Adaptive Topology Reconstruction Mechanism considering Routing Hop and Battery Consumption for Computing Offloading of IoT

Hyun-Woo Kim, Eun-Ha Song

Abstract*: *In this paper, we propose ATRM which handles computing offloading considering routing hops connected in IoT topology environment. ATRM measures battery consumption of other connected devices to computing offloading computational tasks that are difficult to handle in a single device in an IoT environment. In addition, the lower connected routing device is changed dynamically considering the number of routing hops. In the past, studies have been conducted on computing resources such as CPU, memory, and storage, which are dynamic performance of IoT devices, for computing offloading in the IoT environment, or using external resources according to arbitrarily set computing offloading weights. There is a lack of research on distributed processing. High-throughput computing is possible by reconfiguring according to the routing hop of the lower connected IoT device. In this paper, it is possible to set thresholds for arbitrary routing hops and battery consumption of connected IoT devices for computing offloading. This improves IoT topology lifetime and enables many computing offloading processes.

Index Terms*: *Adaptive Topology Reconstruction, Internet of Things, IoT Topology, Routing Hop, Computing Offloading

I. INTRODUCTION

Recently, various researches have been actively conducted for realizing the Internet of Things (IoT) based on real life. IoT allows objects to be recognized and tracked using tags or GPS functions on objects, and data collected through network connections is used for various services. IoT devices are miniaturized mobile devices with single or multi-sensing functions such as temperature, humidity, illumination, gyro, acceleration, and pressure. By using these IoT devices, the efficiency of individuals and corporations is increased, and convenience for convenience of mobility and location is increased. However, IoT devices have a problem that they depend on limited computing capacity, storage capacity, and built-in battery. Storage can be stored over 2TB by SD card, but it cannot be provided smoothly because it needs to be charged or replaced when the built-in battery is exhausted [1, 2, 3, 4, 5, 6]. Thus, computing offloading using a limited battery is very important. Computing offloading refers to the task of computing in a single device, or in the event that the remaining battery power cannot perform the computing task, the computing work is distributed to the connected IoT device. This enables a lot of work to be done by collaborating between IoT devices. However, an IoT device that routes work allocation and processing results of intensively connected IoT devices has a problem that it becomes a dead node only by routing [2, 6, 7, 8, 9, 10, 11, 12, 13, 14]. In [6], a traffic threshold is set for computing offloading of IoT devices, and the mobile cloud is used when the threshold is higher than the threshold. When the threshold is lower than the threshold, a method of processing computing offloading using the computing resources of the IoT device is proposed. Although there is a short time for selection of the object to be processed by a simple threshold, there is a problem that the performance of the IoT device cannot be performed when the performance of the IoT device is not considered and many operations are being performed or the remaining battery capacity is insufficient. In [7], we proposed Mobile Resource Management (MRM) for computing offloading in a resource-integrated environment consisting of only IoT devices. MRM is divided into configuration, management and operation to form a computing resource pool and to perform computing offloading. In consideration of the performance information of IoT devices such as Storage, Memory, and CPU, tasks requested from other IoT devices are allocated. Fault-tolerance is proposed in order to cope with the exception of IoT device's inability to perform the operation. Performance evaluation showed high availability of servers that continuously operate as long as IoT topology is operated. However, focusing on server performance rather than maintaining a connected IoT topology, other IoT devices can experience drastic battery drain and lower throughput of computing offloading. In [8], we proposed a CDS reconstruction algorithm in IoT environment. In this case, when the CDS node is set to balance the load, which is a problem caused by the critical node occurrence of the CDS-based routing protocol, the topology reconfiguration due to the critical node is extended to the two-hop node. If the inter-node connectivity is within the range, the efficiency depends on the connection reconfiguration time. However, reconfiguration caused by frequent threshold node occurrence may adversely affect overall topology lifetime.

Revised Manuscript Received on May 22, 2019.

Hyun-Woo Kim, Professor, Department of Information Security, Baewha Women's University, Seoul, Republic of Korea.
Eun-Ha Song, Corresponding author, Professor, Collage of Convergence & Liberal Arts, Wonkwang University, Iksan, Jeonbuk, Republic of Korea.

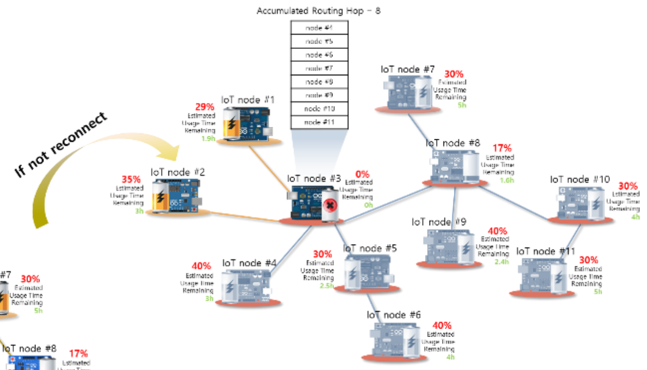
Therefore, this paper proposes an improved topology considering the entire topology and the battery consumption rate of each IoT device without considering only threshold nodes.

In [9], we use a method to remove unnecessary CDS configuration nodes to approximate the minimum number of gateway nodes constituting CDS in IoT environment. It is proposed to increase the overall lifetime of the topology by reducing the energy consumption of the mobile nodes and fixed nodes, but it is not suitable for frequent data collection environments such as big sensing data. In this paper, we reconfigure the topology for computing offloading by providing proper routing according to battery consumption rate considering overall connectivity rather than simply making the number of connected nodes coherent.

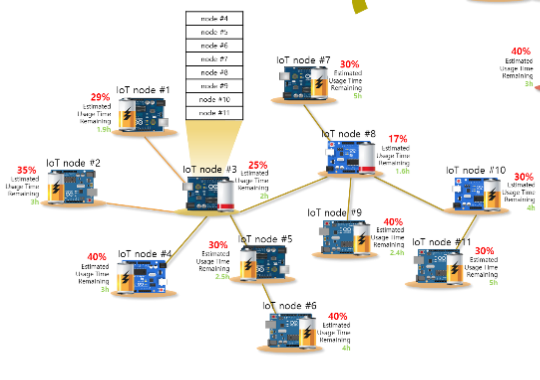
In this paper, we propose an adaptive topology reconstruction mechanism (ATRM) that reconfigures the topology adaptively considering current battery capacity and routing hops that are connected to each other. The composition of this paper is as follows. In Section 1, computing offloading of existing IoT is examined, and in Section 2, operation flow and design of ATRM are explained. Section 3 describes the analyzed performance evaluation of the ATRM. And finally, Section 4 summarizes of overall.

II. ATRM Design

The environment without ATRM proposed in this paper is shown in Fig. 1, and the environment with ATRM is shown in Fig. 2. Fig. 1 (a) shows the current state of IoT topology in which intensive routing occurs. When such continuous routing is performed, Fig. 1 (b) IoT topology cannot receive the information of the lower connected IoT device. Fig. 2 (a) is the same as Fig. 1 (a), and it reconfigures through the ATRM by changing the minimum connection state as shown in Fig. 2 (b) IoT topology. ATRM analyzes the available time of each device by using cumulative battery consumption. The cumulative battery consumption is periodically calculated using the battery consumption measured in a predetermined cycle and the battery consumption measured in the next cycle. At this time, the cumulative number of routing hops is calculated and compared with the average number of routing hops. Identify an IoT device with a large number of routing



(b) IoT Topology (as time goes on)

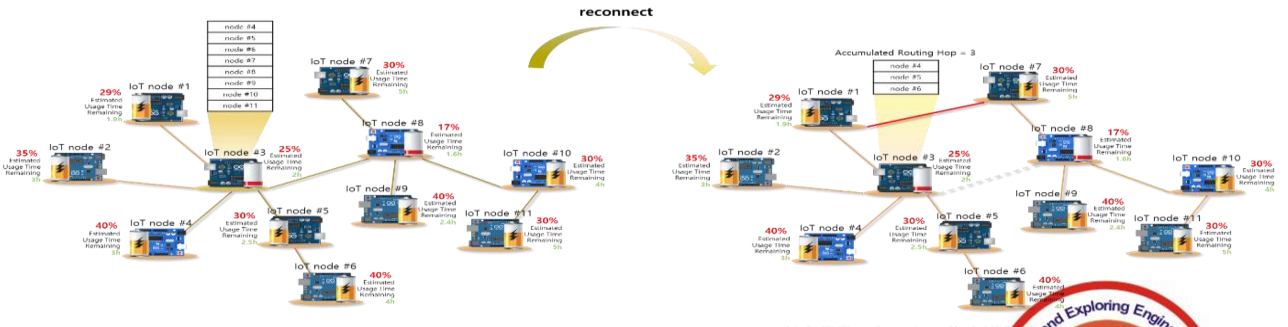


(a) IoT Topology (current status)

Node name	Remaining Volume	Remaining Time	Near Node
node #1	29%	1.9h	node #2, node #3, node #7
node #2	35%	3h	node #1, node #3, node #4
node #3	25%	2h	node #1, node #2, node #5
node #4	40%	3h	node #2, node #3, node #5
node #5	30%	2.5h	node #3, node #4, node #6
node #6	40%	4h	node #5
node #7	30%	5h	node #1, node #8
node #8	17%	1.6h	node #7, node #9, node #10
node #9	40%	2.4h	node #5
node #10	30%	4h	node #11
node #11	30%	5h	node #10

[Reconnect Criteria Table]

Fig. 1 : Existing IoT Topology



(a) IoT Topology (current status)

(b) IoT Topology (applied ATRM)

Fig. 2 : ATRM overview

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

hops while accumulating a large amount of battery power, and reconfigure the IoT device to connect to an IoT device with a small number of routing hops while consuming less battery power among adjacent IoT devices.

ATRM configuration is subdivided into User Interface, Map Manager, Log Manager, Interaction Broker, Node Manager, Map Controller, and Viewer as shown in Fig. 3.

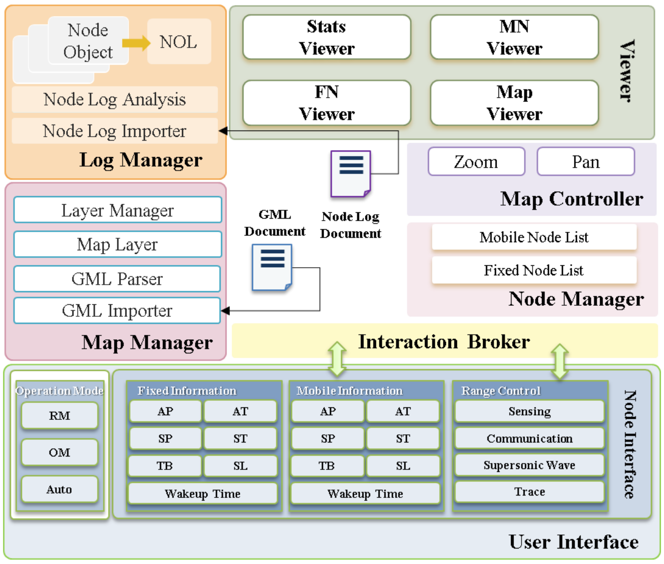


Fig. 3 : ATRM Architecture

User Interface consists of Range Control, Mobile Information, Fixed Information and Operation Mode in detail. Mobile Information and Fixed Information are composed of Active Power (AP), Active Time (AT), Sleep Power (SP), Sleep Time (ST), Total Battery (TB), SL (Sensor Location) and Wakeup Time. Range Control consists of Sensing, Communication, Supersonic Wave, and Trace for interconnect and data acquisition in IoT devices. Operation Mode is composed of RM (Routing Mode), which is the task processing or sensing information of lower connected IoT devices in IoT topology environment with ATRM applied, and OM (Operation Mode), which is a function for processing tasks for computing offloading, it consists of Auto that adaptively operates according to battery consumption and cumulative routing hop count.

Node Manager collectively manages the mobile node list and the fixed node list according to whether the deployed IoT devices can be moved or not. Each IoT device updates its routing hop count and battery consumption information via Log Manager.

The Interaction Broker acts as a broker to send the operation mode input from the User Interface and the changed settings of the IoT device to the Map Controller, Node Manager, and Log Manager.

Map Manger applies ATRM to the location of the IoT device to the actual terrain through the GML document. GML Importer for adding GML document selected by user to ATRM in detail, GML parser for analyzing GML document added, and Map objects for determining obstacles according to objects of analyzed GML terrain data, A Map Layer for sending, and a Layer Manager for mapping and managing the topology information received from the Map Layer and the Mobile Node List of the Node Manager and the SL of the Fixed Node List.

Log Manager is a Node Log Collector for periodically measuring battery consumption of IoT devices, Node Log Analysis for analyzing the lifetime of IoT devices by comparing collected battery consumption and routing hops, Node for analyzing each IoT device individually Object and NOL (Node Object List).

Map Controller is responsible for the reduction and enlargement of the IoT topology visualized by the Viewer.

Viewer consists of MN (Mobile Node) Viewer, FN (Fixed Node) Viewer, Map Viewer and Stats Viewer to check IoT topology situation which operates internally

III. ATRM PERFORMANCE EVALUATION

In order to evaluate the performance of ATRM, we confirmed the changing state when periodically requesting computing offloading in IoT topology with ATRM applied and IoT topology not applied. Fig. 4 shows the change in the number of IoT devices in the IoT topology when the workload is increased to 1, 10, 20, 30, 40, and 50 over time. Artificially, computing offloading performed a multiplication of 500 by 500 metrics. Each device was run on raspberry pi 2 and pi 3 models. In the case of 1 hour to 2 hours, there is no significant difference. However, since 3 hours, dead nodes are generated in IoT topology which does not apply ATRM due to intensive routing. Also, in 5 hours, the IoT devices connected to the subordinate cannot be connected, and two IoT device resources are lost. The IoT

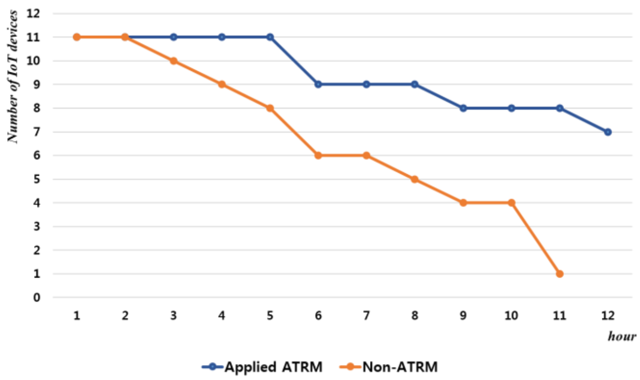


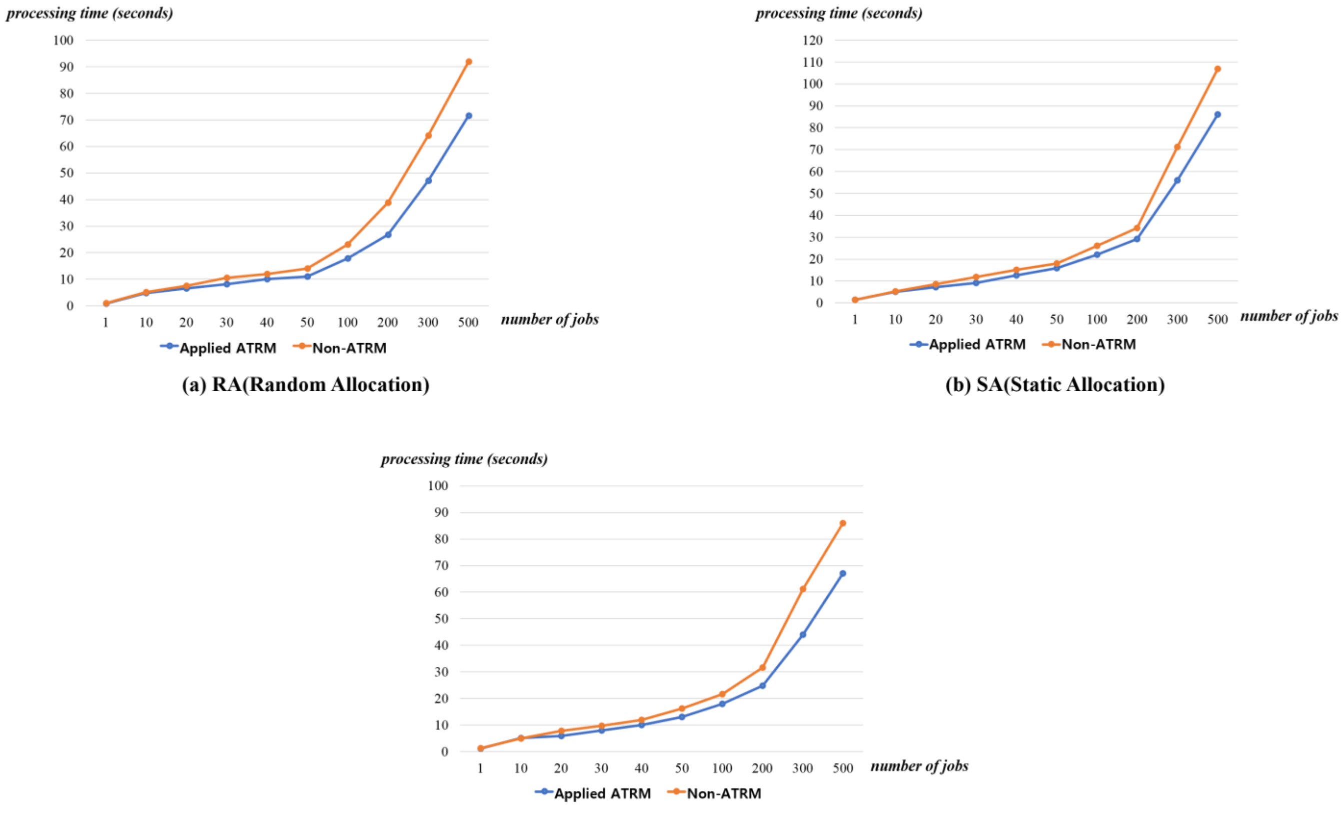
Fig. 4 : Comparison of IoT topology with ATRM

topology with ATRM was confirmed to be efficient because 8 connections were maintained in 11 hours.

As a result of applying the ATRM proposed in this paper, the performance analysis results of the task performance are shown in Fig. 5.

In order to analyze the results of the job performance, Fig. 5 shows the case where RA(random allocation) is allocated randomly as job, SA(static allocation) is assigned job based on static performance information, DA(dynamic allocation) is assigned as job based on dynamic performance information, Otherwise, the case was analyzed.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication



(c) DA(Dynamic Allocation)

Fig. 5 : Comparing jobs processing time by job allocation method in IoT topology with ATRM and IoT topology of non-ATRM

RA generated Java-based system-dependent random values. At this time, the generated random value is set not to exceed the total number of IoT devices. SA allocates tasks based on IoT basic performance information using the amount of memory required for the task. To accomplish this, performance information of IoT devices is collected and sorted in ascending order so that sequential task allocation is possible. DA collects the performance information of the IoT device, together with the basic performance information, together with the performance information currently in use. The residual performance information was calculated through this. We used the residual performance information to sort the list of IoT devices and assign tasks to IoT devices with the best performance. This RA task allocation collects and recalculates the performance information of IoT devices every time a task is requested from the user. This is because the dynamic performance information of the IoT device changes when the task is assigned. As shown in Fig. 5, there is no significant difference when the number of jobs is less than 50 in a (RA), b (SA), and c (DA). From the time when more than 50 jobs are requested, it is confirmed that the routing process of frequently processed data occurs according to intensive work requests. In this case, it is confirmed that ATRM is applied more quickly than when ATRM is not applied. This is because, in the case where ATRM is not applied, a dead node is generated due to an intensive routing process of an assigned task and the connection is released. In addition, the delay time for assigning a new IoT device task and the delay time due to a lot of routing process occur when the task is released.

IV. CONCLUSION

In this paper, we proposed ATRM that reconfigures the topology adaptively by considering the current battery amount and routing hops that are connected to each other for computing offloading in IoT topology. ATRM slowed the imbalance of IoT topology caused by intensive routing and maintained maximum connectivity. However, when the threshold is not set, frequent IoT topology reconfiguration causes a delay. In order to apply efficient ATRM, it is important to set cumulative routing hops and battery consumption ratio to the user's threshold value relative to the remaining battery capacity.

In the future, we will study not only to set the threshold value from the user for optimal computing offloading in the IoT topology environment using ATRM, but also to enable self - determination by using the existing log data. In addition, we intend to conduct research that can divide various tasks in computing offloading. If the task is algorithm-dependent, we want to distribute all of the specific algorithms and study the input of segmentable data from the user.

ACKNOWLEDGMENT

This paper was supported by Wonkwang University in 2019.

REFERENCES

1. Tie Qiu, Yuan Lv, Feng Xia, Ning Chen, Jiafu Wan, Amr Tolba, "ERGID: An efficient routing protocol for emergency response Internet of Things," Journal of Network and Computer Applications, Vol. 72, 2016, pp. 104-112.
2. Hakan Erdol, Sedat Gormus, Mehmet Cemil Aydogdu, "A Novel Energy Aware Routing Function for Internet of Things Networks," In Proceedings of 2017 10th International Conference on Electrical and Electronics Engineering, IEEE, 2017, pp. 1314-1318.
3. Gaojie Chen, Jinchuan Tang, Justin P. Coon, "Optimal Routing for Multihop Social-Based D2D Communications in the Internet of Things," IEEE Internet of Things Journal, Vol. 5, No. 3, 2018, pp. 1880-1889.
4. Samia Allaoua Chelloung, "Energy-Efficient Content-Based Routing in Internet of Things," Journal of Computer and Communications, Vol. 3, 2015, pp. 9-20.
5. Zhaolong Ning, Peiran Dong, Xiangjie Kong, Feng Xia, "A Cooperative Partial Computation Offloading Scheme for Mobile Edge Computing Enabled Internet of Things," IEEE Internet of Things Journal, Early Access, 2018, pp. 1-11.
6. Lu Z, Zhao J. Wu Y, Cao G, "Task Allocation for Mobile Cloud Computing in Heterogeneous Wireless Networks," In Proceedings of 24th International Conference on Computing Communication and Networks (ICCCN), IEEE, 2015, pp. 1-9.
7. Kim B, Byun H, Heo Y.-A, Jeong Y.-S, "Adaptive Job Load Balancing Scheme on Mobile Cloud Computing with Collaborative Architecture," Symmetry, Vol. 9, No. 5, 2017, pp. 1-14.
8. Youn-Sik Hong, Hwa-Seok Lim, "A Load-balanced Topology Maintenance with Partial Topology Reconstruction," The Journal of Korean Institute of Communications and Information Sciences, Vol. 35, No.12, 2010, pp. 1188-1197.
9. Jie Wu, Fei Dai, Ming Gao, Ivan Stojmenovic, "On Calculating Power-Aware Connected Domination Sets for Efficient Routing in Ad Hoc Wireless Networks," Journal of Communications and Networks, Vol. 4, No.1, 2002, pp. 59-70.
10. Tomás Sánchez López, Alexandra Brintrup, Marc-André Isenberg, Jeanette Mansfeld, "Resource Management in the Internet of Things: Clustering, Synchronization and Software Agents," Architecting the Internet of Things, pp.159-193, 2011.
11. Laith Farhan, Rupak Kharel, Omprakash Kaiwartya, Mohammad Hammoudeh, Bamidele Adebisi, "Towards green computing for Internet of things: Energy oriented path massage scheduling approach," Sustainable Cities and Society, Vol. 38, 2018, pp. 195-204.
12. Alessio Botta, Walter de Donato, Valerio Persico, Antonio Pescapé, "Integration of Cloud computing and Internet of Things: A survey," Future Generation Computer Systems, Vol. 56, 2016, pp. 684-700.
13. Jianli Pan, James McElhannon, "Future Edge Cloud and Edge Computing for Internet of Things Applications," IEEE Internet of Things Journal, Vol. 5, Issue. 1, 2018, pp. 439-449.
14. Hongzhi Guo, Jiajia Liu, Jie Zhang, Wen Sun, Nei Kato, "Mobile-Edge Computation Offloading for Ultradense IoT Networks," IEEE Internet of Things Journal, Vol. 5, Issue. 6, 2018, pp. 4977-4988.

AUTHORS PROFILE

Hyun-Woo Kim received the B.S in Computer Engineering WonKwang University in 2012. He has been studied the M.S & Ph.D combined degree in Department of Multimedia Engineering, Dongguk University, Korea in 2016. He had been a research professor the Department of Multimedia Engineering, Dongguk University, Korea. He is now a professor at the Department of Information Security, Baewha Women's University, Korea. His interests are cloud computing, malware analysis, computer security threat, ubiquitous computing, virtualization, big data, Internet of Things.

Eun-Ha Song received her B.S. degree in Statistics and her M.S. and Ph.D. degrees in Computer Engineering from Wonkwang University in Iksan, Korea in 1997, 2000, and 2006, respectively. Dr. Song works in the Collage of Convergence & Liberal Arts of Wonkwang University, Korea. Her research interests include cloud computing, mobile computing, the Internet of Things (IoT), wireless sensor network applications, and Educational Programming Language(EPL).