ISSN 1816-6075 (Print), 1818-0523 (Online)
Journal of System and Management Sciences
Vol. 10 (2020) No. 3, pp. 45-57
DOI:10.33168/JSMS.2020.0304

A Sensor Data Management using Fog-Edge Gateway Model and Column-based Compression for Reliable IoT Networks

Siwoo Byun

Department of Software, Anyang University, South Korea
swbyun@anyang.ac.kr

Abstract. Internet of Things(IoT) sensor network consists of many sensor nodes that combine physical sensing capabilities with networking capabilities. The sensors send data to a central monitoring server where the data is aggregated. The objective of this research is to introduce the base architectures of IoT sensor network, data control schemes, fog computing and edge computing technology.

In this paper, a new approach of handling IoT data is proposed using combined IoT gateways and column-based compression for efficient and reliable data service in IoT environment. Proposed fog-edge gateway model exploits two different gateways to improve data service bandwidth and data reliability. Fog-edge gateway model can transmit compressed sensor data by reducing overhead in advance through column-based data compression scheme as compared to traditional row-based data management schemes. Experiments were conducted to measure the compression and transmission effects of the proposed scheme, and the results show that the data can be compressed to about 37% by column-based compression.

Keywords: Sensor Data, Column-based Compression, Resource Control, Edge Computing

1. Introduction

With the rapid development of the IoT (Bonnet, et al., 2001; Ghapar, et al., 2018; Kim and Lee, 2017), a lot of smart devices and sensors (Fig. 1) are increasingly connected at building. However, due to the diversity of sensors and IoT devices, it is difficult to build an integrated smart system work together. Moreover, some heavy tasks that require a large amount of computing and storage resources, such as real-time video analytics, are unable to execute due to limited hardware capabilities. IoT edge computing(Kim, 2015; Yu, et al., 2017) is used to solve these problems which integrates all components into a single platform, giving them sufficient and elastic resources, enabling smart applications to operate smoothly.

IoT edge computing is defined as a distributed computing infrastructure that includes many sensor devices that are well connected to each other. It is a new paradigm that provides high computing resources close to distributed IoT devices (Fig. 1). Therefore, IoT edge node can collect, classify, and analyse raw data streams locally instead of transferring them to the remote cloud, significantly reducing traffic overheads and speeding up the IoT big data process. However, where IoT edge nodes should be placed to facilitate communication between IoT devices and the edge nodes is still an important issue.

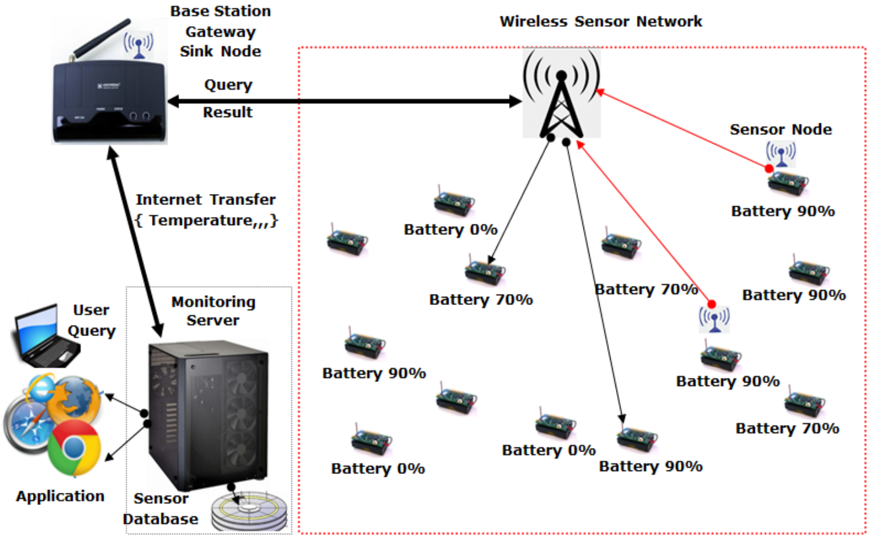

Fig. 1: Example of IoT Sensor Environment.

To use IoT application as an example (Fig. 2), a widely deployed security sensor consists of a smart lock, a video/audio recorder, and various sensor monitors (e.g. lighting sensors, occupancy classification sensors, motion sensors, etc.). These security devices are difficult to connect unless they are from the same vendor (Naidu 2019). However, edge computing can seamlessly services security applications. That is, edge computing can provide related service such as 1) unified interface for integrating all kinds of independent devices, 2) flexible resources to support storage and computation, and 3) real-time processing and low-latency response.

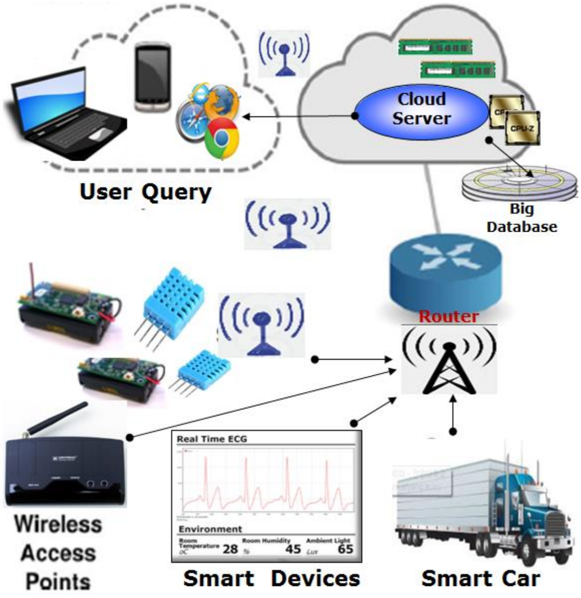

Fig. 2: Example IoT application.

Once the corresponding platform is installed, each sensor is connected to the client. These service applications can be installed on independent virtual machines, and they can also perform advanced processing logic in VMs, which can handle the data shared by these secure monitor applications. For example, if a motion sensor detects suspicious movements in a room, a cleaning robot equipped with a video camera is ordered to check out the exact location. Real-time video analytics will process these videos and check whether it is a false alarm. Then, notification and report will be sent to the homeowner, and the security system will report it to the police station.

The main purpose of IoT devices is to support IoT applications with low latency. Therefore, the IoT computing model must support efficient bandwidth and high reliability even for intensive requests.

2. Fog-based Computing Model

Since the cloud system is located within the Internet, there is no central control. Moreover, there are many problems with network topology and heterogeneous networks. A major problem that seriously affects service quality is the network delay (Fig. 3). Real-time applications are significantly affected by delay and jitter due to the network latency. However, it is very difficult to control delay and jitter from latency in large heterogeneous networks on an Internet scale. Therefore, the problems caused by latency are not going to be resolved soon.

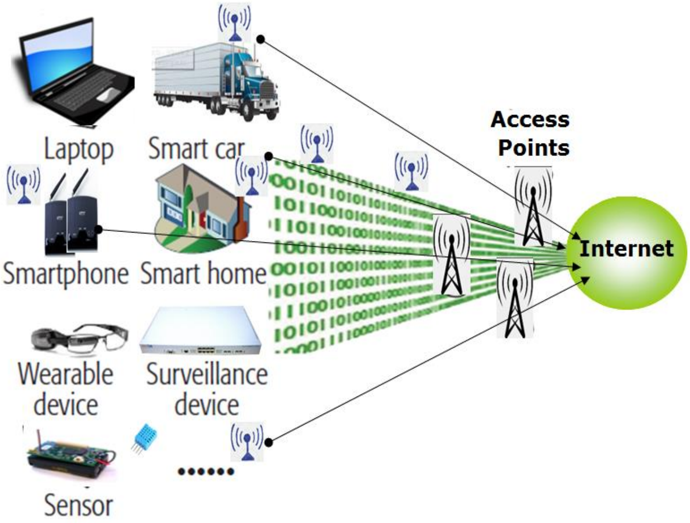

Fig. 3: Example IoT Network Delay.

Another problem with cloud computing is data security and privacy (Yi, et al., 2015). In cloud platforms, data transmission and system responses traverse many intermediate networks. Thus, if customer data is located in the cloud over the public Internet, there is a risk that customer integrity and confidentiality will be compromised. That is, cloud systems now face a variety of security threats due to their implementation within the Internet.

Edge computing pushes data and computing power from centralized host to network edges. Edge computing offers high levels of scalability, reliability and fault tolerance as well as sub-second response to end users. Advantages of edge computing include a significant reduction in data transfer across the network, reduced congestion, reduced latency, elimination of centralized bottlenecks, improved security of encrypted data, and reduced exposure to hostile elements.

Fog Computing is the model introduced by Cisco Systems to ease wireless data transmission in IoT network environments. The reason for using this term is that fog is a cloud closer to end-users and a data center. Therefore, fog computing is more advantageous in terms of delay, power consumption..

A key feature of fog computing is its ability to support applications that require low latency, location recognition, and mobility. Therefore, the fog computing nodes must have enough computing power and storage capacity to handle resource intensive user requests. Other similar concepts proposed to bring computing resources closer to users to overcome the limitations of cloud computing include cloudlet and edge computing.

3. Proposed IoT Computing Model

3.1 IoT Network Environment

In general, an IoT network consists of three components: a sensor device, an IoT gateway, and a cloud network, each meaning a data source, a data communication network, and data processor (Yi, et al., 2015; Premsankar, et al., 2018).

Fig. 4 shows the general IoT architecture based on the edge computing (Rahmani, et al., 2018; Byun, 2020). IoT Edge computing servers are closer to end users than cloud servers which is located far away. Although IoT edge computing servers provide weaker computing power than cloud servers, it has better quality of service (QoS) and lower network latency for end users. In general, the structure of IoT edge computing can be divided into three types: front-end, near-end and far-end (Premsankar, et al., 2018).

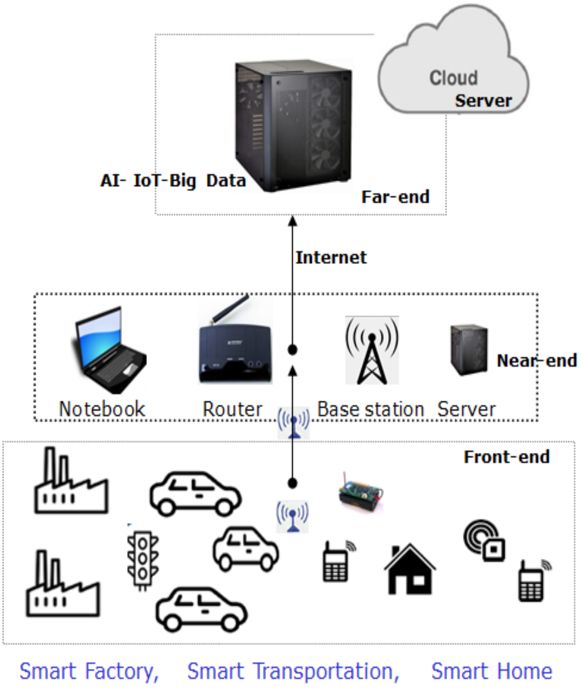

Fig. 4: Architecture of IoT Network Model.

In front-end environment, the end devices such as sensors and actuators are placed at the front-end in IoT edge computing. Front-end computing can provide end users with more interaction and better responsiveness. Because of the limited capacity of the end devices, most of the requirements cannot be satisfied at the front-end, so the end devices must forward the request to the server.

In near-end environment, IoT gateway supports most network services. Edge gateway servers have a various requirements such as real-time data processing, data caching, and computation offloading. By migrating most of the data processing and storing to this near-end environment, users can achieve better performance on data computing and storage, with a slight increase in the latency.

In far-end environment, the transfer delay is severe since the far-end server is deployed farther away from the end device. The far-end server, on the other hand, offers much more computing power and data storage. The far-end servers can provide powerful services such as massive parallel data processing, big data mining, big data management and machine learning.

3.2 IoT Data Management

Data compression can be used to reduce data communication delay and transmission energy. Both lossless and lossy compression methods are widely used in IoT environment. At the device layer of the IoT system, both lossy and lossless compression methods are useful.

If sensor nodes suffer from severe resource limitations such as battery and processing power, lossy data compression is more suitable. However, in general case of sensor data monitoring, lossless data compression is preferable to ensure that all features of the signals all observable. Therefore, in sensor data environments, column-based technology can be used to maximize lossless data compression (Byun, 2020).

Traditional row-based storage systems have to read many unrelated data from data repository, so they have performance limitations when handling a number of concurrent, diverse queries. However, in proposed storage model, only the columns related to IoT query need to be taken from the data repository, which is densely compressed to improve read efficiency like the column-store (Ahn and Kim, 2017).

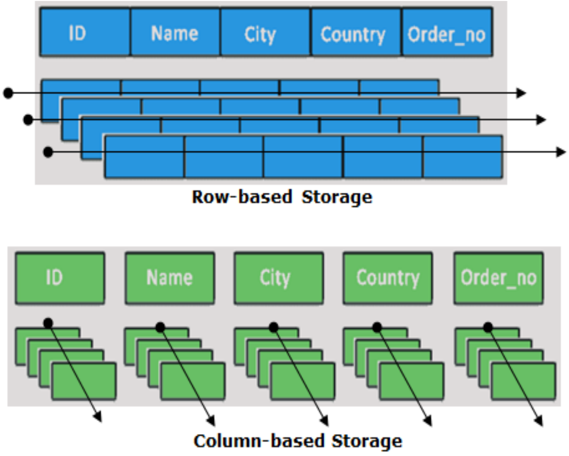

Fig. 5: Architecture of Column-based Storage.

Although current database can be applied to general network or mobile environment, it is difficult to apply to IoT sensor network environment that has incomplete characteristics. Proposed storage model reduces the potential for this undesirable data service by replicating IoT data across multiple gateways.

3.3 Efficient Data Control Model using IoT Gateway

Since IoT devices have very limited storage space, all data collected or generated should be sent to the storage server. If all devices simultaneously transfer data to cloud storage, this will be a significant burden to the network. Instead, sending data to several edge storage based on the characteristics of the edge topology reduces the long-distance traffic.

For this, load balancing and resource allocation technologies for edge-based storage are essential. It monitors different storage demand rates and uses data replication to reduce data traffic with storage balancing. Additionally, selecting the nearest edge storage node and weighted data control, can reduce the storage access time.

The simplest way to manage replicated data is *Read-One-Write-All* scheme. Since this old scheme naturally limits the data availability, the polling algorithms were proposed (Vukolic, 2012). The polling process requires write transactions to update a predefined number of copies (for example, [1/2+1]).

The idea of polling has been generalized to general quorum-consensus scheme which is recognized as the main abstraction skill to ensure data consistency in reliable distributed databases. Recently, quorum algorithms have evolved to fault-tolerant event logger and block chain system (Edson, 2017; Cachin, 2017).

IoT edge-gateway shares fast local network with front-end sensor devices. Thus, IoT edge gateways provide much faster transmissions as compared to cloud servers connected to the slow Internet. However, the IoT edge gateway has limited memory and storage space as compared to the cloud servers. Therefore, it is important to design efficient data management by taking advantage of superior network conditions taking into account limited resource conditions.

Storage devices lead to heavy system workloads because they constantly have to store large amounts of data regardless of its importance from many sensor devices. Therefore, the burden of storing this intensive data can be greatly reduced by offloading unnecessary data in advance. In other words, if non-critical data is not prevented from being put into the next operation, the overloaded storage operations degrade edge-gateway performance severely.

However, although the gateway may determine the importance of the data right away, the strategic judgment of the cloud should be included. Therefore, proposed scheme prefers not to use immediate removing strategy, but to use delayed removing strategy that waits the confirmation of the cloud. For example, all the sensor data is kept in a gateway's bulk storage such as hard disk during one day at very low cost, and the data that is not needed later is linked to garbage places such as trash cans. That is, the invalidated data is transferred to the waste repository, erased at idle time, and recycled later.

Although most sensor data are initially processed with the maximum I/O performance, if the storage device or memory becomes overloaded, the following data is not processed at that speed. Therefore, it is needed to prevent this overloading and to keep data availability and reliability. The approach of the study aims to increase data distribution speed across the entire network and to increase data availability and reliability by adding an IoT gateway called fog-gateway in the entry point of the cloud servers.

Edge gateway and fog gateway have similar hardware design and specification but have different placement locations. While edge gateways are connected to local network that has broad bandwidth, their storage efficiency is low since the acquired data are stored without compression. Fog gateways on the other hand utilize data compression for high storage efficiency while they are connected to slow Internet.

If the edge and fog gateways are efficiently combined, they provide faster network transmission and higher data efficiency for overall IoT network. That is, it is effective to increase the communication efficiency of edge gateways to increase the storage efficiency of fog gateways, to offload sensor data that is not needed on edges, and to send it to the cloud.

In this respect, this study proposes a new management framework called Data-based Resource Management to improve the availability and reliability of IoT data (Fig. 6).

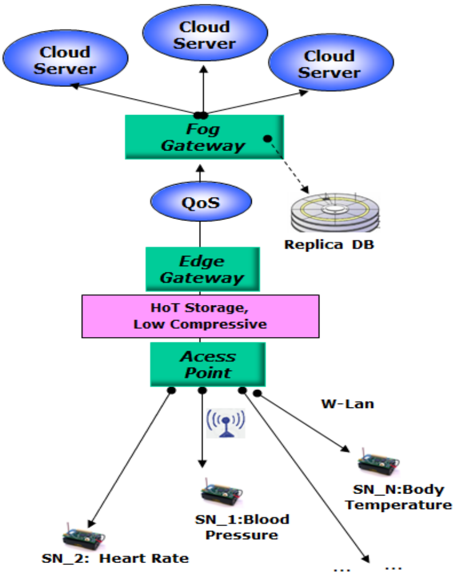

Fig. 6: Architecture of Proposed IoT Management Model

First, edge gateways perform data-based offloading to reduce system overheads such as I/O delay, severe transmission delay, memory shortage, and CPU overload. Second, to enhance the data availability and reliability, the edge gateways are interconnected to the fog gateways below the cloud layer.

Proposed scheme analyses the compression ratio of sensor data and recognizes the compression characteristics. Then, the sensor data are classified to two classes - compressible and incompressible - according to the compression efficiency. Proposed scheme also controls the transfer rate according to the queue length of a task in edge gateway. Proposed scheme utilizes column-based compression skill and uses lzo data compression library (Lzo, 2020), which is not only easy to handle source code but also controls compression ratio and speed by changing compression levels.

For critical data that need to be replicated, the same data is maintained on both edge and fog gateways. Among the replication management techniques, the delayed methods have the disadvantage of less data consistency due to their asynchronous nature. On the other hand, eager techniques ensure strong consistency.

In addition, for reliable and guaranteed data transfer, the network bandwidth between the two gateways should be guaranteed. In this study, it maintains each priorities and minimum bandwidth, ensuring that urgent class of data are reliably sent even in the event of communication congestion as like general QoS model (Fig. 7).

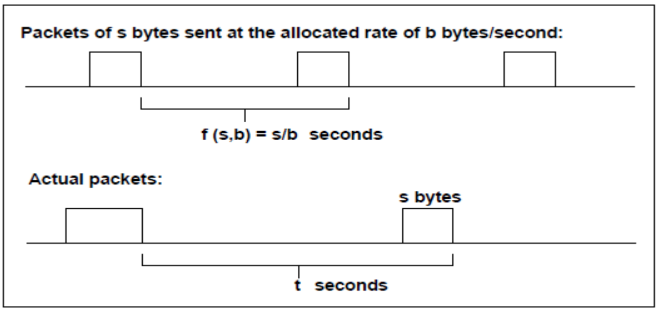
Packets of s bytes sent at the allocated rate of b bytes/second:
f (s,b) = s/b seconds
Actual packets:
s bytes
t seconds

Fig. 7: IoT Resource Management for Guaranteed Bandwidth

In proposed model, Gateway Resource Controller responsible for controlling network bandwidth determines the allowable limit of packet transmission for each service level. In Fig. 7, the upper picture illustrates the time difference between the front packet and next packet when they have been sent continuously at the assigned speed. The lower picture illustrates the time difference measured between the two packets. Based on these two pictures, the speed and frequency of packet transmission can be adjusted.

The transmission speed estimates are calculated using an *exponential weighted moving average* (Kim 2018). In Fig. 7, s is the recently transmitted packet size in bytes and b is the transfer bandwidth allocated to this class in bytes/second. And t is the difference in actually measured time between the front and next packets.

For example, if IoT edge gateway sends a packet of size s using bandwidth *b*, it can be predicted that it will take the time as follows.

$$\mathrm{f(s,b)} = s / b$$

Then, the time difference between the estimate and the actual measurement is as follows.

$$TimeDiff = t - \mathrm{f(s,b)}$$

If a IoT device exceeds the allowable bandwidth, *TimeDiff* is negative. Otherwise *TimeDiff* is positive. If *TimeDiff* is negative, it is necessary to slow down the packet transmission of the device so that it does not exceed the allowable bandwidth.

Experiments were conducted to measure the compression and transmission effects of the proposed technique. Table 1. shows the efficiency of proposed model based on room temperature data and the data can be compressed to about 37% by numerical repetition and token mapping skills. The column-wise compression modules are constructed by multiple chain connections with small compression segments rather than full block compression, in order to reduce both compression and decompression overhead as much as possible.

Table 1: Results of Fog-Edge Data Service (Room Temperature)

Lzo Mode Selection	**Compressed to (%)**	**Transmission per Module (MB/s)**
Speed_Max	51.4	45.753
Speed_Min	42.7	12.967
Size-Min	37.0	02.310

It is also possible to reduce the amount of the transmitted data by using variable cycle transfer. For example, if a sensor node sends data in the normal range every 3 seconds and sends data in the abnormal range every 1 second, it can reduce the amount of sending data by one third. In fact, most of the data is normal, so this may reduce it further. That is, suppressing repetitive similar data can enhance performance much more.

4. Conclusions and Future Work

In this paper, a new resource management scheme is devised to provide reliable data service for IoT information processing. In order to allow the IoT users to use the reliable data services, proposed model used compressed data-based gateway which exploits both fog and edge gateway model.

Proposed scheme can efficiently transmit IoT sensor data by reducing data volume overhead in advance through fog-edge gateway model and column-based data compression method. Experiments were conducted to measure the compression and transmission effects of the proposed scheme, and the results showed that the overhead of sensor data can be reduced to 37% by column-based compression and transmission.

Further study includes sensor data security and privacy data issues and shows algorithms for implementing a secure fog-edge gateway model for efficient and safe data services.

References

Amir M. Rahmani, Gia T.N., Negash B., Anzanpour A., Azimi I., Jiang M., Liljeberg P. (2018), “Exploiting smart e-Health gateways at the edge of healthcare Internet-of-Things: A fog computing approach,” *Future Generation Computer Systems*, 78(2), 641-658. DOI:10.1016/j.future.2017.02.014

Ghapar A.A., Yussof S. and Bakar A.A. (2018), "Internet of Things (IoT) Architecture for Flood Data Management", *International Journal of Future Generation Communication and Networking*, 11(1), 2018, 55-62. DOI:10.14257/ijfgcn.2018.11.1.06.

Cachin C. and Vukolic M. (2017), “Blockchain consensus protocols in the wild”, Technical Report arXiv: 1707.01873, IBM Research - Zurich.

Camargo E.T., Elias P. Duarte Jr. and Pedonez F. (2017), “A Consensus-based Fault-Tolerant Event Logger for High Performance Applications,” Euro-Par 2017: Parallel Processing, 415-427.

Premsankar G., Francesco M.D., and Taleb T. (2018), “Edge Computing for the Internet of Things: A Case Study,” *IEEE Internet of Things Journal*, 5(2), 1275-1284, DOI: 10.1109/JIOT.2018.2805263

Appala Naidu H., and Hymavathi J. (2019). “A Survey on Solutions and Applications of Cloud Service and Storage Security.” *International Journal of Urban Design for Ubiquitous Computing*, 7(1), 17-26.

Kim H.Y., Kim S.C., Park H.J. (2018). “Priority and Delay Aware packet transmission MAC Protocol for Wireless Sensor Networks.” *International Journal of Security Technology for Smart Device*, 5(2), 9-14.

Kim J.H. and Lee Y.S. (2017), "A Study on Context Synchronization Method for Offloading on IoTCloud Fusion Virtual Machine", *International Journal of Grid and Distributed Computing*, 10(12), 149-160, DOI:10.14257/ijgdc.2017.10.12.14.

Kim J.T. (2015), "Requirement of Security for IoT Application based on Gateway System", *International Journal of Security and Its Applications*, 9(10), 201-208, DOI:10.14257/ijsia.2015.9.10.18.

Lzo, http://www.oberhumer.com/products/lzo-professional/, Feb. 12, 2020.

Bonnet P., Gehrke J., and Seshadri P. (2001), “Towards Sensor Database Systems,” *The 2nd International Conference on Mobile Data Management*, 1-12, DOI: 10.1007/3-540-44498-X.

Byun S. (2020), “Dynamic Sensor Data Management Scheme for IoT Servers,” *Journal of Critical Reviews*, 7(14), 396-401

Ahn S., and Kim K. (2013). “A Join Technique to Improve the Performance of Star Schema Queries in Column-Oriented Databases,” *Journal of Korean Institute of Information Scientist and Engineers*, 40(3), 209-218.

Byun S. (2020), “Resource Management for Reliable IoT Environments”, *International Journal of Reliable Information and Assurance*, GVPress 8(1), 7-14, DOI:10.21742/ijria.2020.8.1.02

Yi S., Hao Z., Qin Z., and Li Q. (2015), “Fog Computing: Platform and Applications,” *The 3rd IEEE Workshop on Hot Topics in Web Systems and Technologies*, 12-13, DOI: 10.1109/HotWeb.2015.22.

Vukolic (2012), “Quorum Systems: With Applications to Storage and Consensus,” *Synthesis Lectures on Distributed Computing Theory*. Morgan & Claypool 2012.

Yu W., Liang F., He X., Hatcher W.G., Lu C., Lin J., Yang X.(2017), “A Survey on the Edge Computing for the Internet of Things”, *IEEE Access*, 6, 6900-6919, DOI: 10.1109/ACCESS.2017.2778504