controller is used.

(iv) *Human-Vehicle Interface*. Driving wheel, throttle, and brake are installed for manual driving. Logitech G25 model [24] is used for the interface.

Using the measurement environment, we actually measure the maximum system error as varying periods and

FindOptimalPhases:
Input: $\{\tau_1, \tau_2, \ldots, \tau_n\}$, $\tau_i = (C_i, J_i(T_i, D_i))$
Output: T and D
begin procedure
(1) Set each τ_i at $(0, 0)$
(2) **while** (1) **do**
(3) Choose τ_i with the lowest $J_i(T_i, D_i)$
(4) Check the schedulability for upper and right directions
(5) **if** Case in Figure 3(a) **then**
(6) Move τ_i to the lower slope direction
(7) **end if**
(8) **if** Case in Figure 3(b) **then**
(9) Move τ_i to the lower slope direction
(10) **end if**
(11) **if** Case in Figure 3(c) **then**
(12) Move τ_i to the lower slope direction
(13) **end if**
(14) **if** Case in Figure 3(d) **then**
(15) Visit every cell in the lower slope direction and compare J_i
(16) Move τ_i to the lower J_i direction
(17) **end if**
(18) **if** the system is schedulable **then**
(19) **break**
(20) **end if**
(21) **end while**
end procedure

Procedure 1: Procedure for finding optimal T and D.

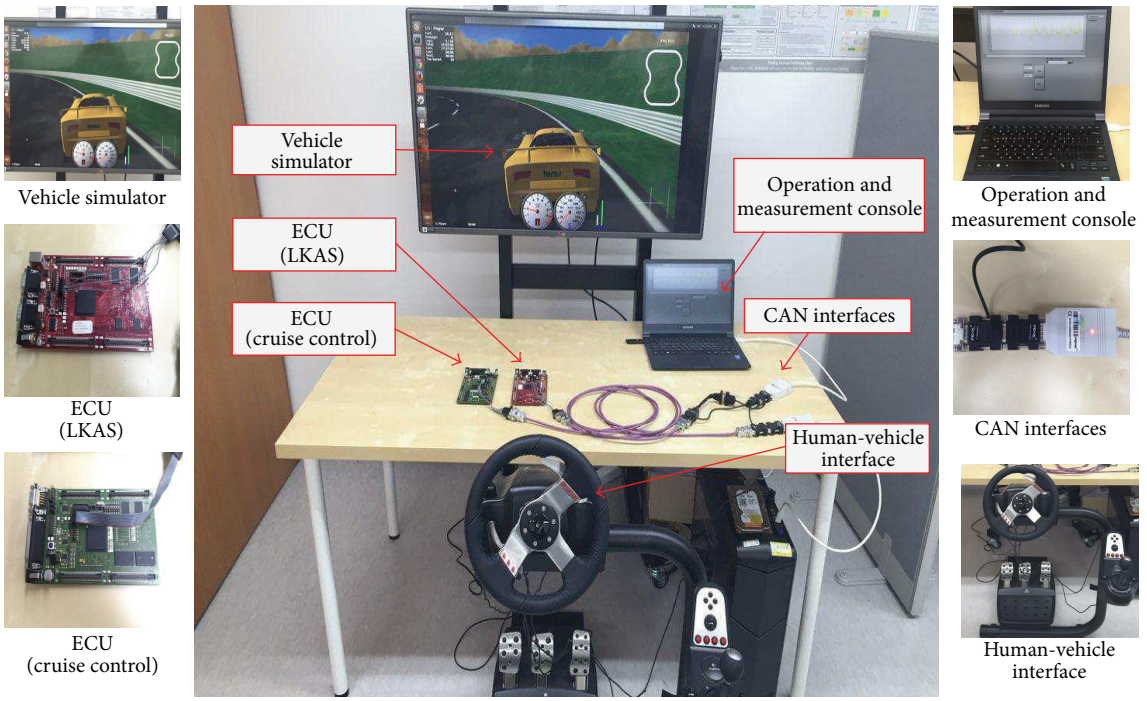


Figure 4: Measurement environment with vehicle dynamics simulator and automotive control ECUs.

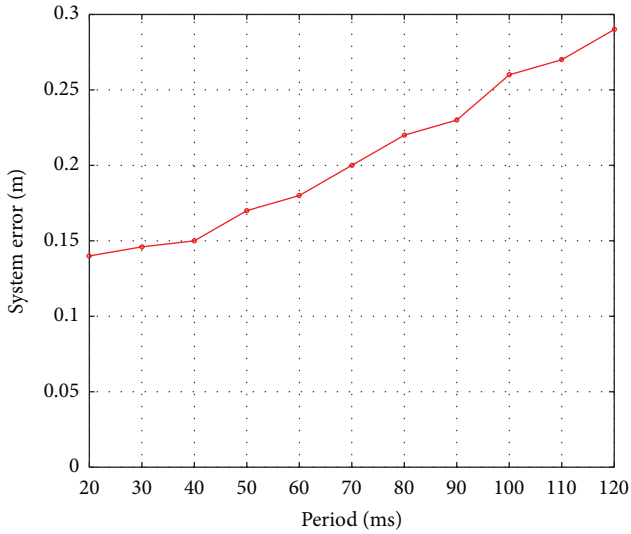


FIGURE 5: System error as varying periods with a fixed deadline.

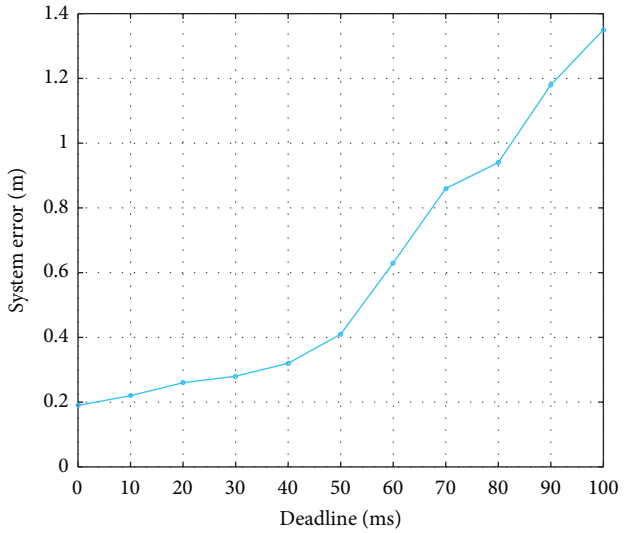


FIGURE 6: System error as varying deadlines with a fixed period.

delays. Period ranges within [0 ms, 120 ms] and deadline ranges from 0 ms to its period. The timing granularity is 10 ms for both period and delay. Figure 5 shows the system errors as varying periods with a fixed deadline at 20 ms. As shown in the figure, the system error monotonically increases as period increases, which means that larger periods have a negative impact on the control performance. Comparing the shortest period (20 ms) and the longest period (120 ms), the system error is almost doubled. Figure 6 shows a different configuration where the period is fixed at 100 ms and the delay is varying from 0 ms to its period, that is, 100 ms. By looking at the trends, we can conclude that larger deadlines have also a negative impact. Comparing Figures 5 and 6, the measured data shows that deadlines have a more significant effect on the performance than periods. Figure 7 shows the system errors as varying periods and deadlines in a 3D graph. The two axes on the floor are period and delay,

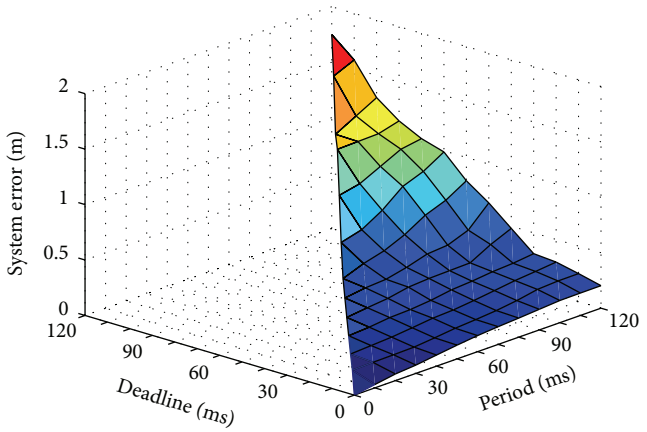


FIGURE 7: System error as varying periods and deadlines.

respectively. The vertical axis is the measured system error. From the figure, it is clearly shown that the system error is monotonically increasing in both period and delay and the delay has a more significant impact on the control performance compared to the period.

6.2. Evaluation of Our Proposed Algorithm. This subsection evaluates our heuristic algorithm in terms of optimality and computational feasibility with synthesized task sets. When generating task sets, for each task, the following were considered:

(i) C_i is an integer value which is randomly selected from the uniform distribution in the interval from 1 ms to 10 ms.

(ii) $J(T_i, D_i)$ is generated as a monotonically increasing function in T_i and D_i. The minimum and maximum of both T_i and D_i are 0 ms and 100 ms, respectively, with a granularity of 10 ms. Since we only assume the cases with $D_i \leq T_i$, a total of 66 values should be generated for each T_i and D_i pair, which are randomly chosen real values uniformly distributed in the interval from 0 to 1.

Figure 8 is an example task set $\{\tau_1, \tau_2, \tau_3\}$ with $n = 3$. For each of them, C_i is simply set to 10 ms. In the figure, note that the cells with $D_i > T_i$ are not generated since we only consider constrained deadlines. The figure also depicts how our heuristic algorithm iteratively finds the solution with an example. The initial solution is set to

$$\{\tau_1(0,0), \tau_2(0,0), \tau_3(0,0)\}, \quad (7)$$

where each tuple is (T_i, D_i) pair. Then, for each iteration, the task with the lowest $J_i(T_i, D_i)$ is chosen and the task is moved to the proper direction as explained in Section 5. Each circled number means the movement of the consecutive search iterations. The move continues until the system becomes schedulable. In this example, after 16 moves, the final solution is found, that is,

$$\{\tau_1(30,30), \tau_2(30,20), \tau_3(30,20)\}. \quad (8)$$

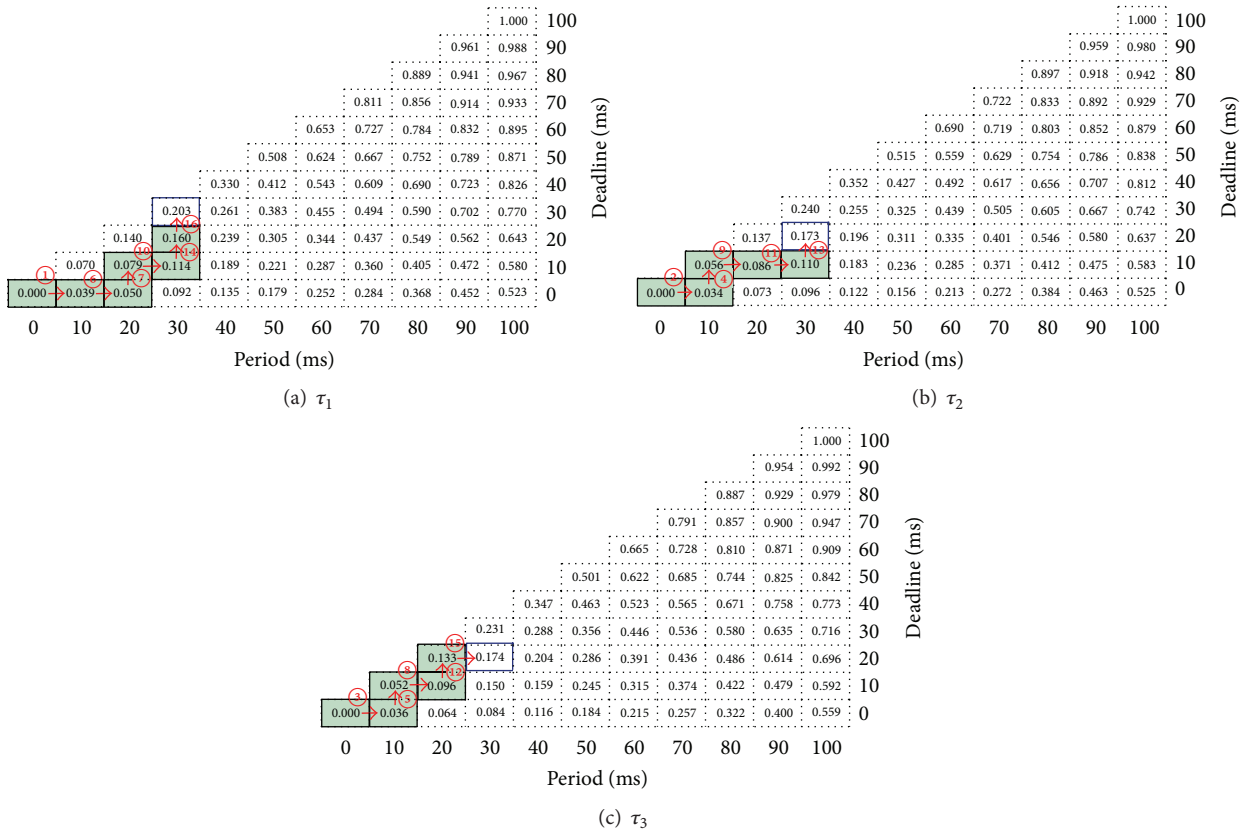


FIGURE 8: The process of heuristic algorithm.

At the final solution, $J(T, D) = 0.55 = 0.203 + 0.173 + 0.174$. Note that when deciding the overall system error, the weights w_i's are simply given 1 in the entire experiment.

With the above exemplified task set generation method and heuristic algorithm, we compare the following three approaches:

(i) Searching every possible combination of periods and deadlines checking the schedulability and overall system error. The result is optimal for the entire solution space. This approach is denoted by *Exhaustive*.

(ii) Searching only the combinations with $P_i = D_i$, which significantly reduces the solution space compared to *Exhaustive*. The result is only optimal for the solution space with implicit deadlines. This approach is denoted by *Implicit*.

(iii) Search guided by our heuristic algorithm as explained in Section 5. The result may not be optimal compared to *Exhaustive*. This approach is denoted by *Ours*.

Figure 9 shows the overall system error with the above three approaches. The number of tasks is varied from 1 to 6. For each experiment, 100 task sets are generated and the result is the average of 100 task sets. As shown in the figure, *Exhaustive* shows the best result even though the algorithm almost never ends when the number of tasks exceeds 4. The results for 5 and 6 are approximated using a curve fitting method with quadratic equations. Comparing *Implicit* and *Ours*, *Ours* wins for every six experiments. Though the difference is marginal when the number of tasks is small, note that the difference is increasing as the number of tasks increases.

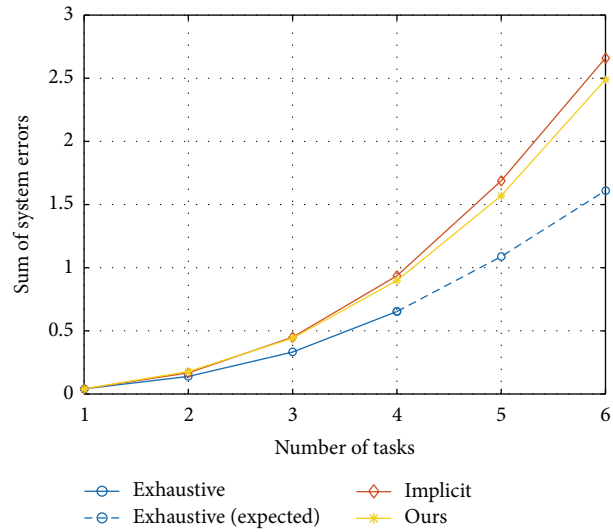


FIGURE 9: Comparison of our approach with optimal results.

Table 1 shows the required computing time for the three approaches when the number of tasks varies from 1 to 10.

Table 1: The required computing times for the three approaches with varying number of tasks.

Number of tasks	Exhaustive	Implicit	Ours
1	0.6835 ms	0.7885 ms	0.69 ms
2	13.684 ms	0.926 ms	0.88 ms
3	1305 ms	6 ms	0.75 ms
4	2 min	72 ms	0.74 ms
5	(3 hours)	986 ms	1.65 ms
6	(12 days)	1100 ms	2.44 ms
7	(1228 days)	(2 min)	16.19 ms
8	(3 years)	(26 min)	121 ms
9	(30577 years)	(5 hours)	609 ms
10	(290635 years)	(2 days)	1121 ms

The numbers in parenthesis are estimated values whereas the other numbers are actually measured. From the table, *Exhaustive* requires more than a year when the number of tasks is only 7. Even for *Implicit*, when the number of tasks is 10, which is relatively small in practice, the required computing time exceeds 2 days. Therefore, we can conclude that both *Exhaustive* and *Implicit* cannot be used as practical size task sets whereas *Ours* finds solutions even with larger task sets.

In order to prove that our heuristic algorithm produces a high-quality solution compared to other methods, we also compare *Ours* with the following two other heuristic algorithms:

(i) At each iteration, the approach chooses the moving direction with the higher slope of $J(T_i, D_i)$. This approach is denoted by *Higher*.

(ii) At each iteration, the approach chooses the moving direction with the lower slope of $J(T_i, D_i)$. This approach is denoted by *Lower*.

Note that *Ours* is an extension of *Lower*, which additionally considers the resulting schedulability as well as the slope when deciding the moving direction.

Figure 10 compares the performance of *Ours* with *Higher* and *Lower*. The result is the average of 100 synthesized task sets for each number of tasks from 1 to 6. As shown in the figure, *Ours* shows the best result compared to *Higher* and *Lower*. By comparing *Higher* and *Lower*, *Lower* shows a better result compared to *Higher*. The result first explains that taking the lower slope produces a better result than taking the higher slope. Meanwhile, *Ours* further enhances the performance by taking the schedulability into consideration as well as the slope at each iteration.

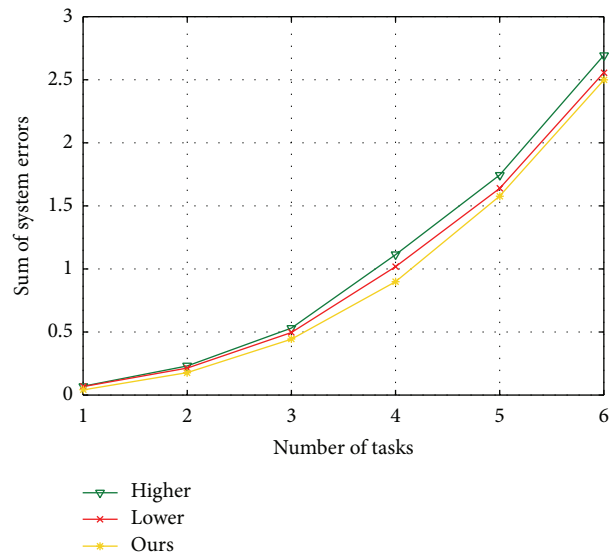


Figure 10: Comparison of our approach with other heuristic methods.

7. Conclusion

By exploiting the trade-off relation of task periods and deadlines, this paper proposes a novel task set synthesis algorithm for maximizing the overall system performance. For conducting a measurement study regarding the control performance, a simulation environment is developed, which can easily gather the performance variations of automotive control applications while applying various task periods and deadlines. Starting from the measured data, which confirms the trade-off relation, our problem is formally defined as a period and deadline selection problem. The input to our problem is each task's WCET and its measured control performance matrix with various periods and deadlines. For the scheduling, DM fixed-priority scheduling is assumed. Since it becomes quickly intractable to find the optimal solution with even relatively small number of tasks, this paper proposes a heuristic algorithm with a linear complexity that finds a high-quality suboptimal solution. Our heuristic algorithm is based on a gradient descent method with its initial solution at the smallest period and deadline for each task. Starting from the initial solution, our algorithm iteratively increases period or deadline one at a time until the task set is schedulable. For each iteration, the task and its moving direction are chosen comparing the control performance reductions.

In our future work, we consider a new system configuration where multiple implicit deadline periodic tasks collectively control a single plant. For such systems, a chain of tasks actually controls a plant from sensing to actuation. By controlling each task's sampling period, we have to indirectly control the sampling frequency and input-output delay the plant actually experiences.

Competing Interests

The authors declare that there are no competing interests regarding the publication of this paper.

References

[1] K.-E. Årzén, A. Cervin, J. Eker, and L. Sha, "An introduction to control and scheduling co-design," in *Proceedings of the 39th IEEE Confernce on Decision and Control*, pp. 4865–4870, IEEE, December 2000.

[2] F. Xia and Y. Sun, "Control-scheduling codesign: a perspective on integrated control and computing," *Dynamics of Continuous, Discrete and Impulsive Systems—Series B*, vol. 13, supplement 1, pp. 1352–1358, 2006.

[3] A. Cervin and J. Eker, "Control-scheduling codesign of real-time systems: the control server approach," *Journal of Embedded Computing*, vol. 1, no. 2, pp. 209–224, 2005.

[4] F. Xia and Y.-X. Sun, *Control and Scheduling Codesign: Flexible Resource Management in Real-Time Control Systems*, Springer Science & Business Media, 2008.

[5] A. Cervin, D. Henriksson, B. Lincoln, J. Eker, and K.-E. Årzén, "How does control timing affect performance? Analysis and simulation of timing using jitterbug and truetime," *IEEE Control Systems*, vol. 23, no. 3, pp. 16–30, 2003.

[6] Y. Wu, G. Buttazzo, E. Bini, and A. Cervin, "Parameter selection for real-time controllers in resource-constrained systems," *IEEE Transactions on Industrial Informatics*, vol. 6, no. 4, pp. 610–620, 2010.

[7] D. Seto, J. P. Lehoczky, L. Sha, and K. G. Shin, "On task schedulability in real-time control systems," in *Proceedings of the 1996 17th IEEE Real-Time Systems Symposium (RTSS '96)*, pp. 13–21, December 1996.

[8] D. Seto, J. P. Lehoczky, and L. Sha, "Task period selection and schedulability in real-time systems," in *Proceedings of the 19th IEEE Real-Time Systems Symposium (RTSS '98)*, pp. 188–198, December 1998.

[9] E. Bini and M. Di Natale, "Optimal task rate selection in fixed priority systems," in *Proceedings of the 26th IEEE Real-Time Systems Symposium (RTSS '05)*, 409, 399 pages, December 2005.

[10] E. Bini and A. Cervin, "Delay-aware period assignment in control systems," in *Proceedings of the Real-Time Systems Symposium (RTSS '08)*, pp. 291–300, December 2008.

[11] L. Tan, C. Du, and Y. Dong, "Control-performance-driven period and deadline selection for cyber-physical systems," in *Proceedings of the 10th Asian Control Conference (ASCC '15)*, pp. 1–6, Kota Kinabalu, Malaysia, May 2015.

[12] C. Du, L. Tan, and Y. Dong, "Period selection for integrated controller tasks in cyber-physical systems," *Chinese Journal of Aeronautics*, vol. 28, no. 3, pp. 894–902, 2015.

[13] G. Buttazzo, M. Velasco, and P. Marti, "Quality-of-control management in overloaded real-time systems," *IEEE Transactions on Computers*, vol. 56, no. 2, pp. 253–266, 2007.

[14] H.-J. Cha, S.-W. Park, W.-H. Jeong, and J.-C. Kim, "Performance tradeoff between control period and delay: lane keeping assist system case study," *Journal of the Korea Society of Computer and Information*, vol. 20, no. 11, pp. 39–46, 2015.

[15] B. Lincoln and A. Cervin, "Jitterbug: a tool for analysis of real-time control performance," in *Proceedings of the 41st IEEE Conference on Decision and Control*, vol. 2, pp. 1319–1324, IEEE, Las Vegas, Nev, USA, December 2002.

[16] D. Henriksson, A. Cervin, M. Andersson, and K.-E. Årzén, "Truetime: simulation of networked computer control systems," in *Proceedings of the 2nd IFAC Conference on Analysis and Design of Hybrid Systems*, Alghero, Italy, June 2006.

[17] N. Audsley, A. Burns, M. Richardson, K. Tindell, and A. Wellings, "Applying new scheduling theory to static priority preemptive scheduling," *Software Engineering Journal*, vol. 8, no. 5, pp. 284–292, 1993.

[18] EVIDENCE, "Erika enterprise manual," http://erika.tuxfamily.org/drupal/.

[19] B. Wymann, "Torcs manual installation and robot tutorial," http://www.berniw.org/aboutme/publications/torcs.pdf.

[20] Infineon, "Tc1797 user's manual," http://www.infineon.com/cms/en/product/.

[21] Tc1796 user's manual, http://www.infineon.com/cms/en/product/.

[22] National Instruments, Labview user manual, http://www.ni.com/labview/ko/.

[23] PEAK-System, "Pcan-basic parameters description," http://www.peak-system.com/PCAN-USB.199.0.html?&L=1.

[24] Logitech, Logitech g25 user manual, http://support.logitech.com/enau/product/g25-racing-wheel.

Hindawi Publishing Corporation
Mobile Information Systems
Volume 2015, Article ID 406327, 12 pages
http://dx.doi.org/10.1155/2015/406327

Research Article

A Remote Medical Monitoring System for Heart Failure Prognosis

Liangqing Zhang,[1] Cuirong Yu,[2] Chunrong Jin,[2] Dajin Liu,[2] Zongwen Xing,[3] Qian Li,[1] Zhinan Li,[4] Qin Li,[4] Yingxiao Wu,[5] and Jie Ren[6]

[1]*Shanxi Cardiovascular Hospital, Taiyuan 030000, China*
[2]*First Hospital of Shanxi Medical University, Taiyuan 030000, China*
[3]*Taiyuan City Central Hospital, Taiyuan 030000, China*
[4]*Taiyuan Maixinyun Healthcare Management Co. Ltd., Taiyuan 030000, China*
[5]*Huaxin Consulting Co., Ltd., Hangzhou 310014, China*
[6]*Shanxi Academy of Medical Sciences and Shanxi Dayi Hospital, Taiyuan 030000, China*

Correspondence should be addressed to Yingxiao Wu; wuyingxiao@126.com and Jie Ren; renjie1011@163.com

Received 16 June 2015; Revised 25 August 2015; Accepted 14 September 2015

Academic Editor: Wenyao Xu

Remote monitoring of heart disease provides the means to keep patients under continuous supervision. In this paper, we introduce the design and implementation of a remote monitoring medical system for heart failure prediction and management. The three-part system includes a patient-end for data collection, a medical data center as data storage and analysis, and a doctor-end to diagnosis and intervention. The main objective of the system is to prognose the occurrence risk of heart failure (HF) confirmed by the level of N-terminal prohormone of brain natriuretic peptide (NT-proBNP) based on the changes of the patients' (systolic and diastolic) blood pressure and body weight that are measured noninvasively in a home environment. The prediction of HF and non-HF patients was achieved by a structured support vector machine (SVM) classification algorithm. With the present system, we also proposed a scoring method to interpret the long-term risk of HF. We demonstrated the efficiency of the system with a pilot clinical study of 34 samples, where the NT-proBNP test was used to help train the prediction model as well as check the prediction results for our system. Results showed an accuracy of 79.4% for predicting HF on day 7 based on daily body weight and blood pressure data acquired over 30 days.

1. Introduction

Heart failure (HF), often known as congestive or chronic heart failure (CHF), is a common condition that develops after the heart becomes damaged or weakened by heart disease such as myocardial infarction, coronary artery, and rheumatic heart diseases [1]. HF is a life-threatening disease and addressing it should be considered a global health priority [2]. At present, approximately 26 million people worldwide are living with HF according to the statistics reported by Ponikowski et al. [3]. In many countries, population-based HF studies have shown that about 1 to 2% of people have HF and even higher proportions have been reported in single-centre studies [4]. Although many advanced techniques have been designed and used to treat HF, the risk of death is still about 35% in the year following diagnosis [5]. In China, according to the Chinese Cardiovascular Disease Report for 2013 [6], there were 4.5 million individuals living with HF and the morbidity was 0.9% for individuals between 35 and 74 years. The population of adults in the world with heart disease is continuously growing and ageing [7].

The signs and symptoms of HF are nonspecific (e.g., dyspnea, exercise intolerance, fatigue, and weakness) and often relate to other conditions such as pulmonary disease, anemia, hypothyroidism, depression, and obesity [8]. Several traditional tests and procedures are usually utilized for diagnostic assessment of HF such as blood test, B-type natriuretic peptide (BNP) [9] or N-terminal fragment of

the prohormone BNP (NT-proBNP) [10] test, electrocardiogram (ECG) [11], chest X-ray [12], echocardiogram [13], and coronary angiogram [14]. These methods require expertise from physicians, biologists, and clinicians. For example, NT-proBNP is an endogenously produced neurohormone primarily secreted by the ventricles in the heart as a response to left ventricular stretching or wall tension that occurs when HF develops and worsens. The level of NT-proBNP has been found to positively associate with age and negatively correlate to renal function [15]. No clear consensus has emerged for NT-proBNP as a diagnostic screening tool, but the age-adjusted cutting points (450 pg/mL for patients of <50 years, 900 pg/mL for patients of 50 to 75 years, and 1,800 pg/mL for patients of >75 years) appear promising and merit greater scrutiny and validation [16]. In clinical practice, the NT-proBNP test is considered an effective diagnostic method for HF and it provides guidance for HF therapy for patients with or without systolic dysfunction [17].

As known in the literature, HF is often a long-term (chronic) condition that generally worsens over time [18]. It requires frequent and costly hospitalization of patients for follow-up monitoring. As the number of patients with HF increases, clearly, there is a need to predict the occurrence of HF and thus provide early interventions with a home-based remote monitoring system to avoid long-term hospitalization or frequent NT-proBNP tests. Further, to reduce the morbidity and mortality of patients with HF and improve clinical outcome, its early prediction and prolonged monitoring are important in determining when to initiate specific therapies, in particular for patients with severe HF, such as cardiac transplantation and mechanical circulatory support [19]. Since therapies for HF are becoming more sophisticated, efforts are spent on developing more efficient prognosis of HF risk and its changes so that personalized medical needs and goals of care for each patient are coordinated and communicated in order to provide the best solution of treatments. In addition, due to the inconvenience of geographic barriers and/or economic constrains, monitoring heart health remotely appears to be a promising solution that can work at scale to improve HF prediction and reduce associated costs to both patients and hospitals. A major challenge to the realization of a remote monitoring system is the ability to collect, store, and process a large amount of data gathered from sensors in an effective, robust, and automated fashion. On the other hand, how to analyze the collected data to support the HF diagnosis or treatment is also a critical problem. The use of traditional NT-proBNP for timely diagnoses of HF usually requires a certain period of hospital stay (or frequent hospitalization) and it may not be appropriate for predicting impending HF during clinical follow-up.

To achieve remote monitoring of HF and potentially reduce patient hospitalization for testing, some physiological data that can be measured in a home environment using noninvasive sensors is required. For instance, Chaudhry et al. [20] showed that changes in body weight precede hospitalization for HF. Haider et al. [21] found that systolic blood pressure (SBP), diastolic blood pressure (DBP), and pulse pressure (PP, the difference between SBP and DBP) are predictors of risk for CHF. These studies point out that the use of body weight and blood pressure is promising for predicting future occurrence of HF and they are well-fitted to the requirements of our remote monitoring solution.

In this paper, we present the design and implementation of a remote medical monitoring system for HF prediction. The system design is an end-to-end solution including data collection, data storage and access, data analytics, and intervention feedback. The system provides prognoses of HF by estimating NT-proBNP level based on changes in blood pressure and body weight using machine learning methods, where a total of 29 features are extracted. To verify the effectiveness of the system, the NT-proBNP test is used as an aid to model the HF predictor estimates and to check the prognosis results. With this system, patients' heart health can be remotely monitored.

The rest of this paper is organized as follows: Section 2 introduces related work on the remote monitoring for HF diagnosis/prognosis. Section 3 provides an overview of the system and its design principles. In Section 4, data collection, feature extraction and analysis, HF prediction algorithms, computation of a risk score for HF, and a pilot clinical trial are described. Section 5 presents the results of the prediction of HF and the computation of its risk score using our system and Section 6 discusses this study. Finally, Section 7 concludes this work.

2. Related Work

The conventional concept of a remote monitoring system in healthcare is often achieved by means of a telephone-based interactive voice response system. It usually collects patients' daily information such as associated symptoms, feelings, and habits by asking questions. Afterwards, the collected information is then reviewed by clinicians. Chaudhry et al. [22] established a large trial on telemonitoring with 1653 patients enrolled, where 826 were randomly assigned to undergo telemonitoring and 827 to receive usual care. Telemonitored patients were required to make daily calls for six months. During each call, the patients were asked a series of questions about their general health and HF symptoms. The results showed that the telemonitoring strategy on a multicenter trial with a large database failed to provide a benefit over usual care and further strategies were needed to improve HF outcomes.

Suh et al. [23, 24] developed a system to remotely monitor patients with CHF, which had a three-tier architecture consisting of pervasive biosensors, a web server, and a back-end database. It acquired four health-related measures: weight, blood pressure, physical activity, and the Heart Failure Somatic Awareness Scale (HFSAS) [25] that reflect the most common signs and symptoms of CHF. Besides, their system can help the patients with guidance and feedback via text messages or emails.

Other researchers used a CardioMEMS [26] heart sensor in HF patients (Class-III according to the New York Heart Association [NYHA] guidelines [27]) to undertake a

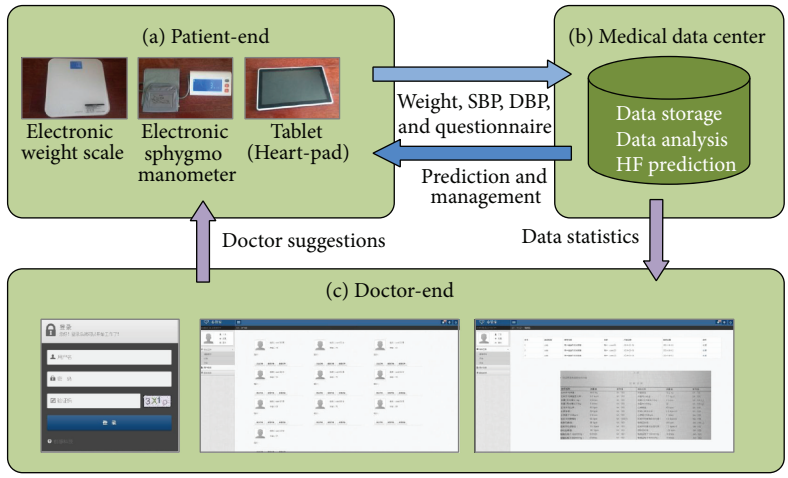


Figure 1: Remote medical monitoring system architecture, where the system is an end-to-end system consisting of (a) a patient-end, (b) a medical data center, and (c) a doctor-end. SBP: systolic blood pressure, DBP: diastolic blood pressure.

single-blind trial [28]. The patients were managed with a wireless implantable hemodynamic monitoring system that collected pulmonary artery pressure. All patients in both treatment and control groups took daily pressure readings and then these measurements were transmitted through a modem to a database. Subsequently, the rate of HF-related hospitalizations was reduced by 37% in the treatment group. The potential limitations of the trial included the maintenance of patient masking and minimization of the effects of investigator-patient and device-patient interactions on HF outcomes.

A conceptual model for HF disease management (HFDM) was proposed in 2014 by Andrikopoulou et al. [29]. HFDM encompassed ongoing patient education and enhancement of self-care behaviour, complemented by data derived from device diagnostics. It had three basic mechanisms: (1) implementation of strategies that modify patients' baseline risk; (2) monitoring of worsening signs and symptoms; and (3) encouragement of patient participation in their own care. This model has been proposed as a way to systematically identify the risk factor for HF and, accordingly, prevent associated mortality.

Bui and Fonarow [30] reviewed some clinical trials and tested different HF monitoring strategies. They suggested several future challenges and opportunities in home-based hemodynamic monitoring such as evaluating HF monitoring in a broader population and in more diverse clinical settings, better defining optimal population for monitoring, studying long-term reliability and safety, and analyzing cost-effectiveness of home-based monitoring. Similarly, Bhimaraj [31] made an evidence-based review of various home-based monitoring systems for HF patients, including monitoring with telephone, portable technology, wearable sensors, and implantable cardioverter defibrillators (ICD) or cardiac resynchronization therapy (CRT) devices. They mentioned that the explosion of social media and smart-phone applications is a potentially untapped resource in creating a patient-centered system in the future.

Currently, many remote monitoring systems exist for HF assessment, where some of them use invasive and implanted sensors [28] and others monitor daily information to analyze HF via the Internet. Our proposed three-part monitoring system not only builds on existing technologies but also introduces new functions that circumvent their shortfalls. The remote system can achieve the diagnosis/prognosis of the occurrence of HF and provide interventions to patients for self-care treatment.

3. System Overview and Design Principles

3.1. System Architecture. The remote medical monitoring system for HF is built with three parts as shown in Figure 1. The first part is patient-end for acquiring data and sending/receiving feedback. It includes noninvasive sensing devices used for measuring body weight and systolic/diastolic blood pressure and a tablet with an "end-user" application (App) used for collecting questionnaire answers and interacting with patients. The second part is medical data center, to which data are sent from the patient-end. The medical data center stores the collected data, performs statistical data analysis and HF prediction, and generates data statistics and patent reports that can be delivered to doctors and patients. The third part is doctor-end, through which patient reports are sent to medical doctors who can then provide interventions/suggestions to patients according to their (current and previous) data statistics and reports.

These three parts form a circle so that the remote medical monitoring system can achieve early detection of HF as well as providing timely interventions. The system has several notable features. First, it can assess the patient's condition by intelligently analyzing the physiological data. Second, it provides a real-time platform for asking and answering questions between doctors and patients. Third, it offers historical tracking for the patient's heart health-related

Figure 2: Example pages of patient GUI, implemented in an Android-based tablet: (a) a main page, (b) a data collection page, (c) an online questionnaire page, and (d) a report page.

data and medical records. Below are more details related to implementation of the system.

3.2. System Hardware and Software

3.2.1. System Hardware. As illustrated in Figure 1, the system includes three hardware devices:

(i) An electronic weight scale measuring body weight (kg).

(ii) An electronic sphygmomanometer measuring systolic blood pressure (SBP, mmHg) and diastolic blood pressure (DBP, mmHg).

(iii) A tablet "Heart-pad" which is a platform between patients and the system/doctors with an Android operation system including functions of collecting physiological data from the other two devices, collecting questionnaire data from patients, asking questions to doctors (by patients), and sending feedback to patients (by the system/doctors).

Note that all three hardware devices are connected using embedded Bluetooth 4.0 modules.

3.2.2. System Software. Several different types of software are implemented to achieve the prediction of HF occurrence. They are

(i) a date collection software in the tablet that serves to control the acquisition of body weight and blood pressure data and automatically upload the data to the medical data center;

(ii) a questionnaire generation software that automatically generates a certain number of questions for patients every day;

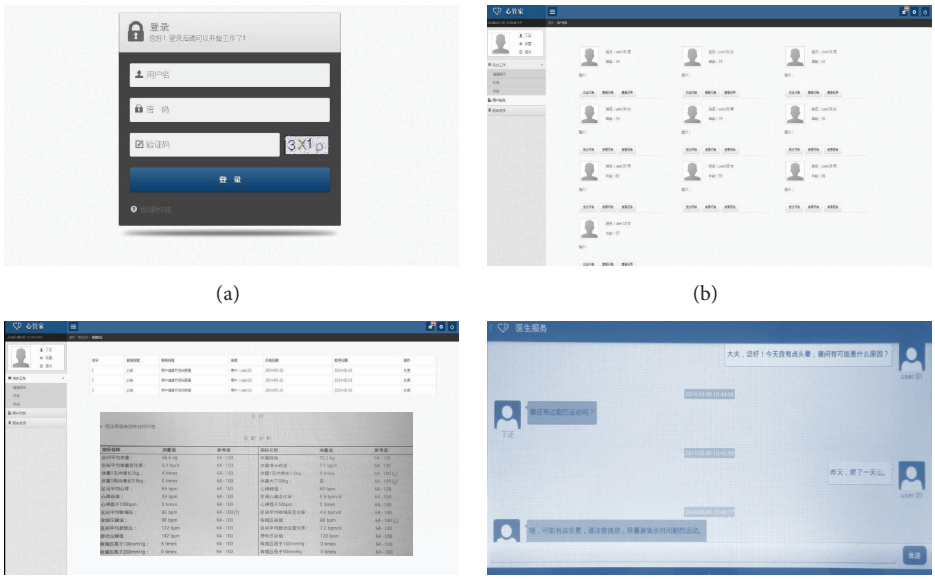
(iii) a prediction software that employs data mining algorithms (comprising feature extraction and structured support vector machine [SVM] classification) to analyze historical data and predict the future likelihood of the HF occurrence and thereafter computes a score indicating the HF risk (see Section 4.2);

(iv) a reporting software that visualizes some data and generates reports delivered to patients, where each report includes, for example, questionnaire statistics, daily changes of body weight and blood pressure (SBP and DBP), HF risk score (HFRS) variation, and suggestions by medical doctors and/or the system itself;

(v) a doctor-patient interaction software that provides a real-time platform for asking and answering questions between doctors and patients.

3.2.3. Graphical User Interface. In the system, we design two graphical user interfaces (GUIs), one for patients (patient-end) and the other for doctors (doctor-end).

(i) Patient GUI: Figure 2 illustrates some example pages of our patient GUI, implemented in an Android-based tablet. In the main page, patients can see different functions of the system and choose what they want to perform (Figure 2(a)). For example, they can measure their weight and blood pressure by following a voice instruction (Figure 2(b)), answer questions online (Figure 2(c)), or read the report



(a) (b) (c) (d)

Figure 3: Example pages of doctor GUI, implemented in a desktop computer: (a) a login page, (b) a patient management page, (c) a data checking page, and (d) a doctor-patient interaction page.

with suggestions generated by the system or a doctor (Figure 2(d)).

(ii) Doctor GUI: some example pages are shown in Figure 3, where doctors can log in to the system through a desktop computer (Figure 3(a)), search and manage patients (Figure 3(b)), check the patients' data (Figure 3(c)), and communicate with patients instantly where they may receive messages from their patients (Figure 3(d)).

4. Heart Failure Prediction and Heart Failure Risk Score

4.1. HF Prediction

4.1.1. Physiological Data. As mentioned above, we consider physiological data including body weight, SBP, and DBP on a daily basis for predicting the risk of occurring HF in consecutive 7. The prediction is based on modeling the objective collected during a 30-day period. To examine our prediction model, patients need to be classified as with or without a high risk of HF occurrence according to their age-adjusted NT-proBNP obtained on day 7 after 30 days of blood pressure and body weight monitoring. Therefore, the HF prediction is considered as a binary (predictive) classification problem, that is, classification of HF and non-HF patients using their past data. Figure 4 compares the body weight, SBP, and DBP for a period of 30 days for HF and non-HF patients. It illustrates that the variations of body weight, SBP, and DBP for HF patients seem greater than those for non-HF patients.

4.1.2. Questionnaire. In addition to the physiological data, our remote medical monitoring system also collects questionnaire data where 5 to 8 questions are sent to patients (i.e., to their Heart-pad terminal). The questions are selected from a large questionnaire pool of more than 100 questions. The questions are chosen by doctors or nurses according to the patient's medical history. The patients are required to respond to them and the answers are then automatically uploaded to the medical data center. The use of a questionnaire is to assist doctors with understanding the objective data, provide suggestions, and make recommendations for further actions. Some example questions related to HF symptoms and identified as relevant based on medical data from clinical practice are shown as follows.

An Example of Questionnaire

(1) Do you suffer from lack of appetite?

(2) Do you sweat or feel nauseous?

(3) Do you experience sudden confusion or have trouble speaking?

(4) Do you experience severe headache?

(5) Do you have shortness of breath (dyspnea)?

(6) Do you feel fatigued?

(7) Do you have physical restrictions or limitations?

(8) Do you have symptoms of depression, loss of interest in activities, changes in sleep, loss of energy, and feelings of hopelessness?

4.1.3. Prediction Framework. With the acquired physiological data, we designed a framework to automatically predict/classify the occurrence of HF and test our classifier. It comprises a training and a classification stage. In the training stage, a set of features are extracted from training data and are used to train the classification model. In the

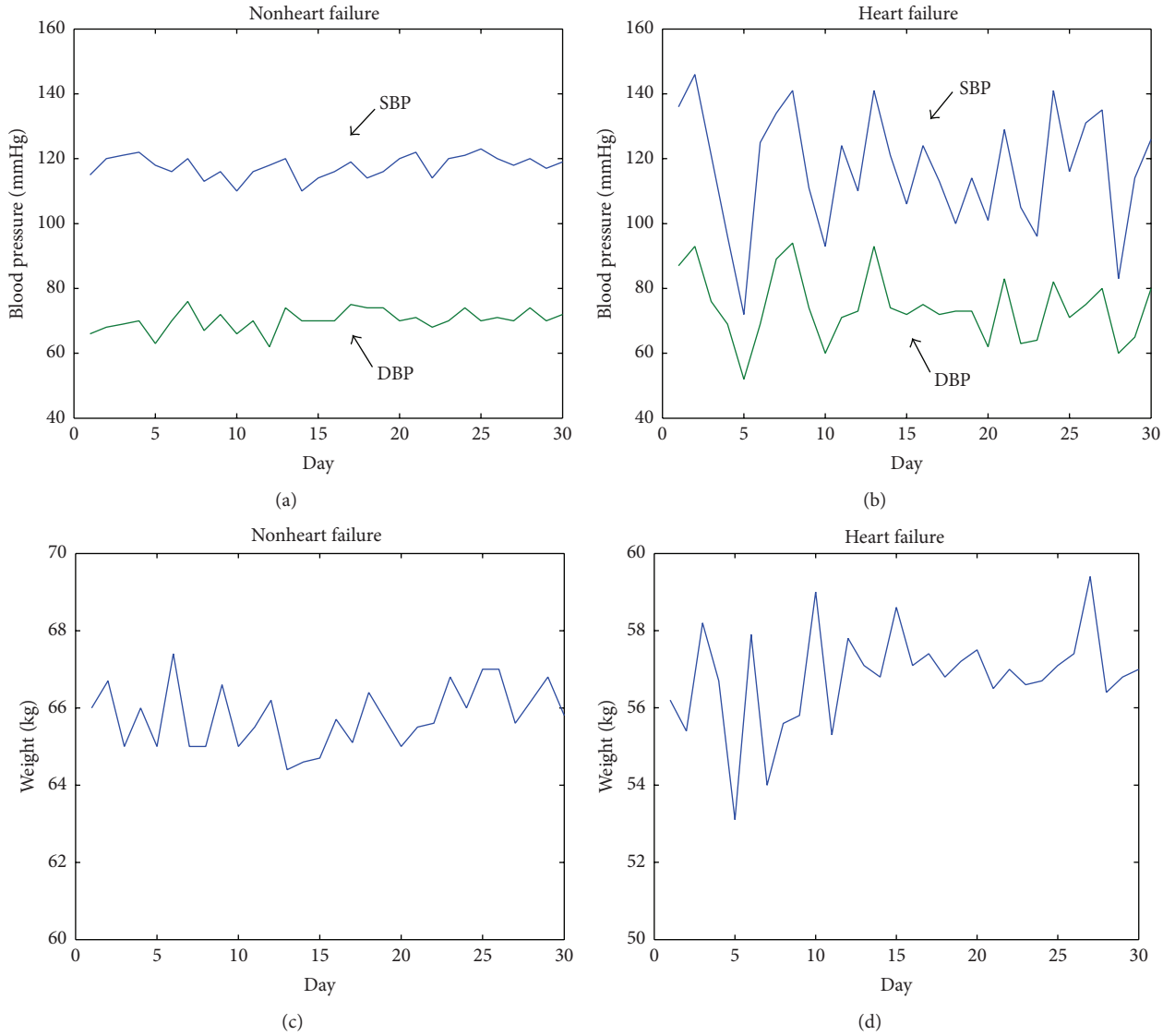


Figure 4: Comparison of body weight, SBP, and DBP measures of 30 days for non-HF and HF patients.

classification stage, features are extracted and classification is performed based on the trained model. After that, results are compared with the annotations of the test samples. The prediction framework with different blocks is illustrated in Figure 5.

4.1.4. Features Extraction and Discriminative Capability. In order to predict the occurrence of HF, we extract a total of 29 features that are characteristic for HF. They are extracted from body weight, SBP, DBP, and PP (pulse pressure, computed as the difference between SBP and DBP) values over a certain period. Table 1 lists all of the 29 features and definitions.

It is important to understand which features are more informative in detecting HF. To quantify the feature discriminative capability, a Mahalanobis distance (MD) metric [32] can be used. For a single feature, it computes the absolute standardized mean difference between two classes. The formula for calculating MD is given by

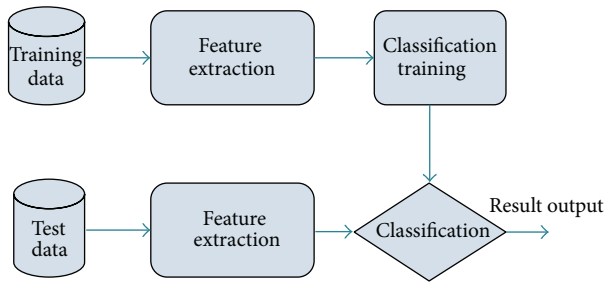


Figure 5: HF prediction framework.

$$\mathrm{MD} = \frac{\left|\mu_p + \mu_n\right|}{\sigma}, \tag{1}$$

Table 1: List of features for HF prediction.

Index	Feature	Description
		Body weight features
1	W_{mean}	Mean weight
2	W_{max}	Maximum weight
3	W_{std}	Std. of weight
4	W_{in2}	Daily weight increase > 1 kg
5	W_{in3}	Daily weight increase > 1.5 kg
6	W_{in}	Mean of daily weight increase
7	W_{instd}	Std. of daily weight increase
8	W_{inmax}	Maximum of daily weight increase
		Systolic blood pressure (SBP) features
9	SBP_{mean}	Mean SBP
10	SBP_{max}	Maximum SBP
11	SBP_{min}	Minimum SBP
12	SBP_{std}	Std. of SBP
13	SBP_{200}	SBP > 200 mmHg
14	SBP_{160}	SBP > 160 mmHg
15	SBP_{100}	SBP > 100 mmHg
16	SBP_{90}	SBP < 90 mmHg
17	SBP_{85}	SBP < 85 mmHg
18	SBP_{in}	Mean daily SBP increase
19	SBP_{instd}	Std. of daily SBP increase
		Diastolic blood pressure (DBP) features
20	DBP_{mean}	Mean DBP
21	DBP_{max}	Maximum DBP
22	DBP_{min}	Minimum DBP
23	DBP_{std}	Std. of DBP
24	DBP_{in}	Mean daily DBP increase
25	DBP_{instd}	Std. of daily DBP increase
		Pulse pressure (PP) features
26	PP_{mean}	Mean PP
27	PP_{std}	Std. of PP
28	PP_{corr}	Corr. between SBP and DBP
29	PP_{incorr}	Corr. between daily SBP and DBP increase

where μ_p is the mean of positive class, μ_n is the mean of negative class, and σ is the population standard deviation. A larger MD means that the feature is more informative in separating the two classes. Note that we consider HF and non-HF as positive and negative class, respectively.

4.1.5. Prediction Model. As stated, a well-known structured SVM classifier [33] is employed to classify HF and non-HF patients in this work. The training principle behind SVM is to find the optimal linear hyperplane such that the expected classification error for unseen samples is minimized. Here the SVM kernel for classification is a polynomial with order 3. More details about the structured SVM classification algorithms can be found elsewhere [33].

4.1.6. Evaluation Metrics and Cross-Validation. The evaluation metrics of classification performance used in this work include overall accuracy, sensitivity (or recall), specificity, and precision (or positive predictive value, PPV). Overall accuracy is computed as the ratio of correctly classified patients to the total number of patients; specificity is a measure that indicates the proportion of correctly classified actual negatives; sensitivity is the proportion of correctly classified actual positives; and precision is computed as the ratio of true positives to true positives plus false positives. These metrics can be computed from a confusion matrix (Table 2) generated based on the classification results such as

Table 2: Confusion matrix.

Confusion matrix	Prediction	
	Positive	Negative
Real class		
Positive	True positive (TP)	False negative (FN)
Negative	False positive (FP)	True negative (TN)

(i) accuracy = (TP + TN)/(TP + TN + FP + FN) × 100%;

(ii) sensitivity = TP/(TP + FN) × 100%;

(iii) specificity = TN/(FP + TN) × 100%;

(iv) precision = TP/(TP + FP) × 100%.

Additionally, a receiver operating characteristic (ROC) curve (plotted as sensitivity versus one minus specificity) of the results can be used to offer an overview of the classifier's performance. In general, a larger area under the ROC curve (AUROC) corresponds to a better performance.

To validate the classifier without biasing the prediction results, we apply a leave-one-out cross-validation (LOOCV) with 34 iterations. During each iteration of the LOOCV procedure, 33 patients are used to train the classifier and the remaining one is used for testing. The above-mentioned evaluation metrics are then computed based on the classification results after LOOCV.

4.2. Heart Failure Risk Score. For the purpose of providing meaningful information to patients regarding their HF risk, we create a scoring system that is able to quantify the risk of the future occurrence of HF, yielding an HF risk score (HFRS). In fact, the structured SVM classifier also delivers posterior probability when assigning classes for decision-making. This posterior probability indicates the probability of a sample being classified as a specific class, which associates with the likelihood of the occurrence of HF. Therefore, this output can then be used to generate a score that indicates the HF risk.

This work proposes to determine HFRS values based on the analysis of classification results (i.e., specificity and sensitivity). We use a simple piecewise linear function with thresholding three "levels" (1 to 3) to map a posterior probability into an HFRS ranging from 0 to 100. The posterior probability range of the first level should ensure that all the patients with their posterior probabilities falling in this range are correctly classified as non-HF, indicating a low risk of HF. Similarly, the posterior probability range of the third

level should make sure that all the patients who have their posterior probabilities in this range are correctly classified as HF, indicating a high risk of HF. The other posterior probability values outside the first and the third level ranges are in the second level, indicating a moderate risk of HF where misclassifications may occur. The purpose of using an HFRS range between 0 and 100 is to provide a comprehensive score to patients so that they can easily understand and assess their risk level.

4.3. A Pilot Study. To evaluate our system, a pilot clinical trial was performed based on a total of 34 Chinese patients (18 females, age 67.2 ± 8.3 years) who have been clinically diagnosed to have heart disease (coronary artery, pulmonary, or rheumatic heart disease, dilated cardiomyopathy, cardiac arrhythmia, or myocardial infarction). The confirmation of heart disease diagnosis was done through blood tests, chest X-ray, echocardiogram, ECG, ejection fraction, angiogram, cardiac computerized tomography scan, and/or magnetic resonance imaging. In clinical practice, the patients were classified as having different severity levels of heart function (from stage-A to stage-D) according to the updated guidelines of the American College of Cardiology/American Heart Association (ACC/AHA guidelines) [34], where one patient was in stage-A (at high risk for HF in the future but no structural heart disease or HF symptoms), seven were in stage-B (with structural heart disease but no HF signs or symptoms), 19 were in stage-C (underlying structural heart disease with previous or current HF symptoms), and seven were in stage-D (refractory HF requiring specialized inventions or hospital-based support).

In this pilot study, we continuously measured the patients' body weight, SBP, and DBP as well as questionnaire data on a daily basis for 30 days, where the measurements were performed before their breakfast in the morning in order to minimize the effect of food intake. Patients were then asked to measure their NT-proBNP value in a hospital 6 days later (on day 7), yielding annotated 22 HF and 12 non-HF patients. The data was collected in Shanxi Cardiovascular Hospital, Taiyuan, China, during the period between April and August, 2014. The devices in our remote medical monitoring system with installed software were provided by Sennotech Inc., China [35].

Here we considered two prediction schemes including predicting HF based on the physiological data during the past 30 days and during the past 7 days. This served to investigate the effect of using different historical data lengths on the ultimate classification performance. In other words, we intended to know if we could obtain acceptable prediction results when patients participated their remote monitoring for only a week. As mentioned, the prediction results were verified by an NT-proBNP test. The purpose of this pilot study was to preliminarily examine the usefulness of our system from HF prediction and implementation perspectives.

Table 3: Results of HF prediction in a pilot study.

Data	Precision	Sensitivity	Specificity	Overall accuracy
7 days	83.4%	63.6%	75.0%	67.6%
30 days	83.3%	77.3%	89.5%	79.4%

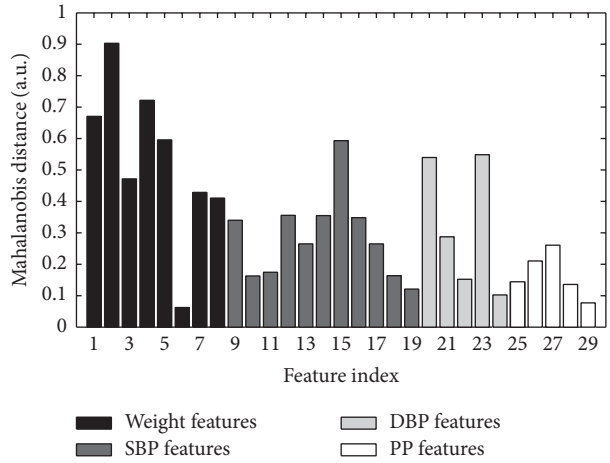


Figure 6: Feature discriminative capability.

5. Results

Figure 6 shows the discriminative capability (as measured by Mahalanobis distance [MD]) of all the 29 physiological features (computed based on past 30 days). It indicates that the mean and maximum body weights, the daily weight increases larger than 1 and 1.5 kg, the number of SBP larger than 100 mmHg, and the mean and standard deviation of DBP have higher discriminative capabilities than the other features. This is because patients with a higher weight, a larger daily weight increase, and larger variations in blood pressure will have a higher possibility for HF to occur after a week.

As shown in Table 3, our system with an SVM classifier achieved an HF prediction accuracy of 79.4% when using the daily weight and blood pressure measures collected during the past 30 days, where the specificity, sensitivity, and precision are also presented. However, the accuracy decreased to 67.6% when only using the past 7 days' data, which indicates that including more historical data with a longer period can improve the prediction performance. This can also be observed by looking at their ROC curves (Figure 7), where the curve obtained with 30 days' data has a larger "area under the ROC curve" compared with using data from the past 7 days.

As stated before, the HFRS scoring can be implemented by converting the SVM posterior probability outputs to HFRS values via a piecewise linear mapping method. Based on the posterior probabilities and the prediction results, we generated the linear mapping for the three different levels in the following. For the first level (Level 1), the posterior probability from 0 to 0.15 was linearly mapped to HFRS from 0 to 50, where the specificity was 100%. Patients classified at this level have a low risk of HF and they are

Table 4: Heart failure risk score.

Level	Posterior probability	HFRS	Status of weight	Status of blood pressure	Suggestions or interventions
1 low risk	0–0.15	0–50	Small variability in weight change and daily weight increase/loss	Small variability in SBP and DBP changes and their daily increases; normal PP (around 40 mmHg)	Keep on monitoring and provide lifestyle-related suggestions (e.g., improving living habits) according to questionnaire data
2 moderate risk	0.15–0.25	50–80	Moderate and increasing variability in weight changes and daily weight increases	Moderate and increasing variability in SBP and DBP changes and their daily increases; occurrence of some large daily increases in SBP or DBP; relatively low PP (<40 mmHg)	Suggestions and interventions are needed from doctor after checking patient's physiological statistics and questionnaire data
3 high risk	0.25–1	80–100	Large variability in weight change and daily weight increase/loss	Large variability in SBP and DBP changes and their daily increases; occurrence of many large daily increases in SBP or DBP; fast SBP drops; low PP (<20 mmHg) and large PP variability	Strongly suggest patients to be hospitalized

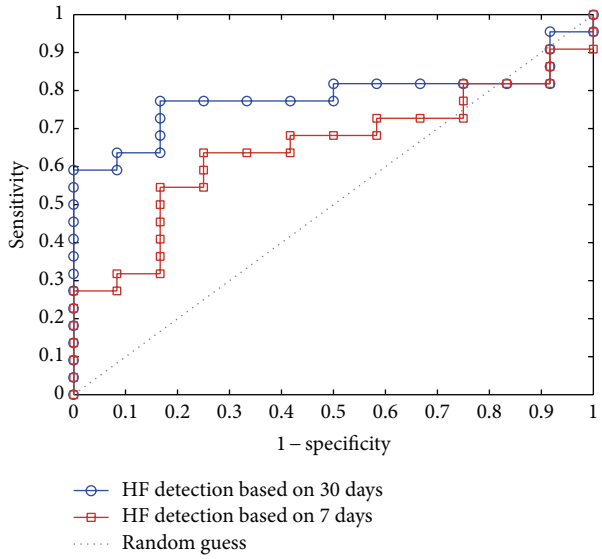


Figure 7: ROC curves of HF prediction using 30 days' and 7 days' data.

asked to continue monitoring. In the third level (Level 3), the posterior probability between 0.25 and 1 was positively linearly correlated to an HFRS value between 80 and 100 where the sensitivity was 100% in this range. This is a high risk level where it is recommended that patients go to the hospital. The second level (Level 2) corresponds to the linear mapping between posterior probability (0.15 to 0.25) and HFRS from 50 to 80, indicating a moderate risk of HF so that patients will receive feedback and interventions from data according to their measured (physiological and questionnaire) data. In these three levels, the body weight and blood pressure also behaved differently. Table 4 summarizes the HFRS levels.

For each patient, as long as the occurrence of HF is predicted and thereafter the HFRS is computed, a report (called HFRS report) with the HFRS score, some visualized (body weight and blood pressure) data, and suggestions either from our remote medical monitoring system or from a medical doctor will be delivered to his/her Heart-pad via (wireless) Internet connection. Figure 8 shows an example of an HFRS report.

6. Discussion

In our clinical trial, we executed a pilot study with only 34 patients. This small data set might lead to limitations on getting an accurate HF predictor and adequate ranges for computing HFRS levels. Therefore, an expanded clinical trial with more patents involved is necessary for future studies. Additionally, including a longer period of data (e.g., 3 to 6 months) may help improve the prediction results and the number of days is needed to obtain a converged performance.

Although the prediction results (Table 3) are far better than random guess, they are still under our expectations. Extracting more advanced features that can express more HF-related pathophysiological information is ideal in predicting its future risk, such as nonlinear entropy-based measures for single source [36] and multiple sources [37]. Here the SVM-based algorithm was used to perform HF and non-HF classification. Although it is an advanced algorithm and has been applied successfully in many different areas, other classifiers (e.g., logistic regression, random forest, and neural networks) still merit further investigation and comparison.

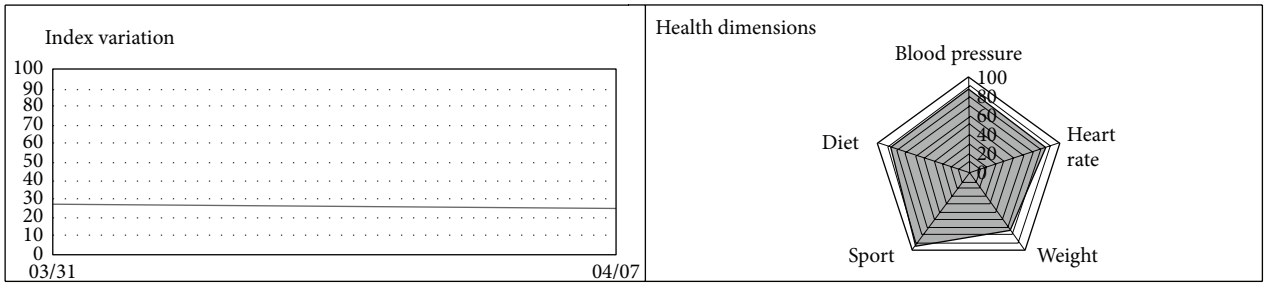
Our monitoring system requires patients to interact with it regularly (daily). Although the data collection and transmission are automatic with patient input, it will still be a challenge for patients to comply even if they have risk of HF. Patient compliance with our system should be further studied and analyzed. To reduce variations from patient to patient and between days when collecting their physiological data, we required them to measure their data at a fixed time (i.e., in the morning before lunch). However, this may result

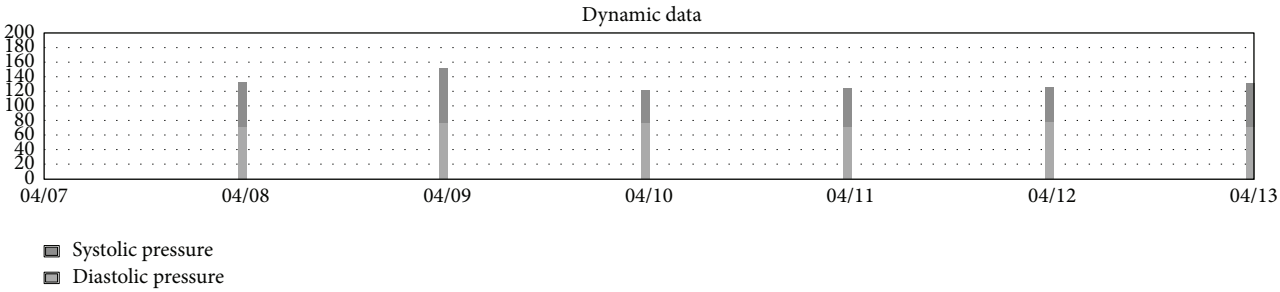
HFRS report

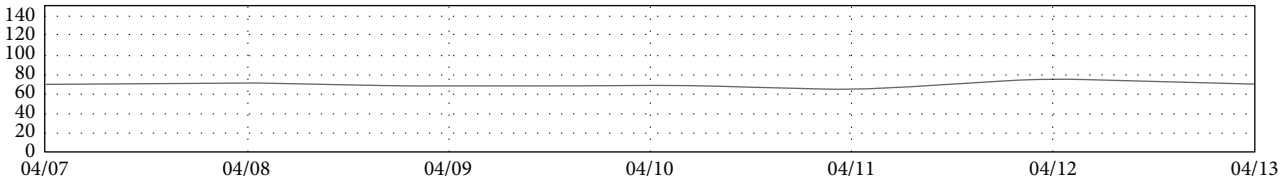
No.: XXXX1404140001

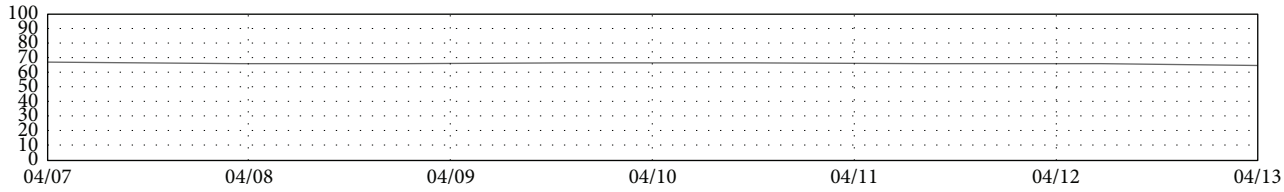
Name: Shengzhou Guo	Gender: Male	Age: 72	ID: 140102194005271819
Questionnaire	Did you drink last week? Yes: 1 time, no: 6 times How is your aerobic exercise frequency (e.g., walking, running, bicycle riding) last week? 2–4 times a week Did you feel dizzy last week? Yes: 3 times, no: 4 times Did you feel short of breath last week? Yes: 2 times, no: 5 times		

HRFS









Doctor suggestion

(I) Reduce the use of high-fat food and sweet food.
(II) Quit smoking and wine.
(III) Keep routine and a good mood.

Time of report: 14-4-2014 Nurse: Mei Li Doctor: Xiaodong Zhang

Figure 8: An example of HFRS report (translated from Chinese to English).

in inconvenience for patients or patient failure to measure data, therefore yielding missing data. Offering alternatives for the time of day patients which are required to collect data may help resolve this time effect in our prediction model. For this purpose, using a larger data set with more data for each patient measured at different times is suggested for future studies.

As stated, this present study only focused on HF for patients with diagnosed heart diseases. However, with our system, it is still unclear how to simultaneously process all the clinical data, many of which might lead to potentially conflicting alerts when the alerts relate to various unexpected comorbid conditions, resulting from medications. For example, for elderly patients with atrial fibrillation and concomitant HF, a rate-limiting calcium channel blocker would be effective for the first condition but potentially dangerous for the second one.

The future occurrence of HF in this study was estimated with NT-proBNP text. This would not be the most accurate parameter as a "ground-truth" of the occurrence of HF. Future study must validate our system based on the historical noninvasively measured data (body weight and blood pressure) for patients who actually have HF rather than using the NT-proBNP-based parameter that only indicates HF risk.

Finally, since the HFRS is based on the HF prediction results which are a data-driven scoring method, it needs to be further validated with more long-term clinical data and corrected by doctor's expertise in clinical practice. As mentioned, the patients enrolled in this study included those with several different types of heart diseases which should be taken into account during HF prediction in future work. In addition to the heart disease type, the patients might not statistically represent the whole population, where ethnic group and patient demographics (e.g., age and gender) would likely influence the prediction models. In order to improve the HF prediction performance and to provide patient-specific interventions by achieving a personalized (prediction and intervention) system, analyzing a broader range of more ethnically diverse groups needs to be further investigated.

7. Conclusion

An effective remote medical monitoring system for heart failure (HF) prediction and management was designed and implemented. The system realized early prediction (or prognosis) of future HF occurrence by estimating future NT-proBNP level based on a patient's historical data (body weight and blood pressure), where data were obtained remotely using noninvasive devices (i.e., a Bluetooth-based weight scale and sphygmomanometer). A Heart Failure Risk Score (HFRS) was proposed to evaluate the risk of the future occurrence of HF based on the prediction results, where the HFRS scoring was designed so that it could be easily understood and perceived by patients. This system optimizes early stage delivery of multiple suggestions/interventions to patients in different risk levels. This end-to-end system can also be used to manage patients and their data by doctors or by the system's data center. To validate our system, a set of real-life data from 34 patients was collected in a pilot clinical trial. Our HF prediction algorithms achieved an overall accuracy of 79.4% and 67.6% when using the data collected over a 30-day and 7-day period, respectively. Therefore, from a clinical perspective, this system is promising to help reduce morbidity and mortality caused by HF and therefore improve the clinical outcome.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] K. Swedberg, J. Cleland, H. Dargie et al., "Guidelines for the diagnosis and treatment of chronic heart failure: executive summary (update 2005)," *European Heart Journal*, vol. 26, no. 11, pp. 1115–1140, 2005.

[2] A. L. Clark, "Origin of symptoms in chronic heart failure," *Heart*, vol. 92, no. 1, pp. 12–16, 2006.

[3] P. Ponikowski, S. D. Anker, K. F. AlHabib et al., "Heart failure: preventing disease and death worldwide," *ESC Heart Failure*, vol. 1, no. 1, pp. 4–25, 2014.

[4] M. R. Cowie, S. D. Anker, J. G. F. Cleland et al., *Improving Care for Patients with Acute Heart Failure: Before, During and After Hospitalization*, Heart Failure Association of the ESC, Oxford, UK, 2014, http://www.oxfordhealthpolicyforum.org/files/reports/ahf-report.pdf.

[5] National Clinical Guideline Centre (UK), *Chronic Heart Failure: National Clinical Guideline for Diagnosis and Management in Primary and Secondary Care*, National Clinical Guideline Centre, Oxford, UK, 2010.

[6] National Center for Cardiovascular Diseases, *Report on Cardiovascular Disease in China 2013*, National Center for Cardiovascular Diseases, Beijing, China, 2014, http://www.healthyheart-china.com/.

[7] Chronic diseases and their risk factors, http://www.smartglobalhealth.org/issues/entry/chronic-diseases.

[8] C. Gutierrez and D. G. Blanchard, "Diastolic heart failure: challenges of diagnosis and treatment," *American Family Physician*, vol. 69, no. 11, pp. 2609–2616, 2004.

[9] J. A. Doust, P. P. Glasziou, E. Pietrzak, and A. J. Dobson, "A systematic review of the diagnostic accuracy of natriuretic peptides for heart failure," *Archives of Internal Medicine*, vol. 164, no. 18, pp. 1978–1984, 2004.

[10] J. L. Januzzi, R. Van Kimmenade, J. Lainchbury et al., "NT-proBNP testing for diagnosis and short-term prognosis in acute destabilized heart failure: an international pooled analysis of 1256 patients—the International Collaborative of NT-proBNP Study," *European Heart Journal*, vol. 27, no. 3, pp. 330–337, 2006.

[11] A. P. Davie, C. M. Francis, M. P. Love et al., "Value of the electrocardiogram in identifying heart failure due to left ventricular systolic dysfunction," *British Medical Journal*, vol. 312, no. 7025, article 222, 1996.

[12] N. Gadsboll, P. F. Hoilund-Carlsen, G. G. Nielsen et al., "Symptoms and signs of heart failure in patients with myocardial infarction: reproducibility and relationship to chest X-ray,

radionuclide ventriculography and right heart catheterization," *European Heart Journal*, vol. 10, no. 11, pp. 1017–1028, 1989.

[13] N. M. Wheeldon, T. M. MacDonald, C. J. Flucker, A. D. McKendrlck, D. G. McDevitt, and A. D. Struthers, "Echocardiography in chronic heart failure in the community," *Quarterly Journal of Medicine*, vol. 86, no. 1, pp. 17–23, 1993.

[14] F. Shamsham and J. Mitchell, "Essentials of the diagnosis of heart failure," *American Family Physician*, vol. 61, no. 5, pp. 1319–1330, 2000.

[15] J. Dai, Q. Tang, and W. Deng, "Effect of renal function on level of serum nt-probnp in dilated cardiomyopathy patients," *Journal of Medical Forum*, vol. 8, 2010.

[16] S. A. Hill, R. A. Booth, P. L. Santaguida et al., "Use of BNP and NT-proBNP for the diagnosis of heart failure in the emergency department: a systematic review of the evidence," *Heart Failure Reviews*, vol. 19, no. 4, pp. 421–438, 2014.

[17] A. Palazzuoli, M. Gallotta, I. Quatrini, and R. Nuti, "Natriuretic peptides (BNP and NT-proBNP): measurement and relevance in heart failure," *Vascular Health and Risk Management*, vol. 6, no. 1, pp. 411–418, 2010.

[18] M. R. Cowie, A. Mosterd, D. A. Wood et al., "The epidemiology of heart failure," *European Heart Journal*, vol. 18, no. 2, pp. 208–225, 1997.

[19] S. A. Hunt, D. W. Baker, M. H. Chin et al., "ACC/AHA guidelines for the evaluation and management of chronic heart failure in the adult: executive summary," *Journal of the American College of Cardiology*, vol. 38, no. 7, pp. 2101–2113, 2001.

[20] S. I. Chaudhry, Y. Wang, J. Concato, T. M. Gill, and H. M. Krumholz, "Patterns of weight change preceding hospitalization for heart failure," *Circulation*, vol. 116, no. 14, pp. 1549–1554, 2007.

[21] A. W. Haider, M. G. Larson, S. S. Franklin, D. Levy, and Framingham Heart Study, "Systolic blood pressure, diastolic blood pressure, and pulse pressure as predictors of risk for congestive heart failure in the Framingham Heart study," *Annals of Internal Medicine*, vol. 138, no. 1, pp. 10–16, 2003.

[22] S. I. Chaudhry, J. A. Mattera, J. P. Curtis et al., "Telemonitoring in patients with heart failure," *The New England Journal of Medicine*, vol. 363, no. 24, pp. 2301–2309, 2010.

[23] M.-K. Suh, C.-A. Chen, J. Woodbridge et al., "A remote patient monitoring system for congestive heart failure," *Journal of Medical Systems*, vol. 35, no. 5, pp. 1165–1179, 2011.

[24] M. Lan, L. Samy, N. Alshurafa et al., "WANDA: an end-to-end remote health monitoring and analytics system for heart failure patients," in *Proceedings of the Conference on Wireless Health (WH '12)*, pp. 1–3, ACM, San Diego, Calif, USA, October 2012.

[25] C. Y. Jurgens, J. A. Fain, and B. Riegel, "Psychometric testing of the heart failure somatic awareness scale," *Journal of Cardiovascular Nursing*, vol. 21, no. 2, pp. 95–102, 2006.

[26] CardioMEMS HF System, http://www.sjm.com/cardiomems/.

[27] The Criteria Committee of the New York Heart Association, *Nomenclature and Criteria for Diagnosis of Diseases of the Heart and Blood Vessels*, Little Brown, Boston, Mass, USA, 9th edition, 1994.

[28] W. T. Abraham, P. B. Adamson, R. C. Bourge et al., "Wireless pulmonary artery haemodynamic monitoring in chronic heart failure: a randomised controlled trial," *The Lancet*, vol. 377, no. 9766, pp. 658–666, 2011.

[29] E. Andrikopoulou, K. Abbate, and D. J. Whellan, "Conceptual model for heart failure disease management," *Canadian Journal of Cardiology*, vol. 30, no. 3, pp. 304–311, 2014.

[30] A. L. Bui and G. C. Fonarow, "Home monitoring for heart failure management," *Journal of the American College of Cardiology*, vol. 59, no. 2, pp. 97–104, 2012.

[31] A. Bhimaraj, "Remote monitoring of heart failure patients," *Methodist Debakey Cardiovascular Journal*, vol. 9, no. 1, pp. 26–31, 2013.

[32] R. De Maesschalck, D. Jouan-Rimbaud, and D. L. Massart, "The mahalanobis distance," *Chemometrics and Intelligent Laboratory Systems*, vol. 50, no. 1, pp. 1–18, 2000.

[33] J. A. K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Processing Letters*, vol. 9, no. 3, pp. 293–300, 1999.

[34] S. A. Hunt, W. T. Abraham, M. H. Chin et al., "ACC/AHA 2005 guideline update for the diagnosis and management of chronic heart failure in the adult," *Circulation*, vol. 112, no. 12, pp. e154–e235, 2005.

[35] Sennotech Inc, Heart Failure Tracer, http://www.sennotech.com/EN/products/HFTracer.php/.

[36] J. S. Richman and J. R. Moorman, "Physiological time-series analysis using approximate entropy and sample entropy," *The American Journal of Physiology—Heart and Circulatory Physiology*, vol. 278, no. 6, pp. H2039–H2049, 2000.

[37] P.-T. De Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of Operations Research*, vol. 134, no. 1, pp. 19–67, 2005.