# Distributed Server Network Design for Delay-Sensitive Applications with Mobile Users

Haruki Izumiya
*Computer Science and Engineering,*
*Toyohashi University of Technology,*
Aichi, Japan
izumiya.haruki.yf@tut.jp

Eiji Oki
*Graduate School of Informatics,*
*Kyoto University,*
Kyoto, Japan
oki@i.kyoto-u.ac.jp

Akio Kawabata
*Computer Science and Engineering,*
*Toyohashi University of Technology,*
Aichi, Japan
kawabata.akio.oc@tut.jp

***Abstract*—This paper proposes a network design scheme for delay-sensitive services that minimizes the number of excluded users who cannot satisfy the application's delay requirements. In the proposed scheme, each mobile user selects an optimal server from multiple candidate servers, considering the processing time incurred when switching to another server. The optimal server to maintain the application's delay requirements is periodically recalculated for user mobility. We formulate the proposed scheme as an integer linear programming problem. Numerical results on our examined network demonstrate that the proposed scheme reduces the average number of excluded users by about 90% compared with a conventional scheme that always selects the nearest server. These results indicate that the proposed approach is effective for delay-sensitive applications for mobile users.**

***Index Terms*—Distributed processing, mobile device, delay-sensitive applications, event-order correction, integer linear programming problem.**

## I. Introduction

In recent years, the proliferation of high-performance mobile devices such as smartphones has been accelerated by advances in mobile communication technologies, such as 5G [1]. Consequently, services requiring low-delay and seamless connectivity, such as online games [2] and online meetings [3] have seen an increase. Providing a high-bandwidth, low-delay, and seamless network for numerous mobile users is essential in these applications. Furthermore, many telecommunications carriers and cloud providers are offering low-delay processing platforms on networks utilizing Multi-access Edge Computing (MEC) [4]. This enables real-time service delivery over networks that were previously limited by communication delay such as telemedicine, autonomous driving, and real-time online gaming.

Among the real-time services, certain applications, such as online trading and competitive games, require strict event ordering. In these applications, event processing must proceed exactly in the order in which events occur for each user. In applications provided via a network, users closer to the application server experience lower network delay, while users farther away experience higher delay. This disparity can cause events to arrive at the server out of order relative to their actual sequence of occurrence. To address this issue, many applications implement countermeasures. For competitive shooting games, for example, measures like collision detection based on past states are employed to account for network delay.

Research on distributed processing networks has been conducted to address this issue. The work in [5] presented an approach enabling event processing in the original order even when network delay varies among users by aligning all users' delays. In this approach, all users' delays are corrected to the maximum delay between users and servers by performing queuing processing before application processing. However, this study assumes users are on fixed lines and does not consider mobile devices. With the proliferation of MEC and advances in mobile communication, technologies that process events in their order of occurrence for mobile users are expected. However, communication delay with servers may fluctuate significantly as mobile devices move, potentially degrading the operability and quality of low-delay applications.

A study [6] addressed assigning mobile users to multiple service points (multiple distributed servers). In this study, to maintain the quality of service (QoS) for mobile users, users are assigned to the service point by balancing network delay and the processing cost of session migration. However, these studies do not address event-order correction. As a result, user actions may be processed later than intended, disrupting the fairness and consistency of applications such as real-time games. This raises the question: How can servers be assigned to mobile users while ensuring event-order correction and minimizing communication delay simultaneously?

To address this issue, we propose a distributed server network design for delay-sensitive applications with mobile users that ensures application quality by maintaining communication delay within application-specific delay requirements. The proposed scheme enables a distributed processing network to process events in their original order, even for mobile devices, where user mobility induces fluctuating delays. Due to user mobility, the accommodating server is periodically recalculated to re-select the optimal server during movement. If no server can satisfy the acceptable delay for a user, that user is excluded from the service to maintain the application's latency quality. The optimal server selection problem in the proposed scheme is formulated as an integer linear program-

This work was supported by ROIS NII Open Collaborative Research 251S1-22586 and JSPS KAKENHI JP23K16867. The authors would like to thank the Information Media Center (IMC) at Toyohashi University of Technology for providing computer resources and the kind support of its staff.

ming (ILP) problem that minimizes the number of excluded users with a constraint on the application's acceptable delay. The proposed scheme is evaluated with 100 mobile devices and multiple servers located at the node of TMN23 [7], which is a metropolitan area network topology model designed based on Tokyo, Japan. Numerical results show that the proposed scheme reduces the number of excluded users by an average of 90.2% compared with the benchmark, where each user connects to the nearest server, demonstrating more stable and optimal server assignment.

## II. Related Research

### A. Distributed real-time delay-sensitive communication

A network design scheme for distributed systems has been developed (conventional scheme) [5] to reduce end-to-end communication delay in distributed processing environments. This scheme targets applications in which multiple user devices communicate bidirectionally and execute events in their actual order of occurrence. In the conventional scheme, all servers share a common virtual clock, which reproduces the actual event order at user devices. The offset between the current time and the virtual clock represents the common end-to-end delay for all users, and the network design (connections between users' assigned servers and inter-server links) that minimizes this delay is determined. The conventional scheme assumes fixed-line connections and does not account for mobility. Our research extends this scheme to scenarios with mobile user devices.

### B. Dynamically assigning service points to moving users

Bortnikov et al. [6] proposed dynamically and optimally assigning service points, such as application servers, to mobile users. In this approach, real-time, scalable applications, such as large-scale online games or multi-party push-to-talk voice communication, are processed on multiple service points, and each user selects a nearest service point based on location. To determine the optimal service point, they proposed algorithms that minimize the total cost, defined as the sum of the setup cost (e.g., signaling overhead and state transfer when reassigning users) and hold cost (e.g., processing capacity, network delay, and bandwidth while connected). Their approach was evaluated using chat room services provided via an urban wireless network. This approach is similar to our proposed scheme in dynamically assigning application servers to mobile users, but our proposed scheme differs from this approach in that it handles delays consisting of communication delays and server re-select processing delays.

## III. Proposed scheme

### A. Overview

In the proposed scheme, an application is processed on multiple distributed servers in the network, and each user connects to the optimal server from them. As shown in Fig. 1, each mobile device connects to all servers via a mobile network supporting handover: the device can connect to a cellular base station (BS), which then connects to a server via a fixed network. We assume that a network delay between the device and each server can be measured in real time. Based on these measurements, the optimal server for each user is determined, and each device is notified of the assigned server and connects to it. When determining the optimal server for each user, users who cannot satisfy the application's acceptable delay are excluded from the service to ensure application delay quality.

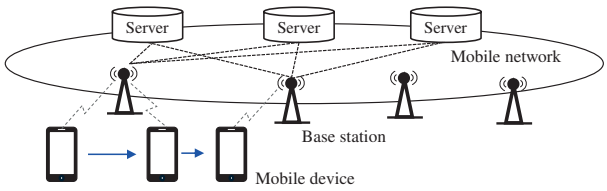


Fig. 1. Communication model.

In the proposed scheme, it periodically recomputes the optimal server for each user based on the latest delay of the devices and servers. By continuously performing this reassignment process, the proposed scheme adapts to user movement and ensures that end-to-end delay remains within the application's acceptable delay. The recalculation interval is set shorter when considering high-speed mobility, allowing the proposed scheme to track delay variations in real time. In the distributed processing servers, events from each user are reordered by their occurrence order before being delivered to the application. To achieve this, incoming events are queued based on their timestamps, ensuring that the delay of all users is the same value and that events are reordered in their original order. Similarly, events sent from the servers are queued before transmission so that they arrive simultaneously at each with the same delay. The optimal server assignment problem is formulated as an ILP problem that minimizes the overall delay while keeping the number of excluded users to a minimum.

### B. Calculation scheme for delay

In the proposed scheme, the delay between users and servers is assumed to arise from two factors. The first is network delay, which increases with communication distance. Specifically, delay is determined by wireless communication between the device and the BS, and by the transmission distance over the fixed network between the BS and the server. The mobile devices are assumed to support handover, allowing them to maintain ongoing communication with a server while moving. An upper bound on user-server delay ($D_{\mathrm{lim}}$) is set as the constraint to prevent the selection of excessively distant servers. In some use-cases, this constraint can also be interpreted as prohibiting the selection of servers (or BS) located outside the radio coverage area, such as in cases where servers are co-located with BS. The second factor is the processing delay ($D_{\mathrm{add}}$) of creating a new connection when a user switches servers. This delay results from connection-

establishment procedures during server switching and occurs only when transitioning to a new server. For server-to-server communication, connections are assumed to be persistently established, and only network delay is considered.

### C. Communication model

In the proposed scheme, the application is processed on distributed servers, such as MEC servers, deployed within a wide-area network. Fig. 2 shows an example of event communication between users and distributed servers. Each user selects one server from among the available servers and exchanges application-related events with it. Each server receives events directly from the users it accommodates, and events from users accommodated on other servers are received via other servers. Each server receives events for all users, and the same application processes with the same events in parallel at distributed servers. Processing results are returned only to the users associated with the corresponding server. For example, User $A$ is associated with Server 1; thus, User $A$'s event $a$ is first sent to Server 1, then multicast by Server 1 to Servers 2 and 3. Conversely, Server 1 receives events $b$ and $c$ from Users $B$ and $C$ via the other servers. After receiving these events, the application at Server 1 processes events $a$, $b$, and $c$, and the resulting output is returned exclusively to User $A$.

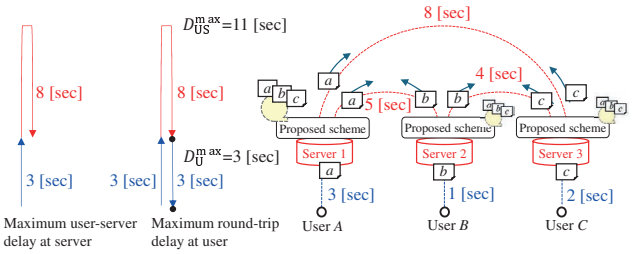


Fig. 2. Prerequisites for distributed processing model.

### D. Event order correction scheme

The event order correction scheme in the proposed approach is based on the conservative synchronization algorithm (CSA) [8]. Before application processing, all events received from users are reordered in their original occurrence order. To achieve this, the system waits until the latest event has arrived at the server and then reorders events using their timestamps. Furthermore, when returning processing results, each server controls the transmission time so that all users receive them simultaneously.

As shown in Fig. 2, the round-trip delay is determined by two parameters: the maximum delay between a user and any server ($D_{\mathrm{US}}^{\max}$), and the maximum delay between a user and its accommodating server ($D_{\mathrm{U}}^{\max}$). For illustration, consider the example in Fig. 2, where $D_{\mathrm{US}}^{\max} = 11$ [sec] and $D_{\mathrm{U}}^{\max} = 3$ [sec]. First, the waiting time required for order correction is explained. As shown in Fig. 2, each event arrives at all servers via the server that accommodates its user. The event with the largest communication delay is that of User $A$, with $D_{\mathrm{US}}^{\max} = 11$ [sec]. To align all users' delays, events are queued before application processing so that their delays equal $D_{\mathrm{US}}^{\max}$. For example, User $B$ has a user-server delay of 1 [sec] and an inter-server delay of 4 [sec]; therefore, a queuing delay of 6 [sec] ($= 11 - 1 - 4$) is added. By this alignment, the application can process all events in their correct occurrence order, as if they all arrive at the current time + $D_{\mathrm{US}}^{\max}$.

Next, the return of processing results is explained. Each processing result is sent from the server that accommodates the user. Among all users, the largest communication delay is $D_{\mathrm{U}}^{\max}$. Thus, if all results are delivered with a delay equal to $D_{\mathrm{U}}^{\max}$, every user receives them simultaneously. For instance, in Fig. 2, User $B$ waits for 2 [sec] before Server 2 transmits the result, so that all users receive it after 3 [sec]. As a result, all users experience a common end-to-end delay of current time + $D_{\mathrm{US}}^{\max} + D_{\mathrm{U}}^{\max}$.

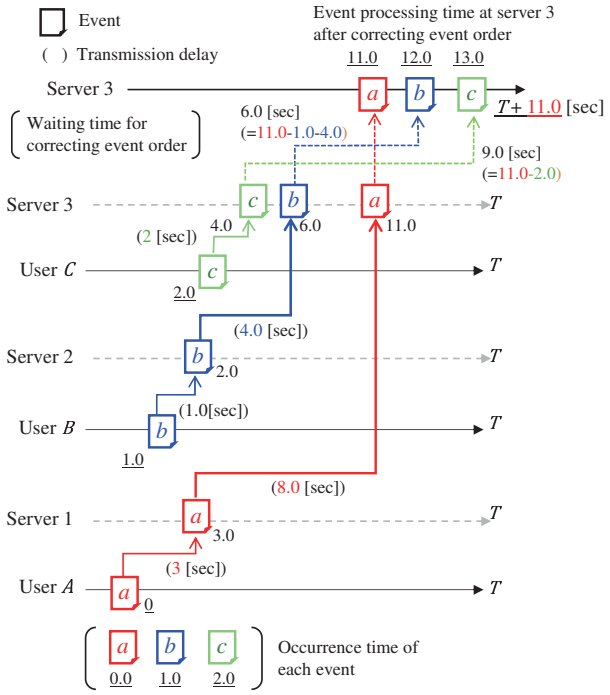


Fig. 3. Example of event order correction.

## IV. Formulation

We describe the ILP problem used to determine the distributed server network design in the proposed scheme. The primary objective function is the number of excluded users. Under the condition that the primary objective function is minimized, the secondary objective function is to minimize the total distance between users and servers. The proposed scheme is designed so that users should select the closest possible server under the condition of minimizing the number of excluded users. Consider an undirected graph $G(V, E)$ as the network model. $V$ is the set of nodes, and $E$ is the set of links. $V_{\mathrm{U}} = \{1, \cdots, |V_{\mathrm{U}}|\}$ is the set of nodes representing users. $V_{\mathrm{S}} = \{1, \cdots, |V_{\mathrm{S}}|\}$ is the set of nodes representing

servers. Since nodes are only user nodes and server nodes, $V_{\mathrm{U}} \cup V_{\mathrm{S}} = V$ and $V_{\mathrm{U}} \cap V_{\mathrm{S}} = \emptyset$. $E_{\mathrm{U}}$ is the set of user-server links, where a link between user $u \in V_{\mathrm{U}}$ and server $i \in V_{\mathrm{S}}$ is denoted as $(u, i) \in E_{\mathrm{U}}$. $E_{\mathrm{S}}$ is the set of server-server links, where a link between server $i \in V_{\mathrm{S}}$ and server $j \in V_{\mathrm{S}}$ is denoted as $(i, j) \in E_{\mathrm{S}}$. Since links consist only of user-server links and server-server links, $E_{\mathrm{U}} \cup E_{\mathrm{S}} = E$ and $E_{\mathrm{U}} \cap E_{\mathrm{S}} = \emptyset$.

We describe the given parameters assigned in the proposed scheme. Table I lists them. We assume that a virtual link exists between each user and all servers. The delay for a user-server link is denoted as $d_{ui}, (u, i) \in E_{\mathrm{U}}$. This delay includes all delays along the path, encompassing the wireless segment delay between the mobile device and the BS, the fixed network delay between the BS and the server, and delays at all other devices on the route. Similarly, we assume that virtual links exist between servers, and denote the delay between server links as $d_{ij}, (i, j) \in E_{\mathrm{S}}$. The delay for server-to-server links includes delays at all devices along the path, including switches and optical fibers. The maximum delay for links between users and servers that users can connect to is set to $D_{\mathrm{lim}}$. If the delay exceeds $D_{\mathrm{lim}}$, the user cannot connect to the server. The processing delay in creating a connection when a user switches to a new server due to user movement is denoted as $D_{\mathrm{add}}$. Considering each server's performance, the maximum number of users that can be connected to server $i \in V_{\mathrm{S}}$ is denoted as $C_i$. To consider this processing delay when a user switches servers, the parameter $p_u \in V_{\mathrm{S}}$ is introduced. $p_u$ denotes the previously connected server number of user $u$, and it is used to detect whether the user switches servers ($c_u$). The acceptable delay specified per application is denoted as $D_{\mathrm{cap}}$.

The decision variables of the proposed scheme are described below. Table I lists them. $n_{ui}$ is a binary variable for link $(u, i) \in E_{\mathrm{U}}$. If link $(u, i) \in E_{\mathrm{U}}$ is selected, $n_{ui} = 1$; otherwise, $n_{ui} = 0$. $n_{ij}$ is a binary variable for link $(i, j) \in E_{\mathrm{S}}$. If link $(i, j) \in E_{\mathrm{S}}$ is selected, $n_{ij} = 1$; otherwise, $n_{ij} = 0$. $y_i$ is a binary variable indicating whether a server is selected. If at least one user selects server $i \in V_{\mathrm{S}}$, then $y_i = 1$. If no user selects server $i$, then $y_i = 0$. $e_u$ is a binary variable indicating whether a user is excluded. If user $u \in V_{\mathrm{U}}$ is excluded, $e_u = 1$; otherwise, $e_u = 0$. $c_u$ is a binary variable indicating server switching. If user $u \in V_{\mathrm{U}}$ selects a different server than last time, $c_u = 1$; otherwise, $c_u = 0$. $D_{\mathrm{U}}^{\max}$ denotes the maximum delay between the users who are not excluded and the connected servers. $D_{\mathrm{US}}^{\max}$ denotes the maximum delay between the users who are not excluded and all servers. It is the delay for reordering all events in the occurrence order for the proposed scheme. $U_{\mathrm{excl}}$ indicates the number of excluded users.

The proposed scheme is formulated as the following ILP problem:

$$\text{Objective} \quad \min \; U_{\mathrm{excl}} + \alpha \sum_{(u,i) \in E_{\mathrm{U}}} d_{ui}\, n_{ui} \tag{1a}$$

$$\text{s.t.} \quad \sum_{i \in V_S} n_{ui} = 1, \forall u \in V_{\mathrm{U}} \tag{1b}$$

TABLE I
GIVEN PARAMETERS AND DECISION VARIABLES.

| | | |
|---|---|---|
| Given parameters | $V_{\mathrm{U}}$ | Set of users |
| | $V_{\mathrm{S}}$ | Set of servers |
| | $E_{\mathrm{U}}$ | Set of links between users and servers |
| | $E_{\mathrm{S}}$ | Set of links between servers |
| | $d_{ui}$ | Delay of link between user $u \in V_{\mathrm{U}}$ and server $i \in V_{\mathrm{S}}$ |
| | $d_{ij}$ | Delay of link between server $i \in V_{\mathrm{S}}$ and server $j \in V_{\mathrm{S}}$ |
| | $D_{\mathrm{lim}}$ | Maximum delay for servers accessible by a user |
| | $D_{\mathrm{add}}$ | Processing delay during server switching |
| | $C_i$ | Maximum number of users that can be connected to server |
| | $p_u$ | Server number previously connected to by user $u \in V_{\mathrm{U}}$, where $p_u \in V_{\mathrm{S}}$ |
| | $D_{\mathrm{cap}}$ | Maximum end-to-end delay tolerated by the application |
| Decision variables | $n_{ui}$ | Binary variable: 1 if link $(u, i) \in E_{\mathrm{U}}$ is used; 0 otherwise |
| | $n_{ij}$ | Binary variable: 1 if link $(i, j) \in E_{\mathrm{S}}$ is used; 0 otherwise |
| | $y_i$ | Binary variable: 1 if server $i \in V_{\mathrm{S}}$ is used by at least one user; 0 if not used by any user |
| | $e_u$ | Binary variable: 1 if user $u \in V_{\mathrm{U}}$ is excluded; 0 if not excluded |
| | $c_u$ | Binary variable: 1 if user $u \in V_{\mathrm{U}}$ connects to a new server; 0 if connecting to the same server as before |
| | $D_{\mathrm{U}}^{\max}$ | Maximum delay between a user and their assigned server |
| | $D_{\mathrm{US}}^{\max}$ | Maximum delay between a user and all servers |
| | $U_{\mathrm{excl}}$ | Number of excluded users |

$$\sum_{u \in V_{\mathrm{U}}} n_{ui} \le C_i, \forall i \in V_{\mathrm{S}} \tag{1c}$$

$$d_{ui}(n_{ui} - e_u) \le D_{\mathrm{lim}}, \forall (u, i) \in E_{\mathrm{U}} \tag{1d}$$

$$y_i \ge n_{ui}, \forall (u, i) \in E_{\mathrm{U}} \tag{1e}$$

$$y_i + y_j - 1 \le n_{ij}, \quad \forall i \in V_{\mathrm{S}}, j \in V_{\mathrm{S}}, (i, j) \in E_{\mathrm{S}} \tag{1f}$$

$$n_{ij} \le y_i, \forall i \in V_{\mathrm{S}}, j \in V_{\mathrm{S}}, (i, j) \in E_{\mathrm{S}} \tag{1g}$$

$$n_{ij} \le y_j, \forall i \in V_{\mathrm{S}}, j \in V_{\mathrm{S}}, (i, j) \in E_{\mathrm{S}} \tag{1h}$$

$$d_{ui}(n_{ui} - e_u) \le D_{\mathrm{U}}^{\max}, \forall (u, i) \in E_{\mathrm{U}} \tag{1i}$$

$$d_{ui}(n_{ui} - e_u) + D_{\mathrm{add}}\, c_u + d_{ij} n_{ij} \le D_{\mathrm{US}}^{\max}, \forall (u, i) \in E_{\mathrm{U}}, (i, j) \in E_{\mathrm{S}} \tag{1j}$$

$$D_{\mathrm{US}}^{\max} + D_{\mathrm{U}}^{\max} \le D_{\mathrm{cap}} \tag{1k}$$

$$\sum_{u \in V_{\mathrm{U}}} e_u = U_{\mathrm{excl}} \tag{1l}$$

$$p_u - \sum_{i \in V_{\mathrm{S}}} (i\, n_{ui}) \le M c_u, \forall u \in V_{\mathrm{U}} \tag{1m}$$

$$\sum_{i \in V_{\mathrm{S}}} (i\, n_{ui}) - p_u \le M c_u, \forall u \in V_{\mathrm{U}} \tag{1n}$$

$$n_{ui} \in \{0, 1\}, \forall (u, i) \in E_{\mathrm{U}} \tag{1o}$$

$$n_{ij} \in \{0, 1\}, \forall (i, j) \in E_{\mathrm{S}} \tag{1p}$$

$$y_i \in \{0, 1\}, \forall i \in V_{\mathrm{S}} \tag{1q}$$

$$e_u \in \{0, 1\}, \forall u \in V_{\mathrm{U}} \tag{1r}$$

$$c_u \in \{0,1\}, \forall u \in V_{\mathrm{U}}. \tag{1s}$$

Equation (1a) shows that the total delay between users and servers is minimized as the secondary objective function, with the number of excluded users as the primary objective function. Since the first objective function $U_{\mathrm{excl}}$ takes integer values, $\alpha$ is set to satisfy $\alpha < \frac{1}{\sum_{(u,i)\in E_{\mathrm{U}}} d_{ui}}$, where the second objective function $\sum_{(u,i)\in E_{\mathrm{U}}} d_{ui} n_{ui}$ takes fractional values. Equation (1b) indicates that a user selects one server from multiple servers. Equation (1c) indicates that the number of users selecting server $i \in V_{\mathrm{S}}$ does not exceed $C_i$. Equation (1d) indicates that each user cannot select a server with a delay exceeding $D_{\mathrm{lim}}$. Equation (1e) indicates that $y_i = 1$ if at least one user has selected the server. Equations (1f)–(1h) are a linear representation of $y_i \cdot y_j = n_{ij}$, indicating that if servers $i \in V_{\mathrm{S}}$ and $j \in V_{\mathrm{S}}$ are selected, the inter-server link $(i,j) \in E_{\mathrm{S}}$ is utilized. Equation (1i) indicates that the maximum delay between a user and the connected server is $D_{\mathrm{U}}^{\max}$. $D_{\mathrm{U}}^{\max}$ is calculated as network delays. Equation (1j) indicates that the maximum delay between the user and all servers is $D_{\mathrm{US}}^{\max}$. $D_{\mathrm{US}}^{\max}$ is calculated as a value that considers both the network and processing delays. Equation (1k) indicates that the maximum round-trip delay, as shown in Fig. 2, $D_{\mathrm{US}}^{\max} + D_{\mathrm{U}}^{\max}$ does not exceed the application's acceptable delay $D_{\mathrm{cap}}$. Equation (1l) indicates that the number of excluded users is $U_{\mathrm{excl}}$. Equations (1m)–(1n) represent that $c_u$ = 1 when a user switches the server and $c_u$ = 0 when the user selects the same server. $M$ is a sufficiently large constant, set to one greater than the number of servers ($M = |V_{\mathrm{S}}| + 1$). Equations (1o) to (1s) indicate that $n_{ui}$, $n_{ij}$, $e_u$, and $c_u$ are binary variables, respectively.

## V. Numerical Results

### A. Experimental settings

To evaluate the effectiveness of the proposed scheme, we evaluate the number of excluded users $U_{\mathrm{excl}}$, beyond the application's acceptable delay. The evaluation is conducted under conditions where 100 mobile devices are moving. We compare $U_{\mathrm{excl}}$ of the proposed scheme with that of the benchmark, where each user selects the nearest server. The parameter $C_i$ representing server processing capacity is set to 100 and 6 (for all $i \in V_{\mathrm{S}}$), verifying that the proposed scheme's effectiveness is maintained even with changes in processing capacity. The processing delay of server switching ($D_{\mathrm{add}}$) is varied from 0 [$\mu$sec] to 100 [$\mu$sec] in 50 [$\mu$sec] increments for evaluation. $D_{\mathrm{lim}}$ is set to 350 [$\mu$sec], and $D_{\mathrm{cap}}$ is set to 1000 [$\mu$sec]. Servers are deployed at each node in TMN23, totaling 23 locations. Each user can select any server. Servers are connected via links in TMN23, and if not directly connected, they are connected via the shortest path on the links. The delay between users and servers is assumed to be 5 [$\mu$sec] per kilometer of distance based on latitude and longitude information. Delays between servers are calculated based on length of link in TMN23.

The number of users is set to 100, divided into four groups of 25 each. The distribution of each group is shown in Fig. 4. Within each of the four regions, divided at latitude 37.675 degrees and longitude 139.75 degrees, 25 users are randomly placed. Each group sequentially moves through six locations, reassigning servers at each transition. For the first location, since there is no previous allocation, the variables and constraints related to server switching are not applied. The proposed scheme in the first location is formulated as the following ILP problem:

$$\text{Objective} \quad \min \; U_{\mathrm{excl}} + \alpha \sum_{(u,i)\in E_{\mathrm{U}}} d_{ui}\, n_{ui} \tag{2a}$$

$$d_{ui}(n_{ui} - e_u) + d_{ij} n_{ij} \le D_{\mathrm{US}}^{\max}, \quad \forall (u,i) \in E_{\mathrm{U}}, (i,j) \in E_{\mathrm{S}} \tag{2b}$$

$$(1b) - (1i), (1k), (1l), (1o) - (1r). \tag{2c}$$

From the second location, the previous selected server of each user is used as input parameters $p_u$, the optimal server considering switching servers is determined using ILP problem (1a)–(1s). Movement is as follows: Group 1 (green) changes longitude by $+0.03°$, Group 2 (orange) changes latitude by $-0.03°$, Group 3 (purple) changes longitude by $-0.03°$, and Group 4 (blue) changes latitude by $+0.03°$. The parameter $\alpha$ is set to 0.00001. The formulated optimization problem is solved using the linear programming solver CPLEX [9].

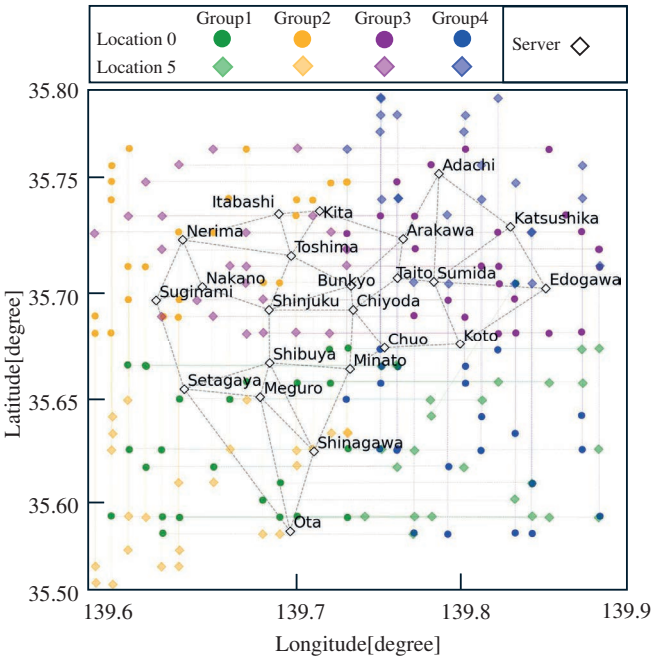


Fig. 4. Server locations and location distribution of each user group.

### B. Number of excluded users

*1) $C_i$=100, $\forall i \in V_{\mathrm{S}}$:* Figure 5 shows the number of excluded users for the proposed scheme and the benchmarks

when $C_i$ = 100, $i \in V_S$, and $D_{add}$ is 0, 50, and 100 [$\mu$sec]. Both the proposed scheme and the benchmark show a decrease in the number of excluded users from user locations 0 to 2, and an increase from locations 2 to 5, regardless of the value of $D_{add}$. As users move from location 1 to location 2, the number of excluded users decreases because many move closer to the nearest server and can connect to the server within $D_{cap}$.

Conversely, as users move from location 3 to location 5, the number of excluded users increases because many move away from the nearest server and cannot connect to any server within $D_{cap}$. As shown in Fig. 5, both the benchmark and the proposed scheme exhibit an increase in the number of excluded users as $D_{add}$ increases. However, the proposed scheme maintains a lower rate of increase than the benchmark because the optimal server is selected considering the processing delay with server switching in the proposed scheme. In the proposed scheme, if the delay, including the processing delay, increases for a new server, the connection to the currently connected server is maintained without switching to the new server.

We have confirmed that the proposed scheme consistently excludes fewer users than the benchmark, regardless of $D_{add}$. On average, the proposed scheme reduced the number of excluded users by 93.0% compared with that of the benchmark. Even under the least effective conditions ($D_{add}$ = 50 [$\mu$sec] and user location 5), a 72.5% reduction is observed. Furthermore, at user locations 1 and 2, the proposed scheme resulted in zero excluded users. These results clearly demonstrate that the proposed scheme can suppress the number of users excluded compared with the benchmark, even when the application's acceptable delay is identical to that of the benchmark.

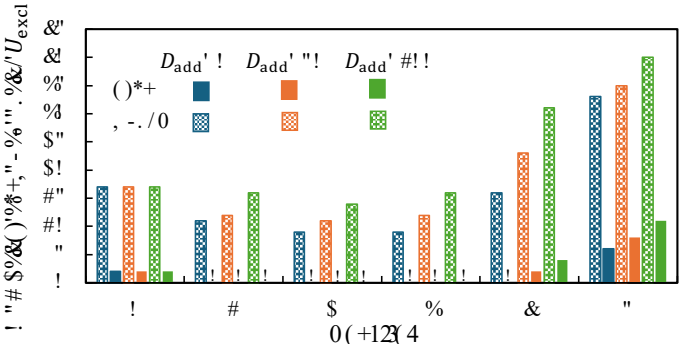


Fig. 5. Number of excluded users when $C_i = 100, \forall i \in V_S$.

*2) $C_i$=6, $\forall i \in V_S$:* Figure 6 shows the number of excluded users for the proposed scheme and the benchmark when $C_i$ = 6 and $D_{add}$ is 0, 50, and 100 [$\mu$sec]. Similar to the case of $C_i$ = 100, as $D_{add}$ increased, both the benchmark and proposed scheme showed an increase in the number of excluded users. Regardless of the $D_{add}$ value, the proposed scheme consistently excluded fewer users than the benchmark. On average, the proposed scheme reduced the number of excluded users by 84.6%. Even under the least effective conditions ($D_{add}$=50 [$\mu$sec] and location 5), a 57.5% reduction is confirmed. Averaging with the case for $C_i$ = 100, the proposed scheme can reduce the number of excluded users by 88.8%. These results clearly demonstrate that even under conditions where the server is imposed with an upper limit on the number of users, the proposed scheme can suppress the number of excluded users compared to the benchmark.

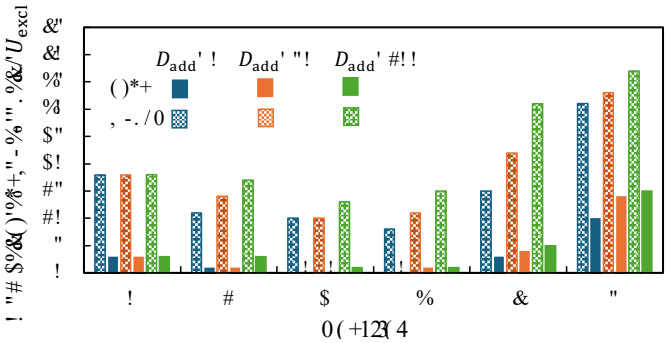


Fig. 6. Number of excluded users when $C_i = 6, \forall i \in V_S$.

## VI. Conclusion

This paper proposes a network design scheme that maintains application communication quality for distributed servers and mobile users, considering constraints such as the application's acceptable delay and the maximum number of connecting users in the server. Using a network topology based on TMN23, we evaluated the proposed scheme. The results showed that, compared with a scheme in which users select the nearest server, the proposed scheme can reduce the number of users excluded by an average of 88.8%. These results demonstrated that the proposed scheme enables a network design that achieves optimal server assignment for mobile users and can accommodate more users.

## References


[1] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. K. Soong *et al.*, "What will 5G be?," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 6, pp. 1065–1082, 2014.

[2] K. T. Chen, P. Huang, and C. L. Lei, "How sensitive are online gamers to network quality?," *Commun. ACM*, vol. 49, no. 11, pp. 34–38, 2006.

[3] Zoom Video Communications, Inc., "Zoom video conferencing platform," https://zoom.us/, online; accessed 23 October 2025.

[4] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—A key technology towards 5G," *ETSI White Paper*, no. 11, pp. 1–16, 2015.

[5] A. Kawabata, B. C. Chatterjee, S. Ba, and E. Oki, "A real-time delay-sensitive communication approach based on distributed processing," *IEEE Access*, vol. 5, pp. 20235-20248, 2017.

[6] E. Bortnikov, I. Cidon, and I. Keidar, "Nomadic service assignment," *IEEE Trans. Mob. Comput.*, vol. 6, no. 8, pp. 915-928, 2007.

[7] T. Tachibana, Y. Hirota, K. Suzuki, T. Tsuritani, and H. Hasegawa, "Metropolitan area network model design using regional railways information for beyond 5G research," *IEICE Trans. Commun.*, vol. E106-B, no. 4, pp. 296-306, 2023.

[8] A. Kawabata, B. C. Chatterjee, and E. Oki, "Optimal server selection scheme with optimistic synchronization for delay sensitive services," *IEEE 18th Annu. Consumer Commun. & Netw. Conf.* (CCNC), Las Vegas, USA, 2021.

[9] IBM, "IBM ILOG CPLEX Optimizer," http://www.ibm.com/software/commerce/optimization/cplex-optimizer/, online; accessed 23 October 2025.