Computer Systems
Science and Engineering

Tech Science Press

DOI: 10.32604/csse.2023.040473
Article

Edge Cloud Selection in Mobile Edge Computing (MEC)-Aided Applications for Industrial Internet of Things (IIoT) Services

Dae-Young Kim[1], SoYeon Lee[2], MinSeung Kim[2] and Seokhoon Kim[1],*

[1]Department of Computer Software Engineering, Soonchunhyang University, Asan, 31538, Korea
[2]Department of Software Convergence, Soonchunhyang University, Asan, 31538, Korea
*Corresponding Author: Seokhoon Kim. Email: seokhoon@sch.ac.kr
Received: 20 March 2023; Accepted: 15 May 2023; Published: 28 July 2023

Abstract: In many IIoT architectures, various devices connect to the edge cloud via gateway systems. For data processing, numerous data are delivered to the edge cloud. Delivering data to an appropriate edge cloud is critical to improve IIoT service efficiency. There are two types of costs for this kind of IoT network: a communication cost and a computing cost. For service efficiency, the communication cost of data transmission should be minimized, and the computing cost in the edge cloud should be also minimized. Therefore, in this paper, the communication cost for data transmission is defined as the delay factor, and the computing cost in the edge cloud is defined as the waiting time of the computing intensity. The proposed method selects an edge cloud that minimizes the total cost of the communication and computing costs. That is, a device chooses a routing path to the selected edge cloud based on the costs. The proposed method controls the data flows in a mesh-structured network and appropriately distributes the data processing load. The performance of the proposed method is validated through extensive computer simulation. When the transition probability from good to bad is 0.3 and the transition probability from bad to good is 0.7 in wireless and edge cloud states, the proposed method reduced both the average delay and the service pause counts to about 25% of the existing method.

Keywords: Industrial Internet of Things (IIoT) network; IIoT service; mobile edge computing (MEC); edge cloud selection; MEC-aided application

1 Introduction

Internet of Things (IoT) services exploit measured sensor data from devices based on data transmission and analysis from numerous tiny devices. The data are sent to the Internet cloud and can be used to extract knowledge about services through data processing in the Internet cloud. The edge cloud has recently made it possible to process data for services within the IoT domain [1–4]. Edge clouds are implemented by mobile edge computing (MEC) servers that integrate with gateways. The edge cloud (i.e., MEC server) can reduce the amount of data processing in the Internet cloud and provide computing resources to IoT devices. It is not located at the network core, but at the gateway of radio access, enabling immediate support of services within the IoT domain [5–8]. Thus, the edge cloud receives data from the IoT domain and finds data insights using data analysis and artificial intelligence (AI). The data insight can be applied to IoT services. The data insight can be applied in industrial IoT (IIoT) services [9,10] and improve factory productivity. The network architecture of IIoT is as follows: Devices in the IIoT domain transmit sensor data to the edge cloud via the gateway, and the edge cloud processes the received data. IIoT domains have several gateways and local edge clouds. The local edge cloud is integrated into the gateway, and the device must select the appropriate local edge cloud before sending data. Fig. 1 illustrates the network architecture for the IIoT services.

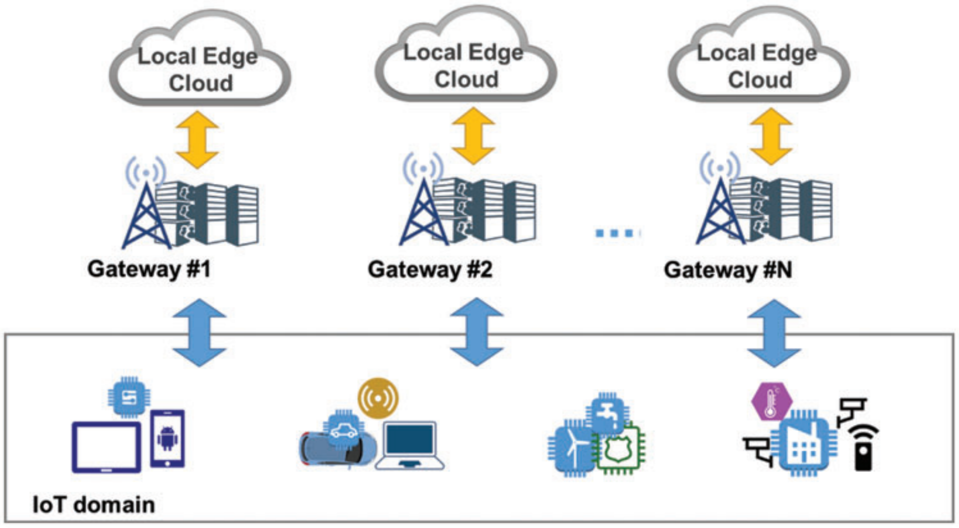


Figure 1: Network architecture for IIoT services

The edge cloud is vital to numerous services, such as data processing and caching points. Because IIoT sensors have insufficient computing resources, they only deliver their sensor data to the selected local edge cloud. The local edge cloud performs data processing and computing. The data insight obtained by the computing results can apply to IIoT services. Each sensor device transmits data by determining a local edge cloud and setting a routing path. Although this IoT network is an evolved environment of the traditional wireless sensor network, it has a different architecture from the wireless sensor network. In a typical wireless sensor network, the network architecture for a service is simple. Sensor data are delivered to the given gateway system for a sensor field, which relays the data to a central cloud on the Internet [11,12]. Therefore, communication costs are incurred to transfer data from the sensor to the Internet cloud through the routing path in the wireless sensor network.

However, for IoT services, especially IIoT services, there are more factors to consider than sensor data transfer. As described previously, the data insight for IIoT services is extracted in the local edge cloud. The computing state of the local edge cloud can also affect service efficiency. When data traffic arrives at the local edge cloud, excessive computational load on the edge cloud delays data processing. In this situation, even though data is transmitted quickly in the wireless network, service delays inevitably occur. Thus, if the IIoT system considers only the communication cost for data transfer, it is challenging to achieve adequate service efficiency. The computing cost should be considered along with the communication cost when an IIoT device chooses its local edge cloud. Service efficiency can be achieved when both transmission efficiency in wireless and computing efficiency in the edge cloud are achieved. Therefore, this paper proposes a local edge cloud selection method that considers both communication and computing costs in the IIoT domain where multiple edge clouds exist. The proposed method measures transmission delays in wireless and service waiting time in edge clouds. Based on the measurement results, a local edge cloud is selected. The proposed method can improve service efficiency by simultaneously achieving transmission efficiency and computing efficiency.

The remainder of this paper is organized as follows: The related work for the local edge cloud selection is described in Section 2, and Section 3 explains the proposed method considering both the communication and computing costs. The performance evaluation for the proposed method is provided in Section 4. Finally, Section 5 concludes this paper.

2 Related Work

MEC-aided applications using the edge cloud are growing in IoT services. The edge cloud can compensate for the device's insufficient computing power. It can perform data processing on behalf of tiny devices in a radio access network. In IIoT services that require frequent feedback from edge clouds, it is important to select an appropriate edge cloud for service efficiency. Because the edge cloud is integrated with a gateway, the edge cloud selection can also be considered as selecting the gateway.

Local edge cloud can also be selected based on service discovery considering network service functions. Service discovery is important in a network composed of various edge clouds with different functions, where functions are determined by service attributes [13]. However, when data transmission occurs in a wireless network, path selection metrics play a more important role than service discovery by attribute functions. The data transmission path is determined through the gateway selection. There are several metrics in selecting a gateway in IoT devices: distance, received signal strength, link quality, throughput, latency, energy consumption, and load balancing. Many gateway selection algorithms use these metrics [14].

Lai et al. [15] considers interference in the wireless channel to select a gateway. They attempt to choose a gateway that minimizes the interference power and maximizes the desired signal power. They use link quality as a metric. The taxi-sharing method [16] performs data grouping and compression in the gateway to reduce the amount of traffic between the gateway and the cloud. In this method, IoT devices check the queues of gateways and send data to the gateway with a proper queue to join. Then, the gateway compresses the data in the queue. If there is no proper gateway, the device uses network latency to select its gateway. The game theory method [17] uses the result of a utility function to select a gateway. This utility function exploits wireless link information such as received signal strength, signal-to-noise ratio, and channel capacity. In the algorithm, a game to maximize the utility function is performed. Based on the results, devices select their gateway. The distance-based method [18] is usually used for a gateway location selection in the network. In the algorithm, a graph for the network devices is constructed. The gateway location is determined by using the shortest path to all devices in the graph. Thus, the device forwards the data to the nearest gateway. Accordingly, existing studies have focused on gateway selection within a wireless network. However, as described previously, the mobile edge cloud is widely used in data learning and analysis in current IoT services. The computing state of the mobile edge cloud can directly affect IoT services. The problem of selecting a gateway in IoT is a problem of selecting a local edge cloud as a mobile edge cloud. When a device selects its gateway (i.e., local edge cloud), it should consider both the network state and the local edge cloud state to improve service efficiency. Thus, in this paper, the proposed method approaches local edge cloud selection considering both the communication cost as the network state and the computing cost as the local edge cloud state.

3 Proposed Approach for Local Edge Cloud Selection

IoT services with MEC-aided applications primarily generate uplink data traffic. This data traffic goes through processing in the local edge cloud. Factors affecting uplink data transmission are the state of the radio link and the state of the local edge cloud. Therefore, an uplink path must be determined by considering both factors. The proposed method takes an approach that enables path selection by selecting a local edge cloud in consideration of the factors affecting uplink path selection.

3.1 System Model

We consider an IIoT service network enabled by multiple local edge clouds, as depicted in Fig. 1. In the network, the measured data from IIoT devices are delivered to the local edge cloud via a gateway for data processing. There are two service costs during data transmission and processing: communication and computing. In the IIoT system, the communication and computing costs can be defined as a transmission delay and a waiting time for data processing. These costs represent the system load. High cost indicates that a system has heavy loads (i.e., long service time) for a service. The transmission delay for the communication cost is calculated by applying a data rate to the given data size. If the measured data are delivered through multiple hops, the transmission delay is increased based on the hop counts. Eq. (1) calculates the communication cost (t_{cm}).

$$t_{cm} = \sum_{i \in G} \frac{X}{r_i}, \tag{1}$$

where i is a routing link in a set of the routing paths (G) between a device and a gateway. r_i is a data rate of the routing link i. X is the measured data in the device.

The transmitted data packets are enqueued and processed sequentially in the local edge cloud. Thus, the computing cost can be defined as the waiting time in the local edge cloud. When the local edge cloud is assumed in the *MM1* queueing system, computing intensity (ρ) in the local edge cloud is represented as

$$\rho = \frac{\text{mean service time (S)}}{\text{mean arrival time (A)}} = \frac{1/\mu}{1/\lambda} = \frac{\lambda}{\mu}, \tag{2}$$

where λ and μ are the mean arrival and service rates of a data packet in the local edge cloud. Because the waiting time refers to the time until the service is performed in the local edge cloud. It can be represented through the service time in the response time of the local edge cloud. Eq. (3) calculates the response time.

$$R = \frac{1/\mu}{1-\rho} = \frac{\text{average service time}}{\text{probability that the local edge cloud is idle}}, \tag{3}$$

Then, the waiting time (W) as the computing cost (t_{cp}) is represented as Eq. (4).

$$t_{cp} = W = R - S = \frac{1}{\mu\,(1-\rho)} - \frac{1}{\mu} = \frac{1}{\mu}\left(\frac{\rho}{1-\rho}\right), \tag{4}$$

3.2 Local Edge Cloud Selection

Applying data learning and analysis are importantly handled in current IIoT systems. This type of data processing is performed in the local edge cloud integrated with a gateway. For efficient data processing of the measured data in the IIoT domain, an appropriate local edge cloud should be selected. Devices choose a proper local edge cloud because the state of the local edge clouds affects data learning and analysis. This can decrease service efficiency. The local edge cloud selection should also consider the communication cost for wireless data transmission. Thus, we define a selection metric for a local edge cloud using t_{cm} and t_{cp} in the previous section. Eq. (5) represents the selection metric (C_j).

$$C_j = \alpha \cdot t^j_{cm} + (1 - \alpha) \cdot t^j_{cp}, \tag{5}$$

where j is a local edge cloud and α is a system parameter. An IoT device can receive the status message from several local edge clouds. The local edge clouds issue the status messages periodically. Based on these messages, the device computes the selection metric in Eq. (5) for each local edge cloud. Then, the device finds the lowest value of the selection metric and selects its local edge cloud. Algorithm 1 represents the pseudo-code for the local edge cloud selection and Table 1 lists the notations for Algorithm 1.

Table 1: Algorithm 1 notation

Notation	Description
j	A gateway j in the network
E	A set of gateways for local edge clouds
v	A node v in the network
N	A set of nodes in the network
t_{cm}	Communication cost
t_{cp}	Computing cost in the local edge cloud
C_j	Local edge cloud selection metric
g_v	A selected gateway

Algorithm 1: The proposed local edge cloud selection algorithm

```
1: loop for $j \in E$:
2:   $j$ computes $t_{cp}$
3:   $j$ constructs a message for $t_{cp}$ with the expiry time
4:   $j$ broadcasts the message of $t_{cp}$ to a network
5: end loop
6: loop for $v \in N$:
7:       loop for $j \in E$:
8:          $v$ computes $t_{cm}$ for $j$
9:          $v$ computes $C_j$
10:      end loop
11: end loop
12: $g_v \leftarrow \mathrm{argmin}_j\ C_j$
```

Each node in a network computes the communication and computing cost to select its local edge cloud. For computing cost, gateways for local edge clouds calculate their t_{cp} through data processing status, construct container messages for t_{cp} with expiry time, and then broadcast the messages to the network (*lines* 1–5). By receiving the broadcasted message, a node can obtain computing costs of local edge clouds in the network. If the broadcast message is expired, t_{cp} is set to infinity (∞). t_{cp} is updated by the periodic broadcast message from the gateways. Furthermore, each node measures the data rate of a routing path using the broadcasted message. It piggybacks the information to the broadcast message and rebroadcasts the message. Then, a node computes t_{cm} in a routing path for a gateway.

Given t_{cp} and t_{cm}, a node calculates the selection metric in Eq. (5) (*lines* 6–11). Among the selection metrics for local edge clouds, a node determines the lowest value as a path for its local edge cloud (*line* 12). As described previously, considering the computing cost of a local edge cloud is essential for IIoT services. Intelligent services of IIoT are based on the collected data from numerous sensor devices and data processing in the local edge cloud. Therefore, if a node transmits data to the local edge cloud with very high data processing loads, IoT service efficiency may be decreased by data processing delay. Providing an efficient IoT service is challenging when considering only wireless data transmission in the network.

The proposed method can also support avoiding a local edge cloud with faults. The edge clouds (i.e., the integrated gateways) in the proposed method periodically broadcast their computing status information using t_{cp} to inform the network. This information is updated by receiving a new broadcast message. If a local edge cloud has a fault such as an operating problem, the edge cloud's gateway cannot broadcast t_{cp} information. Then, the t_{cp} value in a device (i.e., an IIoT device) is set to infinity and the edge cloud is not selected by the device. Thus, the local edge cloud with faults can be avoided in the network. Service quality using data analysis results can be maintained.

4 Performance Evaluation

The proposed scheme was intended for efficient IIoT services using data learning among multiple local edge clouds. In this section, the performance of our scheme is evaluated through extensive computer simulations by comparing it with the proposed scheme and the typical wireless network approach. The simulator is implemented by the event-driven library, SMPL [19], based on the C-language. The SMPL provides an event scheduler and C-functions for probability distributions. Using the SMPL library, scheduled events occur at the times specified in the event scheduling. The event times can be obtained from a probability distribution. For performance evaluation, we perform the event-driven simulation by applying simulation parameters.

4.1 Simulation Environments

For computer simulations, a device is located a 2-hop distance from local edge cloud #1 and a 3-hop distance from local edge cloud #2. A device can select one of them as its local edge cloud. As the device generates data, it sends the data to the local edge cloud for data analysis. The local edge cloud processes the data and then feeds back the process result to the device. LTE Cat. M1 is considered for wireless communications, designed for IoT communication with a 1 Mbps data transmission rate for both the uplink and downlink [20]. Fig. 2 illustrates the network environment for simulations. In the simulation, the evaluation performs through the service delay including the data transmission and processing.

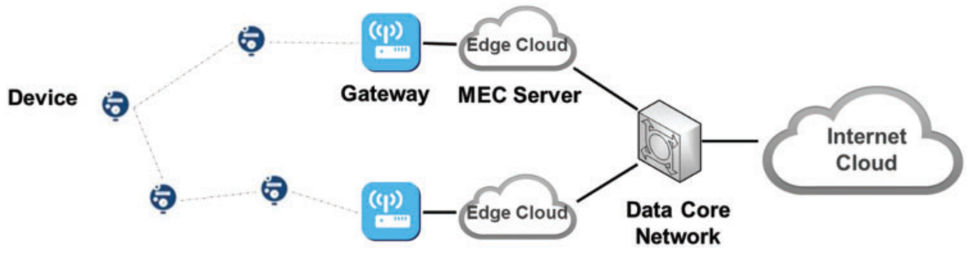


Figure 2: Network environment for simulations

The service delay of a network (T) consists of t_{send} (time to send data in a wireless network) + $t_{process}$ (time to process data in the MEC server including delivery time between the gateway and the MEC server) + $t_{receive}$ (time to receive data in a wireless network).

$$T = t_{send} + t_{process} + t_{receive}, \tag{6}$$

In Eq. (6), t_{send} is the uplink time for multi-hop transmissions from the end device and $t_{receive}$ is the downlink time for multi-hop transmissions from the gateway of the local edge cloud. In LTE Cat. M1, the uplink data rate is identical to the downlink data rate. However, t_{send} and $t_{receive}$ can differ depending on communication conditions (e.g., interference) or the difference in data rate between uplink and downlink. $t_{process}$ is data processing time in the local edge cloud. In the simulation, if T is greater than 1 s, it is assumed that a service pause has occurred.

For the wireless network, the network conditions change according to the two-state Markov chain model [21]. It has good and bad states. p is the transition probability from a good state to a bad state, and q is the transition probability from a bad state to a good state. By the p and q, the wireless network conditions for the uplink and downlink are determined. Fig. 3 represents the two-state Markov chain. A two-state Markov chain is also used to determine the MEC server state in local edge clouds. In the MEC server, a bad state indicates a busy state of the local edge cloud, and a good state indicates the ideal state of the edge cloud. The busy state is where the number of transactions to be processed in the local edge cloud increases rapidly. In the simulation, the state is determined and maintained for 60 s. According to the Markov chain model, the wireless and MEC server states can be changed every 60 s.

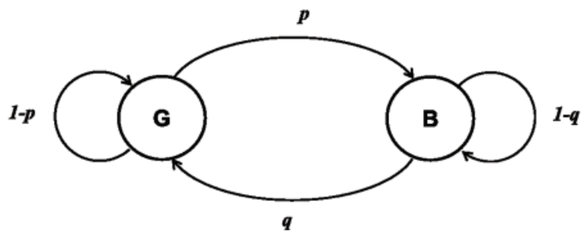


Figure 3: Two-state Markov state for determining network and edge cloud state

Data traffic in a device is generated in exponential distribution with a mean of 5 min. As described previously, the data rate is set to a maximum of 1 Mbps for the uplink and downlink. Under good wireless conditions, the data rate is randomly selected between 512 Kbps and 1 Mbps. Under bad wireless conditions, the data rate is randomly selected between 256 and 512 Kbps. The packet size is assumed to be 256 bytes. In the local edge cloud, it is assumed that the MEC server processes a maximum of 2,000 transactions per second. Under a bad state, incoming transactions to the MEC server are generated in an exponential distribution with a mean of 1,800 transactions per second. Under a good state, they are generated in an exponential distribution with a mean of 1,000 transactions per second. Furthermore, it is assumed that the edge cloud spends 5 ms for data processing. For route information and local edge state information, each gateway issues a broadcast message every 60 s. The simulation time is set to 22 h. Table 2 lists the simulation parameters.

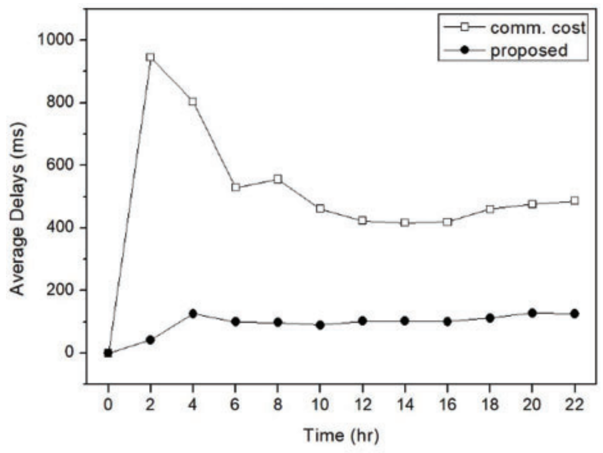
Table 2: Simulation parameters

Parameter	Value
Simulation time	22 h
Traffic generation	5 min (exponential distribution)
Packet size	256 bytes
Broadcast time	60 s
p	0.3, 0.7
q	0.7, 0.3
Wireless state interval	60 s
Data rate	512 Kbps ~ 1 Mbps (good state)
	256 Kbps ~ 512 Kbps (bad state)
Edge state interval	60 s
MEC server outgoing transaction	2,000 per second
MEC server incoming transaction	1,800 per second (bad state)
	1,000 per second (good state)
Edge processing time	5 ms
α	0.5

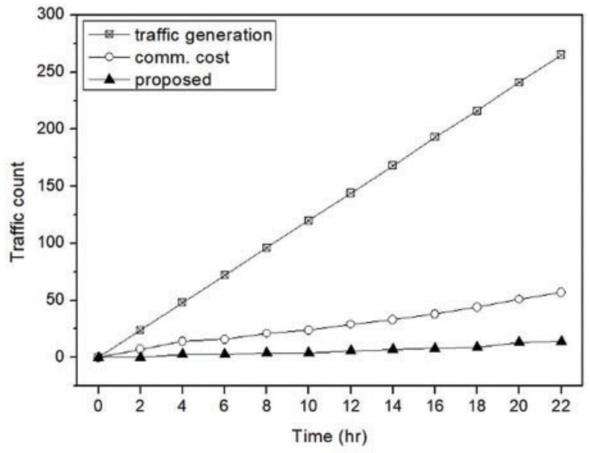
4.2 Simulation Results

Fig. 4 illustrates the average service delay and service pause count when p and q for wireless and local edge state transitions are 0.3 and 0.7. Based on the simulation, the average service delay is transient until 6 h and then becomes steady-state. The average service delays in the proposed scheme are 125.4 and 484.1 ms in communication-cost-only mode (i.e., existing scheme). For simulation time, the device generated data 265 times and requested service processing, which resulted in 14 service pauses. In contrast, 57 service pauses occurred in communication-cost-only mode. As described earlier, many IIoT services require local edge cloud processing for generated data to provide intelligent services (such as applying AI or data analysis). Thus, it is necessary to consider not only communication costs but also information about the state of the edge cloud for local edge cloud selection. In the proposed scheme described in Sections 3 and 4, a device selects its local edge cloud by reflecting the state of the edge cloud. Consequently, the proposed scheme can reduce the average service delay and the service pause count.

Fig. 5 illustrates the average service delay and service pause count when p and q are 0.3 and 0.7 for wireless state transition and 0.7 and 0.3 for local edge state transition. The average service delay is 202.9 ms in the proposed scheme and 627.5 ms in communication-cost-only mode. In the proposed scheme, 24 service pauses occurred in 265 service requests by data generation in a device. However, 80 service pauses occurred in the communication-cost-only mode. Even if the state of the local edge cloud changes frequently, the average service delay increases slightly. However, the service delay can be maintained lower than the communication-cost-only mode.

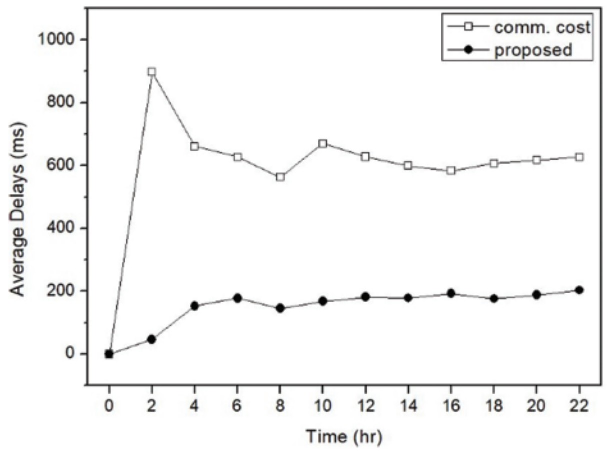


(a) Average delay

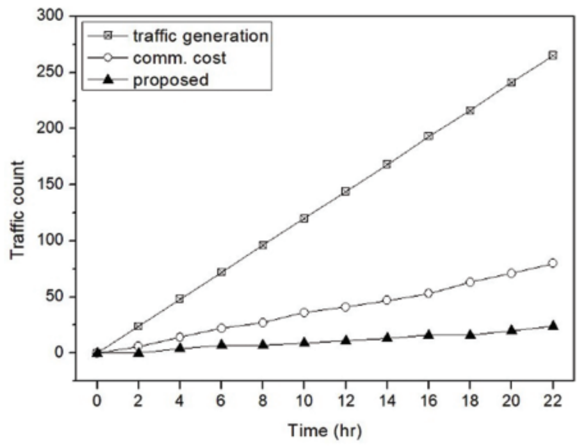


(b) Pause count

Figure 4: Simulation result: $p = 0.3$, $q = 0.7$ in wireless and local edge cloud



(a) Average delay

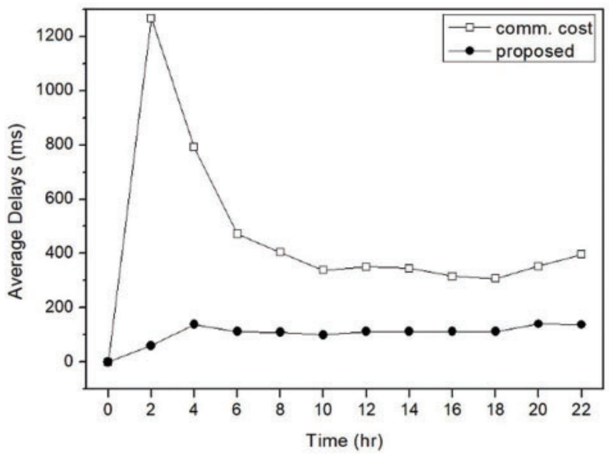


(b) Pause count

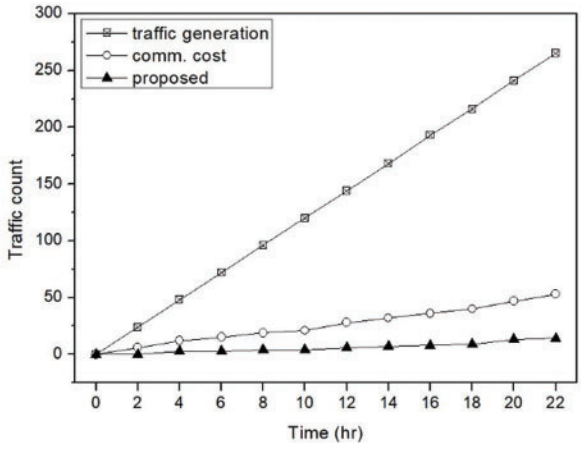
Figure 5: Simulation result: $p = 0.3$, $q = 0.7$ in wireless and $p = 0.7$, $q = 0.3$ in local edge cloud

Fig. 6 illustrates the average service delay and service pause count when p and q are 0.7 and 0.3 for wireless state transition and 0.3 and 0.7 for local edge state transition. The average service delay is 138.1 ms in the proposed scheme and 396.5 ms in the communication-cost-only mode. The service pause count is 14 in the proposed scheme and 53 in the communication-cost-only mode. Even if the wireless state changes frequently, the proposed scheme outperforms the existing scheme. Based on the simulation results, the performance tends to depend heavily on the state of the edge cloud. With the expansion of smart services in IoT, the use of local edge clouds expanding. Therefore, route determination considering the state of the edge cloud has become more critical. The proposed scheme provides a reliable approach for edge-cloud-aided IoT services like IIoT.

Table 3 represents the simulation results according to the state transition probability p and q. The state where p is 0.7 and q is 0.3 means either the edge cloud is busy, or the wireless is bad. The proposed method shows better performance in all cases.



(a) Average delay



(b) Pause count

Figure 6: Simulation result: $p = 0.7$, $q = 0.3$ in wireless and $p = 0.3$, $q = 0.7$ in local edge cloud

Table 3: Simulation results

Transition probability		Average delay		Pause count	
Wireless	Edge cloud	Proposed	Comm. cost	Proposed	Comm. cost
$p = 0.3/q = 0.7$	$p = 0.3/q = 0.7$	125.4 ms	484.1 ms	14	57
$p = 0.3/q = 0.7$	$p = 0.7/q = 0.3$	202.9 ms	627.5 ms	24	80
$p = 0.7/q = 0.3$	$p = 0.3/q = 0.7$	138.1 ms	396.5 ms	14	53

5 Conclusion

In this study, we proposed a local edge cloud selection approach to provide a path with improved service efficiency to IoT devices. The proposed approach selects the local edge cloud with the lowest service cost based on communication and data processing costs. The communication cost reflects transmission efficiency in wireless and the data processing cost reflects computing efficiency in edge clouds. Thus, the proposed method improves service efficiency. Through extensive computer simulation, the proposed approach reduced the average service delay and pause count by 26% and 25% of the conventional approach (i.e., using only communication cost), respectively. Even if the state of edge clouds or wireless condition frequently changes, the proposed scheme improved performance. Based on extensive computer simulations, we found that an IoT service that uses the edge cloud should provide a solution that considers the state of the edge cloud. Thus, the proposed scheme establishes guidelines to design network solutions tightly integrated with the edge cloud.

Terminal devices using local edge clouds in various IoT services can move to other service areas. Handoff can occur when the terminal device moves. For practical service scenarios of IoT, we can consider handoff between local edge clouds. This can be addressed as a future work.

Funding Statement: This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea Government (MSIT) (No. 2021R1C1C1013133), and this work was supported by the Institute of Information and Communications Technology Planning and Evaluation (IITP) grant funded by the Korea Government (MSIT) (RS-2022-00167197, Development of

Intelligent 5G/6G Infrastructure Technology for The Smart City), and this work was supported by the Soonchunhyang University Research Fund.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] T. Salam, W. U. Rehman and X. Tao, "Data aggregation in massive machine type communication: Challenges and solutions," *IEEE Access*, vol. 7, pp. 41921–41946, 2019.

[2] J. Gubbi, R. Buyya, S. Marusic and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Elsevier Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[3] X. Xiong, K. Zheng, R. Xu, W. Xiang and P. Chatzimisios, "Low power wide area machine-to-machine networks: Key techniques and prototype," *IEEE Communications Magazine*, vol. 53, no. 9, pp. 64–71, 2015.

[4] P. Tam, S. Math and S. Kim, "Intelligent massive traffic handling scheme in 5G bottleneck backhaul networks," *KSII Transactions on Internet and Information Systems*, vol. 15, no. 3, pp. 874–890, 2021.

[5] D. Y. Kim and S. Kim, "Network-aided intelligent traffic steering in 5G mobile networks," *Computers, Materials & Continua*, vol. 65, no. 1, pp. 243–261, 2020.

[6] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[7] C. Zhang, P. Patras and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2224–2287, 2019.

[8] D. Y. Kim and S. Kim, "Data aggregation-based transmission method in ultra-dense wireless networks," *Intelligent Automation & Soft Computing*, vol. 35, no. 1, pp. 727–737, 2023.

[9] X. Zhang, C. Xia, T. Ma, L. Zhang and Z. Jin, "Optimizing energy-latency tradeoff for computation offloading in SDIN-enabled MEC-based IIoT," *KSII Transactions on Internet and Information Systems*, vol. 16, no. 12, pp. 4081–4097, 2022.

[10] P. Papcun, E. Kajati, D. Cupkova, J. Mocnej, M. Miskuf *et al.,* "Edge-enabled IoT gateway criteria selection and evaluation," *Concurrency and Computation Practice and Experience*, vol. 32, no. 13, pp. e5219, 2020.

[11] J. Yick, B. Mukherjee and D. Ghosal, "Wireless sensor network survey," *Computer Networks*, vol. 52, no. 12, pp. 2292–2330, 2008.

[12] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, "Wireless sensor networks: A survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[13] I. Dimolitsas, D. Dechouniotis, S. Papavassiliou, P. Papadimitriou and V. Theodorou, "Edge cloud selection: The essential step for network service marketplaces," *IEEE Communications Magazine*, vol. 59, no. 10, pp. 28–33, 2021.

[14] A. Ritu, P. M. Tanha, W. N. Islam, S. Islam and A. Chakrabarty, "A comparative study on gateway selection in IoT," in *Proc. the Fourth Int. Conf. on IoT in Social, Mobile, Analytics and Cloud (I-SMAC)*, Palladam, India, 2020.

[15] X. Lai, J. Xia, L. Fan, T. Q. Duong and A. Nallanathan, "Outdated access point selection for mobile edge computing with cochannel interference," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 7, pp. 7445–7455, 2022.

[16] W. T. Chai, B. Y. Ooi, S. Y. Liew and S. Shirmohammadi, "Taxi-sharing: A wireless IoT-gateway selection scheme for delay-tolerant data," in *Proc. the IEEE Int. Instrumentation and Measurement Technology Conf. (I2MTC)*, Houston, USA, 2018.

[17] T. Chen and B. Wu, "Gateway selection based on game theory in Internet of Things," in *Proc. the Int. Conf. on Electronics Technology (ICET)*, Chengdu, China, 2018.

[18] R. Risald, A. E. Mirino and S. Suyoto, “Best routes selection using dijkstra and floyd-warshall algorithm,” in *Proc. the 11th Int. Conf. on Information & Communication Technology and System (ICTS)*, Surabaya, Indonesia, 2017.

[19] M. H. MacDougal, *Simulating Computer Systems, Techniques and Tool*. Cambridge, USA: The MIT Press, 1987.

[20] S. Y. Wang, J. E. Chang, H. Fan and Y. H. Sun, “Comparing the performance of NB-IoT, LTE Cat-M1, Sigfox, and LoRa for IoT end devices moving at high speeds in the air,” *Journal of Signal Processing Systems*, vol. 94, no. 1, pp. 81–99, 2022.

[21] S. M. Ross, *Probability Models for Computer Science*. Burlington, USA: Harcourt/Academic Press, 2002.