Edge-Assisted Stream Scheduling Scheme for the Green-Communication-Based IoT

Licheng Wang, Yan Meng, Haojin Zhu, Minxing Tang, and Kaoru Ota

***Abstract*—The consumer Internet of Things (IoT), which exploits wireless personal area network (WPAN) technology, is undergoing rapid growth. Although the consumer IoT enables users to control many devices and offers conveniences and benefits for daily life, its long-term operation capabilities are subject to a bottleneck related to power management. To save energy and prolong the lifetime of an IoT system, the basic idea is to allow idle devices to go to sleep. Because excessively frequent switching between the awake and asleep phases will consume a significant amount of power, it is essential to properly schedule the order of multiple communication streams among multiple devices such that the total number of wake-up events is as small as possible. Based on the typical communication protocols deployed in IoT systems, this problem can be divided into two cases: 1) the intersuperframe case and the 2) intrasuperframe case. The former case has been well studied in existing works, whereas research on the latter case is currently immature. In this paper, we propose an efficient scheme for addressing the stream order scheduling (SOS) problem in the intrasuperframe case. Mobile edge computing technology is utilized in the proposed scheme to reduce the network load, and three heuristic algorithms are proposed to improve the scheme's performance. We report various tests conducted on 4800 random original IoT topologies and 19 000 random Hamiltonian edge-dual topologies, and the experimental results demonstrate that our scheme achieves optimal solutions with a very high success probability.**

***Index Terms*—Edge computing, green Internet of Things (IoT), stream order scheduling (SOS).**

I. Introduction

THE consumer Internet of Things (IoT), as an emerging technological paradigm, establishes connectivity between the digital world and the physical world. It enables users to intelligently interact with and control their smart devices, such as domestic appliances and portable medical devices, via various wired and wireless communication protocols. In particular, IoT technologies are widely deployed in the areas of intelligent home networking [2]–[4], delay-tolerant networking [5]–[7], energy-aware data collection systems [8], device-to-device (D2D) communications [9]–[11], cellular networks [12], and mobile ad hoc networks [13]–[16]. However, the massive volume of communication among the smart devices in an IoT platform results in extremely high power consumption, which has become a major bottleneck for IoT devices since the devices in IoT platforms are typically powered by batteries and thus have limited power resources. To solve this problem, the green IoT concept has been proposed, which exploits various energy-saving communication protocols to reduce the power consumption of IoT devices.

At present, most wireless communication protocols (e.g., Wi-Fi, ZigBee, Z-Wave, and IEEE 802.15.3) utilized by IoT platforms possess power management capabilities. These protocols utilize time-division multiple access (TDMA) technology [17], meaning that in these protocols, time is divided into multiple superframes (SFs). To prolong the lifetimes of battery-powered devices, several power-saving modes (e.g., the device synchronized power save (DSPS) mode in IEEE 802.15.3 [18], [19]) have been defined in these protocols; these modes allow participating devices to sleep for one or more SFs to reduce their energy consumption. Thus, these modes schedule devices from the *intersuperframe* perspective. However, there is also potential to further reduce devices' power consumption within a given SF, although corresponding *intrasuperframe*-based energy-saving technologies have not yet been extensively studied.

In this paper, we propose a system for power management from the intrasuperframe perspective based on mobile edge computing. As shown in Fig. 1, within an SF, a single TDMA-based channel time allocation (CTA) period, which is shared by multiple streams among multiple devices, can be further divided into multiple time slots called CTAs. In the proposed system, to reduce the network load within an SF, an edge device in the IoT platform is preselected as a coordinator. Then, the selected edge device assigns a CTA for each stream and announces the position and duration of each CTA through a beacon at the beginning of the SF. Since each device knows its CTA position in the current SF from this beacon, it can go to sleep during phases when no CTAs are assigned for its streams and wake up only during phases when CTAs are assigned for its streams. As a result, the relative positioning of the streams within a TDMA SF might have a significant effect on the total number of wake-up events. Considering that each transformation between the awake and asleep phases

Manuscript received February 13, 2019; revised April 12, 2019; accepted May 4, 2019. Date of publication May 9, 2019; date of current version July 31, 2019. This work was supported in part by the National Key Research and Development Program of China under Grant 2016YFB0800602, in part by the Shandong Provincial Key Research and Development Program under Grant 2018CXGC0701, in part by the National Natural Science Foundation of China under Grant 61672350, in part by the 111 Project under Grant B08004, and in part by JSPS KAKENHI under Grant JP19K20250. An extended abstract was published in WASA 2015 [1]. *(Corresponding author: Haojin Zhu.)*

L. Wang is with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China.

Y. Meng and H. Zhu are with the School of Electronic Engineering, Shanghai Jiaotong University, Shanghai 200240, China (e-mail: zhu-hj@cs.sjtu.edu.cn).

M. Tang is with the Department of Computer Science, Humboldt University of Berlin, 10099 Berlin, Germany.

K. Ota is with the Department of Information and Electronic Engineering, Muroran Institute of Technology, Muroran 050-8585, Japan.

Digital Object Identifier 10.1109/JIOT.2019.2915793

2327-4662 © 2019 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 12:01:37 UTC from IEEE Xplore. Restrictions apply.

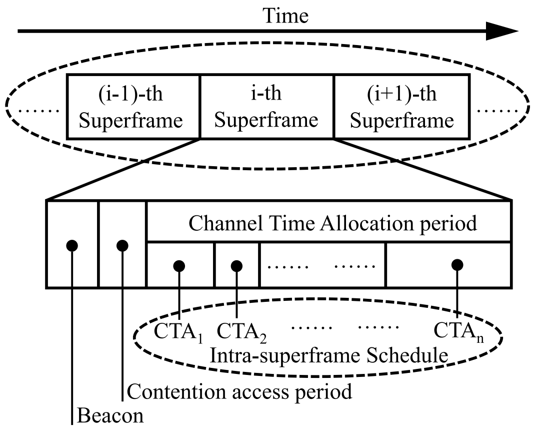


Fig. 1. Two scopes of scheduling: inter-superframe versus intra-superframe.

consumes a significant amount of power [20], the coordinator should schedule the multiple streams within an SF such that the total number of wake-up events is as small as possible.

With the introduction of an edge device as the coordinator, the power management problem is transformed into a problem of stream order scheduling (SOS) within an SF. However, the traditional solutions [2], [17], [21]–[25] to the SOS problem are not suitable in our scenario. These solutions treat the SOS problem within an SF as a Hamilton path problem to be solved over the corresponding edge-dual graph. However, the hardness of the Hamilton problem prevents these solutions from finding the optimal order of the streams. To solve this problem, in this paper, we approach the problem from two perspectives. On the one hand, we falsify the equivalence between the optimal solution to the SOS problem in the original topology and the existence of a Hamilton path in the corresponding edge-dual topology (see Section II-C for details). Instead, we find that an optimal solution to the SOS problem can be derived from several partial Hamilton paths. On the other hand, even if it is theoretically difficult to solve the SOS problem by finding Hamilton paths, a nearly optimal solution can be found in practice by resorting to certain heuristic strategies. Based on the above considerations, in this paper, we propose a heuristic algorithm for solving the SOS problem within an SF.

The contributions of this paper are summarized as follows.

1) We propose a novel power management system for the IoT from the intrasuperframe perspective. We propose a mobile-edge-computing-based architecture to reduce the network load and a heuristic algorithm to find the optimal order of the streams within an SF.
2) Our heuristic algorithm is efficient in terms of time. We theoretically demonstrate that for a given instance of the SOS problem with m streams, the time complexity is $\mathcal{O}(m^2)$.
3) We report comprehensive simulations conducted for 4800 random original topologies and 19 000 random Hamiltonian edge-dual topologies to evaluate the performance of the proposed system. The experimental results show that our system finds the optimal solution with a very high probability and that the system remains stable as the numbers of devices and streams increase.

To the best of our knowledge, this is the first mobile-edge-computing-based system designed for power management from the intrasuperframe perspective. The remainder of this paper is organized as follows. In Section II, we introduce some background on mobile edge computing and review the definitions and models related to the SOS problem within an SF. Section III presents the detailed design of our proposed system, including the architecture and the heuristic algorithm for solving the SOS problem, and presents an analysis of the time complexity. The simulations are presented and analyzed in Section IV. The limitations of our system are discussed in Section V. Finally, concluding remarks are given in Section VI.

II. Preliminaries and Related Work

A. Mobile Edge Computing

Mobile edge computing refers to technology that moves computations to edge devices in a mobile network [26]. Compared with a traditional network in which computations are performed in the cloud, edge computing has the following advantages in our scenario. First, computing at the network edges saves bandwidth resources since it is not necessary for a massive number of devices to send messages to a cloud server. Second, since the duration of a single SF is short, edge computing can provide better real-time performance and thus is suitable for the intrasuperframe scenario. Third, edge computing is secure because it prevents data from being leaked through an untrustworthy cloud server.

B. Stream Order Scheduling Problem Within SF

The multiple streams among multiple devices within a TDMA-based SF can be modeled as an undirected graph $G = (V, E)$, where each node in V represents a device and each edge in E represents a stream between two different devices. A schedule for $m = |E|$ streams can be viewed as a permutation $\pi \in S_m$. Given two streams $e_1, e_2 \in E$, let the notation $e_1 \cap e_2$ denote the intersection of the nodes defining e_1 and e_2. Suppose that we have $e_1 = \{1, 3\}$ and $e_2 = \{2, 3\}$, such that $e_1 \cap e_2 = \{3\}$. Then, the *number of wake-up events* for each stream, denoted by $w(e_{\pi(i)})$, with respect to the schedule π can be calculated as follows:

$$w(e_{\pi(i)}) = \begin{cases} 2, & i = 1 \\ 2 - |e_{\pi(i)} \cap e_{\pi(i-1)}|, & 2 \le i \le m. \end{cases} \quad (1)$$

Naturally, the number of wake-up events for a schedule π is the sum of the numbers of wake-up events for all m scheduled streams, that is, $w(\pi) = \sum_{i=1}^{m} w(e_{\pi(i)})$, and the objective of the SOS problem is to find $\pi \in S_m$ such that $w(\pi)$ is minimized.

C. Traditional SOS Solutions

The stream scheduling problem is a topic that is, widely followed but often new, considering that the optimizing objectives and scenarios are various. In 2002, Stine and Veciana [21] proposed an algorithm for dealing with the stream schedule problem in a centrally controlled wireless data network. But its performance is outperformed by the so-called MDS algorithm due to Guo *et al.* [17] in 2005. At IS3C 2012,

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 12:01:37 UTC from IEEE Xplore. Restrictions apply.

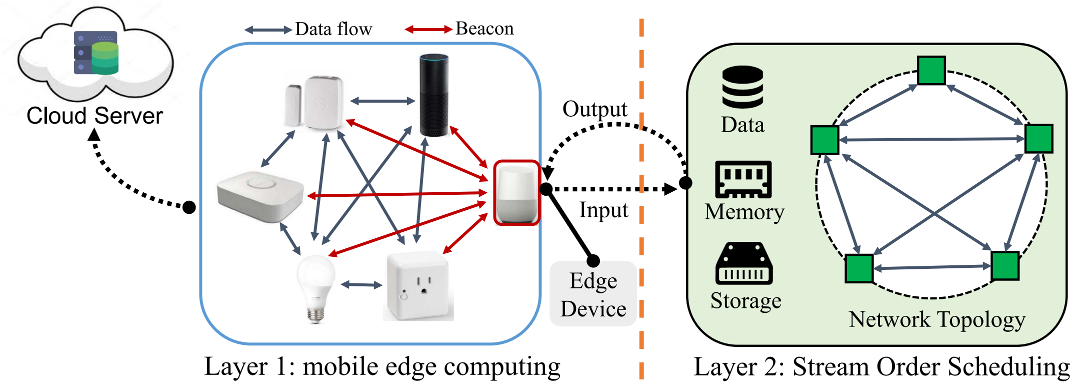


Fig. 2. Architecture of the proposed system.

Wang and Wen [2] conducted an experimental studied on the performance of MDS algorithm from the perspectives of wakeup times, device and stream scales, and stream densities. The result demonstrates that MDS algorithm outperforms the recommendations from LAN/MAN standards committee [18], [19], therefore, MDS algorithm is regarded as a good benchmark in studying SOS problem. In 2014, they [22] also extended the MDS algorithm to multiaerial wireless communication scenarios. From 2016 to 2017, Gai *et al.* [27], [28] developed two energy-aware dynamic task assignment schemes for green/cloud computing scenarios. In 2018, Chen *et al.* [23] proposed a real-time scheduling algorithm using task duplication for minimizing both the completion time and monetary cost of processing big data workflows in clouds. At ISCAS 2018, Chiang and Hsiao [24] proposed an algorithm for scheduling layered video streaming over wireless broadband networks. In 2019, Kim and Chung [25] proposed a segment scheduler for efficient bandwidth utilization in multipath environments.

Among these solutions, two typical strategies are always adopted [2], [17], [22]. The first is to associate an optimal solution of an SOS problem to finding an Eular path, and the second is to finding a Hamiltonian Path on the edge-dual tologoies.[1] However, we find that both of these two strategies are over-constrained: the existence of an Eular path on the original topology or a Hamilton path on its edge-dual topology is merely a *sufficient but not necessary* condition for deriving the optimal SOS solution. An instance for supporting this observation was given in [1].

III. Proposed Scheme

A. System Overview

In this section, we propose our mobile-edge-computing-based power management system. Fig. 2 illustrates the architecture of our system, which consists of two layers. In the mobile edge computing layer, a mobile device is preselected as the edge coordinator, which then interacts with the other devices to perform power management within one SF. In the second layer, the edge coordinator calculates the optimal stream order among the devices. We will describe these layers in detail in the following sections.

[1]Given a graph G, its edge-dual graph G^* is defined as follows: each edge in G induces a node in G^*, and whenever two edges in G are adjacent, they induce an edge in G^*.

B. Mobile Edge Computing Layer

As shown in Fig. 2, the devices in an IoT system can perform peer-to-peer communication or send/receive beacons to/from the edge coordinator. In our proposed system, before each SF starts, the devices that will be active in this SF send beacons to the edge coordinator. Then, the edge coordinator obtains the network topology, utilizes the SOS layer to calculate the optimal SOS solution, and converts this solution into a beacon for the SF (as illustrated in Fig. 1). Subsequently, the edge coordinator broadcasts the beacon into the network, and the devices perform the corresponding actions to communicate while minimizing their power consumption.

C. Stream Order Scheduling Layer

1) Our Starting Point and Heuristic Strategy: Instead of using the aforementioned two traditional strategies, we try to associate the (near) optimal solutions of SOS problems to (partial) Hamilton path problem on the edge-dual topology. In fact, Guo *et al.* [17] once considered to model the SOS problem as a Hamilton path problem. However, being aware of the NP-hardness of the well-known Hamilton problem, they *gave up* this strategy finally.

From a historical perspective regarding algorithm development, it is evident that NP-hardness has never stymied researchers' efforts to find (nearly) optimal solutions to many well-known NP-complete problems [29]. Instead, over the past decades, many powerful algorithms have been proposed for approximately or intelligently solving various NP-complete problems, such as approximation algorithms for knapsack problems [30], Steiner tree problems [31] and tree-like network construction [32]; genetic algorithms and cuckoo search algorithms for traveling salesman problems [33]–[36]; particle swarm algorithms for integer programming problems [37]; support vector machine algorithms for steganalysis [38] and data regression [39]; and machine learning algorithms for vehicle classification [40]. In fact, many applications have been successfully addressed based on the ability to solve the related NP-complete problems at moderate scales [32], [38], [40]–[44].

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 12:01:37 UTC from IEEE Xplore. Restrictions apply.

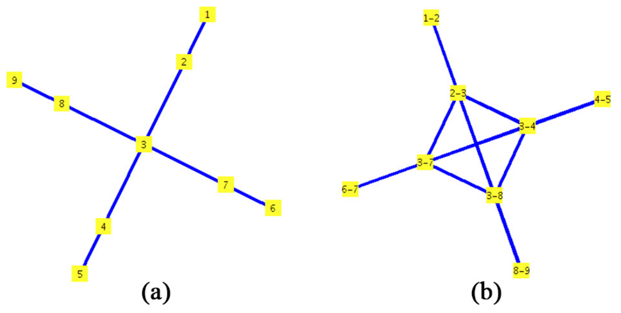


Fig. 3. Original topology G and its edge-dual topology G^*. (a) Original topology. (b) Edge-dual topology.

The aforementioned successful cases encourage us to reconsider the effort that Guo *et al.* [17] once abandoned. Specifically, our starting point is to try to obtain nearly optimal solutions to SOS problems by using heuristic strategies. In fact, we find that optimal SOS solutions can be easily derived by piecing together disjoint partial Hamilton paths on the edge-dual graph. For instance, for the original graph shown in Fig. 3(a), the following two partial Hamilton paths can be found on the edge-dual graph, as shown in Fig. 3(b):

$$e_{12} \to e_{23} \to e_{37} \to e_{67} \quad e_{45} \to e_{34} \to e_{38} \to e_{89}. \tag{2}$$

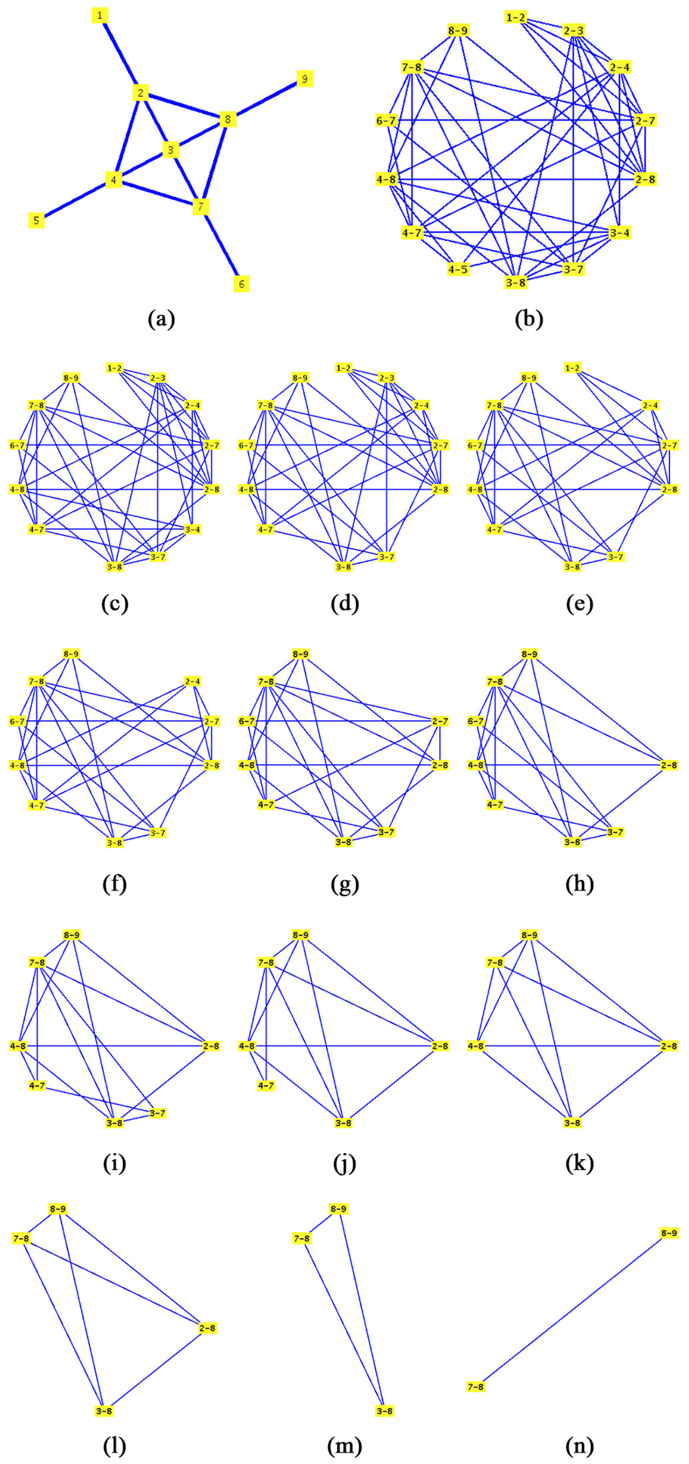
By piecing these paths together, one can finally obtain the optimal solution. Therefore, the remaining problem is how to find such partial Hamilton paths efficiently.

Our heuristic strategy, called VDF for short, is based on two intuitive conditions.

1) Whenever a node is *visited*, it is immediately *dropped* so that it can never be revisited.
2) The remaining adjacent nodes that have *fewer* connections with others should be visited as soon as possible.

The VDF strategy is illustrated in a detailed, step-by-step manner in Fig. 4. More explanations are given below.

1) Label each node of the original topology as $1, 2, \dots, v$ where $v = |V|$. This leads to Fig. 4(a).
2) Label each node of the edge-dual topology as e_{ij} where the edge $(i, j) \in E$ belongs the original topology. This leads to Fig. 4(b).
3) The optimal SOS solution, $e_{45} \to e_{34} \to e_{23} \to e_{12} \to e_{24} \to e_{27} \to e_{67} \to e_{37} \to e_{47} \to e_{48} \to e_{28} \to e_{38} \to e_{78} \to e_{89}$, is obtained via the following "visiting-and-dropping" process over the edge-dual topology Fig. 4(b).
   a) Find a node with *fewer* connections with others (i.e., minimal degree) in Fig. 4(b). Since $\deg(e_{45}) = \deg(e_{12}) = \deg(e_{89}) = \deg(e_{67}) = 4$ and all of them reach the minimal degree, we pick one of them, say e_{45}, at random.
   b) Among nodes adjacent to e_{45} in Fig. 4(b), find a node with *fewer* connections with others. Since $\deg(e_{34}) = 7 < \deg(e_{47}) = \deg(e_{48}) = \deg(e_{24}) = 8$, the node e_{34} is picked without controversial.
   c) The node e_{45} in Fig. 4(b) is visited and dropped, leading to Fig. 4(c).
   d) Among nodes adjacent to e_{34} in Fig. 4(c), find a node with *fewer* connections with others. Since



$$e_{45} \to e_{34} \to e_{23} \to e_{12} \to e_{24} \to e_{27} \to e_{67} \to$$
$$\to e_{37} \to e_{47} \to e_{48} \to e_{28} \to e_{38} \to e_{78} \to e_{89}$$

Fig. 4. Illustration of the VDF strategy. (a) Original. (b) Edge-dual. (c) After visiting e_{45}. (d) After visiting e_{34}. (e) After visiting e_{23}. (f) After visiting e_{12}. (g) After visiting e_{24}. (h) After visiting e_{27}. (i) After visiting e_{67}. (j) After visiting e_{37}. (k) After visiting e_{47}. (l) After visiting e_{48}. (m) After visiting e_{28}. (n) After visiting e_{38}.

now $\deg(e_{23}) = \deg(e_{24}) = \deg(e_{37}) = \deg(e_{38}) = \deg(e_{47}) = \deg(e_{48}) = 7$, the node e_{23} is picked at random.

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 12:01:37 UTC from IEEE Xplore. Restrictions apply.

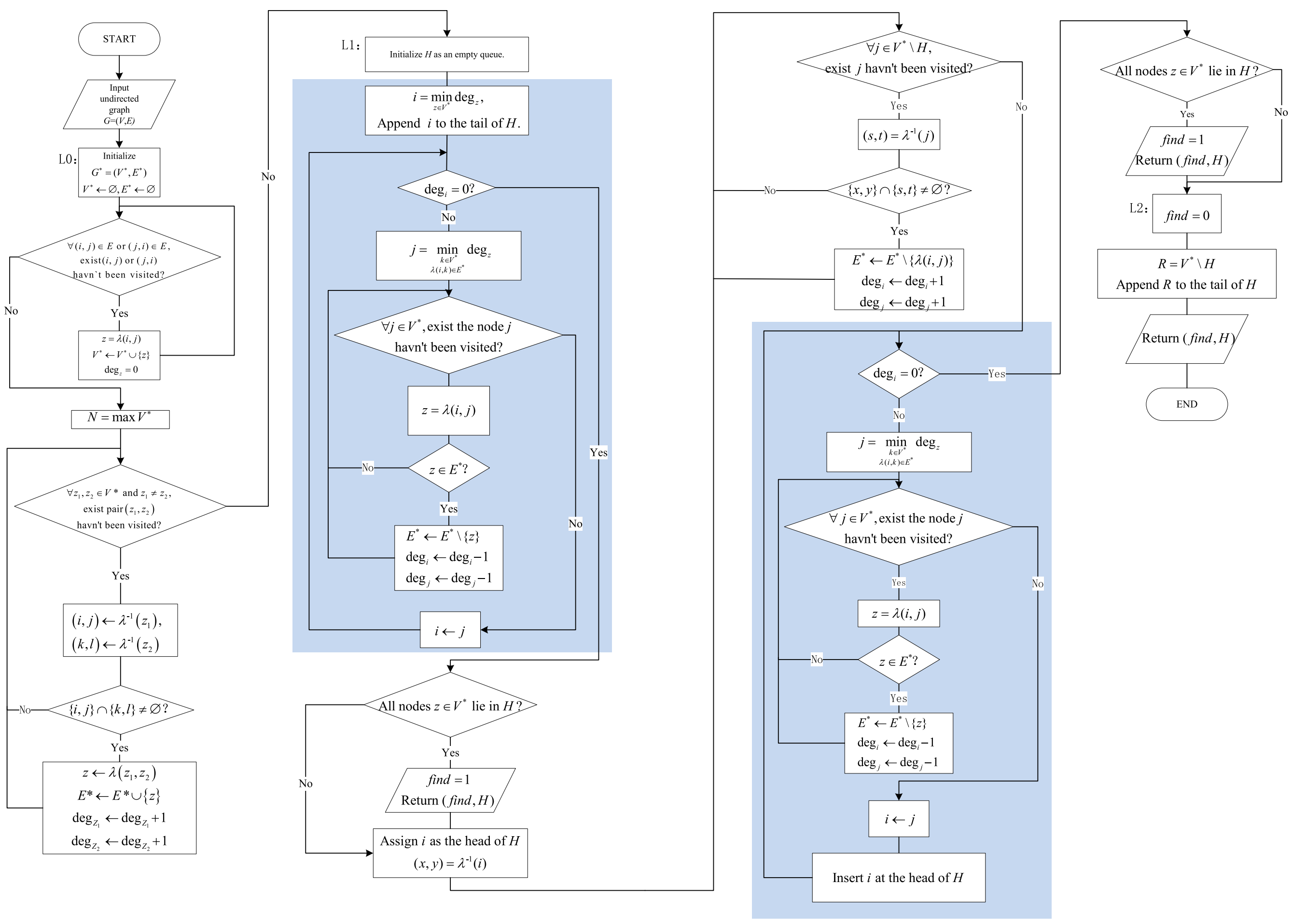

START
Input undirected graph G=(V,E)
L0:
Initialize $G^*=(V^*,E^*)$ $V^*\leftarrow\varnothing, E^*\leftarrow\varnothing$
$\forall(i,j)\in E$ or $(j,i)\in E$, exist (i,j) or (j,i) havn't been visited?
Yes
No
$z=\lambda(i,j)$ $V^*\leftarrow V^*\cup\{z\}$ $\deg_z=0$
$N=\max V^*$
$\forall z_1,z_2\in V^*$ and $z_1\neq z_2$, exist pair (z_1,z_2) havn't been visited?
Yes
No
$(i,j)\leftarrow\lambda^{-1}(z_1)$, $(k,l)\leftarrow\lambda^{-1}(z_2)$
$\{i,j\}\cap\{k,l\}\neq\varnothing$?
No
Yes
$z\leftarrow\lambda(z_1,z_2)$ $E^*\leftarrow E^*\cup\{z\}$ $\deg_{z_1}\leftarrow\deg_{z_1}+1$ $\deg_{z_2}\leftarrow\deg_{z_2}+1$
L1:
Initialize H as an empty queue.
$i=\min_{z\in V^*}\deg_z$, Append i to the tail of H.
$\deg_i=0$?
No
$j=\min_{k\in V^*, \lambda(i,k)\in E^*}\deg_z$
$\forall j\in V^*$, exist the node j havn't been visited?
$z=\lambda(i,j)$
$z\in E^*$?
No
Yes
$E^*\leftarrow E^*\setminus\{z\}$ $\deg_i\leftarrow\deg_i-1$ $\deg_j\leftarrow\deg_j-1$
No
Yes
$i\leftarrow j$
All nodes $z\in V^*$ lie in H?
No
Yes
$find=1$ Return $(find,H)$
Assign i as the head of H $(x,y)=\lambda^{-1}(i)$
$\forall j\in V^*\setminus H$, exist j havn't been visited?
Yes
No
$(s,t)=\lambda^{-1}(j)$
$\{x,y\}\cap\{s,t\}\neq\varnothing$?
No
Yes
$E^*\leftarrow E^*\setminus\{\lambda(i,j)\}$ $\deg_i\leftarrow\deg_i+1$ $\deg_j\leftarrow\deg_j+1$
$\deg_i=0$?
Yes
No
$j=\min_{k\in V^*, \lambda(i,k)\in E^*}\deg_z$
$\forall j\in V^*$, exist the node j havn't been visited?
Yes
No
$z=\lambda(i,j)$
$z\in E^*$?
No
Yes
$E^*\leftarrow E^*\setminus\{z\}$ $\deg_i\leftarrow\deg_i-1$ $\deg_j\leftarrow\deg_j-1$
$i\leftarrow j$
Insert i at the head of H
All nodes $z\in V^*$ lie in H?
Yes
No
$find=1$ Return $(find,H)$
L2:
$find=0$
$R=V^*\setminus H$ Append R to the tail of H
Return $(find,H)$
END

Fig. 5. Flowchart of the VDF algorithm.

e) The node e_{34} in Fig. 4(c) is visited and dropped, leading to Fig. 4(d).

f) The above process is repeated, and the nodes $e_{23}, e_{12}, e_{24}, e_{27}, e_{67}, e_{37}, e_{47}, e_{48}, e_{28}$ are picked, visited, and dropped one by one, leading to Fig. 4(e)–(m), respectively. And before dropping the node e_{28}, the node e_{38} is picked according to the VDF strategy.

g) Among nodes adjacent to e_{38} in Fig. 4(m), find a node with *fewer* connections with others. Since now $\deg(e_{78}) = \deg(e_{89}) = 2$, the node e_{78} is picked at random.

h) The node e_{38} in Fig. 4(m) is visited and dropped, leading to Fig. 4(n).

i) Among nodes adjacent to e_{78} in Fig. 4(n), there is only one node e_{89} unvisited. Thus, e_{89} is picked without controversial.

j) The node e_{78} in Fig. 4(n) is visited and dropped.

k) The node e_{89} is visited and dropped. This is the end of the VDF algorithm.

The simulations reported in Section IV suggest that the VDF strategy works well for most random instances. In particular, if the edge-dual topology contains a Hamilton path, the proposed strategy tends to lead to an optimal SOS solution for the corresponding original topology with a very high probability (see Section IV-B).

2) Compact Edge Representation and Algorithm Description: For convenience in associating nodes and edges based on not only the original topology but also the edge-dual topology, we adopt a compact edge representation method to simplify our algorithmic descriptions. Given an undirected graph $G = (V, E)$, all nodes are first labeled with $|V|$ unique positive integers, i.e., $V = \{\ell_1, \dots, \ell_{|V|}\}$. Then, an $M \geq \max V$ is selected, and each edge $(i, j) \in E$ is labeled with a positive integer $\lambda(i, j)$, where

$$\lambda(i,j) \triangleq \min\{i,j\}\cdot M + \max\{i,j\}. \tag{3}$$

Moreover, the representation has the following inverse:

$$\lambda^{-1}(z) \triangleq \left(\left\lfloor \frac{z}{M} \right\rfloor, z - \left\lfloor \frac{z}{M} \right\rfloor \cdot M\right) \tag{4}$$

where $\lfloor x \rfloor$ returns the greatest integer $\leq x$.

Based on the VDF strategy and the compact edge representation introduced above, the flowchart of our heuristic algorithm for solving the SOS problem is depicted in Fig. 5.

Theorem 1: The time complexity of the VDF algorithm is quadratic in the number of streams. More precisely, for a given SOS instance with m streams, the time complexity of the VDF algorithm is $\mathcal{O}(m^2)$.

Proof: A detailed analysis of the time complexity of the VDF algorithm is presented in Appendix A. ■

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 12:01:37 UTC from IEEE Xplore. Restrictions apply.

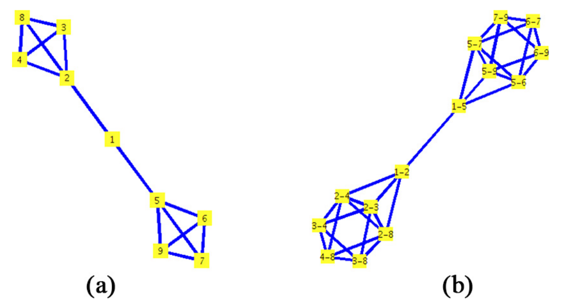


Fig. 6. SOS instance for which VDF outperforms MDS. (a) Original topology. (b) Edge-dual topology.

Remark 1: Note that the proposed VDF algorithm is fundamentally different from Guo *et al.*'s MDS algorithm [17]. The MDS algorithm selects an edge in the original graph G in each step such that both of the associated nodes have the "minimum" degree. According to [17], this step is executed by means of the following two substeps: first, all candidate nodes with the minimum degree are found, and then, for each of them, an adjacent node with the minimum degree is found. By contrast, in each step of our algorithm, it is necessary to select only one node with the minimum degree in the edge-dual graph G^*. Although it cannot be strictly proved that the VDF algorithm outperforms MDS algorithm for all SOS instances, it outperforms MDS in most typical cases. A good example of an SOS instance in which our VDF algorithm outperforms the MDS algorithm is shown in Fig. 6. In this instance, the VDF algorithm will output a Hamilton path over the edge-dual graph with a total of 15 wake-up events, while the MDS algorithm will fall in reaching optimal SOS solutions and output a scheduling sequence that requires 16 wake-up events in total. The detailed execution of the MDS algorithm in this instance is described in Appendix B.

IV. Simulations and Evaluations

Our simulations were organized into two groups. In the first group of simulations, we tested the success probability and run time for finding optimal SOS solutions for 4800 random original topologies. In the second step, we tested the success probability and run time for finding optimal SOS solutions with different node numbers and edge densities. We simulate our proposed system using Thinkpad X1 Carbon laptop with Intel i5-4300U CPU and 8-GB RAM, and process data using MATLAB.

A. Simulations for Original Topologies

We considered a number of devices varying from 5 to 10—the typical number of devices within an SF according to the simulations presented in [2], [17], [22]—and a density of streams varying from 0.2 to 0.9 (in increments of 0.1). For each of the corresponding combinations of device number and stream density, 100 random SOS instances were generated. In other words, a total of 4800 random SOS instances were generated, which could be divided into 48 distinct cases. For each case, the success probability of our algorithm for reaching optimal SOS solutions and the total run time over the 100 random instances are depicted in Fig. 7. From Fig. 7(a), the following results are obtained.

1) The higher the stream density is, the higher the probability of reaching an optimal SOS solution tends to be, although one exceptional point is observed at a stream density of 0.4 and device number of 5. In particular, when the stream density is higher than 0.5, our algorithm finds optimal SOS solutions in all cases with probability over 95%.
2) However, there is no apparent relationship between the number of devices and the probability of reaching optimal SOS solutions.

From Fig. 7(b), the following results are obtained.

1) On the one hand, our algorithm is very efficient, as seen from the fact that the total run time over 100 random instances is no longer than 12 s.
2) On the other hand, except for the cases of 9 and 10 devices, the total run time increases approximately linearly as the stream density increases.

B. Simulations for Edge-Dual Topologies

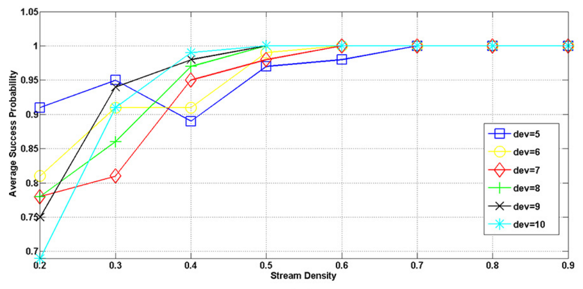
We then further tested our algorithm on various edge-dual topologies. To test the success probability of our algorithm for achieving (nearly) optimal SOS solutions, we first generated random graphs such that we knew optimal solutions existed and then tested whether our algorithm could find either those solutions or other but equivalently good solutions. There are two possible methods of doing so. The first is to generate random Euler graphs and then compute their edge-dual graphs, and the second is to directly generate random graphs that contain at least one Hamilton path. For simplicity, we choose the second method.

Given an edge density $\rho \in (0, 1)$ and a number of nodes n, we can produce a random graph $G = (V, E)$ with the required properties via the following steps.

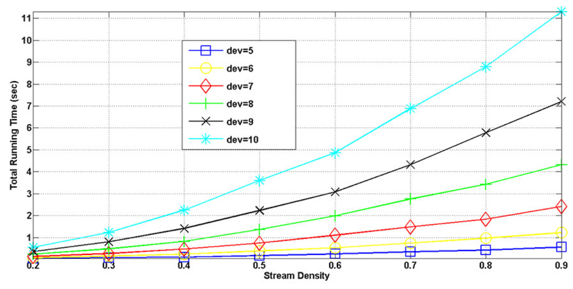
1) Let $V = \{v_1, \ldots, v_n\}$ and $E = \emptyset$.
2) Sample a random permutation $\pi \in S_n$, and assign a random Hamilton path $H = v_{\pi(1)} \cdots v_{\pi(n)}$ accordingly. That is, E will be updated as follows:
$$E \leftarrow E \cup \{(v_{\pi(i)}, v_{\pi(i+1)}) : 1 \leq i \leq n-1\}.$$
3) Add other edges randomly into E with probability ρ. That is, for each pair (i, j) $(i, j = 1, \ldots, n; i \neq j)$, if neither (v_i, v_j) nor (v_j, v_i) is contained in E, then E will be updated, with probability ρ, as follows:
$$E \leftarrow E \cup \{(v_i, v_j)\}.$$

We at first performed new simulations for edge-dual topologies with a fixed edge density of $\rho = 1/3$. The results are summarized in Table I and depicted in Fig. 8. It is evident that these results are quite good. Among $1000 \times 9 = 9000$ random instances, our algorithm reaches an optimal SOS solution (i.e., a Hamilton path in the corresponding edge-dual topology) in 8930 instances. That is, the average success probability (ASP) is 99.22%. Fig. 8 also suggests that when the node number is larger than 20, the success probability is almost to 100%. Besides, the total run time over 1000 random instances is within 2 s and is nearly linear with respect to the number of nodes (i.e., the number of streams in the corresponding original topology).

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 12:01:37 UTC from IEEE Xplore. Restrictions apply.



(a)



(b)

Fig. 7. Simulations for original topologies. (a) ASP. (b) TRT.

TABLE I

SIMULATIONS ON 1000 RANDOM HAMILTONIAN GRAPHS WITH EDGE DENSITY $\rho = 1/3$

Node Number	Simu. Times	Succ. Times	Succ. Prob.	Total Running Time (s)
10	1000	930	93%	0.1090
20	1000	1000	100%	0.1998
30	1000	1000	100%	0.3108
40	1000	1000	100%	0.4467
50	1000	1000	100%	0.5805
60	1000	1000	100%	0.7346
70	1000	1000	100%	0.9091
80	1000	1000	100%	1.0977
90	1000	1000	100%	1.3012

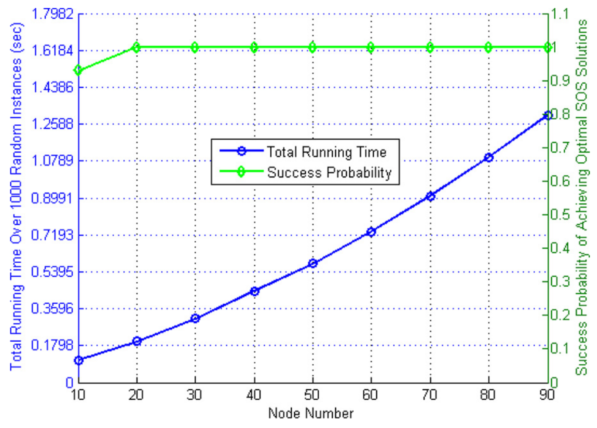


Fig. 8. Simulations on random Hamiltonian graphs with small node scales and a fixed edge density $\rho = 1/3$.

Next, we conducted simulations for Hamiltonian edge-dual topologies with even larger numbers of nodes and various edge densities.

1) The node number n scales from 100 to 1000 by step 100.
2) The edge-density d is divided into 10 levels, varying within the interval $[(2/n), [2/\sqrt{n}])$ by step $(1/10)([2/n] - [2/\sqrt{n}])$. Here, the lower bound of edge density $(2/n)$ comes from the fact that to maintaining a graph Hamiltonian, it needs at least $n-1$ edges, leading to that $d \geq ([n-1]/[n(n-1)/2]) = (2/n)$. The upper bound $(2/\sqrt{n})$ comes from our intuition: when the edge density exceeds this upper bound, our algorithm would perform even well.
3) For each different choice on node number and edge-density, we run the VDF algorithm for 100 randomly generated Hamiltonian graphs.

Then, the total running time (TRT) and ASP of reaching optimal SOS solutions are collected in Tables II and III, and also depicted in Fig. 9.

1) The TRT over 100 random Hamiltonian instances is no more than 20 s. When edge density level is fixed, the TRT increases, *piecewise-linearly*, with the increasing of scale of nodes. This result is consistent with the theory result given by Theorem 1, where m^2 is just the scale of the scale of nodes of the edge-dual topology.
2) The ASP of achieving optimal SOS solutions increases with the increasing of edge density levels. In particular, when edge density level is over 4, the ASP of our simulation is near to 100%. This is reasonable considering that the denser the edges, the less opportunity that the VDF algorithm is caved into a locally non-Hamiltonian path.

In summary, over a total of 19 000 random Hamiltonian edge-dual topologies, our algorithm reaches optimal SOS solutions in 17 166 instances. That is, the ASP of finding optimal SOS solutions for random Hamiltonian topologies is as high as 90.34%.

V. LIMITATIONS

A. Counterexample of the VDF Strategy

However, there is no silver bullet with which to settle the matter once and for all. A counterexample to the VDF strategy is presented in Fig. 10. In this example, our strategy will output the following three disjoint paths:

$$e_{1,10} \rightarrow e_{1,2} \rightarrow e_{2,7} \rightarrow e_{6,7} \rightarrow e_{6,11} \quad (5)$$

$$e_{5,12} \rightarrow e_{4,5} \rightarrow e_{4,8} \rightarrow e_{8,9} \rightarrow e_{9,13} \quad (6)$$

and

$$e_{2,3} \rightarrow e_{3,4} \rightarrow e_{3,7} \rightarrow e_{3,8}. \quad (7)$$

By piecing them together, we merely obtain a nearly optimal solution with 17 wake-up events in total. However, we can

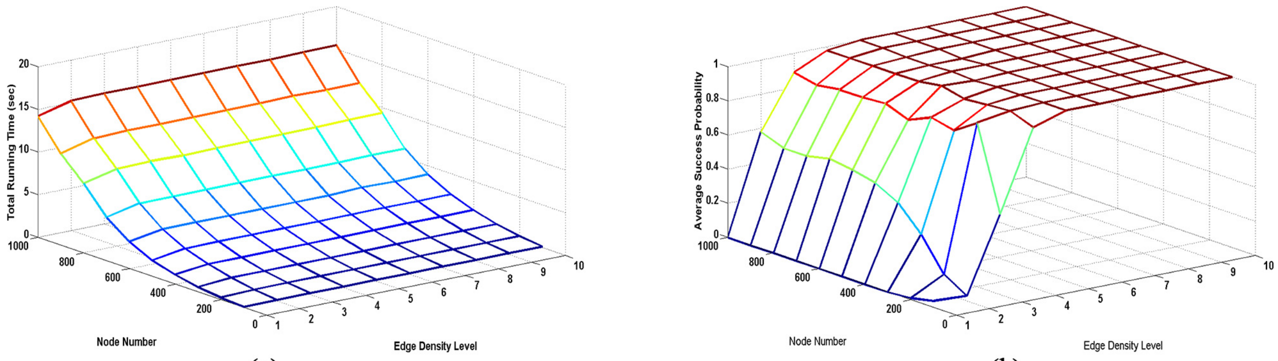
Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 12:01:37 UTC from IEEE Xplore. Restrictions apply.

TABLE II
TRT OVER 100 RANDOM INSTANCES (s)

Node Number	Edge Density Level									
	1	2	3	4	5	6	7	8	9	10
100	0.1099	0.1122	0.1304	0.1329	0.1371	0.1370	0.1369	0.1419	0.1392	0.1408
200	0.2839	0.4004	0.4310	0.4336	0.4375	0.4408	0.4427	0.4480	0.4492	0.4511
300	0.5832	0.9003	0.8902	0.8939	0.9010	0.9064	0.9144	0.9162	0.9224	0.9340
400	1.1183	1.5798	1.5844	1.5891	1.6039	1.6132	1.6229	1.6738	1.6413	1.6551
500	1.8620	2.4815	2.5280	2.5156	2.5250	2.5336	2.5698	2.5877	2.5760	2.5857
600	3.1369	3.8458	3.8532	3.8614	3.8693	3.8798	3.8786	4.0420	3.9736	4.0245
700	5.0901	5.9389	5.9736	5.9809	6.0318	6.0341	6.0414	6.0380	6.1340	6.1247
800	8.1379	9.0690	9.1772	9.0876	9.1847	9.2031	9.2138	9.2775	9.3420	9.4171
900	10.7253	11.7548	11.8638	11.8387	11.8596	11.8966	11.9203	12.0177	11.9668	11.9929
1000	14.2106	15.2250	15.3299	15.2811	15.3306	15.3663	15.4328	15.3815	15.4820	15.5309

TABLE III
ASP OVER 100 RANDOM INSTANCES

Node Number	Edge Density Level									
	1	2	3	4	5	6	7	8	9	10
100	0.0400	0.0300	0.4700	0.9400	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
200	0.0100	0.1100	0.9600	0.9900	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
300	0	0.3000	0.8700	0.9800	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
400	0	0.4400	0.9000	0.9700	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
500	0	0.5100	0.8400	1.0000	0.9900	1.0000	1.0000	1.0000	1.0000	1.0000
600	0	0.5400	0.8900	0.9800	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
700	0	0.5600	0.8800	0.9800	0.9900	0.9900	1.0000	1.0000	1.0000	1.0000
800	0	0.5300	0.8800	0.9800	0.9900	1.0000	1.0000	1.0000	1.0000	1.0000
900	0	0.5300	0.8600	0.9800	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1000	0	0.5800	0.8900	0.9800	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000



(a) (b)

Fig. 9. Simulations on random Hamiltonian graphs with large node scales and leveled edge densities. (a) TRT. (b) ASP

easily see that an optimal solution requiring only 16 wake-up events can be obtained by piecing together the following two disjoint paths:

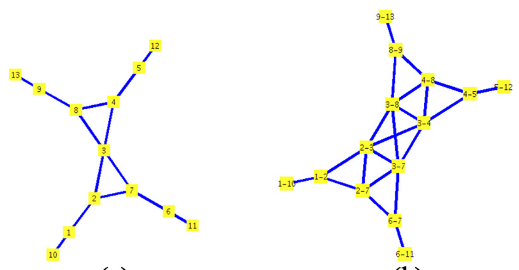
$$e_{1,10} \rightarrow e_{1,2} \rightarrow e_{2,3} \rightarrow e_{2,7} \rightarrow e_{3,7} \rightarrow e_{6,7} \rightarrow e_{6,11} \quad (8)$$

and

$$e_{5,12} \rightarrow e_{4,5} \rightarrow e_{3,4} \rightarrow e_{4,8} \rightarrow e_{3,8} \rightarrow e_{8,9} \rightarrow e_{9,13}. \quad (9)$$

Remark 2: Note that this limitation is essential to our method.

1) The intuition behind our proposal is to *find a partial but longest Hamiltonian path H* by using the VDF strategy in two directions—appending new unvisited nodes at the tail and the head of H iteratively, and then piecing H with a randomly arranged other unvisited nodes. On one hand, our VDF strategy falls in finding the following longer partial Hamiltonian path H': $e_{1,10} \rightarrow e_{1,2} \rightarrow e_{2,3} \rightarrow e_{3,8} \rightarrow e_{8,9} \rightarrow e_{4,8} \rightarrow e_{4,5} \rightarrow e_{3,4} \rightarrow e_{3,7} \rightarrow e_{2,7} \rightarrow e_{6,7} \rightarrow e_{6,11}$ (with 13 times device wake-up). On the other hand, even if there is algorithm that can find H', it is less helpful for finding optimal SOS solutions toward this instance. In fact, piecing H' with other unvisited nodes $e_{9,13}$ and $e_{5,12}$ (with additional four times device wake-up) will lead to a schedule with in total 17 times wake-up, which is not optimal apparently.
2) It is well-known NP-hard to *find a partial but longest Hamiltonian path*, so is to find optimal SOS solutions.



(a) (b)

Fig. 10. Counterexample to the VDF strategy. (a) Original topology. (b) Edge-dual topology.

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 12:01:37 UTC from IEEE Xplore. Restrictions apply.

Thus, we cannot hope to work out a heuristic algorithm that solves all cases efficiently, unless P=NP happens.
3) The high success probability of our simulations in Section IV suggests that this limitation is not widespread.

B. Roubustness of the Proposed System

It is well known that the security of coordinator device plays a key role in the SOS of IoT system. To enhance the robustness of our system against attacks aiming at coordinator, two potential countermeasures could be considered. The first countermeasure is incorporating our proposed system into customized IoT communication protocol, and implementing the protocol in the firmware level of IoT device. In this case, attacking coordinator needs to change the device's hardware design, while this strong requirement of physical proximity in this attack make the threat less concern. The second countermeasure is dynamic coordinator selection. In this case, the system dynamically chooses some IoT devices as coordinators, and monitors the SOS running state. When the SOS success probabilities is decreasing, it could mean that some of the current coordinators are suffering from attacks. Then, our system will change the coordinator devices, try to find and disable the malicious coordinator and alert the user.

VI. Conclusion

Let the involved idle devices go to sleep is a fundamental idea for enlarging the lifetime of an IoT system. But too frequent switching between awake phase and asleep phase will also consume significant amount of powers. Thus, it is interesting to schedule the order of multiple communication streams among multiple devices so that the total wake-up times achieve as small as possible. In this paper, a heuristic algorithm for SOS is developed based on some intuitive insights on the edge dual graph. Then, based on simulations toward 4800 random original topologies and 19 000 random Hamiltonian edge-dual topologies, we find that on one hand, the denser the stream the higher the success probability of reaching optimal SOS solutions. In particular, if the edge-dual contains a Hamiltonian path, our algorithm can find the optimal SOS solution with a very high success probability, nearly 96%. Considering that some previous work was blocked by the difficulty for finding Hamiltonian paths, our algorithm performs considerably well. On the other hand, there in no apparent relationship between the scale of device numbers and the success probability of our algorithm reaching optimal SOS solutions.

Appendix A
Proof of Theorem 1

The VDF algorithm can be divided into three stages.
1) Building the edge-dual graph, from step L0 to the beginning of step L1.
2) Finding a (nearly) optimal solution (i.e., partial Hamilton paths), from step L1 to the beginning of step L2.
3) Scheduling the remaining streams (if any), from step L2 to the end.

Suppose that $|V^*| = |E| = m$ and $|E^*| \leq |V^* \times V^*| = m^2$ (since $E^* \subset V^* \times V^*$). It is easy to see that the complexities of the first and third stages are bounded by $\mathcal{O}(m^2)$ and $\mathcal{O}(m)$, respectively.

For the second stage, the main task can also be divided into three substeps.
(2.1) *Iteratively* select a candidate node from $V^* \setminus H$ that can be appended to the tail of H (i.e., the light blue box on the left) or inserted at the head of H (i.e., the light blue box on the right).
(2.2) Vertex visiting and dropping. This means to visit the selected node in the substep (2.1), and after visiting, all edges associated to this node should be dropped.
(2.3) Edge recovering. This occurs only one time when the substep (2.1) cannot find new candidate node while some nodes have not been visited yet. So, we need to recover edges that are dropped at the first time execution of the substep (2.2) (i.e., these edges are associated to the head node in the queue H).

Apparently, the time complexity of substep (2.3) is bounded by maximal degree [$\ll \mathcal{O}(m)$ in general] of the edge-dual graph since it executes only one-time. The total complexity of substep (2.2) is exactly $\mathcal{O}(|V^*|+|E^*|) = \mathcal{O}(m^2)$ since neither a node nor an edge can be dropped more than twice, considering that some nodes and edges will be recovered in the substep (2.3) for only one-time. Thus, the left thing is to count the total number of comparisons for selecting the candidate nodes in substep (2.1).

Suppose that the sequence of degrees in the edge-dual graph $G^* = (V^*, E^*)$ is $\{\deg_i\}_{i=1}^m$. For choosing the first head node of H, we need no more than $m-1$ comparisons. In subsequent iterations, for choosing the $(i+1)$th node in the queue H, the required number of comparisons is no more than $\deg_i$. Without loss of generality, suppose that a node of degree $\deg_1 > 0$ is the final node selected for inclusion in H. Then, the total number of comparisons needed for seeking and piecing together partial Hamilton paths in H is bounded by

$$(m-1) + \sum_{i=2}^{m} \deg_i = (m-1) + 2|E^*| - \deg_1 < 2(m + |E^*|)$$

where the first equality arises from the fact that $\sum_{i=1}^m \deg_i = 2|E^*|$. By combining this with the fact $|E^*| \leq m^2$, we conclude the theorem.

Appendix B
Detailed Execution of the MDS Algorithm

According to [2], [17], [22], the MDS algorithm proceeds as follows.
1) For each edge $e = (i, j)$ in the original topology, assign a weight to that edge equal to the sum of the degrees of its two ends, i.e., $w_e \triangleq \deg_i + \deg_j$.
2) Suppose that the current visited edge is e. Then, select the next edge e^* in accordance with the following optimization: $e^* = \min_{e' \cap e \neq \emptyset} w_{e'}$, where $e' \cap e \neq \emptyset$ indicates that e and e' are adjacent to each other.

Then, for the example of the SOS instance given in Fig. 6(a), the sequence of (e/w_e) (i.e., edges over weights) is as listed

$$\frac{e_{12}}{6} \quad \frac{e_{15}}{6} \quad \frac{e_{23}}{7} \quad \frac{e_{24}}{7} \quad \frac{e_{28}}{7} \quad \frac{e_{34}}{6} \quad \frac{e_{38}}{6}$$
$$\frac{e_{48}}{6} \quad \frac{e_{56}}{7} \quad \frac{e_{57}}{7} \quad \frac{e_{59}}{7} \quad \frac{e_{67}}{6} \quad \frac{e_{69}}{6} \quad \frac{e_{79}}{6}.$$

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 12:01:37 UTC from IEEE Xplore. Restrictions apply.

Therefore, if the MDS algorithm happens to start from edge e_{12}, then it will output the following scheduling sequence, with a total of 16 wake-up events:

$$e_{12} \rightarrow e_{15} \rightarrow e_{56} \rightarrow e_{67} \rightarrow e_{69} \rightarrow e_{79} \rightarrow e_{57} \rightarrow e_{59}$$
$$\dashrightarrow e_{34} \rightarrow e_{38} \rightarrow e_{48} \rightarrow e_{28} \rightarrow e_{23} \rightarrow e_{24}$$

where the symbol "$\dashrightarrow$" indicates that after visiting e_{59}, because there is no adjacent unvisited edge, the MDS algorithm selects a new set of edges with the minimum weight from among the remaining unvisited edges. As seen from the scheduling sequence given above, the total number of wake-up events is 16.

References

[1] L. Wang, Y. Pan, M. Jia, and A. Haseeb, "A heuristic stream order scheduling algorithm for intra-superframe power management in WPANs," in *Wireless Algorithms, Systems, and Applications* (LNCS 2015), K. Xu and J. Zhu, Eds. Cham, Switzerland: Springer Int., 2015, pp. 539–549.

[2] N.-C. Wang and C.-C. Wen, "A performance study for power management schemes in WPANs," in *Proc. Int. Symp. Comput. Consum. Control (IS3C)*, Jun. 2012, pp. 427–430.

[3] Y. Meng, W. Zhang, H. Zhu, and X. S. Shen, "Securing consumer IoT in the smart home: Architecture, challenges, and countermeasures," *IEEE Wireless Commun.*, vol. 25, no. 6, pp. 53–59, Dec. 2018.

[4] Y. Meng *et al.*, "WIVo: Enhancing the security of voice control system via wireless signal in IoT environment," in *Proc. 18th ACM Int. Symp. Mobile Ad Hoc Netw. Comput. (Mobihoc)*, 2018, pp. 81–90. [Online]. Available: http://doi.acm.org/10.1145/3209582.3209591

[5] H. Zhu, X. Lin, R. Lu, Y. Fan, and X. Shen, "SMART: A secure multilayer credit-based incentive scheme for delay-tolerant networks," *IEEE Trans. Veh. Technol.*, vol. 58, no. 8, pp. 4628–4639, Oct. 2009.

[6] H. Zhu, S. Du, Z. Gao, M. Dong, and Z. Cao, "A probabilistic misbehavior detection scheme toward efficient trust establishment in delay-tolerant networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 1, pp. 22–32, Jan. 2014.

[7] M. Dong, K. Ota, A. Liu, and M. Guo, "Joint optimization of lifetime and transport delay under reliability constraint wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 1, pp. 225–236, Jan. 2016.

[8] M. Dong, K. Ota, L. T. Yang, S. Chang, H. Zhu, and Z. Zhou, "Mobile agent-based energy-aware and user-centric data collection in wireless sensor networks," *Comput. Netw.*, vol. 74, pp. 58–70, Dec. 2014.

[9] Z. Zhou, M. Dong, K. Ota, J. Wu, and T. Sato, "Energy efficiency and spectral efficiency tradeoff in device-to-device (D2D) communications," *IEEE Wireless Commun. Lett.*, vol. 3, no. 5, pp. 485–488, Oct. 2014.

[10] J. Liu and N. Kato, "Device-to-device communication overlaying two-hop multi-channel uplink cellular networks," in *Proc. 16th ACM Int. Symp. Mobile Ad Hoc Netw. Comput. (MobiHoc)*, Jun. 2015, pp. 307–316.

[11] H. Zhu, R. Lu, X. Shen, and X. Lin, "Security in service-oriented vehicular networks," *IEEE Wireless Commun.*, vol. 16, no. 4, pp. 16–22, Aug. 2009.

[12] J. Liu, S. Zhang, H. Nishiyama, N. Kato, and J. Guo, "A stochastic geometry analysis of D2D overlaying multi-channel downlink cellular networks," in *Proc. IEEE INFOCOM*, 2015, pp. 46–54.

[13] J. Liu, X. Jiang, H. Nishiyama, and N. Kato, "Exact throughput capacity under power control in mobile ad hoc networks," in *Proc. IEEE INFOCOM*, 2012, pp. 1–9.

[14] W. Zhang, Y. Meng, Y. Liu, X. Zhang, Y. Zhang, and H. Zhu, "HoMonit: Monitoring smart home apps from encrypted traffic," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security (CCS)*, 2018, pp. 1074–1088. [Online]. Available: http://doi.acm.org/10.1145/3243734.3243820

[15] T. Wang, G. Zhang, A. Liu, M. Z. A. Bhuiyan, and Q. Jin, "A secure IoT service architecture with an efficient balance dynamics based on cloud and edge computing," *IEEE Internet Things J.*, to be published.

[16] S. Basu *et al.*, "An intelligent/cognitive model of task scheduling for IoT applications in cloud computing environment," *Future Gener. Comput. Syst.*, vol. 88, pp. 254–261, Nov. 2018.

[17] Z. Guo, R. Yao, W. Zhu, X. Wang, and Y. Ren, "Intra-superframe power management for IEEE 802.15.3 WPAN," *IEEE Commun. Lett.*, vol. 9, no. 3, pp. 228–230, Mar. 2005. doi: 10.1109/LCOMM.2005.03023.

[18] *LAN MAN Standards Committee. Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Wireless Personal Area Networks (WPANs)*, IEEE Comput. Soc., Washington, DC, USA, 2002.

[19] *LAN MAN Standards Committee. Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for High Rate Wireless Personal Area Networks (WPANs)*, IEEE Standard 802.15.3-2003, 2003.

[20] D. Li, P. H. Chou, and N. Bagherzadeh, "Mode selection and mode-dependency modeling for power-aware embedded systems," in *Proc. 7th Asia South Pac. 15th Int. Conf. VLSI Design (VLSID)*, Bengaluru, India, Jan. 2002, pp. 697–704.

[21] J. A. Stine and G. de Veciana, "Improving energy efficiency of centrally controlled wireless data networks," *Wireless Netw.*, vol. 8, no. 6, pp. 681–700, 2002. doi: 10.1023/A:1020379326558.

[22] N.-C. Wang and C.-C. Wen, "An efficient power management scheme with SDMA for IEEE 802.15.3 WPANs," *Int. J. Commun. Syst.*, vol. 27, no. 12, pp. 4092–4102, 2014. doi: 10.1002/dac.2600.

[23] H. Chen, J. Wen, W. Pedrycz, and G. Wu, "Big data processing workflows oriented real-time scheduling algorithm using task-duplication in geo-distributed clouds," *IEEE Trans. Big Data*, to be published.

[24] Y.-J. Chiang and H.-F. Hsiao, "Expectation model and scheduling for video streaming," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2018, pp. 1–5.

[25] H. Kim and K. Chung, "Segment scheduling scheme for efficient bandwidth utilization of HTTP adaptive streaming in multipath environments," *IEEE Access*, vol. 7, pp. 36910–36920, 2019.

[26] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[27] K. Gai, M. Qiu, Z. Hui, L. Tao, and Z. Zong, "Dynamic energy-aware cloudlet-based mobile cloud computing model for green computing," *J. Netw. Comput. Appl.*, vol. 59, pp. 46–54, Jan. 2016.

[28] K. Gai, M. Qiu, and Z. Hui, "Energy-aware task assignment for mobile cyber-enabled applications in heterogeneous cloud computing," *J. Parallel Distrib. Comput.*, vol. 111, pp. 126–135, Jan. 2018.

[29] M. Vomlelová and J. Vomlel, "Troubleshooting: NP-hardness and solution methods," *Soft. Comput.*, vol. 7, no. 5, pp. 357–368, 2003. doi: 10.1007/s00500-002-0224-4.

[30] E. L. Lawler, "Fast approximation algorithms for knapsack problems," *Math. Oper. Res*, vol. 4, no. 4, pp. 339–356, 1979.

[31] N. Garg and R. Khandekar, "Fast approximation algorithms for fractional Steiner forest and related problems," in *Proc. 43rd Annu. IEEE Symp. Found. Comput. Sci. (FOCS)*, Vancouver, BC, Canada, Nov. 2002, pp. 500–509.

[32] S. Xie and Y. Wang, "Construction of tree network with limited delivery latency in homogeneous wireless sensor networks," *Wireless Pers. Commun.*, vol. 78, no. 1, pp. 231–246, 2014. doi: 10.1007/s11277-014-1748-5.

[33] H. Braun, "On solving travelling salesman problems by genetic algorithms," in *Proc. 1st Workshop Parallel Prob. Solving Nature (PPSN)*, vol. 496, Oct. 1991, pp. 129–133.

[34] M. Grötschel and O. Holland, "Solution of large-scale travelling salesman problems," *Math. Program.*, vol. 51, no. 2, pp. 141–202, 1991.

[35] S. Maity, A. Roy, and M. Maiti, "A modified genetic algorithm for solving uncertain constrained solid travelling salesman problems," *Comput. Ind. Eng.*, vol. 83, pp. 273–296, May 2015. doi: 10.1016/j.cie.2015.02.023.

[36] A. Ouaarab, B. Ahiod, and X.-S. Yang, "Discrete cuckoo search algorithm for the travelling salesman problem," *Neural Comput. Appl.*, vol. 24, nos. 7–8, pp. 1659–1669, 2014. doi: 10.1007/s00521-013-1402-2.

[37] M. G. H. Omran, A. P. Engelbrecht, and A. A. Salman, "Barebones particle swarm for integer programming problems," in *Proc. IEEE Swarm Intell. Symp. (SIS)*, Honolulu, HI, USA, Apr. 2007, pp. 170–175.

[38] Z. Xia, X. Wang, x. Sun, Q. Liu, and N. Xiong, "Steganalysis of LSB matching using differences between nonadjacent pixels," *Multimedia Tools Appl.*, vol. 75, no. 4, pp. 1947–1962, 2016. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01096138

[39] B. Gu, V. S. Sheng, Z. Wang, D. Ho, S. Osman, and S. Li, "Incremental learning for ν-support vector regression," *Neural Netw.*, vol. 67, pp. 140–150, Jul. 2015. doi: 10.1016/j.neunet.2015.03.013.

[40] X. Wen, L. Shao, Y. Xue, and W. Fang, "A rapid learning algorithm for vehicle classification," *Inf. Sci.*, vol. 295, pp. 395–406, Feb. 2015. doi: 10.1016/j.ins.2014.10.040.

[41] D. Bienstock and S. Mattia, "Using mixed-integer programming to solve power grid blackout problems," *Discr. Optim.*, vol. 4, no. 1, pp. 115–141, 2007.

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 12:01:37 UTC from IEEE Xplore. Restrictions apply.

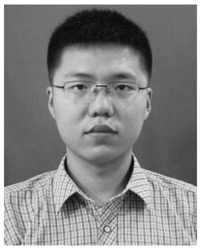
[42] R. Montemanni, “Integer programming formulations for maximum lifetime broadcasting problems in wireless sensor networks,” *Wireless Sensor Netw.*, vol. 2, no. 12, pp. 924–935, 2010. doi: 10.4236/wsn.2010.212111.

[43] M. M. Paydar, I. Mahdavi, and K. A. Szabat, “Application of single depot multiple travelling salesman method to cell formation problems,” *Int. J. Appl. Decis. Sci.*, vol. 3, no. 4, pp. 390–399, 2010. doi: 10.1504/IJADS.2010.036853.

[44] A. Baltz, M. E. Ouali, G. Jäger, V. Sauerland, and A. Srivastav, “Exact and heuristic algorithms for the travelling salesman problem with multiple time windows and hotel selection,” *J. Oper. Res. Soc.*, vol. 66, no. 4, pp. 615–626, 2015. doi: 10.1057/jors.2014.17.

Licheng Wang received the B.S. degree in engineering from Northwest Normal University, Lanzhou, China, in 1995, the M.S. degree in mathematics from Nanjing University, Nanjing, China, in 2001, and the Ph.D. degree in engineering from Shanghai Jiaotong University, Shanghai, China, in 2007.

He is currently an Associate Professor with the Beijing University of Posts and Telecommunications, Beijing, China. His current research interests include cryptography, blockchain, and future Internet architecture.



Yan Meng received the B.S. degree in electronic and information engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2016, where he is currently pursuing the Ph.D. degree at the Department of Computer Science and Engineering.

His current research interests include wireless network security and Internet of Things security.

Haojin Zhu received the B.Sc. degree in computer science from Wuhan University, Wuhan, China, in 2002, the M.Sc. degree in computer science from Shanghai Jiao Tong University, Shanghai, China, in 2005, and the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2009.

Since 2017, he has been a Full Professor with the Computer Science Department, Shanghai Jiao Tong University. He has authored or coauthored over 40 international journal papers, including in the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, the IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEM, the IEEE TRANSACTIONS ON MOBILE COMPUTING, the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, and the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, and 60 international conference papers, including in the ACM CCS, ACM MOBICOM, ACM MOBIHOC, IEEE INFOCOM, and IEEE ICDCS. His current research interests include network security and privacy enhancing technologies.

Dr. Zhu was a recipient of a number of awards, including the IEEE ComSoc Asia–Pacific Outstanding Young Researcher Award in 2014, the Top 100 Most Cited Chinese Papers Published in International Journals in 2014, the Supervisor of Shanghai Excellent Master Thesis Award in 2014, the Distinguished Member of the IEEE INFOCOM Technical Program Committee in 2015, the Outstanding Youth Post Expert Award for Shanghai Jiao Tong University in 2014, the SMC Young Research Award of Shanghai Jiao Tong University in 2011, and the Young Scholar Award of Changjiang Scholar Program from the Ministry of Education of China in 2016. He was a co-recipient of the Best Paper Award of IEEE ICC in 2007 and Chinacom in 2008, the IEEE GLOBECOM Best Paper Nomination in 2014, and the WASA Best Paper Runner-Up Award in 2017.

Minxing Tang received the B.E. degree in information security from the Beijing University of Posts and Telecommunications, Beijing, China, in 2016, and the M.S. degree in computer science from the Humboldt University of Berlin, Berlin, Germany, in 2018. She is currently pursuing the Ph.D. degree at the Institute of Computer Science, Humboldt University of Berlin, Berlin.

She is currently with the Department of Computer Science, Humboldt University of Berlin. Her current research interests include software testing and debugging.

Kaoru Ota was born in Aizuwakamatsu, Japan. She received the B.S. degree in computer science and engineering from Oklahoma State University, Stillwater, OK, USA, in 2006, the M.S. degree in computer science from Oklahoma State University, Stillwater, OK, USA, in 2008, and the Ph.D. degree in computer science and engineering from the University of Aizu, Aizuwakamatsu, Japan, in 2012.

She is currently an Assistant Professor with the Department of Information and Electronic Engineering, Muroran Institute of Technology, Muroran, Japan. From 2010 to 2011, she was a Visiting Scholar with the University of Waterloo, Waterloo, ON, Canada. She was also a Japan Society of the Promotion of Science Research Fellow with the Kato-Nishiyama Laboratory, Graduate School of Information Sciences, Tohoku University, Sendai, Japan, from 2012 to 2013. Her current research interests include wireless networks, cloud computing, and cyber-physical systems.

Dr. Ota was a recipient of the Best Paper Award of ICA3PP 2014, GPC 2015, IEEE DASC 2015, IEEE VTC 2016-Fall, and FCST 2017, the 2017 IET Communications Premium Award, the IEEE ComSoc CSIM Best Conference Paper Award in 2018, the IEEE TCSC Early Career Award in 2017, and the 13th IEEE ComSoc Asia–Pacific Young Researcher Award in 2018. She is an Editor of the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE COMMUNICATIONS LETTERS, *Peer-to-Peer Networking and Applications* (Springer), *Ad Hoc & Sensor Wireless Networks*, *International Journal of Embedded Systems* (Inderscience), and Smart Technologies for Emergency Response and Disaster Management (IGI Global), as well as a Guest Editor of the *ACM Transactions on Multimedia Computing, Communications and Applications* (leading), the IEEE INTERNET OF THINGS JOURNAL, *IEEE Communications Magazine*, *IEEE Network*, *IEEE Wireless Communications Magazine*, IEEE ACCESS, *IEICE Transactions on Information and Systems*, and *Ad Hoc & Sensor Wireless Networks* (Old City Publishing).

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 12:01:37 UTC from IEEE Xplore. Restrictions apply.