Accurate Delay Measurement for Parallel Monitoring of Probe Flows

Kohei Watabe*, Shintaro Hirakawa* and Kenji Nakagawa*
*Graduate School of Engineering, Nagaoka University of Technology
Kamitomiokamachi 1603–1, Nagaoka, Niigata, 940-2188 Japan
Email: k_watabe@vos.nagaokaut.ac.jp

***Abstract*—In this paper, we propose an accurate parallel flow monitoring method using active probe packets. Although multiple probe flows are monitored to measure delays on multiple paths in parallel for most measurement applications, information of only one probe flow of the multiple probe flows is utilized to measure an end-to-end delay on a path in conventional active measurement. In addition to information observed by the flow along the path, information of other flows is also utilized for the measurement in the proposed method. Delays on a flow are accurately measured by partially converting the observation results of a flow to those of another flow. Simulations are performed to confirm that the observation results of 72 parallel flows of active measurement are appropriately converted between each other in the proposed method. When the 99th-percentile of an end-to-end delay for each flow are measured, the proposed method achieves up to 95% reduction of the error, and the error of the worst flow among all flows are reduced by 28%.**

I. Introduction

End-to-end metrics are fundamental for a network performance evaluation. Traditionally, end-to-end throughput is one of the most common performance metrics for many applications. Real-time applications, such as audio/video conferencing and IP telephony, require a low delay and stable bandwidth on an end-to-end path, compared to traditional applications such as e-mail, web browsing, etc. For interactive Voice over IP (VoIP) applications, ITU-T Recommendation G.114 [1] mentions that an end-to-end packet delay of more than or equal to 150 [ms] harmfully affects the communication quality. Hence, for both network managers and application developers, an accurate measurement technique for end-to-end metrics is required.

An active measurement in which probe packets are injected into a network for measurement is a common method for end-to-end metrics. Various tools have been proposed that measure end-to-end delays [2], [3], [4], packet losses [5], [6], available bandwidth [7], [8]. Moreover, large-scale measurement in real networks has allowed a better understanding of various characteristics of the networks [9], [10].

In active measurement for end-to-end delays, researchers have tried to achieve accurate measurement without increasing the number of probe packets [11], [12], [13] since a large number of probe packets leads to communication overheads and an intrusiveness problem [14], [15]. Since a large delay (that exceeds 150 [ms] as mentioned in ITU-T Recommendation G.114 [1]) is a rare event in the modern Internet, however, it is difficult to capture a large delay using the limited number of the probe packets on the path. Therefore, high quantile of delay distribution is still hard to measure.

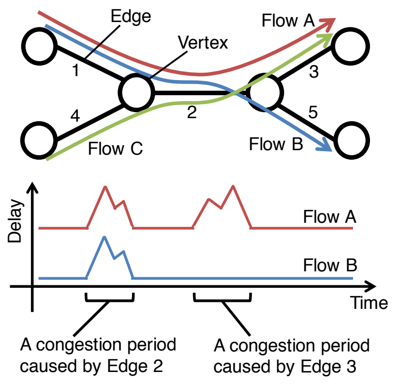
Edge
Flow A
Vertex
1
3
2
4
5
Flow C
Flow B
Delay
Flow A
Flow B
Time
A congestion period caused by Edge 2
A congestion period caused by Edge 3

Fig. 1. Parallel monitoring of multiple flows. The paths of Flow A and B share Edge 1 and 2. If a delay is caused by Edge 2, similar delays are observed in Flow A and B.

Although multiple probe flows are monitored to measure delays on multiple paths in parallel for most measurement applications, only one probe flow of the multiple probe flows is utilized to measure the end-to-end delay on a path. In daily operations, Internet service providers are partly or wholly monitoring end-to-end delays of the paths on their network. Network researchers and practitioners often measure end-to-end delays of multiple paths on a network to clarify the characteristics of the entire network. Parallel monitoring of multiple flows is usual for network monitoring tasks. Since paths of flows share common parts in parallel monitoring, observation results of a flow can include information of delays on another path. In Fig. 1, since both paths of Flow A and B include Edge 2, packets of flow B experience similar delay with those of Flow A when a delay is mainly caused by Edge 2. Therefore, the information concerning Flow B can be utilized supplementary for improving the accuracy of the delay measurement of Flow A.

In this paper, we propose a parallel flow monitoring method in which delays on a flow are accurately measured by partially converting the observation results of another flow into the results of the flow. Congestion periods are taken from each observation result, and they are divided into clusters for each common edge that causes a delay. Observation results of flows in a cluster are converted between each other. A clustering technique in machine learning is utilized to divide them into clusters. The proposed method does not require any internal information of a measured network, including a topology, and it only uses the delay of each flow. We evaluate the proposed method though simulation, and confirm that the proposed method achieves accurate measurement of end-to-end delays in parallel monitoring of probe flows.

The remainder of this paper is organized as follows. Section II explains a network model and several assumptions of the proposed measurement method. In Section III, we summarize a conversion process for parallel flow monitoring and show the algorithm of the proposed method. Section IV explains end-to-end metrics for delay using results of the proposed method. We evaluate the proposed method using simulations in Section V. Section VI summarizes related works. Finally, Section VII concludes the paper and presents future research directions.

II. Network Model and Assumptions

We are interested in measuring end-to-end delays in wired packet networks. A network considered within the scope of this work is represented by a directed graph. An *edge* of a directed graph represents a physical/virtual link and interfaces at both ends of the link. Note that an interface includes input/output packet queue. A *vertex* represents a part of a network device other than its interfaces (e.g., a forwarding element). A *path* is defined as a sequence of vertices and edges. A packet is delivered from a source to a destination along a path. Paths are stable in a measurement period (generally within several minutes) since paths are not changed frequently.

Packets are delayed at vertices or edges on a path. An end-to-end network delay experienced by a packet consists of four elements: propagation delay, queueing delay, transmission delay, and processing delay. Processing delay occurs when a packet is on vertices. The other delays occur on edges. In the modern Internet, propagation delay and queueing delay are dominant, and transmission delay and processing delay are negligible [10]. In this paper, we assume that an end-to-end delay is consisted of propagation delay and queueing delay. Propagation delay can be regarded as a constant for a path while queueing delay dynamically changes reflecting traffic status. Both delays experienced by a packet on a single edge are independent for a path that the packet passes. We assume that edges with large queueing delay are sparse among all edges in a network, and a ratio of periods with large queueing delay on an edge to the other periods is small. The validity of the assumption can be confirmed since the average link utilization of the modern Internet is maintained low [16]. Note that we do not assume a congested edge is unique.

Network researchers or practitioners measure end-to-end delay on each path. To measure delay on paths, probe packets are periodically injected for all or a part of paths on a network. A delay experienced by a probe packet can be obtained from the values of timestamps recorded at the source and the destination. Time synchronization of source and destination devices is required if network researchers or practitioners measure one-way delay. They do not need to know the topology of a network. We first tackle development of a measurement technique without the knowledge of the network topology since it is more applicable, though the proposed method may be accurate utilizing a network topology. Development of a measurement technique with a network topology is left for our future work.

III. Sample Conversion Technique Using Parallel Flow Monitoring

A. Overlap of Queueing Delay Processes

In active measurement for delay of the modern Internet, it is important to sample information regarding congestion periods with large delay since the ratio of the periods to the other periods are extremely small. A delay experienced by a probe packet of Flow A can be regarded as a sample of a virtual delay process $\chi_{\mathrm{A}}(t)$, which is the delay experienced by a virtual packet injected from the source into the path of Flow A at time t. We denote m samples by the probe packets of Flow A as $X_{\mathrm{A}} = \{(t^i_{\mathrm{A}}, x^i_{\mathrm{A}})\,;\, i = 1, \ldots, m\}$, where t^i_{A} and x^i_{A} are the injection time of the ith probe packet of Flow A and the delay observed by the ith probe packet, respectively. Note that x^i_{A} corresponds to $\chi_{\mathrm{A}}(t^i_{\mathrm{A}})$. Since probe packets are injected with constant interval, the number of probe packets injected into a path within congestion periods are few. Although high quantile is a key metric for delay sensitive applications such as VoIP, fewer probe packets within congestion periods lead less accurate measurement for high quantile of end-to-end delay.

Queueing delay processes within a congestion period on multiple paths that have common edges often coincide with each other. A virtual delay process $\chi_{\mathrm{A}}(t)$ is the sum of propagation delay $\bar{\chi}_{\mathrm{A}}$ and queueing delay $\hat{\chi}_{\mathrm{A}}(t)$. A *congestion period* can be defined as a period where a large delay is included in the period and a queueing delay is nonzero. Queueing delay processes $\hat{\chi}_{\mathrm{A}}(t)$ and $\hat{\chi}_{\mathrm{B}}(t)$ tightly overlap if the following three conditions are satisfied:

1) The two paths of Flow A and B have the same source;
2) Routes from the source to the last congested edge on the paths are common like Flow A and B in Fig. 1;
3) A queueing delay that packets experience on edges after the last congested edge can be negligible.

When the above conditions 1) − 3) hold, we see the difference $\chi_{\mathrm{A}}(t) - \chi_{\mathrm{B}}(t)$ is constant $\bar{\chi}_{\mathrm{A}} - \bar{\chi}_{\mathrm{B}}$ in a congestion period since $\hat{\chi}_{\mathrm{A}}(t) = \hat{\chi}_{\mathrm{B}}(t)$. The overlap of queueing delay processes is likely to occur when the sparsity of edges with large queueing delay holds.

B. Conversion Process

If the queueing delay processes $\hat{\chi}_{\mathrm{A}}(t)$ and $\hat{\chi}_{\mathrm{B}}(t)$ tightly overlap, namely the above three conditions hold, samples of these processes can be converted mutually as shown in Fig. 2. It is, however, difficult to discriminate the conditions 2) and

3) without using topology and queue information. We should design a method that uses information from probe packets.

First of all, we show how to detect congestion periods from samples obtained by probe packets. We can estimate the propagation delay $\bar{\chi}_{\mathrm{A}}$ on a path of Flow A by the minimum value $\bar{x}_{\mathrm{A}} \equiv \min_{1 \le i \le m} x_{\mathrm{A}}^i$ since a delay is non-negative and we assume a propagation delay is a constant. The jth congestion period of a Flow A is observed as the jth sequence of x_{A}^i that is larger than $\bar{x}_{\mathrm{A}} + x_{\mathrm{th}}$, where the threshold x_{th} is a control parameter in the proposed method. The start time of the jth congestion period is estimated as the jth injection time among the injection times $\{t_{\mathrm{A}}^i\ ;\ x_{\mathrm{A}}^{i-1} < \bar{x}_{\mathrm{A}} + x_{\mathrm{th}} \vee \bar{x}_{\mathrm{A}} + x_{\mathrm{th}} \le x_{\mathrm{A}}^i\}$. The end times of the jth congestion period is also estimated as the jth injection time among the injection times $\{t_{\mathrm{A}}^i\ ;\ \bar{x}_{\mathrm{A}} + x_{\mathrm{th}} \le x_{\mathrm{A}}^i \vee x_{\mathrm{A}}^{i+1} < \bar{x}_{\mathrm{A}} + x_{\mathrm{th}}\}$.

In the proposed method, congestion periods of two flows whose start and end times are respectively the same, and samples within the congestion periods of each flow are mutually converted. If we can assume that a congested edge is at most one in the entire network, i.e., strong sparsity of congested edges can be assumed, paths with congestion periods that start and end at the same time satisfy the conditions 2) and 3) shown in Section III-A (We will relax the assumption later). We consider that samples $X_{\mathrm{A},j}$ within the jth congestion period of Flow A and samples $X_{\mathrm{B},k}$ within the kth congestion period of B can be converted between each other if the two flows satisfy the following conditions:

i) The two flows have the same source;
ii) The interval between the packet injection times of the first samples in $X_{\mathrm{A},j}$ and $X_{\mathrm{B},k}$ is smaller than δ;
iii) The interval between the packet injection times of the last samples in $X_{\mathrm{A},j}$ and $X_{\mathrm{B},k}$ is smaller than δ.

δ denotes the injection interval of probe packets, and, in the above conditions, it is used to discriminate whether the congestion periods of two flows start/end at the same time. Each sample $(t_{\mathrm{B}}^i, x_{\mathrm{B}}^i)$ in $X_{\mathrm{B},k}$ is converted into a sample of Flow A by $(t_{\mathrm{B}}^i, x_{\mathrm{B}}^i - \bar{x}_{\mathrm{B}} + \bar{x}_{\mathrm{A}})$, since propagation delays of Flow A and B are different even if queueing delay process are tightly overlap.

C. Conversion Process Based on Destination's Time

Discussions similar to Section III-A and III-B can hold when we consider a virtual delay process $\psi_{\mathrm{A}}(t)$ based on destination's time, which is the delay experienced by a packet that reaches the destination at time t. A queueing delay process based on destination's time is also defined by $\hat{\psi}_{\mathrm{A}}(t) = \psi_{\mathrm{A}}(t) - \bar{\chi}_{\mathrm{A}}$. $\chi_{\mathrm{A}}(t)$ and $\psi_{\mathrm{A}}(t)$ can be translated each other since $\chi_{\mathrm{A}}(t) = \psi_{\mathrm{A}}(t + \chi_{\mathrm{A}}(t))$. We also denote m samples of $\psi_{\mathrm{A}}(t)$ on the path of Flow A as $Y_{\mathrm{A}} = \{(u_{\mathrm{A}}^i, x_{\mathrm{A}}^i); i = 1, \ldots, m\}$, where u_{A}^i is the receive time of the ith probe packet at the destination d, and x_{A}^i is the delay observed by the ith probe packet as we defined above.

The conditions for tightly overlapping queueing delay processes $\hat{\psi}_{\mathrm{A}}(t)$ and $\hat{\psi}_{\mathrm{C}}(t)$ are as follows:

1) The two paths of Flow A and C have the same destination;
2) Routes from the first congested edge to the destination on the paths are common like Flow A and C in Fig. 1;
3) A queueing delay that a packet experiences on edges before the first congested edge can be negligible.

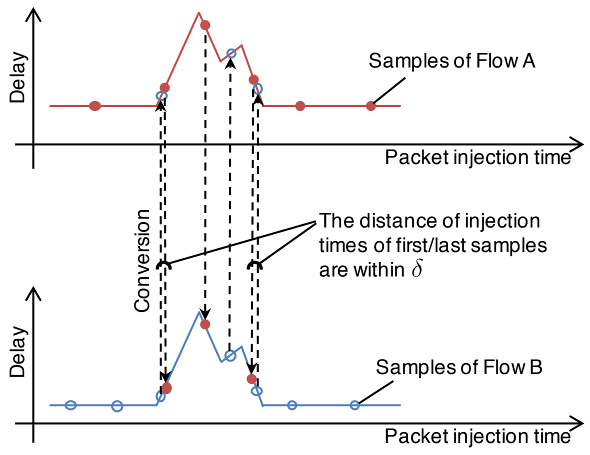

Fig. 2. A conversion process of two flows with congestion periods that started and ended at the same time.

Similarly, by indicating samples within the jth congestion period of Flow A as $Y_{\mathrm{A},j}$, the conditions for discriminating whether the congestion periods of two flows start/end at the same time are as follows:

i) The two flows have the same destination;
ii) The interval between the packet receive times of the first samples in $Y_{\mathrm{A},j}$ and $Y_{\mathrm{C},k}$ is smaller than δ;
iii) The interval between the packet receive times of the last samples in $Y_{\mathrm{A},j}$ and $Y_{\mathrm{C},k}$ is smaller than δ.

Samples within congestion periods that satisfy the above conditions are mutually converted. The converted samples of $\psi_{\mathrm{A}}(t)$ are translated into samples of $\chi_{\mathrm{A}}(t)$ by the equation $\chi_{\mathrm{A}}(t) = \psi_{\mathrm{A}}(t + \chi_{\mathrm{A}}(t))$.

D. Clustering Process

If multiple edges are congested at the same time, the conversion process we shown in Section III-B and III-C may convert inappropriate samples. The samples within congestion periods expected to have the same start and end time are converted in the conversion process. As we mentioned above, the conditions for discriminating whether the congestion periods of two flows start/end at the same time are different from these for tightly overlapping queueing delay processes (the former is shown in Section III-B and the latter is shown in Section III-A). Therefore, even if we convert samples based on the conditions shown in Section III-B, the queueing delay processes behind the samples do not necessarily overlap.

For instance, in Fig. 1, if a congestion period caused by Edge 3 starts and ends within a congestion period caused by Edge 2, we should not convert the samples of Flow B into those of Flow A (see Fig. 3). In this case, since the virtual queueing delay processes for Flow A and C tightly overlap,

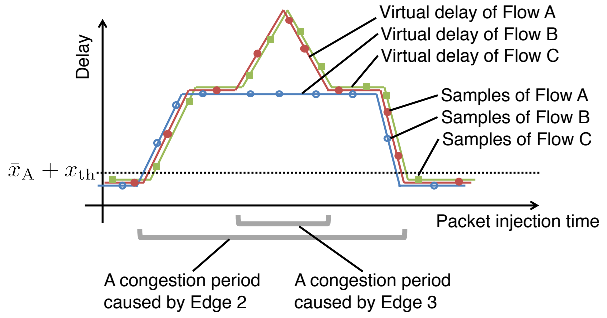

Fig. 3. A case of multiple congested edges. A congestion period caused by Edge 3 starts and ends within a congestion period caused by Edge 2 on a network shown in Fig. 1. The virtual delay processes for Flow A and B do not overlap though those for Flow A and C overlap tightly. The samples of Flow B should not be converted into the samples of Flow A.

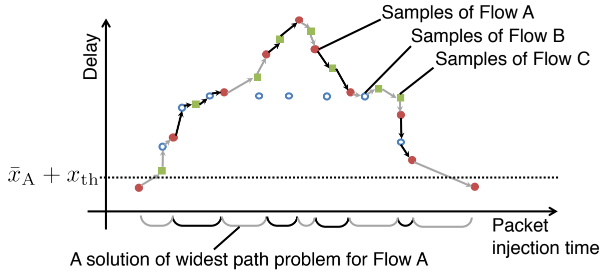

Fig. 4. Solutions of the widest path problem for Flow A. By solving the problem 9 times, the first vertex is connected to the last vertex via all vertices of Flow A.

we can convert the samples of Flow C into those of Flow A. However, the virtual queueing delay processes for Flow A and B do not overlap since packets of Flow B do not experience a delay caused by Edge 3. The conversion process described in the previous sections converts the samples of Flow B into those of Flow A. Hence, we should remove these inappropriately converted samples from samples of Flow A.

To remove inappropriate samples, we utilize a clustering technique in machine learning [17]. Based on samples that are converted, we construct clusters of flows using a clustering technique. In the example of Fig. 3, Flow A and C should be in a cluster, and Flow B should be in another cluster.

Since the number of samples and their intervals vary, we transform samples of each flow into n-dimensional vectors to use general clustering techniques. We let $X_{{\rm A},j}^{{\rm B},k}$ denote the set of the converted samples from the kth congestion period of Flow B into samples of the jth congestion period of Flow A. Let $F_{{\rm A},j}$ be the set of sample sets that are added to the jth congestion period of Flow A, i.e., $F_{{\rm A},j} = \{X_{{\rm A},j}, X_{{\rm A},j}^{{\rm B},k}, X_{{\rm A},j}^{{\rm C},l}, \dots\}$. $\mathcal{F}_{{\rm A},j}$ denotes the set of all samples $\bigcup_{f \in F_{{\rm A},j}} f$ in $F_{{\rm A},j}$. First, we construct a directed graph with vertices $\{(t_{\rm f} - \delta, \bar{x}_{\rm A}), (t_{\rm l} + \delta, \bar{x}_{\rm A})\} \cup \mathcal{F}_{{\rm A},j}$ for each congestion period, where $t_{\rm f}$ and $t_{\rm l}$ denote the first and last injection times in $X_{{\rm A},j}$. In the graph, edges from a vertex $(t_{\rm A}^i, x_{\rm A}^i)$ are toward all vertices with injection times that are larger than $t_{\rm A}^i$. The cost of the edge from $(t_{\rm A}^i, x_{\rm A}^i)$ to $(t_{\rm B}^j, x_{\rm B}^j)$ is set to

$$\frac{1}{\sqrt{\frac{\beta^2}{\delta^2}(t_{\rm A}^i - t_{\rm B}^j)^2 + (x_{\rm A}^i - x_{\rm B}^j)^2}},$$

where β denotes a parameter of the proposed method. β adjusts the ratio of the scale on the horizontal and vertical axes in Fig. 4. For each flow, we search a path from the vertex with the earliest injection time to the vertex with the last injection time via all vertices of the flow (see Fig. 4). The path between vertices of the flow are a solution of the widest path problem [18]. Next, for each flow, we transform the path into an n-dimensional vector by making the vertices evenly spaced. The jth element of the vector is a queueing delay when the injection time is $((j-1)(t_{\rm l} - t_{\rm f}))/(n-1)$ on the path, where n denotes the product of the number $|X_{{\rm A},j}|$ of original samples and multiplicity $|F_{{\rm A},j}|$ of flows. Through the above process, we can express the samples of each flow as an n-dimensional vector.

The proposed method constructs clusters of n-dimensional vectors that represent samples of each flow, and samples that are converted from flows of the different clusters are removed. Prior work on clustering techniques has left us with a rich collection of literature [17]. Among these techniques, we can utilize the techniques that are able to handle high-dimensional vectors and do not have the predefined number of clusters as an input parameter (e.g., DENCLUE [19], G-Means [20], Minimum Entropy Clustering (MEC) [21], etc.). Unfortunately, it is difficult to appropriately divide clusters of flows when the number of samples of each flow is extremely small. When the number of samples in the congestion period per flow is at most 1, the converted samples are removed in the proposed method before the clustering process.

E. Scalability

In this section, we will discuss the scalability of the proposed method. In the conversion process, the proposed method checks whether the start and end times of any pair of congestion periods are respectively the same. The computational complexity of the process is $O(N^2M^2)$, where N and M denote the number of flows and the maximum number of congestion periods of a flow, respectively. The actual converting of samples requires $O(NML)$ operations, where L denotes the maximum number of samples in a congestion period (i.e., $L = \max_{A,j} |F_{{\rm A},j}|$). On the other hand, in the clustering process, the computational complexity of composing n-dimensional vectors are $O(L^3K^3)$, where K denotes the maximum number of flows in an edge. Therefore, the computational complexity of the proposed method other than MEC clustering is $O(N^2M^2 + NML + L^3K^3)$. The average time complexity of each iteration in MEC is usually less than $O(K)$, and the number of iterations is usually less than 20 when $K = 800$ (K in our experience shown in Section V is quite smaller than 800) [21].

F. Limitations

As with all other measurement methods, the proposed method has limitations. This section discusses cases where the proposed method cannot improve accuracy by converting samples (see Fig. 5).

- *Momentary congestion*: The proposal method cannot improve accuracy by converting samples in very short congestion periods. Basically, momentary congestion is hard to detect since the number of probe packets that are included in the period is small. If there is no sample of a flow in the period, samples of the other flows cannot be converted into the samples of the flow. Even if the congestion is detected fortunately, the proposed method does not convert the samples intentionally, as mentioned in Section III-D.
- *Non-sparse congestion*: We have assumed sparsity of congested edges in Section II. If congested edges are not sparse, i.e., queueing delays are always high on most edges, the proposed method cannot detect start and end times of a congestion period. Even if the start and end times are detected, queueing delay processes of flows are not overlapped since the processes are flow specific.
- *Complex routes*: The proposed method cannot convert samples between flows that are once forked and rejoined. The condition 2) in Section III-A for overlapping queueing delay processes requires that the paths are completely common from the source to the last congested edge. Since flows that are once forked and rejoined violate the condition, samples in a congestion period that occurs on an edge after a rejoin cannot be converted between each other.

Since the proposed method cannot convert samples in the above three cases, the result approaches to that of the conventional method. The limitations do not mean that the proposed method can be inaccurate compared with the conventional method.

IV. End-to-end Metrics for Parallel Flow Monitoring

The proposed method increases the number of samples of a virtual delay process, and these samples can be utilized for various metrics regarding end-to-end delay. Most of measurement approaches based on active measurements can be jointly used the proposed method since the proposed method simply adds samples in active measurements. The samples by the proposed method is not uniformly distributed in the time space since samples are added in congestion periods. Hence, it is needed to weight samples depending on multiplicity of flows. In this section, we give examples of mean delay and q-quantile measurement.

A weight of a sample is determined by multiplicity of flows in the congestion period that contains the sample. The weight

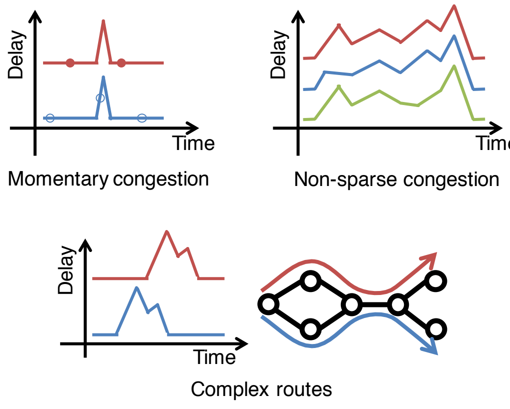

Fig. 5. Cases where the proposed method cannot improve accuracy by converting samples. (Top left) Samples in momentary congestion are difficult to convert since momentary congestion are hard to detect. (Top right) Queueing delay processes caused by non-sparse congestion edges are hard to overlap each other. Therefore, since flows are divided into different clusters, the samples are not converted. (Bottom) If there are multiple routes between source and a congested edge, the queueing delay processes do not overlap since propagation delays before the congested edge differ. Hence samples cannot be converted between each other.

of sample s is given as follows:

$$w_s = \begin{cases} \dfrac{|\mathcal{F}_{\mathrm{A},j}|}{|F_{\mathrm{A},j}|} & s \in \mathcal{F}_{\mathrm{A},j} \qquad (j = 1, 2, \dots), \\ 1 & \text{otherwise.} \end{cases}$$

Note that $F_{\mathrm{A},j}$ is the set of sample sets that added to the jth congestion period of Flow A and $\mathcal{F}_{\mathrm{A},j} = \bigcup_{f \in F_{\mathrm{A},j}} f$, as we defined in Section III-D.

If we want to measure the mean delay on the path of Flow A, it is measured by

$$\frac{1}{|X_{\mathrm{A}}|} \sum_{s \in X_{\mathrm{A}} \cup \mathcal{F}_{\mathrm{A},*}} w_s d_s,$$

where d_s and $\mathcal{F}_{\mathrm{A},*}$ denote delay obtained from sample s and all samples $\bigcup_j \mathcal{F}_{\mathrm{A},j}$ in all congestion periods, respectively. For q-quantile measurement, we first calculate the following:

$$k = \arg\max_j \left\{ \sum_{i=1}^{j} w_{s_i} \leq q|X_{\mathrm{A}}| \right\},$$

where s_i denotes the sample whose delay is the ith smallest among all samples $X_{\mathrm{A}} \cup \mathcal{F}_{\mathrm{A},*}$. Then, q-quantile of end-to-end delay is estimated by $d(s_k)$.

Our estimators are natural extensions of the conventional estimators [12]. The conventional estimators for the mean and q-quantile of end-to-end delay are defined respectively as

$$\frac{1}{|X_{\mathrm{A}}|} \sum_{s \in X_{\mathrm{A}}} d_s,$$

and

$$d\left(s'_{\lfloor q|X_{\mathrm{A}}| \rfloor}\right),$$

where s'_i denotes the sample whose delay is the ith smallest of all samples X_{A}.

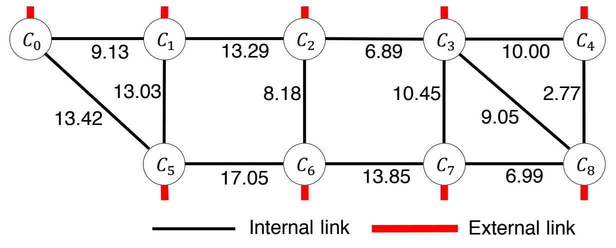

Fig. 6. A simulation topology. The numerical values beside the links indicate propagation delays in ms.

V. Experiments

We perform NS-3 [22] simulations to confirm that samples of parallel flows of active measurement are appropriately converted between each other in the proposed method. The topology in our simulations that is shown in Fig. 6 resembles Internet2 topology [23]. There are 9 nodes, and they are connected by physical links whose capacity are 15.552 [Mbps]. The numerical values beside the links in the Fig. 6 indicate propagation delay, and we set them proportional to the distance between the nodes in Internet2. While we have also performed simulation on several simple topologies and similar results are obtained from all simulations, the results are not shown due to the lack of space.

The traffic in our simulation is categorized into 3 types that are listed in Table I. These 3 types of traffic stream between all pairs of 9 nodes (i.e. 72 flows in the entire network). Phases of packet injection are randomized while probe packets are injected periodically. Since the link capacity is uniformly 15.552 [Mbps], traffic intensity on a link temporally exceeds the link capacity if two or more flows of burst traffic are joined at the link. Since the maximum queue size is set significantly large, a buffer overflow does not occur though this temporal capacity shortage causes queueing delays. The simulation time is 42 [s] and we only use the data from 20 [s] to 42 [s].

The parameters of the proposed method are set as follows. The threshold x_{th} and parameter β are set to 0.01 [s] and 0.16, respectively. We use MEC for clustering, and its parameters α in α-entropy and the expected number e of clusters are set to 0.001 and 10, respectively. Although we have tried using DENCLUE and G-Means, the same result as that of the conventional method is obtained since all flows are divided into different clusters. This is because DENCLUE cannot appropriately estimate the density of data due to the small number of flows. On the other hand, G-Means assumes that Gaussian distribution of data, though our data do not follow Gaussian distribution.

To confirm that the proposed method appropriately converts samples of the other flows into that of a flow, we depict examples of samples by the conventional and the proposed method in Fig. 7. The examples shown in Fig. 7 are the samples of Flow ID 5-3 and 6-4. In the figure, we only depict a period (from 37.0 [s] to 41.0 [s] for Flow ID 5-3 and from 20.0 [s] to 22.0 [s] for Flow ID 6-4) when one of

TABLE I
Types of traffic in our simulation.

Stationary	Packet size	600 [Byte]
	Traffic pattern	Poisson arrivals
	Traffic intensity	388.8 [Kbps] (4% of a link capacity)
Burst	Packet size	500 [Byte]
	Traffic pattern	On/off process with periodic arrivals in burst periods
	Traffic intensity	8,000 [Kbps] in burst periods
		0 [bps] in idle periods
	Burst period	Exponential distribution with mean 0.1 [s]
	Idle period	Exponential distribution with mean 0.4 [s]
Probe	Packet size	74 [Byte]
	Traffic pattern	Periodic arrivals
	Packet intervals δ	200 [ms]

large delays is observed. We can confirm that the number of samples of the proposed method is larger than those of the conventional method, and the samples tightly approximate the virtual delay. Removed samples in the clustering process are indicated by the green plus marker, and most of them are not on the virtual delay. Hence, it is confirmed that the clustering process removed inappropriate samples.

We also evaluate the accuracy of the proposed method when the 99th-percentile of end-to-end delay is measured. The simulation is repeated 10 times by changing the phase of the probe packet injection time. The true value of 99th-percentile and the number of the converted samples are displayed in Fig. 8 (Top) and (Middle), respectively. We display only flows whose 99th-percentile of delay exceeds 100 [ms]. Flow ID s-d in the figure is composed of source s and destination d of the flow. The number of original samples that are obtained from probe packets is 110 samples, and they are not included in Fig. 8 (Middle). Similarly, the samples that are removed in the clustering process are not included in the figure. Up to 78 samples are converted into samples of a flow, and it is confirmed that the number of the converted samples tends to be larger if a flow has large delay. Root Mean Squared Errors (RMSE) of 99th-percentile measurement of end-to-end delay are calculated, and the result is shown in Fig. 8 (Bottom). The error bars represent 95% confidence intervals. The proposed method provides up to 95% reduction of RMSE (the maximum reduction rate for flows is achieved at Flow ID 8-2). Additionally, the RMSE reduction rate of the worst flow is reduced by 28% (The ratio of RMSE of Flow ID 0-7 in the proposed method to that of Flow ID 5-3 in the conventional method). Compared with the conventional method, nearly equivalent or higher accuracy is achieved for most of the flows. The flows whose RMSE are almost 0 did not experience large delay.

To verify the effect of probe packet intervals δ on the

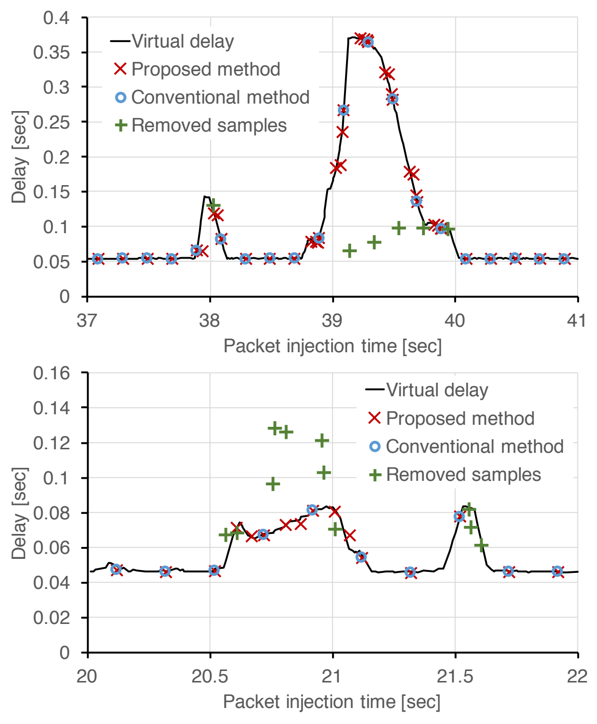
Fig. 7. Samples of the proposed method and the conventional method. While the number of samples in the conventional method is small, the number of samples in the proposed method is larger. Samples that are not on the virtual delay are appropriately removed by the clustering technique. (Top) Flow ID 5-3. (Bottom) Flow ID 6-4.

performance of the proposed method, we compare RMSE of 99th-percentile of end-to-end delay by changing probe packet interval δ from 0.1 to 1.0 [s]. Except for the probe packet intervals, the simulation settings are unchanged from the settings shown in Table I. The results are shown in Fig. 9. It is confirmed that the values of maximum RMSE reduction rate for flows are high, though RMSE reduction rate of the worst flow and median of RMSE reduction rate occasionally decrease to nearly 0.

Next, to confirm the dependency of RMSE on parameters of MEC, we calculate the maximum RMSE reduction rate for flows, the RMSE reduction rate of the worst flow, and median of RMSE reduction rate by changing the parameters. The probe packet intervals δ is set to 0.2 [s], and the other parameters except for α-entropy and the expected number e of clusters are not changed from the first experiment whose results are shown in Fig. 8 in this section. The results when we change α from 10^{-5} to 10^{-1} are shown in Fig. 10 (e is set to 10). It is confirmed that the RMSE is independent of α parameter from the figure. The results when we change e from 1 to 25 are shown in Fig. 11 (the parameter α is set to 0.001). The results are unchanged unless e is set to 1. $e = 1$ means that vectors are not divided into clusters in the clustering process, and that is an unreasonable value for the expected number of clusters.

Finally, we verify the dependency of the performance of

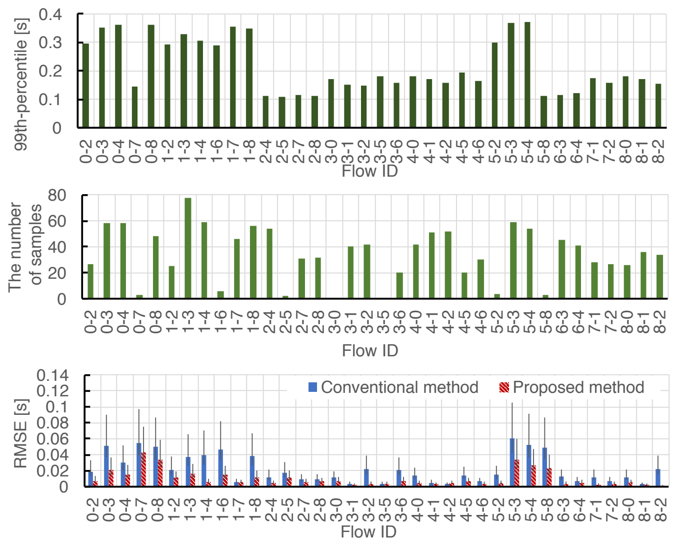
Fig. 8. (Top) The true values of 99th-percentile of delay. Only flows whose 99th-percentile of delay exceeds 100 [ms] are displayed. (Middle) The number of the converted samples. Note that the number of the converted samples does not include the original samples of the flow. The number of the converted samples tends to be larger if a flow has large delay. (Bottom) RMSE of the 99th-percentile measurement for each flow. The error bars represent 95% confidence intervals. The proposed method provides up to 95% reduction of RMSE, and nearly equivalent or higher accuracy is achieved for most of the flows.

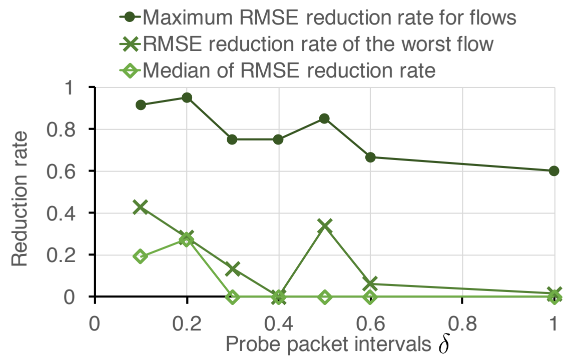
Fig. 9. Dependency of RMSE on probe packet intervals δ. The proposed method maintains high values of maximum RMSE reduction rate for flows, although RMSE reduction rate of the worst flow and median of RMSE reduction rate decrease to nearly 0 on occasion.

the proposed method on parameter β. We calculate RMSE of end-to-end delay by changing β from 0.005 to 2.56. Except of parameter β, the parameters in the experiment are not changed from the first experiment in this section. The result of the experiment is shown in Fig. 12. The maximum RMSE reduction rate for flows increases as parameter β increases, and stabilizes after β becomes 0.16. Although the figure indicates that larger β is better for the maximum RMSE reduction rate for flows, we have observed an inappropriate conversion of samples when β is between 0.64 and 2.56. The RMSE reduction rate of the worst flow and median of RMSE reduction rate decrease due to the inappropriate conversions.

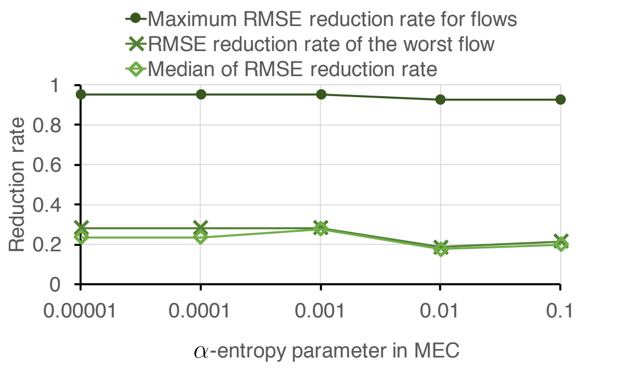

Fig. 10. Independence of RMSE on α-entropy parameter of MEC. The maximum RMSE reduction rate for flows is stable between 92%–95% for all α from 0.0001 to 0.1. Similary, the other indexes are almost unchanged for all α.

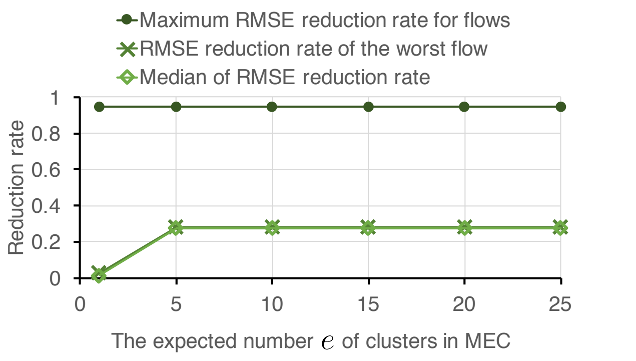

Fig. 11. Dependency of RMSE on the expected number e of clusters of MEC. The maximum RMSE reduction rate for flows is stable at 95% for all e. The other indexes are almost unchanged for all e except for $e = 1$.

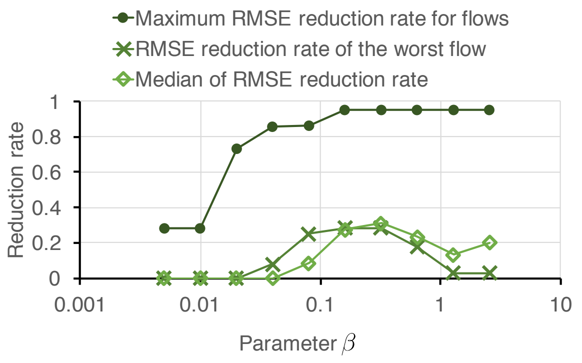

Fig. 12. Dependency of RMSE on parameter β. The performance of the proposed method is equivalant to or better than that of the conventional method for various β though sufficien improvement in accuracy cannot be obtained for small β.

Therefore, the values between 0.08 and 0.32 are good values of β.

VI. Related Works

There is a rich collection of literature that aims at measuring end-to-end delays [2], [3], [11], [12], [4], [15]. Some prior works [11], [12] have tried to estimate high quantile of end-to-end delays by active measurement. Choi *et al.* [11] has proposed a scheme that estimates high quantile with bounded errors. The scheme allows us to know the minimum number of probe packets needed to bound the error of quantile estimation within a prescribed accuracy. Sommers *et al.* [12] also have proposed an estimator of high quantile. Since the estimator provides confidence intervals, we can tune the number of probe packets to achieve the required accuracy. Unlike the proposed method, these prior works utilize only a single flow for an end-to-end delay on a path.

The effect of probe packets on the path quality have been also studied [24], [14], [25], [26], [15]. References [24], [25] have shown that an arrival process of the probe packets affects accuracy of end-to-end delay/loss measurement. Degradation of measurement accuracy caused by probe traffic load have been studied in references [14], [26], [15]. The limitation of a single flow measurement can be understood by these works.

VII. Conclusion

In this paper, we proposed a parallel flow monitoring method that achieves accurate measurement by partially converting the observation results each other. The proposed method adds to samples of a flow from the samples of the other flows, and removes inappropriate samples using a clustering technique in machine learning. We demonstrated that the proposed method can properly add and remove samples through simulations. When the 99th-percentile of end-to-end delay is measured, the proposed method provides up to 95% reduction of errors, and the error of the worst flow among all flows is reduced by 28%.

In future work, we will evaluate our method using real network traffic. Additionally, we will develop a method that utilizes a network topology for the conversion process. By using the knowledge of the topology, the samples that are not added in the proposed method can be added. Moreover, our method can be extended to loss measurement.

Acknowledgment

This work was partly supported by the Telecommunications Advancement Foundation, the Japan Prize Foundation, and JSPS KAKENHI Grant Number JP17K00008.

References

[1] "One-way Transmission Time," *ITU-T Recommendation G.114*, May 2003.
[2] J.-C. Bolot, "Characterizing End-to-End Packet Delay and Loss in the Internet," *Journal of High Speed Networks*, vol. 2, no. 3, pp. 289–298, 1993.
[3] K. P. Gummadi, S. Saroiu, and S. D. Gribble, "King: Estimating Latency between Arbitrary Internet End Hosts," in *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurment (IMW 2002)*, Marseille, France, Nov. 2002, pp. 5–18.
[4] L. De Vito, S. Rapuano, and L. Tomaciello, "One-way Delay Measurement: State of the Art," *IEEE Transactions on Instrumentation and Measurement*, vol. 57, no. 12, pp. 2742–2750, Dec. 2008.

[5] J. Sommers, P. Barford, N. Duffield, and A. Ron, "Improving Accuracy in End-to-End Packet Loss Measurement," *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 4, pp. 157–168, Oct. 2005.

[6] F. Baccelli, S. Machiraju, D. Veitch, and J. Bolot, "Probing for Loss: the Case against Probe Trains," *IEEE Communications Letters*, vol. 15, no. 5, pp. 590–592, Mar. 2011.

[7] J. Strauss, D. Katabi, and F. Kaashoek, "A Measurement Study of Available Bandwidth Estimation Tools," in *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement (IMC 2003)*, Miami, FL, USA, Oct. 2003, pp. 39–44.

[8] V. J. Ribeiro, R. H. Riedi, R. G. Baraniuk, J. Navratil, and L. Cottrell, "pathChirp: Efficient Available Bandwidth Estimation for Network Paths," in *Proceedings of the 4th Passive and Active Measurement Conference (PAM 2003) Workshop*, San Diego, CA, USA, Apr. 2003.

[9] M. Yajnik, S. Moon, J. Kurose, and D. Towsley, "Measurement and Modelling of the Temporal Dependence in Packet Loss," in *Proceedings of the 18th IEEE International Conference on Computer Communication (INFOCOM 1999)*, New York, NY, USA, Mar. 1999, pp. 345–352.

[10] H. Pucha, Y. Zhang, Z. M. Mao, and Y. C. Hu, "Understanding Network Delay Changes Caused by Routing Events," *ACM SIGMETRICS Performance Evaluation Review*, vol. 35, no. 1, p. 73, Jun. 2007.

[11] B.-Y. Choi, S. Moon, R. Cruz, Z.-L. Zhang, and C. Diot, "Quantile Sampling for Practical Delay Monitoring in Internet Backbone Networks," *Computer Networks*, vol. 51, no. 10, pp. 2701–2716, Jul. 2007.

[12] J. Sommers, P. Barford, N. Duffield, and A. Ron, "Accurate and Efficient SLA Compliance Monitoring," *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 4, pp. 109–120, Oct. 2007.

[13] F. Baccelli, S. Machiraju, D. Veitch, and J. Bolot, "On Optimal Probing for Delay and Loss Measurement," in *Proceedings of the 7th ACM Conference on Internet Measurement (IMC 2007)*, San Diego, CA, USA, Oct. 2007, pp. 291–302.

[14] M. Roughan, "Fundamental Bounds on the Accuracy of Network Performance Measurements," *ACM SIGMETRICS Performance Evaluation Review*, vol. 33, no. 1, pp. 253–264, Jun. 2005.

[15] K. Watabe and K. Nakagawa, "Packet Delay Estimation that Transcends a Fundamental Accuracy Bound due to Bias in Active Measurements," *to appare in IEICE Transactions on Communications*, vol. E100-B, no. 8, Aug. 2017.

[16] "CAIDA: The Cooperative Association for Internet Data Analysis." [Online]. Available: http://www.caida.org/

[17] A. Fahad, N. Alshatri, Z. Tari, A. Alamri, I. Khalil, A. Y. Zomaya, S. Foufou, and A. Bouras, "A Survey of Clustering Algorithms for Big Data: Taxonomy and Empirical Analysis," *IEEE Transactions on Emerging Topics in Computing*, vol. 2, no. 3, pp. 267–279, Sep. 2014.

[18] Z. Wang and J. Crowcroft, "Bandwidth-delay Based Routing Algorithms," in *Proceedings of 1995 IEEE Global Telecommunications Conference (GLOBECOM 1995)*, Singapore, Nov. 1995, pp. 2129–2133.

[19] A. Hinneburg and D. Keim, "DENCLUE : An efficient approach to clustering in large multimedia databases with noise," in *Proceedings of 4th International Conference on Knowledge Discovery and Data Mining (KDD 1998)*, New York, NY, USA, Sep. 1998, pp. 58–65.

[20] G. Hamerly and C. Elkan, "Learning the k in k-means," in *Proceedings of Advances in Neural Information Processing Systems 16 (NIPS 2003)*, 2003.

[21] H. Li, K. Zhang, and T. Jiang, "Minimum Entropy Clustering and Applications to Gene Expression Analysis." in *Proceedings of 2004 IEEE Computational Systems Bioinformatics Conference (CSB 2004)*, Stanford, CA, USA, Aug. 2004, pp. 142–151.

[22] T. R. Henderson, M. Lacage, G. F. Riley, G. Dowell, and J. B. Kopena, "Network Simulations with the ns-3 Simulator," in *Proceedings of ACM SIGCOMM 2008*, Seattle, WA, USA, Aug. 2008, p. 527.

[23] "Internet2 Network NOC." [Online]. Available: https://globalnoc.iu.edu/i2network/

[24] F. Baccelli, S. Machiraju, D. Veitch, and J. Bolot, "The Role of PASTA in Network Measurement," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 4, pp. 231–242, Oct. 2006.

[25] K. Watabe and M. Aida, "Analysis on the Fluctuation Magnitude in Probe Interval for Active Measurement," in *Proceedings of the 30th IEEE International Conference on Computer Communication (INFOCOM 2011) Mini-Conference*, Shanghai, China, Apr. 2011, pp. 161–165.

[26] K. Watabe and K. Nakagawa, "Intrusiveness-aware Estimation for High Quantiles of a Packet Delay Distribution," in *Proceedings of 2015 IEEE International Conference on Communications (ICC 2015)*, London, UK, Jun. 2015, pp. 7787–7792.