DSND: Delay-Sensitive Network Design Scheme for Multi-Service Slice Networks

Akio Kawabata, *Member, IEEE*, Bijoy Chand Chatterjee, *Senior Member, IEEE*, and Eiji Oki, *Fellow, IEEE*

***Abstract*—This letter proposes a delay-sensitive network design scheme, DSND, for multi-service slice networks. DSND contributes to a virtual processing system where users can share the same application space regardless of distance-related delays. DSND introduces the service slice and service virtual time concepts. A service slice is a virtual network comprising user and server nodes. A service virtual time is a time for eliminating the difference in delay caused by distance, and the user's events are reordered in occurrence order. The difference between the current time and the service virtual time is the end-to-end delay shared by all users within the same service slice. We formulate DSND as an integer linear programming problem and compare the delays between DSND to a benchmark scheme where each user selects the closest server. Numerical results indicate that DSND can reduce the delay by 4–38 percent compared to the benchmark scheme.**

***Index Terms*—Distributed processing, network delay, conservative synchronous algorithm, service slice, and network design.**

I. Introduction

THE provision of Internet of Things (IoT) services, which were previously inaccessible due to limitations in delay quality, will now be possible through the utilization of low-delay networks and the expansion of Edge data Center (EC) locations. Furthermore, there have been recent advancements in the development of network games that incorporate virtual reality technology, allowing multiple players to engage in shared virtual space. A forthcoming processing system will be required to integrate low-delay networks and ECs to provide virtual spaces for various real-time applications.

Dealing with the allocation of application space among multiple users poses a considerable challenge. In the context of real-time applications sharing application space, the discrepancies in delay arising from communication distance can present significant hurdles, particularly for applications whose state changes are contingent upon user events. The research into processing events in the order of their occurrence within distributed systems has been a focal point in both parallel and distributed processing investigations. The resolution to this challenge primarily falls into two categories: conservative and optimistic synchronization [1]. In conservative synchronization, time information is given to events, and the events are rearranged in the order of occurrence before processing the application. In optimistic synchronization, events are processed in the order of arrival, and if past events are received, the status is rolled back, and the processing result is corrected. The work in [2] introduced a network design scheme that effectively eliminates delay discrepancies among users based on conservative synchronization and facilitates sharing a unified application space for all users. The work is grounded in a monolithic service framework and makes no assumptions regarding providing multiple services within a singular network. When network or service providers offer an application processing environment for IoT services, it necessitates using a processing platform that can support multiple services.

Next, we discuss the research concerning processing platforms that can support multiple services for widely distributed users in a network. Various network slicing techniques have been addressed to provide multiple services using 5G networks [3]. The work in [4] presented an approach that significantly improves the utilization of network resources consumed by multiple network slices with the same acceptable delay that can be accommodated in the network. The work in [5] introduced a network design approach for efficiently discovering virtual servers in ECs and configuring a network slice delay-awarely. For application allocation to multiple servers, the work in [6] presented a multi-access edge computing (MEC) location problem enhanced with $1:1$ and $1:N$ protection schemes. The network slicing research in these studies mentions delay reduction or efficient use of resources but does not include the elimination of delay differences between multiple users.

This letter, for the first time, proposes a delay-sensitive network design scheme for multi-service slice networks named DSND. The novelty of the proposed scheme is that it achieves delay control to share the same application space for users over the network slice. The scheme contributes to the processing platforms to provide a virtual space for multiple real-time services. DSND introduces two key concepts: service slice and service virtual time. A service slice is a virtual network consisting of user and virtual server nodes. Service virtual time refers to a shared temporal framework universally applicable to all users. The events generated by users are reordered according to the occurrence order at the virtual time of each service. The disparity between the current and virtual times is represented by the shared end-to-end delay experienced by all users. DSND is formulated as an integer linear programming (ILP) problem. The numerical findings demonstrate that DSND performs better than a benchmark scheme in which each user selects the closest server. This observation holds for two distinct networks, namely NSFnet [7] and COST [8].

II. DSND: Proposed scheme

A. Communication model and service slice

In this section, we describe state synchronization and communication models. In DSND, users select one optimal virtual

Manuscript received 29 February, 2024; revised 2 March, 2024.

This work was supported by ROIS NII Open Collaborative Research 2024-24S0101. Akio Kawabata is with Toyohashi University of Technology, Aichi, Japan (e-mail: kawabata.akio.oc@tut.jp). Bijoy Chand Chatterjee is with South Asian University, New Delhi, India (e-mail: bijoycc@ieee.org). Eiji Oki is with Kyoto University, Kyoto, Japan (e-mail: oki@i.kyoto-u.ac.jp).

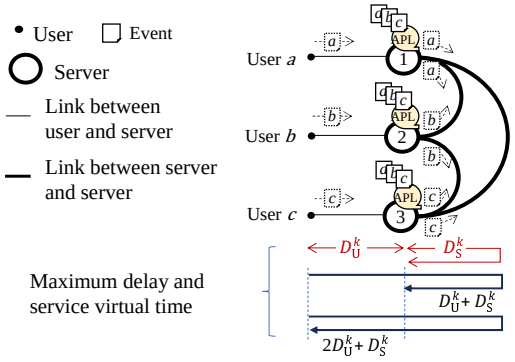

Fig. 1. Example of communication model among servers.

server from multiple candidate servers. Figure 1 illustrates the communication model. Every server broadcasts the accommodated user's events to other servers. The virtual network is referred to as a service slice. The purpose of the service slice design is to establish relationships between users and servers and between servers themselves.

B. Delay synchronization and service virtual time

DSND utilizes a conservative synchronization that reorders events in the order of their occurrence before application processing. Additionally, it employs total ordering for order correction, ensuring that all servers maintain a consistent order [9]. Figure 2 illustrates an instance of reordering events in DSND. This reordering is determined based on the largest value among the delay of all users. We denote the maximum delay between the user and the accommodated server in service k as D_{U}^k and the maximum delay between the servers in service k as D_{S}^k. The maximum delay between the user and all selected servers in service k is $D_{\mathrm{U}}^k + D_{\mathrm{S}}^k$ which is used for reordering the events in service k. The event that experiences the largest delay at Server 3 is associated with User a. This event arrives at Server 3 via Server 1, resulting in a cumulative delay of 0.025 [sec]. If $D_{\mathrm{U}}^k + D_{\mathrm{S}}^k$=0.025 [sec], it is necessary to queue all events before processing them to ensure that the delay for all events is adjusted to 0.025 [sec]. In other words, all events are reordered in the order of their occurrence with a delay of $D_{\mathrm{U}}^k + D_{\mathrm{S}}^k$.

The timing for sending results has been adjusted to guarantee simultaneous receipt of the processed results by all users. The timing is adjusted to ensure that the delay equals the maximum delay, D_{U}^k. In Fig. 2, D_{U}^k is 0.02 [sec], which is the delay between User a and Server 1. The timing is adjusted to arrive at all users with a delay of D_{U}^k=0.02 [sec]. Eventually, as shown in Fig. 1, all users receive the processing results at a time that is delayed by $2D_{\mathrm{U}}^k + D_{\mathrm{S}}^k$ from the current time. All users will use the processing system with the same end-to-end delay (Delay). We name the time of the current time T plus the same delay $2D_{\mathrm{U}}^k + D_{\mathrm{S}}^k$ as the service virtual time. The delay is determined individually for each service slice.

The reordering of events may impose an overhead process. The conservative synchronization method employed in the proposed scheme typically involves inserting a timestamp into the packet and arranging the packets according to the timestamp. In systems that retain time information, retrieving events that correspond to the current time plus the service

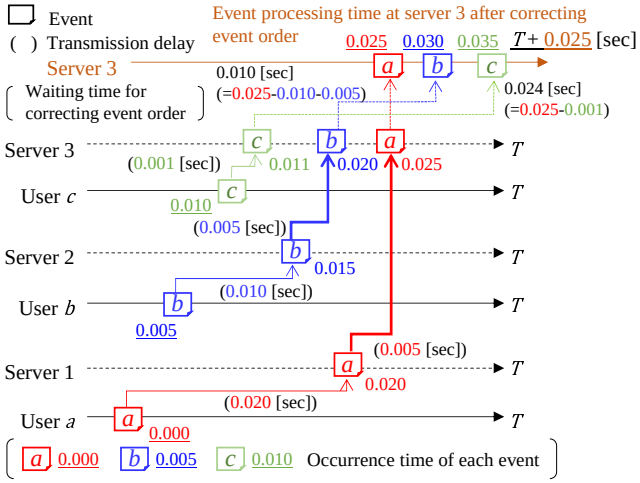

Fig. 2. Example of event reordering.

virtual time becomes feasible, thereby potentially mitigating the complexity associated with event reordering.

C. Determine service slice and service virtual time

In DSND, the configuration of the service slice is designed to minimize the delay. The optimal server for each user is selected among widely distributed ECs. When a terminal interacts with multiple services, numerous client applications run on the terminal, and the network slice attributed to each client application is different. It is assumed that the interconnections among the edge clouds are facilitated by a backbone network with ample circuit capacity. DSND is formulated as ILP to minimize the combined delay of all services and the overall delay of the selected links.

D. Formulation

In DSND, we assume that each user can select servers within all locations and that all servers are connected with a logical connection. For example, if there are two links, Tokyo-Hawaii and Hawaii-California, the logical connection between Tokyo and California is connected via Hawaii.

First, we describe the given parameters of DSND. Consider an undirected graph $G(V, E)$ consisting of a set, V, of nodes and a set, E, of links. Let $p \in V_{\mathrm{U}}$ be the set of users and $i \in V_{\mathrm{S}}$ be the set of servers. It follows that $V_{\mathrm{U}} \cup V_{\mathrm{S}} = V$ and $V_{\mathrm{U}} \cap V_{\mathrm{S}} = \emptyset$. Let $(p, i) \in E_{\mathrm{U}}$ be the set of links between user $p \in V_{\mathrm{U}}$ and server $i \in V_{\mathrm{S}}$ and $(i, j) \in E_{\mathrm{S}}$ be the set of links between server $i \in V_{\mathrm{S}}$ and server $j \in V_{\mathrm{S}}$. It follows that $E_{\mathrm{U}} \cup E_{\mathrm{S}} = E$ and $E_{\mathrm{U}} \cap E_{\mathrm{S}} = \emptyset$. Let d_{pi} be the delay between user $p \in V_{\mathrm{U}}$ and server $i \in V_{\mathrm{S}}$ and d_{ij} be the delay between server $i \in V_{\mathrm{S}}$ and server $j \in V_{\mathrm{S}}$. Let d_{pi} and d_{ij} denote the delay between link $(p, i) \in E_{\mathrm{U}}$ and link $(i, j) \in E_{\mathrm{S}}$. These delays include the processing delay of the network devices and the transmission delay over the optical fiber proportional to the link length. Since the network provides real-time service, sufficient bandwidth is allocated to prevent congestion due to traffic concentration, and there is no variation in delay due to congestion. Since each physical server may have a different processing performance, let M_i be the maximum number of users to accommodate server $i \in V_{\mathrm{S}}$. Let $k \in K$ be the set of services k, and let S_k be the maximum acceptable delay allowed for service k.

Next, we describe the decision variables of DSND. In service k, we introduce x_{pi}^k and x_{ij}^k as variables that indicate whether a link is selected. In service k, $x_{pi}^k = 1$ if the link $(p,i) \in E_{\rm U}$ is selected, otherwise $x_{pi}^k = 0$. In service k, $x_{ij}^k = 1$ if the link $(i,j) \in E_{\rm S}$ is selected, otherwise $x_{ij}^k = 0$. In service k, we introduce y_i^k as a variable that indicates whether a server i is selected. In service k, $y_i^k = 1$ if the server $i \in V_{\rm S}$ is selected, otherwise $y_i^k = 0$. The maximum delay between the user and the accommodated server in service $k \in K$ is $D_{\rm U}^k$ and the maximum delay between the servers in service $k \in K$ is $D_{\rm S}^k$. As described in Section II-B, the delay of service $k \in K$ is $2D_{\rm U}^k + D_{\rm S}^k$, denoted by $D_{\rm vir}^k$.

We formulate DSND as an ILP problem, which is given by:

$$\text{Objective} \quad \min \sum_{k\in K} D_{\rm vir}^k + \alpha \sum_{k\in K} \sum_{(p,i)\in E_{\rm U}} d_{pi} x_{pi}^k \quad (1a)$$

$$\text{s.t.} \quad \sum_{i\in V_{\rm S}} x_{pi}^k = 1, \forall p \in V_{\rm U}, k \in K \quad (1b)$$

$$\sum_{k\in K} \sum_{p\in V_{\rm U}} x_{pi}^k \le M_i, \forall i \in V_{\rm S} \quad (1c)$$

$$D_{\rm S}^k \ge d_{ij} x_{ij}^k, \forall (i,j) \in E_{\rm S}, k \in K \quad (1d)$$

$$D_{\rm U}^k \ge d_{pi} x_{pi}^k, \forall (p,i) \in E_{\rm U}, k \in K \quad (1e)$$

$$D_{\rm vir}^k = 2D_{\rm U}^k + D_{\rm S}^k, \forall k \in K \quad (1f)$$

$$D_{\rm vir}^k \le S^k, \forall k \in K \quad (1g)$$

$$y_i^k \ge x_{ij}^k, \forall i \in V_{\rm S}, (i,j) \in E_{\rm S}, k \in K \quad (1h)$$

$$x_{ij}^k \ge y_i^k + y_j^k - 1, \forall (i,j) \in E_{\rm S}, k \in K \quad (1i)$$

$$x_{ij}^k \le y_i^k, \forall i \in V_{\rm S}, (i,j) \in E_{\rm S}, k \in K \quad (1j)$$

$$x_{ij}^k \le y_j^k, \forall j \in V_{\rm S}, (i,j) \in E_{\rm S}, k \in K. \quad (1k)$$

Equation (1a) indicates the combined value of the end-to-end delay for all services and the total delay of the selected links. The sum of the delays of all services is the first objective function, and the sum of the delays of the selected user-server links is the second objective function. Set α to be $\sum_{k\in K} D_{\rm vir}^k \gg \alpha \sum_{k\in K} \sum_{(p,i)\in E_{\rm U}} d_{pi} x_{pi}^k$. α is a given parameter set to such a sufficiently small value that the decision of the second objective value does not affect that of the first objective value. Equation (1b) indicates that the user selects an optimal server for each service. Equation (1c) indicates that the total number of users that can accommodate server $i \in V_{\rm S}$ is within M_i. Equation (1d) indicates that the maximum delay of the selected link between the user and the accommodated server is $D_{\rm S}^k$ for service $k \in K$. Equation (1e) indicates that the maximum delay of the selected link between servers is $D_{\rm U}^k$ in service $k \in K$. Equation (1f) indicates that the delay in service $k \in K$ is $D_{\rm vir}^k$. Equation (1g) indicates that the delay in service $k \in K$ is within S^k, the maximum delay allowed for that service. Equation (1h) indicates that server $i \in V_{\rm S}$ is selected if the link $(i,j) \in E_{\rm S}$ is selected in service $k \in K$. Equations (1i)–(1k) are linear expressions for $x_{ij} = y_i \times y_j$.

E. Applicable conditions

Delay-sensitive services are usually provided by networks whose delay is guaranteed by service level agreements (SLAs) [10]. Service providers' system resources in network and application services can be controlled to keep the quality of services. In DSND, we assume that an SLA-based network can estimate the delay and the processing performance. However, our considered delay may vary. It includes those mentioned deterministic and variable, i.e., non-deterministic delays; the latter can vary even under a congestion-avoided condition. DSND can handle such a variable delay without explicitly taking it. A network provider conservatively estimates a variable delay as a deterministic value incorporated when it designs the network. This delay is due to the network configuration and cannot be controlled by the user.

For applying DSND for networks where the estimation of the parameters cannot be so precise, if the worst value of each parameter can be calculated, DSND can be applied to the networks. For example, when DSND can be applied to a best-effort network with delay fluctuations, such as the Internet, the virtual time is calculated from the largest delay based on observing delay fluctuations. On the other hand, it does not apply to networks where parameters such as delay and capacity of each server are not estimable.

Note that the proposed scheme offers foundational mechanisms for designing a multi-service slice network. However, its implementation must address numerous challenges, including measuring and updating delays, coordinating the sharing of virtual time among servers, and sorting events at each server.

III. Evaluation

A. Prerequisites and conditions for evaluation

To assess the effectiveness of DSND, we conduct a comparative analysis of the delay between DSND and a benchmark scheme in which each user selects the closest server. The delay of the benchmark scheme can be determined using (2a)–(2b) provided in the Appendix. The delay is assumed to be pre-measured and includes the transmission and equipment delays under the SLA-based network [10].

The assumption is made that the servers and the server-server links are situated at COST [8] and NSFnet [7], while the users are dispersed within the quadrangle area. The distance of each link is calculated as a straight line distance from the latitude and longitude of each city. Figures 3(a) and (b) depict the geographical placements of the users and servers associated with NSFnet and COST. As shown in Fig. 3, the users' location is determined that the users for service 1 are uniformly distributed in the participating area and the users for services 2–5 are locally distributed at the corner in the participating area. Each server is assumed to be connected by the shortest path on the COST and NSFnet links. The distance between the user and the servers is treated as the linear distance between two locations multiplied by 1.5. This assumption is considered because the fact that metro networks are often configured in a ring topology [11], and the distance between two points on the ring is, on average, 1.5 times greater than the distance in a straight line. The delay of links between users and servers and between servers is treated as 1 km equals five microseconds. The ILP problem is solved by CPLEX [12] on the Linux platform using Intel(R) Xeon(R) Gold 6132 CPU 2.60 GHz, 128 GB memory.

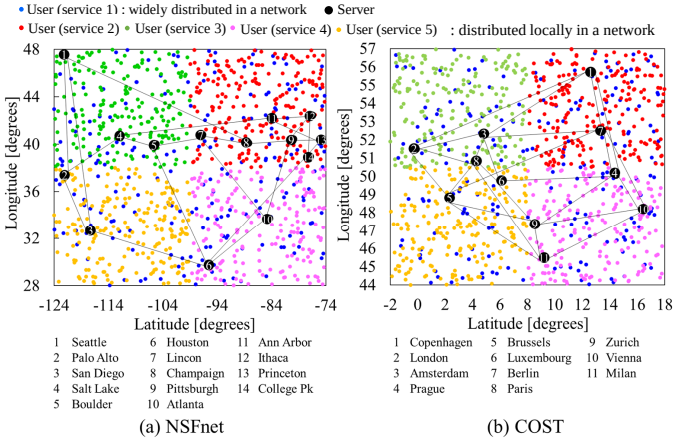

Fig. 3. Location of users and servers.

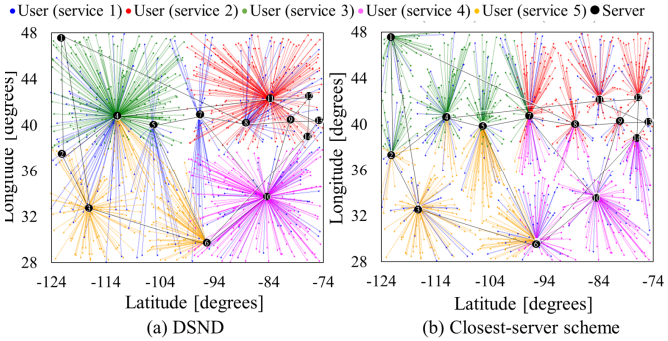

Fig. 4. Selected servers of DSND and benchmark scheme for NSFnet.

B. Server selection and delay characteristics

In this section, we describe the server selection and the delay performance of DSND with no restrictions on server selection based on acceptable delay. Figures 4(a) and (b) depict the links that were selected by individual users in DSND and the benchmark scheme within the NSFnet network. As depicted in Fig. 4(b), the benchmark scheme involves users opting for the server closest to them. As depicted in Fig. 4(a), each user strategically selects an optimal server to avoid selecting long inter-server links in DSND. This results from optimizing server selection so that maximum delay is reduced. Regarding the server selection for each service slice, it can be observed from Fig. 4(a) that the users of service 1, represented by the blue color, distribute themselves across the area and select four servers. The reason is that users of service 1 are widely distributed in the network. In contrast, individuals utilizing services 2–5 in the peripheral region opt for a smaller number of servers to eliminate the escalation of delay among servers. The reason is that users of services 2–5 are located near the corner of the network. In this way, each user selects an optimal server for reducing the delay. The trend in server selection at COST is also the same as at NSFnet.

Figures 5(a) and (b) show the delay for five service slices in NSFnet and COST for DSND and the benchmark scheme. As a reference, the delay of the central processing scheme, in which processing is performed by a single server, is shown by the dotted line. The delay of the central processing scheme is obtained by (3a)–(3b) in the Appendix. The delay for five service slices of DSND is reduced by 9–38 percent for NSFnet compared to that of the benchmark scheme and by 4–36

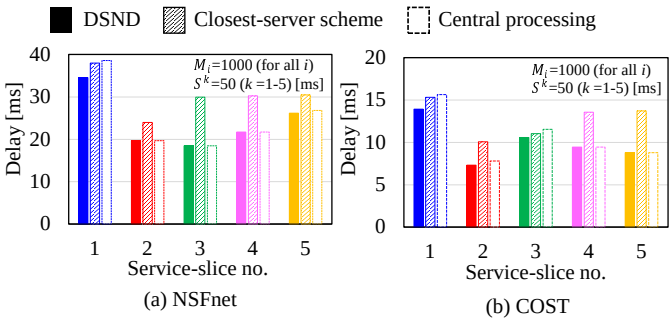

Fig. 5. Delay of DSND and benchmark scheme.

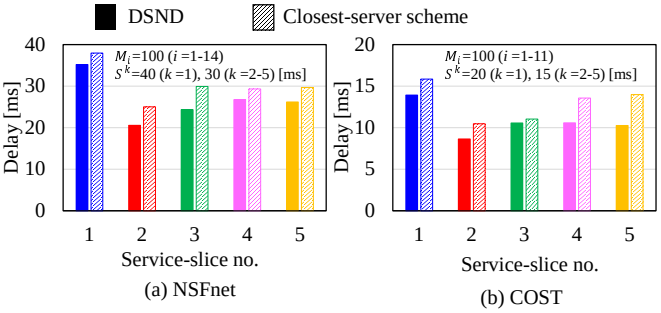

Fig. 6. Delay of DSND and benchmark scheme under constraints.

percent for COST compared to that of the benchmark scheme. In some cases, both NSFnet and COST in Fig. 5, the delay of the benchmark scheme increases more than the delay of the central processing scheme. The weakness of the benchmark scheme lies in the fact that each user selects the closest server irrespective of the delay between servers, which overlooks the influence of both the user-server and server-server links on the delay within a distributed system. From these results, DSND can reduce the delay compared to the benchmark scheme regardless of the network topology (NSF and COST) and whether the user distribution is uniformly (service 1) or concentrated in a specific area (services 2–5). Except for each server's maximum number of users, the server selections are determined without interdependence among each service slice. Therefore, these evaluations are equivalent to several settings of the user's position with statistical validity.

C. Delay performance under constraints

We evaluate the delay under constraints on the number of accommodated users for the server and the acceptable delay for each service. The central processing scheme is not included in this evaluation because it is not feasible unless M_i of a server is 1000 or more users. Figures 6(a) and (b) show the delay of DSND and under the constraints with user and server locations in Fig. 3. The delay for five service slices for DSND is reduced by 7–19 percent for NSFnet compared to that of the benchmark scheme and by 4–27 percent for COST compared to that of the benchmark scheme. The delay reduction for DSND is reduced compared to the results in Fig. 5. This is because the stronger constraint of M_i prevented the selection of servers with a smaller delay. These results indicate that even under conditions where there are constraints with the number of accommodated users for the server and the acceptable delay, the delay reduction effect is reduced, but the delay is reduced more than in the benchmark scheme.

D. Computation time

We evaluate computation time at DSND on the networks with 1000 users. The computation times for the evaluation of NSFnet and COST in Sections III-B (Fig. 5) are 34.0 [sec] and 18.7 [sec], respectively. The computation times for the evaluation of NSFnet and COST in Sections III-C (Fig. 6) are 1618.5 [sec] and 3089.9 [sec], respectively. Each computation time is calculated as the average time over five trials.

From the computational complexity of view, the single-slice version of DSND is NP-complete [2]. We analyze computation time for 100 and 10000 users in addition to 1000 users. M_i is assumed to be the same value as the number of users on all servers. The users are uniformly distributed in the same area in Fig. 5(a). The computation times for the NSFnet under the conditions in Fig. 5(a) for 100 and 10000 (5 slices of 2000 users/slice) are 0.5 and 958.7 [sec], respectively.

E. Discussion

As detailed in Sections III-B and III-C, the proposed scheme outperforms the benchmark scheme in terms of delay performance. The reduction in delay is particularly advantageous for real-time applications, especially considering the stringent requirements of ultra-reliable low-latency communications (URLLC) in 5G networks, where delays are expected to be in the order of single-digit milliseconds [3]. The proposed scheme effectively minimizes the maximum delay per service slice, ensuring that all users experience the same delay as the user with the maximum delay due to the guaranteed event ordering. While this may initially seem like a drawback, as long as the delay experienced by all users falls within the acceptable range for each service, there is no issue in terms of quality of service.

The increase in computation time as the number of users rises is a notable concern. While computation time with limited server resources (M_i constraints) tends to be longer than no constraints, a computation time of 16 minutes (958.7 [sec]) for a 10000-user scale in a virtual private networks service is acceptable for network design processes before service commencement. While several techniques, such as machine learning (ML), offer faster results, the approach employing ILP holds value due to its ability to determine the optimal solution. Considering the above facts, DSND can also be used as a tool for validating the precision of various approaches.

IV. Conclusion and Future Work

In this letter, we proposed a network design scheme for multi-service slice networks, named DSND, to introduce two new concepts: service slice and service virtual time. A service slice is a virtual network of user and server nodes that process the service. In a service slice, events of all users are reordered at the service virtual time in occurrence order before processing by the application. Numerical evaluation indicates that DSND has the potential to apply a practical network design for wide-area distributed systems in terms of computation time (less than one hour) and delay performance for several user patterns in multiple networks.

To apply DSND to networks where the traffic and users dynamically change over time, such as mobile networks or successive participating scenarios with sequential users entering and leaving, issues such as calculating the maximum delay and processing event queuing on the server must be studied further. When a client application on the user terminal is associated with multiple application servers, a potential solution could involve placing these diverse application servers within the same network slice and determining the delay accordingly. However, addressing this aspect is left for future endeavors.

Appendix
Formulation of benchmark and central processing schemes

In the benchmark scheme, where each user selects the closest server, the network design problem is formulated as an ILP problem as follows:

$$\text{Objective} \quad \min \sum_{k\in K} \sum_{(p,i)\in E_{\mathrm{U}}} d_{pi} x_{pi}^{k} \tag{2a}$$

$$\text{s.t.} \quad (1b) - (1k). \tag{2b}$$

In the central processing scheme, each user selects the optimal server under the condition of limiting to one server in each service slice, the network design problem is formulated as an ILP problem as follows, whose objective function is (1a):

$$\text{s.t.} \quad \sum_{i\in V_{\mathrm{U}}} y_i^k = 1, \forall k \in K \tag{3a}$$

$$(1b) - (1k). \tag{3b}$$

References

[1] R. M. Fujimoto, *Parallel and Distributed Simulation Systems*, Wiley-Interscience, 2000.

[2] 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.

[3] S. Wijethilaka and M. Liyanage, "Survey on network slicing for Internet of Things realization in 5G networks," *IEEE Communications Surveys and Tutorials*, vol. 23, no. 2, pp. 957–994, 2021.

[4] I. Kovacevic, A. S. Shafigh, S. Glisic, B. Lorenzo, and E. Hossain, "Multi-domain network slicing with latency equalization," *IEEE Transactions on Network and Service Management*, vol. 17, no. 4, pp. 2182–2196, 2020.

[5] I. Dimolitsas, D. Spatharakis, D. Dechouniotis, and S. Papavassiliou, "A delay-aware approach for distributed embedding towards cross-slice communication" *IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, pp. 13–18, 2022.

[6] H. D. Chantre and N. L. S. da Fonseca, "The location problem for the provisioning of protected slices in NFV-based MEC infrastructure," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 7, pp. 1505–1514, 2020.

[7] K. C. Claffy, G. C. Polyzos, and H. W. Braun, "Traffic characteristics of the T1 NSFNET backbone," *IEEE Infocom'93 The Conference on Computer Communications, Proceedings*, pp. 885–892, 1993.

[8] M. O'Mahony, "Ultrahigh capacity optical transmission network: European research project cost 239," *Information, Telecommunications, Automata Journal*, vol. 12, pp. 33–45, 1993.

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

[10] *Global IP Network.* NTT Communications Corporation. [Online]. Available: https://www.ntt.com/en/services/network/gin/sla.html

[11] S. Matsuoka, "Ultrahigh-speed ultrahigh-capacity transport network technology for cost-effective core and metro networks," *NTT Technical Review*, vol. 9, no. 8, 2011.

[12] *IBM ILOG CPLEX.* IBM. [Online]. Available: https://www.ibm.com/analytics/cplex-optimizer