IEICE TRANS. COMMUN., VOL.E103–B, NO.7 JULY 2020

PAPER

Participating-Domain Segmentation Based Server Selection Scheme for Real-Time Interactive Communication

Akio KAWABATA[†a)], ***Member***, **Bijoy CHAND CHATTERJEE**[††], ***Nonmember***, ***and*** **Eiji OKI**[†††], ***Fellow***

SUMMARY This paper proposes an efficient server selection scheme in successive participation scenario with participating-domain segmentation. The scheme is utilized by distributed processing systems for real-time interactive communication to suppress the communication latency of a wide-area network. In the proposed scheme, users participate for server selection one after another. The proposed scheme determines a recommended server, and a new user selects the recommended server first. Before each user participates, the recommended servers are determined assuming that users exist in the considered regions. A recommended server is determined for each divided region to minimize the latency. The new user selects the recommended available server, where the user is located. We formulate an integer linear programming problem to determine the recommended servers. Numerical results indicate that, at the cost additional computation, the proposed scheme offers smaller latency than the conventional scheme. We investigate different policies to divide the users' participation for the recommended server finding process in the proposed scheme.

key words: *real-time application, distributed processing, edge computing, successive participation, domain partitioning*

1. Introduction

Network function virtualization (NFV) [1], [2] affords numerous functions on servers in cloud computing services. In cloud services [3], various applications are served by using virtualization techniques; these applications also include those that are conventionally processed at on-premise environments or home-network. These applications suffer from large latency caused by wide-area networks (WANs). Real-time interactive applications that need latency values of less than several tens of milliseconds, namely network games, telephone services, and music sessions through networks. In addition, these applications are usually needed to be processed the events under the conditions that the event order follows the occurrence order.

Centralized and distributed processing techniques [4] are used in a WAN to manage real-time interactive applications. Centralized approaches, where applications are worked in a centralized server, may incur latency values that cannot be tolerated by latency-sensitive applications. To overcome the problem of centralized approaches, edge computing approaches [5] perform the applications in a distributed manner, where each user accesses its nearest appropriate server. These technologies are effective for center-client type communication in which a part of the center function can be deployed at the edge, but may be not effective for communication among multiple locations. Therefore, real-time applications working at edges distributed in WANs require a technology to suppress communication latencies among multiple locations.

To suppress the latencies, the works in [6]–[8] introduced a server selection scheme based on the edge computing approach, where each application is worked at its nearest appropriate server. The server selection problem is NP-complete [9]; it is unmanageable when a large number of users participate for server selection simultaneously.

In the communication approach [6]–[8], all events are processed at the actual occurrence order on the distributed servers. Users choose one of the servers among multiple distributed servers. The processed data is multicasted to other distributed servers for synchronization. In this communication approach, a virtual time is introduced, which adds the correction time to the current time T for reproducing the occurrence order. As shown in Fig. 1, the time of each event at the server is modified as the corresponding virtual time to replicate the terminal event occurrence order at the actual time. The latency between user p and the selected server is defined D_p, and the maximum value of D_p over all users is defined $D_{\mathrm{U}}^{\max}$. All events are considered to arrive at any server after $D_{\mathrm{U}}^{\max}$. The event of each user is pro-

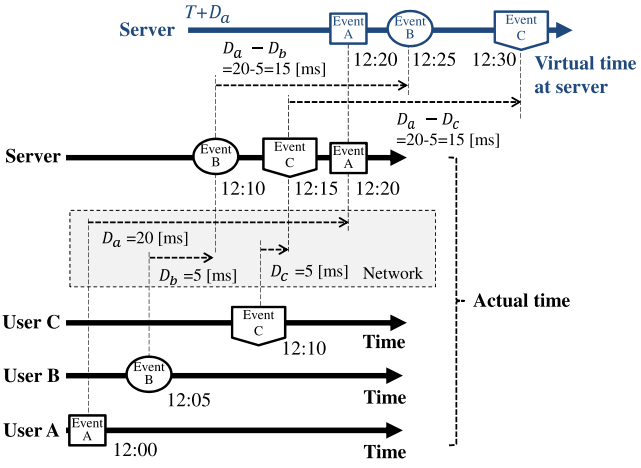
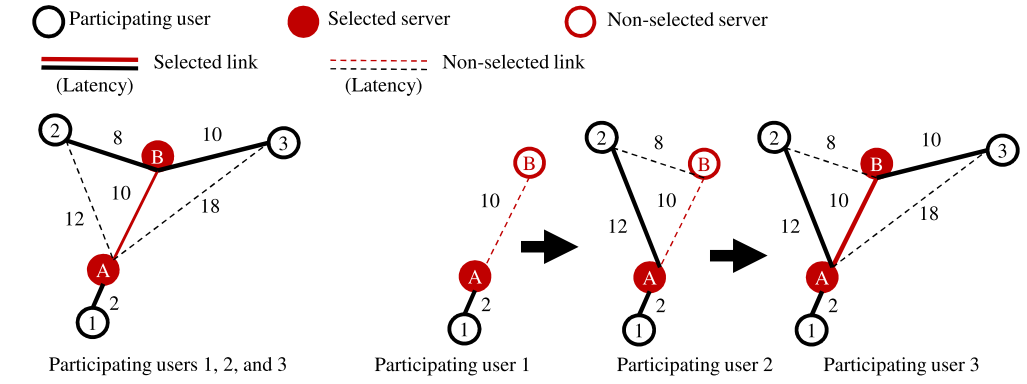


Fig. 1 Example of correction of events in servers using virtual time.

Manuscript received September 12, 2019.
Manuscript revised December 3, 2019.
Manuscript publicized January 17, 2020.

[†]The author is with the NTT Network Service Systems Laboratories, Musashino-shi, 180-8585 Japan.
[††]The author is with the South Asian University, New Delhi, India.
[†††]The author is with the Kyoto University, Kyoto-shi, 606-8501 Japan.
a) E-mail: akio.kawabata.un@hco.ntt.co.jp
DOI: 10.1587/transcom.2019EBP3195



(a) Simultaneous participation scenario

(b) Successive participation scenario

Fig. 2 Examples of server selection in simultaneous and successive participation scenarios.

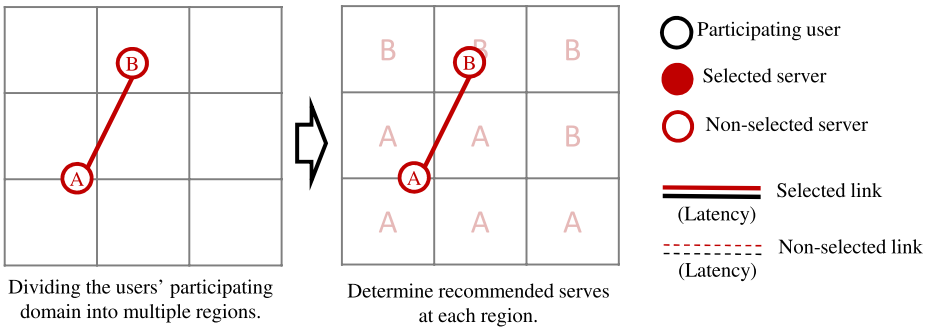
cessed after $D_{\mathrm{U}}^{\max} - D_p$ at the arrival time and is processed on the distributed servers at the virtual time of $D_{\mathrm{U}}^{\max}$. The processed data at each server must be synchronized between distributed servers. The maximum value of latencies over all server-server pairs is defined $D_{\mathrm{S}}^{\max}$. A server that receives the data from another server processes it after at most $D_{\mathrm{S}}^{\max}$. Thus, all events of users on all servers are replicated, in the event occurrence order, at the virtual time after at most $D_{\mathrm{S}}^{\max} + D_{\mathrm{U}}^{\max}$. The virtual time at the server is identical to at most $T + D_{\mathrm{S}}^{\max} + D_{\mathrm{U}}^{\max}$. The virtual time at the user is identical to at most $T + 2D_{\mathrm{S}}^{\max} + D_{\mathrm{U}}^{\max}$ by adding $D_{\mathrm{U}}^{\max}$ to $T + D_{\mathrm{S}}^{\max} + D_{\mathrm{U}}^{\max}$. The value of correction time is $2D_{\mathrm{S}}^{\max} + D_{\mathrm{U}}^{\max}$, as the time difference between current time and virtual time. In this paper, we define the correction time as latency, which is $2D_{\mathrm{U}}^{\max} + D_{\mathrm{S}}^{\max}$.

The work in [10], [11] introduced a server selection scheme according to the segmentation of users' domain. In the server selection scheme, the users' participation domain is divided into several regions to diminish the computational time for resolving the problem of server selection. The approach suppresses the computation time in the simultaneous participating scenario [7].

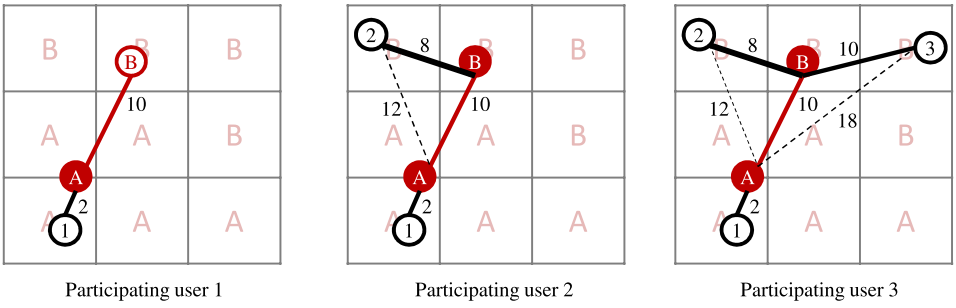
In the successive participation scenario presented in [7], a set of users participates sequentially at various times during processing applications. If a participated user changes its selected server to another one, it may interrupt the application at the user. Therefore, it is desirable when a set of new users joins, the participated users should not change the selected servers. A new user selects either a server that is selected by at least a user or a server that is not selected by any user. Therefore, any other approach than a greedy approach is not acceptable for the server selection at the successive participation scenario. This is because, if we adopt a non-greedy approach, the the user who has participated may change the selected server. The work in [7] introduced the server selection scheme; when a set of users participates, they select the best servers to minimize the latency while keeping that participated users do not change the selected servers, which avoids interrupting the application that is being executed.

The latency considering successive users' participation [7] is typically larger than that of the simultaneous participation, because of the nature of greedy approach. In the sever selection problem [7], the objective is to minimize the latency. Figure 2 shows each example of server selection of the simultaneous and successive scenarios. The latency is obtained by $2D_{\mathrm{S}}^{\max} + D_{\mathrm{U}}^{\max}$. At the simultaneous participation shown in Fig. 2(a), user 1 selects server A, and users 2 and 3 select server B; the latency is $2 \times 10 + 10 = 30$. Since all users select the servers at the same time, each of them selects the optimal server to minimize the latency at the condition that all users participate simultaneously. At the successive participation shown in Fig. 2(b), users 1, 2, and 3 participate sequentially. First, user 1 selects server B, where the latency is $2 \times 2 = 4$; no latency between any two servers is included. Second, user 2 also selects server A, which minimizes the latency at this time, where the latency is $2 \times 12 = 24$. If user 2 selects server B, the latency is $2 \times 8 + 10 = 26$. Thus, server B is not selected by user 2. Then, user 3 selects server B; the latency is $2 \times 12 + 10 = 34$, which is larger than that of the simultaneous participation scenario. Note that the latency in the successive participation scenario depends on the participation order. The reason of latency degradation of successive participation scenario is that user 2 cannot select the server considering user 3. If it is known that user 3 will participate, user 2 selects the server B to minimize the latency. This is because, in the case of successive participation scenario, the user can not select the server considering future participating users.

To overcome the issue of latency degradation using the successive participating scenario [7], this paper proposes a server selection scheme. The proposed scheme applies the basic idea of participating-domain segmentation approach [10], [11], which is used for the simultaneous participating, for the successive participating scenario. A recommended server is introduced and a new user selects the recommended server first. Recommended servers are determined in advance at the condition that users exist in the considered regions. A recommended server is determined for each divided region to minimize the latency when there exists one user at each region. The new user selects the recommended avail-



(a) Recommended server finding process



(b) Server decision process

Fig. 3 Process of proposed scheme at case of Fig. 2.

able server, where the user is located. Figure 3 shows the process of this scheme at the case of Fig. 2. An integer linear programming (ILP) problem is formulated to determine the recommended servers for the proposed scheme.

For the recommended server determining process, we investigate two types of domain partitioning policies. The first policy divides equally the users' domain into regions. The second policy divides equally the users' domain into the regions in which users exist, but excludes those in which users do not exist. We evaluate the proposed scheme in terms of latency and computation time under the condition that the same users participate in different orders and comparing it to the greedy based server selection scheme [7]. The numerical results indicate that smaller latency is obtained using the proposed scheme compared to the conventional greedy based server selection scheme, by employing additional computations for finding the recommended servers before participating of users. We observe that the proposed scheme with second policy suppresses more latency than that of the first policy.

This paper is an extended version of [12] with various additions, which are mainly described as follows. We provide the related works on distributed system for guaranteeing event occurrence order, low latency processing system for network application, and the optimization and approximate algorithms of server selection problems. We employ two different type of policies at the server finding process, and evaluate latency characteristics.

We discuss the structure of the paper as follows. Section 2 provides the related works on distributed systems. Section 3 outlines the conventional successive participation method. The proposed scheme is presented in Sect. 4. We evaluate the proposed scheme in Sect. 5 in terms of latency and computational time. Ultimately, Sect. 6 provides conclusions.

2. Related Work

First, we describe related works of the distributed systems and communication scheme considered in this paper. The application that runs at different distributed servers processes the events by guaranteeing the event occurrence order. Several works have been studied to guarantee the event occurrence order. To guarantee the actual event occurrence order, conventional techniques are typically divided into two major categories, namely (i) conservative synchronization and (ii) optimistic synchronization [13]. In conservative synchronization [14], the event order occurrence is guaranteed by assigning the time information to the event. An example of conservative synchronization is the terminal processing approach. In contrast, the approach of optimistic synchronization does not require to maintain the correct order of events in advance. If the processing function receives a past event, it assurances the event order by rolling back the status and adjusts its outcome.

To reproduce the event order, we typically use the causal

ordering and total ordering methods. The causal ordering [15] assigns a timestamp to event to guarantee the event occurrence order. In contrast, the total ordering [16] is executed at the condition that the same events are processed in the same order at all servers. The latency-sensitive approach [7] is based on the conservative synchronization algorithm and total ordering.

In the context of placement optimization of distributed systems, the server selection problem that minimizes latencies has been studied as the controller placement problem in software-defined networks (SDNs). The work in [17] addressed a specific question on a given topology and investigated the number of required controllers and their placement. The authors examined the fundamental limits to control plane propagation latency on future Internet deployment, and then increase the scope to over 100 openly existing WAN topologies. To estimate the average and worst latencies due to server placement and the number of servers, no placement rule that applies to every network appears when the controller performs distributed processing with multiple servers [17]. The work in [18] presented a server selection, configuration, reconfiguration, and automatic performance verification technology to satisfy the user functional and performance requirements on several types of cloud computing servers.

To reduce the computational time for the server selection problem [7], the work in [19] introduced approximate algorithms at the situation that users select the nearest server, and hence it achieves better scalability compared to the work introduced in [7]. Better scalability is achieved in [19] contrasted to the ILP approach for server selection presented in the simultaneous participation method [6]–[8]. It is observed that the latency using the approximate algorithms in [19] increases as the users' domain becomes large. To address the disadvantage of server selection presented in [19], the work in [10], [11] introduced a server selection scheme according to the segmentation of users' domain.

Considering the low latency processing system for network application, the edge computing [5], [20] has been considered to improve the latency characteristic by processing applications on the servers located near the user. The works in [21], [22] presented two algorithms to suppress the latency variation for client-server and peer-to-peer architectures. Recent research has observed that, when edge computing is incorporated, the performances are improved 20-70 percent compared to the standard web engine [23]. On the other hand, the decentralized approaches require synchronization between servers in the network, which introduce an additional latency.

The research of distributed systems ensuring event occurrence order are mainly focused on the processing of event ordering guarantee; it does not focus on end-to-end latency for communication services and cloud applications [13]–[15]. The research of suppressing end-to-end latency using edge computing is focused on client-server type communication, such as web service [5], [20], [23]. This work is difficult to apply for applications with undirected communication in multiple users. The research of the optimal server selection at SDN controller allocation focuses on the latency between the controller and target devices [17], [18]. This work is not adopted for suppressing the end-to-end latency among the multiple users.

Distributed processing systems typically use greedy based approaches. Taking this direction, the work in [24] presented an algorithm in a heterogeneous environment, which allocates resources using shorter scheduling time. To reduce the scheduling time, the work in [24] adopted an approach that divides jobs into task clusters according to resource properties and requirements. In multi-agent task assignment problems, such as satellite-location assignment in distributed space systems, it is difficult to deploy a centralized coordinator that can access the global information of all tasks and communicates among all agents. To overcome the problem of multi-agent task assignment, the work in [25]. presented an efficient algorithm that can be implemented distributively and asynchronously, which means that there is no centralized coordinator and each agent chooses its task using only local information and local communication based on its own time-clock. These studies are an effective way to improve efficiency by solving complex problems locally, but they do not suppress the error of the value obtained as an objective function associated with the greedy manner.

In this paper, we attempt to improve the performance of greedy manner by suppressing errors of the objective function.

3. Conventional Scheme

The work in [7] introduced the following greedy based server selection scheme, where users participate sequentially without interrupting participated users. The presented scheme assumes that users participate sequentially while processing the application.

The presented scheme in [7] models the network as undirected graph $G(V, E)$, where V and E, respectively, are the sets of nodes and undirected links. The set of all users is represented by $V_U \subseteq V$, and a user is symbolized by $p \in V_U$. The set of servers is represented by $V_S \subseteq V$, and a server is symbolized by $i \in V_S$. $V_U \cup V_S = V$ since a node is either a server or a user, and $V_U \cap V_S = \emptyset$ since there is no node that is both a user and a server. The set of links between user and server is represented by $E_U \subseteq E$, and a link between user $p \in V_U$ and server $i \in V_S$ is symbolized by $(p, i) \in E_U$. $(p, i) \in E_U$ exists between all users $p \in V_U$ and all servers $i \in V_S$. $(p, i) \in E_U$ is a logical link, which is established by using a set of physical links such as wired or wireless ones. The existence of $(p, i) \in E_U$ means that user $p \in V_U$ is able to access server $i \in V_S$ logically. The set of links between server and server is represented by $E_S \subseteq E$, and a link between server i and server j is symbolized by $(i, j) \in E_S$. $(i, j) \in E_S$ exists between all servers $i \in V_S$ and all servers $j \in V_S$ $(i \neq j)$. $(i, j) \in E_S$ is a logical link, which is established by using a set of physical links such as wired or wireless ones. The existence of $(i, j) \in E_S$ means

that servers $i \in V_S$ and $j \in V_S (i \neq j)$ is able to be connected logically. $E_U \cup E_S = E$ since a link is either a server-server or a user-server link, and $E_U \cap E_S = \emptyset$ since there is no link that is both a link among servers and a link between user and server.

The set of nodes representing the participated users is symbolized by V_U^1. V_U^2 denotes a set of nodes that express the newly participating users. E_U^1 represents a set of used links between users and the server. The set of links connecting new users to the servers is symbolized by E_U^2. In this setting, we consider the following. (i) A user is either in V_U^1 or V_U^2. (ii) A link is either in E_U^1 or E_U^2. Considering (i) and (ii), we deduce that $V_U^1 \cup V_U^2 = V_U$, $E_U^1 \cup E_U^2 = E_U$, $V_U^1 \cap V_U^2 = \emptyset$, $E_U^1 \cap E_U^2 = \emptyset$.

d_{pi} represents latency of the link between user $p \in V_U$ and server $i \in V_S$. $D_U^{\max}$ expresses the maximum value of d_{pi} over the selected links. d_{ij} denotes latency of the link between server $i \in V_S$ and server $j \in V_S$. $D_S^{\max}$ represents the maximum value of d_{ij} over the selected links. M_i denotes the maximum number of users that is allowed to belong to server $i \in V_S$. x_{kl} denotes a binary variable for $(k, l) \in E_S$, where $x_{kl} = 1$ if (k, l) is used, and $x_{kl} = 0$ otherwise. y_i is a binary variable for $i \in V_S$, where $y_i = 1$ if server i is used, and $y_i = 0$ otherwise. $x_{kl} = 1$ if $y_k = 1$ and $y_l = 1$, and $x_{kl} = 0$ otherwise. In other words, $y_k \cdot y_l = x_{kl}$ is satisfied.

x_{pi}^{given} represents the value of x_{pi} for link $(p, i) \in E_U^1$ between user $p \in V_U^1$ and server $i \in V_S$, which is a given parameter. The value y_i is determined by whether a participated user is connected to server $i \in V_S^1$, and is symbolized by y_i^{given}. For $i \in V_S^2$, y_i^{given} is defined as $y_i^{\text{given}} = 0$, since the considered servers are inactive. In the following, an ILP is formulated as the server selection problem for successive participation scenario.

$$\text{Objective} \quad \min(2D_U^{\max} + D_S^{\max}) \tag{1a}$$

$$\textbf{s.t.} \quad \sum_{i \in V_S} x_{pi} = 1, \forall p \in V_U^2 \tag{1b}$$

$$x_{pi} \geq x_{pi}^{\text{given}}, \forall (p, i) \in E_U^1 \tag{1c}$$

$$y_i \geq y_i^{\text{given}}, \forall i \in V_S \tag{1d}$$

$$\sum_{p \in V_U} x_{pi} \leq M_i, \forall i \in V_S \tag{1e}$$

$$x_{pi} d_{pi} \leq D_U^{\max}, \forall (p, i) \in E_U \tag{1f}$$

$$x_{ij} d_{ij} \leq D_S^{\max}, \forall (i, j) \in E_S \tag{1g}$$

$$y_i \geq x_{pi}, \forall p \in V_U, i \in V_S \tag{1h}$$

$$y_i + y_j - 1 \leq x_{ij}, \forall (i, j) \in E_S \tag{1i}$$

$$x_{ij} \leq y_i, \forall i \in V_S, (i.j) \in E_S \tag{1j}$$

$$x_{ij} \leq y_j, \forall j \in V_S, (i, j) \in E_S \tag{1k}$$

$$x_{ij} \in \{0, 1\}, \forall (i, j) \in E_S \cup E_U \tag{1l}$$

$$y_i \in \{0, 1\}, \forall i \in V_S \tag{1m}$$

$D_U^{\max}$, $D_S^{\max}$, x_{pi}, x_{ij}, and y_i are used as decision variables. M_i, y_i^{given}, d_{pi}, x_{pi}^{given}, and d_{ij} are used as given parameters. Equation (1a) expresses that the objective function to minimize the latency. Equation (1b) represents that one link is used between a new user and a server. Equation (1c) represents that $x_{pi} = 1$ of selected link at participated users and $x_{pi} = 0$ of non-selected link at participated users. Equation (1d) represents that $y_i = 1$ of selected link at participated users and $y_i = 0$ of non-selected link at participated users. Equation (1e) represents that the sum of number of participated users and new users that belongs to server i does not exceed M_i. Equation (1f) represents that the maximum value of d_{pi} for all selected $(p, i) \in E_U$ does not exceed $D_U^{\max}$. Equation (1g) represents that the maximum value of d_{ij} for all selected $(i, j) \in E_S$ does not exceed $D_S^{\max}$. Equation (1h) represents that server i is used, $y_i = 1$, if it is used by at least one user. Equations (1i)–(1j) indicate $y_k \cdot y_l = x_{kl}$ and $y_k \cdot y_l = x_{kl}$ in the linear model. Equations (1l)–(1m) represents that x_{ij} and y_i are binary variables.

The successive participation scheme is explained in the following.

- Step 1:
  - Initialization, $V_U = 0$, $E_U = 0$.
- Step 2:
  - Participation of a set of new users V_U^2, and generate E_U^2 form V_U^2.
- Step 3:
  - Solve the optimization problem constructed in (1a)–(1l).
- Step 4:
  - The value of x_{kl} for considered user is substituted with x_{kl}^{given}.
  - The value of y_i for considered user is substituted with y_i^{given}.
  - $V_U^1 \leftarrow V_U^1 \cup V_U^2$
  - $E_U^1 \leftarrow E_U^1 \cup E_U^2$
  - Go to Step 2.

4. Proposed Scheme

4.1 Overview

We already noticed that the latency using the successive participation method is higher than that of the simultaneous participation method [7]. The proposed scheme aims to suppress the latency of the successive participation method introduced in [7] by dividing the users' domain. Since it divides the users' domain into several regions and determines the recommended server for each region in advance, the latency and the server for each user are decided considering the recommended server as long as it is available.

The proposed scheme involves two phases, which are (i) recommended server finding process and (ii) sever decision process. It requires less amount of computing time

as the recommended server finding process is limited by restricting the searching space; the proposed scheme considers approximation for server selection. We assume that one user exists in each region in the recommended server finding process. This assumption is according to the condition that users exist in the considered regions in the users' domain. The selected servers in each region are treated as the recommended server for the server decision process. The server decision process considers the outcomes of the recommended server finding process as inputs. Lastly, the server decision process decides the selected server for each user considering the recommended server during successive user participation and estimates latency. In the server decision process, it is necessary to consider that the new user cannot select the recommended server when the number of accommodated users of the recommended server crosses its limit. In that situation, the user selects the server that minimizes the latency without considering the recommended server.

In summary, a recommended server is decided by the first phase of the proposed scheme before user participation, and the second phase the proposed scheme selects which server is used by each user considering the recommended server. We discuss the ILP formulations for the recommended server finding process followed by the server decision process in the following subsections.

4.2 Network Model

We model the network as undirected graph $G'(V', E')$, where V' and E', respectively, represent the sets of nodes and non-directional links. The set of servers is represented by $V_S \subseteq V'$, and a server is symbolized by $i \in V_S$. The set of regions is represented by $V_R \subseteq V'$, and a region is symbolized by $m \in V_R$. $V_R \cup V_S = V'$ since a node is either a region or a server, and $V_R \cap V_S = \emptyset$ since there is no node that is both a user and a server. The set of links between region and server is represented by $E_R \subseteq E'$, and a link between region $m \in V_R$ and server $i \in V_S$ is symbolized by $(m, i) \in E_R$. $(m, i) \in E_R$ exists between regions $m \in V_R$ and server $i \in V_S$. The set of links among servers is represented by $E_S \subseteq E'$, and a link between server $i \in V_S$ and server $j \in V_S$ is symbolized by $(i, j) \in E_S$. A region is permitted to choose one or more servers. $E_R \cup E_S = E'$ is satisfied since a link is either a region-server or server-server link, and $E_R \cap E_S = \emptyset$ since there is no link that is both a link among servers and a link between user and a server.

4.3 Recommended Server Finding Process

We discuss the recommended server finding process of the proposed scheme in this subsection. As the users' domain is divided into several regions, each region has only one user and the user in each region can select any server. We adopt these conditions as the simplest model with which users exist in the considered regions; the recommended server finding process is executed before user participating. We formulate an ILP problem for the recommended server finding process to achieve our goal as follows.

We consider the same notations used in Sect. 3, unless stated otherwise. d'_{mi} represents the latency between $m \in V_R$ and server $i \in V_S$. d'_{ij} expresses the latency between server $i \in V_S$ and $j \in V_S$. The maximum values of d'_{mi} over participated $(m, i) \in E_R$ and d'_{ij} over used $(i, j) \in E_S$ are represented by $D_R^{\max}$ and $D_S^{\max}$, respectively. x'_{xl} is a binary variable for $(k, l) \in E'$, where $x'_{kl} = 1$ if (k, l) is used, and $x'_{kl} = 0$ otherwise. $x'_{kl} = 1$ if $y_k = 1$ and $y_l = 1$, and $x'_{kl} = 0$ otherwise. Videlicet, $y_k \cdot y_l = x'_{kl}$ is satisfied.

$$\text{Objective} \quad \min \{(2D_R^{\max} + D_S^{\max}) + \alpha(\sum_{(m,i)\in E_R} x'_{mi} d'_{mi})\} \tag{2a}$$

$$\text{s.t.} \quad \sum_{i \in V_S} x'_{mi} = 1, \forall m \in V_R \tag{2b}$$

$$\sum_{m \in V_R} x'_{mi} \le M_i, \forall i \in V_S \tag{2c}$$

$$x'_{mi} d'_{mi} \le D_R^{\max}, \forall (m, i) \in E_R \tag{2d}$$

$$x'_{ij} d'_{ij} \le D_S^{\max}, \forall (i, j) \in E_S \tag{2e}$$

$$y_i \ge x'_{mi}, \forall m \in V_R, i \in V_S \tag{2f}$$

$$y_i + y_j - 1 \le x_{ij}, \forall (i, j) \in E_S \tag{2g}$$

$$x'_{ij} \le y_i, (i, j) \in E_S \tag{2h}$$

$$x'_{ij} \le y_j, (i, j) \in E_S \tag{2i}$$

$$x'_{uv} \in \{0, 1\}, \forall (u, v) \in E' \tag{2j}$$

$$y_i \in \{0, 1\}, \forall i \in V_S \tag{2k}$$

The decision variables are $D_R^{\max}$, $D_S^{\max}$, x'_{mi}, x'_{ij}, and y_i. The given parameters are d'_{mi}, d'_{ij}, and M_i. Equation (2a) expresses that the first objective function is the latency at the communication scheme assumed in [7] and the second object function is the sum of latency of all users. The weight of the second term in (2a) is expressed by α that is a constant. To satisfy the constraint, which is $(2D_R^{\max} + D_S^{\max}) \gg \alpha \sum_{(m,i)\in E_R} x'_{mi} d'_{mi}$, the network designer adjusts the parameter of α. In other words, α is an adjustable parameter that can be used to set the first and second terms of (2a) as primary and secondary objective functions, respectively.

One link, $(m, i) \in E_R$, is used between a region-server pair is expressed by (2b). Equation (2c) states that the sum of number of used link that belongs to server i does not exceed M_i. Equation (2d) states that the maximum value of d'_{mi} for all selected $(m, i) \in E_R$ does not exceed $D_R^{\max}$. Equation (2e) represents that the maximum value of d'_{ij} for all selected $(i, j) \in E_S$ does not exceed $D_S^{\max}$. Equation (2f) represents that server i is used, $y_i = 1$, if it is used by at least one region. Equations (2g)–(2i) indicate $y_k \cdot y_l = x'_{kl}$ and $y_k \cdot y_l = x'_{kl}$ in the linear model. Equations (2j)–(2k) states that x'_{mi}, and y_i are binary variables.

For the sake of simplicity, the latency between a region and a server is considered as proportional to the length between the server and the region. Note that, the proposed scheme is still valid by defining the latency between a re-

gion and a server as a function of locations of them, even if this assumption is not maintained. In this paper, we use conservative approximation [26] to determine the length between the region and the server; the location of the region is decided regardless of the number of users and their participation location.

d'_{mi} represents the distance between region $m \in V_R$ and server $i \in V_S$. d'_{mi} is considered as the distance between server i and the furthermost point of region $m \in V_R$, which is shown in Fig. 4. $\boldsymbol{r}_i$ and Q_m represent, respectively, a position vector of server i and the set of position $\boldsymbol{q}$ whose tip position is situated at region $m \in V_R$. The distance between $\boldsymbol{q}$ and $\boldsymbol{r}_i$ is represented by $|\boldsymbol{q} - \boldsymbol{r}_i|$. The furthermost distance from region $m \in V_R$ to server i is stated in (3).

$$d'_{mi} = \max_{\boldsymbol{q} \in Q_m} |\boldsymbol{q} - \boldsymbol{r}_i|. \tag{3}$$

4.4 Server Decision Process

The server decision process determines the server that is selected by each user, and the latency is obtained. The recommended sever finding process indicates that the recommended server of region m is server i when x'_{mi}=1. If the total number of new users and participated users who select server i is less than or equal to M_i, new users select recommended server i. Otherwise, it is necessary to solve the ILP problem in (1a)–(1m). The procedure for the server decision process is given in the following.

- Step 1-1:
  - Dividing the predetermined users' domain into multiple regions.
- Step 1-2:
  - Solve the ILP problem for the recommended server finding process formulated in (2a)–(2k).
  - Decide the recommended servers for each region.
- Step 2-1:
  - Initialization, $V_U = 0$, $E_U = 0$.
- Step 2-2:
  - Participation of a set of new users V_U^2, and generate E_U^2 form V_U^2.
  - Determine the region where new users $p \in V_U^2$ located and the recommended server $i \in V_S$.
  - If the total number of new users and participated users who select server i does not exceed M_i, new users select server i.
  - If the total number of new users and participated users who select server i is larger than M_i, the ILP problem formulated in (1a)–(1m) is solved.
- Step 2-3:
  - The value of x_{pi} for considered user is substituted with x_{pi}^{given}.
  - $V_U^1 \leftarrow V_U^1 \cup V_U^2$.
  - $E_U^1 \leftarrow E_U^1 \cup E_U^2$.
  - Go to Step 2-2.

In Step 2-2, if only one new user participates, there is no need to solve the ILP problem formulated in (2a)–(2k); the server decision process selects the server among all useful servers, which causes the minimum latency.

5. Evaluation

We assess the performance of the proposed scheme in terms of latency and computation time. The outcomes of the proposed scheme are compared with the conventional scheme. We assume the following conditions for evaluation. The latency is considered to be proportionate with the distance of transmission. In advance, we estimate the latency between server-to-server and user-to-server. The latency for processing at server is ignored. We solve the ILP model using Solving Constraint Integer Programs (SCIP) [27]; the computer configured with Intel(R) Core(TM) i7-4790 3.60 GHz 8 core and 32GB memory is used to solve the ILP models.

We assume that the servers are located in Kanto-area of JPN48 model [28] shown in Fig. 5. We assume that 1000, 500, 200, 100, 50, 30, and 10 users are uniformly distributed in the Kanto-area, which are shown in Figs. 6. The users at Figs. 6(b)–6(g) are randomly selected from the users at Fig. 6(a). Each user can connect all servers directly, and all

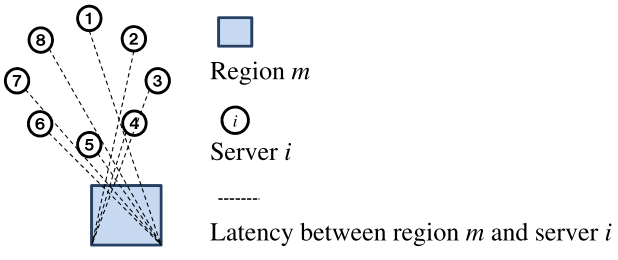


Fig. 4 Distance between server i and region m.

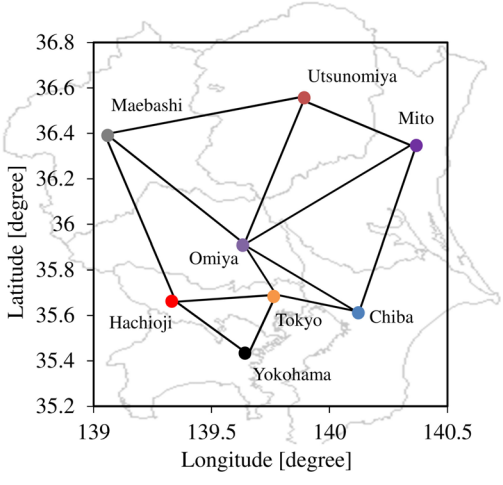


Fig. 5 Server locations of Kanto-area of JPN48 model.

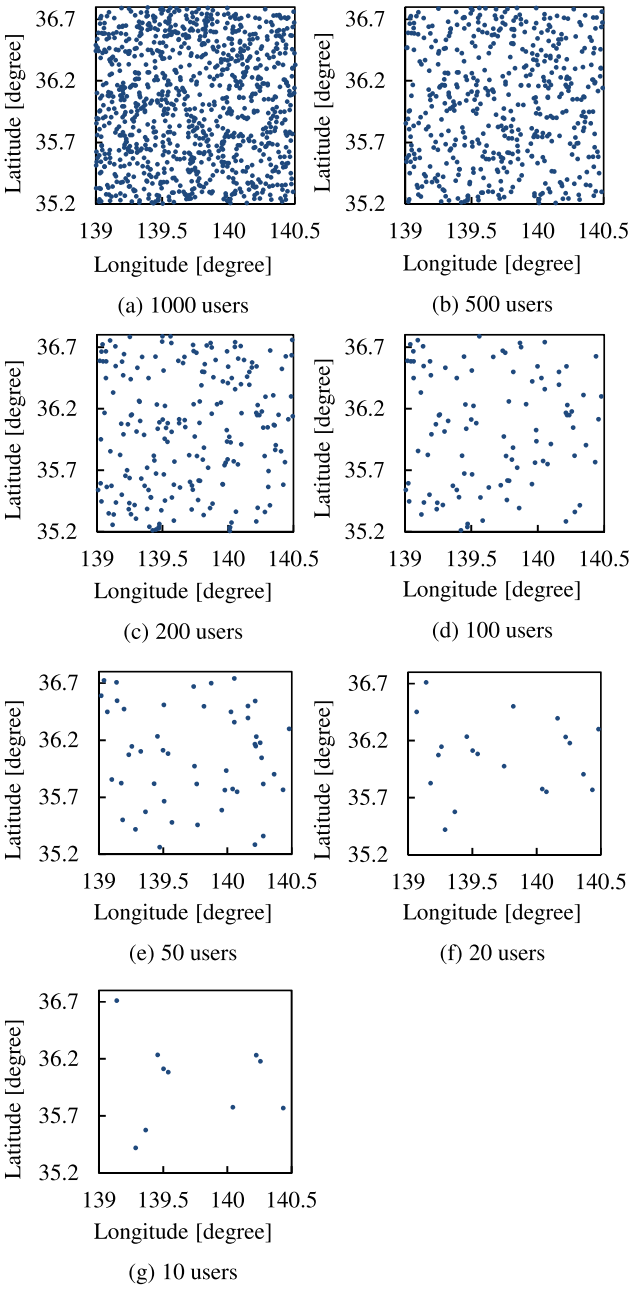


Fig. 6 Uses location patterns of Kanto-area of JPN48 model.

servers logically connect with each other at shortest routes with the physical links. For example, the servers in Tokyo and Maebashi are logically connected via Omiya, using the Tokyo-Omiya physical link and the Omiya-Maebashi physical link.

In the horizontal axis, the considered longitude from 139 to 140.5 corresponds to 150 km, which is equivalent to 0.75 [ms] of latency. In the vertical axis, the considered latitude from 35.2 to 36.8 corresponds to 150 km, which is equivalent to 0.75 [ms] of latency. The distance between a user-server pair is considered as the linear distance on the coordinate multiplied by 1.6 for the effect of latency in access or wireless networks.

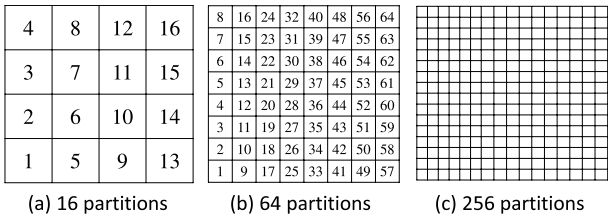


Fig. 7 Partition patterns of the users' domain in the proposed scheme.

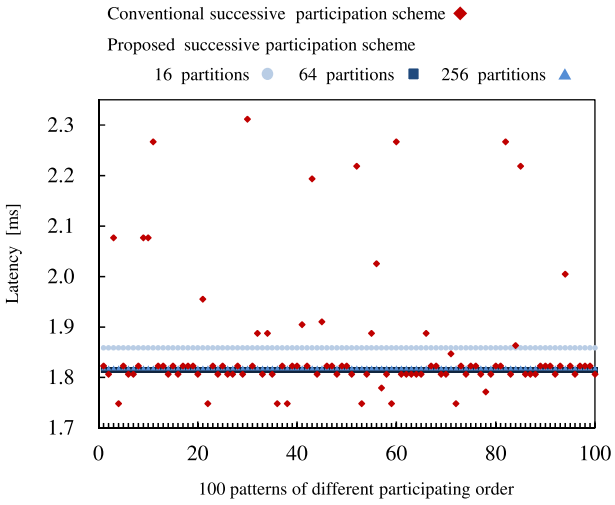


Fig. 8 Latencies using proposed and conventional schemes with different participation orders of 1000 users in each scenario.

We consider that the users' domain is equally divided into 16, 64 and 256 square regions, shown in Fig. 7. In this evaluation, we consider α=1 × 10^{-9}, $|V_R|$ = 16, 64 or 256, $|V_S|$=8, $|V_U|$=10, 20, 50, 100, 200, 500 and 1000, $|E_R|$ = 128, 512 and 2048, $|E_S|$=56, and $|E_U|$=80, 160, 400, 800, 1600, 4000 and 8000. We assume that there is no capacity restriction of users at the server, the value of M_i in all servers are set to the number of users, unless otherwise stated. We evaluate each scheme with 100 different participating order patterns at the same users, shown in Figs. 6(a)–6(g). New users participate one by one and select the server using the conventional method or the proposed method every time.

We employ two different types of domain partitioning polices for the server finding process. The first policy divides equally the users' domain into regions. The second policy divides equally the users' domain into the regions in which users participate, but excludes those where users do not participate.

Figure 8 shows the latencies using the proposed and conventional schemes when 1000 users participate with 100 different participating orders; the same users are considered for each pattern. The latencies using the conventional scheme with different participating orders fluctuate as the conventional scheme tends to select a near server. In our examined all partition patterns, the latencies of the proposed scheme do not fluctuate depending of the user participating order. Table 1 shows the standard deviation of the conven-

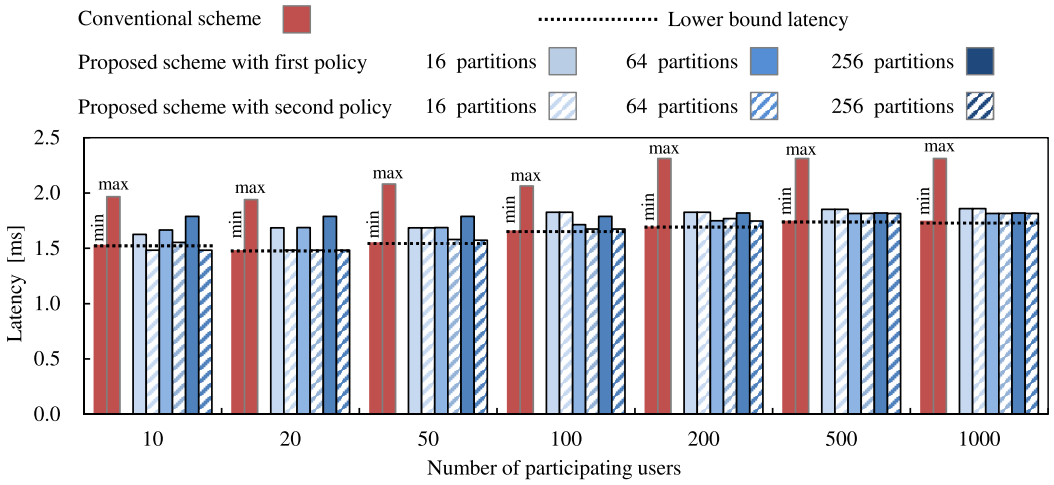


Fig. 9 Latencies using proposed and conventional schemes with 100 different participating orders.

Table 1 Standard deviation for each pattern using different schemes.

Number of users	Conventional	Number of partitions in proposed		
		16	64	256
1000	0.12	0.00	0.00	0.00
500	0.12	0.00	0.00	0.00
200	0.13	0.00	0.00	0.00
100	0.09	0.00	0.00	0.00
50	0.14	0.00	0.00	0.00
20	0.12	0.00	0.00	0.00
10	0.09	0.00	0.00	0.00

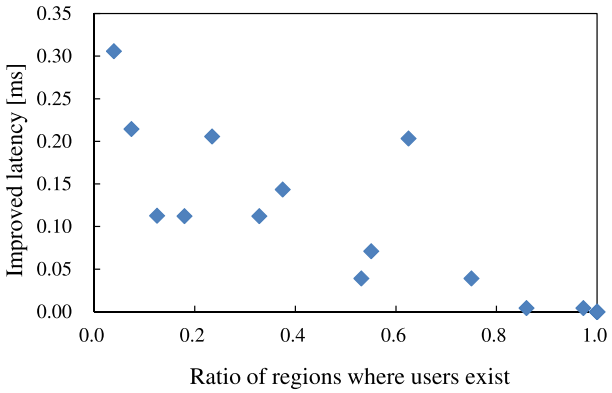


Fig. 10 Dependency of improved latency between two policies on the rate of the regions where users exist.

tional scheme and the proposed scheme, respectively. These results indicate that the proposed scheme overcomes the issue of the conventional scheme in which the latency fluctuates due to the participating order of users.

Figure 9 shows the latencies using the proposed and conventional schemes with 100 different participating orders at 10–1000 users that shown in Fig. 6. The latencies of the conventional scheme show maximum and minimum values out of 100 different participation order results. The latencies of the proposed scheme are the same results for all 100 different participation orders. These results indicate that the latencies of the proposed scheme are improved compared to that of the conventional scheme in each number of users. In the proposed scheme, the latencies of the second policy becomes lower than that of the first policy as the number of users becomes smaller.

In the successive participation scenario, the lower bound of the latency is obtained by solving the sever selection problem in the simultaneous participation scenario. The latency of successive participation scenario must not be lower than that of the simultaneous participation scenario, where all the users participate simultaneously and the optimization problem is solved only one time.

The proposed scheme suppresses the fluctuation of latency and provides better latency performance, compared to the conventional scheme; the latencies obtained by the proposed scheme with the second policy are close to the lower bound of the latency. The value of latency reduction is about 0.5 ms, which is equivalent to transmission latency with 100 km at an optical fiber. For a network service provider, the latency reduction of 0.5 ms is equivalent to extending the service providing area to about 100 km far without service degradation. If the users' domain is more larger, the effect of reduction can be also more larger.

The conventional scheme sometimes achieves smaller latency than the proposed scheme. This is due to (i) the difference between the distance of an actual server and a user and the (ii) distance of a corresponding region and a server. The distance difference may lead to the misselection of servers. It is expected that the latency can be suppressed by dividing the users' domain into more finer regions and determine the recommended servers using the second policy. Addressing this issue is left as part of our future work.

The difference between the second policy and the first policy is whether to determine recommended servers in consideration of regions that has no user. To analyze the difference of the two policies, we compare the latencies between the two policies depending on the number of regions that have no user. Figure 10 shows the dependency of improved

latencies between two policies on the rate of the regions where user exists. The ratio of regions on the horizontal axis is indicated as the ratio of the number of regions without any user to the number of regions for the entire domain. The improved latency on the vertical axis is indicated as the time that the latency with the second policy improved from the first policy. From the results of Fig. 10, the latency improvement effect of the second policy increases as the number of regions that have no user increases. These results indicate that the regions with no user negatively affect the selection of suitable servers, which leads to selecting servers inappropriately and increasing the latency. As shown in Fig. 11, the inappropriate server selection may negatively affect the recommended server finding process. The recommended server of the region with user 2 is server B in the first policy. The recommended server of the region with user 2 is server A in the second policy. In the second policy, the region with user 2 select server A, if the region with user 1 has selects server A. In the first policy, the region with user 2 selects server B, since the recommended servers are selected by considering that users exist all regions.

Figure 12 shows the maximum and minimum latencies of 100 users using the proposed and conventional schemes with 100 different participating orders, when M_i is limited. The maximum and latencies are presented by the right and left bars for each condition, respectively. The lower bound of the latency, which is obtained in the same way as that presented in Fig. 10, is also depicted in Fig. 12 as a reference. If the total number of new users and participated users who select server i is less than or equal to M_i, the new users select the recommended server. Otherwise, each new user selects the server that is determined by the ILP problem formulated in (1a)–(1m). M_i is set to the the number of users, $|V_U|$, multiplied by γ, i.e., $M_i = \gamma|V_U|$, where $0 < \gamma \leq 1$. Figures 12 presents the latencies; γ is set to $\frac{1}{6}$, $\frac{1}{5}$, $\frac{1}{4}$, $\frac{1}{3}$, $\frac{1}{2}$, and 1. Note that, when $\gamma = 1$, there is no limitation of M_i. In the conventional scheme, the latency increases as M_i becomes small, since new users may not select suitable server due to the limitation of M_i. This trend is the same using the proposed scheme. In the proposed scheme, some fluctuations of the latencies are observed depending on the order of user participation. The is because the user who cannot select the recommended server varies depending on the order of user participation. From these results, the effectiveness of the proposed scheme is limited due to the limitation of M_i.

The computation times for the server decision process in proposed and conventional schemes are comparable. The proposed scheme requires additional computations for find-

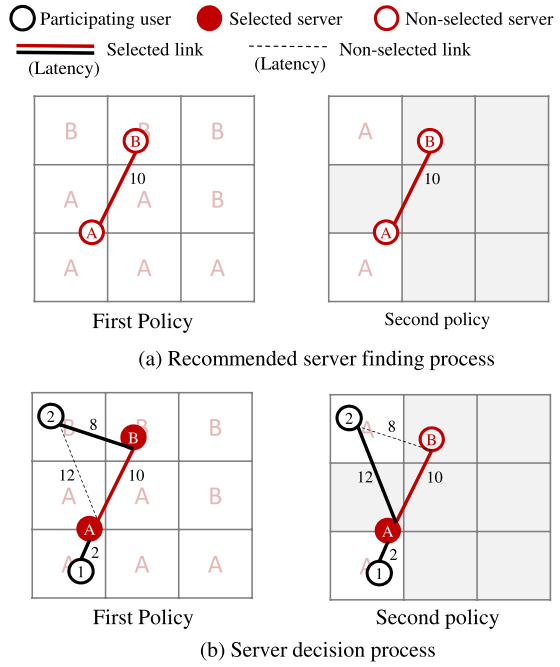


Fig. 11 Example of server selection using first and second policies.

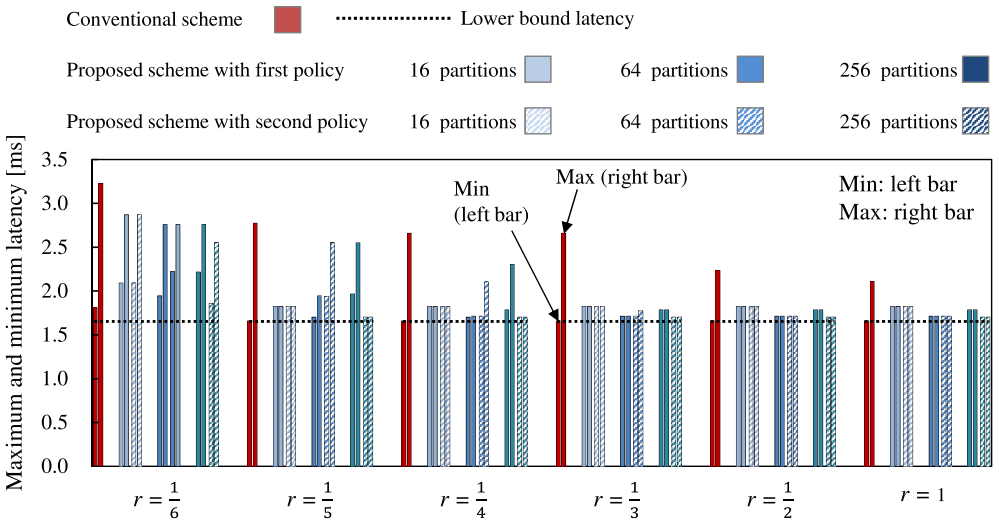


Fig. 12 Maximum and minimum latencies of 100 different participating orders with 100 users with M_i limitation, $M_i = \gamma|V_U|$.

Table 2 Computation time for recommended server finding process.

No. of partitions	Computation time [sec]
16	0.19
64	1.30
256	10.53

ing the recommended servers, and this process is executed before participating of users. Table 2 shows the average computation time using the proposed scheme with the first policy for different partitions to execute the recommended server finding process. Note that the computation time with the second policy is less than that of the first policy. We observe that the computation time increases with increase in the number of partitions.

The above results indicate that the proposed scheme suppresses the latency compared to the conventional scheme by employing additional computations for finding the recommended servers before participating of users. In the proposed scheme, the latencies of 256 partition are not necessarily the lowest latency of the three partition patterns. These results indicate that a latency is not necessarily reduced even if regions are divided into smaller. An optimal region partitioning policy to reduce latency is a forthcoming challenge. Since the server finding process is processed prior to user participation, it may be difficult to guess the region if user does not participates, but it can be useful to reduce latency by excluding the regions, such as mountainous areas and the sea.

In summery, the proposed scheme improves latency in the successive participation scenario in a greedy manner, which avoids interrupting the application that is being executed. the latency in the proposed scheme is less affected by the participation orders of users and reduces the latency, compared with the conventional scheme.

6. Conclusions

A participating-domain segmentation based server selection scheme was proposed; users participate in server selection incrementally to suppress the latency normally created by the server selection process. The proposed scheme consists of two phases. The first phase is responsible for finding recommended servers, and it determines the recommended server for each region. The second phase is responsible for the server decision process, which selects the server. An ILP problem was formulated for the recommended server selection. Numerical results showed that the proposed scheme offers lower latency than the conventional scheme. The proposed scheme with the second policy, which excludes the regions in which users do not participate, yields lower latency than that of the first policy.

References

[1] R. Mijumbi, J. Serrat, J.L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," IEEE Commun. Surveys Tuts., vol.18, no.1, pp.236–262, 2015.

[2] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee, "Network function virtualization: Challenges and opportunities for innovations," IEEE Commun. Mag., vol.53, no.2, pp.90–97, 2015.

[3] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: State-of-the-art and research challenges," J. Internet Services and Applications, vol.1, no.1, pp.7–18, 2010.

[4] X. Ge, F. Yang, and Q.L. Han, "Distributed networked control systems: A brief overview," Inform. Sciences, vol.380, pp.117–131, 2017.

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

[6] A. Kawabata and E. Oki, "Distributed processing communication scheme for real-time application," IEICE Trans. Commun. (Japanese edition), vol.J99-B, no.4, pp.356–367, April 2016.

[7] 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.

[8] A. Kawabata, B.C. Chatterjee, and E. Oki, "Distributed processing communication scheme for real-time applications considering admissible delay," 2016 IEEE International Workshop Technical Committee on Communications Quality and Reliability (CQR 2016), pp.1–6, IEEE, 2016.

[9] S. Ba, A. Kawabata, B.C. Chatterjee, and E. Oki, "Computational time complexity of allocation problem for distributed servers in real-time applications," 2016 18th Asia-Pacific Network Operations and Management Symposium (APNOMS), pp.1–4, IEEE, 2016.

[10] A. Kawabata, B.C. Chatterjee, and E. Oki, "Participating-domain segmentation based server selection scheme in delay-sensitive distributed communication approach," 2018 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), pp.1–4, IEEE, 2018.

[11] A. Kawabata, B.C. Chatterjee, and E. Oki, "Participating-domain segmentation based delay-sensitive distributed server selection scheme," IEEE Access, vol.7, pp.20689–20697, 2019.

[12] A. Kawabata, B.C. Chatterjee, and E. Oki, "Participating-domain segmentation based server selection scheme in successive participation scenario," IEEE GLOBECOM, IEEE, 2019.

[13] R.M. Fujimoto, Parallel and Distributed Simulation Systems, Wiley New York, 2000.

[14] R.M. Fujimoto, "Research challenges in parallel and distributed simulation," ACM Trans. Model. Comput. Simul. (TOMACS), vol.26, no.4, p.22, 2016.

[15] K. Birman, A. Schiper, and P. Stephenson, "Lightweight causal and atomic group multicast," ACM Trans. Comput Syst., vol.9, no.3, pp.272–314, 1991.

[16] R. Baldoni, S. Cimmino, and C. Marchetti, "A classification of total order specifications and its application to fixed sequencer-based implementations," J. Parallel Distr. Com., vol.66, no.1, pp.108–127, 2006.

[17] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," Proc. first workshop on Hot topics in software defined networks, pp.7–12, ACM, 2012.

[18] Y. Yamato, "Server selection, configuration and reconfiguration technology for iaas cloud with multiple server types," J. Netw. Syst. Manage., vol.26, no.2, pp.339–360, 2018.

[19] T. Ito, N. Kakimura, N. Kamiyama, Y. Kobayashi, Y. Okamoto, and T. Shiitada, "Tight approximability of the server allocation problem for real-time applications," International Workshop on Algorithmic Aspects of Cloud Computing, pp.41–55, Springer, 2017.

[20] P. Garcia Lopez, A. Montresor, D. Epema, A. Datta, T. Higashino, A. Iamnitchi, M. Barcellos, P. Felber, and E. Riviere, "Edge-centric computing: Vision and challenges," ACM SIGCOMM Comput. Commun. Rev., vol.45, no.5, pp.37–42, 2015.

[21] Y.R. Chen, S. Radhakrishnan, S.K. Dhall, and S. Karabuk, "Server selection with delay constraints for online games," 2010 IEEE Globe-

com Workshops, pp.882–887, IEEE, 2010.
[22] Y.R. Chen, S. Radhakrishnan, S.K. Dhall, and S. Karabuk, "On the game server network selection with delay and delay variation constraints," 2011 Third International Conference on Communication Systems and Networks (COMSNETS 2011), pp.1–10, IEEE, 2011.
[23] N. Takahashi, H. Tanaka, and R. Kawamura, "Analysis of process assignment in multi-tier mobile cloud computing and application to edge accelerated Web browsing," 2015 3rd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering, pp.233–234, IEEE, 2015.
[24] V. Toporkov, "Flow and greedy algorithms of resource co-allocation in distributed systems," J. Comput. Syst. Sci. Int., vol.46, no.2, pp.269–278, 2007.
[25] G. Qu, D. Brown, and N. Li, "Distributed greedy algorithm for multi-agent task assignment problem with submodular utility functions," Automatica, vol.105, pp.206–215, 2019.
[26] V. Gupta and G. Tachev, Approximation with Positive Linear Operators and Linear Combinations, Springer, 2017.
[27] "Solving constraint integer programs," http://scip.zib.de, 9 2019.
[28] "Japan photonic network model," https://www.ieice.org/cs/pn/jpn/JPNM/, 9 2019.

Akio Kawabata is an executive research engineer, project manager of Network Service Systems Laboratories in Nippon Telegraph and Telephone Corporation (NTT), Tokyo, Japan. He received the B.E., M.E. and Ph.D. degrees from the University of Electro-Communications, Tokyo, Japan, in 1991, 1993 and 2016, respectively. He is also associated with the Department of Communication Engineering and Informatics at the University of Electro-Communications in Tokyo, Japan for research activities. In 1993, he joined Nippon Telegraph and Telephone Corporation (NTT) Communication Switching Laboratories, Tokyo, Japan, where he has been engaging to develop switching systems, and researching network design and switching system architecture. He served as a senior manager of R&D Department at NTT East since 2011 until 2014.

Bijoy Chand Chatterjee received the Ph.D. degree from the Department of Computer Science and Engineering, Tezpur University, in 2014. From 2014 to 2017, he was a Post-Doctoral Researcher with the Department of Communication Engineering and Informatics, The University of Electro-Communications, Tokyo, Japan, where he was involved in researching and developing high-speed flexible optical backbone networks. From 2017 to 2018, he worked as a DST Inspire Faculty at Indraprastha Institute of Information Technology Delhi (IIITD), New Delhi, India. In 2018, he joined South Asian University, New Delhi, India, where he is currently an Assistant Professor in the Department of Computer science. Before joining South Asian University, he was an ERCIM Postdoctoral Researcher at the Norwegian University of Science and Technology (NTNU), Norway. He has authored over 50 journal/conference papers. His research interests include optical networks, QoS-aware protocols, optimization, and routing. He is a senior member of IEEE and a life member of IETE.

Eiji Oki is a Professor at Kyoto University, Kyoto, Japan. He received the B.E. and M.E. degrees in instrumentation engineering and a Ph.D. degree in electrical engineering from Keio University, Yokohama, Japan, in 1991, 1993, and 1999, respectively. He was with Nippon Telegraph and Telephone Corporation (NTT) Laboratories, Tokyo, from 1993 to 2008, and The University of University of Electro-Communications, Tokyo, from 2008 to 2017. From 2000 to 2001, he was a Visiting Scholar at the Polytechnic Institute of New York University, Brooklyn. He is a Professor at Kyoto University, Kyoto, Japan, from 2017. His research interests include routing, switching, protocols, optimization, and traffic engineering in communication and information networks. He was a recipient of several prestigious awards, including the 1999 IEICE Excellent Paper Award, the 2001 IEEE ComSoc Asia-Pacific Outstanding Young Researcher Award, the 2010 Telecom System Technology Prize by the Telecommunications Advanced Foundation, the IEEE HPSR Paper Awards in 2012, 2014, and 2019, the IEEE Globcom 2015 Best Paper Award, the 2016 Fabio Neri Best Paper Award, Runner up, by Elsevier Optical Switching and Networking, and the 2018 Excellent Paper Award of International Conference on Information and Communication Technology Convergence. He is an IEEE Fellow.