IEICE TRANSACTIONS

on Communications

VOL. E102-B NO. 4
APRIL 2019

A PUBLICATION OF THE COMMUNICATIONS SOCIETY

EiC

The Institute of Electronics, Information and Communication Engineers
Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3chome, Minato-ku, TOKYO, 105-0011 JAPAN

PAPER

A Parallel Flow Monitoring Technique That Achieves Accurate Delay Measurement*

Kohei WATABE[†a)], ***Member***, **Shintaro HIRAKAWA**[†], ***Student Member***, ***and*** **Kenji NAKAGAWA**[†], ***Member***

SUMMARY In this paper, a parallel flow monitoring technique that achieves accurate measurement of end-to-end delay of networks is proposed. In network monitoring tasks, network researchers and practitioners usually monitor multiple probe flows to measure delays on multiple paths in parallel. However, when they measure an end-to-end delay on a path, information of flows except for the flow along the path is not utilized in the conventional method. Generally, paths of flows share common parts in parallel monitoring. In the proposed method, information of flows on paths that share common parts, utilizes to measure delay on a path by partially converting the observation results of a flow to those of another flow. We perform simulations to confirm that the observation results of 72 parallel flows of active measurement are appropriately converted between each other. When the 99th-percentile of the end-to-end delay for each flow are measured, the accuracy of the proposed method is doubled compared with the conventional method.

key words: *active measurement, parallel monitoring, probe packets, delay measurement, end-to-end delay*

1. Introduction

In performance evaluation of networks, it is important to accurately measure end-to-end metrics. Service Level Agreements (SLAs) that detail the contractual obligations between Internet Service Providers (ISPs) and users define criteria of end-to-end packet loss and delay. ISPs are forced to comply with SLAs, and validate it. Besides, it is well known that real-time applications, e.g., audio/video conferencing, IP telephony, or telesurgery are sensitive to end-to-end packet loss or delay. Although a loss monitoring depends on a Management Information Base (MIB) is commonly used in current SLAs, advanced SLAs in which metrics along an end-to-end path is guaranteed are needed for delay/loss sensitive applications. Hence, the accurate measurement of end-to-end metrics is a key technology of SLA validation, in order to strictly guarantee low end-to-end packet loss or delay for delay/loss sensitive applications. Especially, a high-quantile measurement of delay distribution is important since large delay critically affects the performance of delay/loss sensitive applications even if the probability of occurrence is small.

An active delay measurement in which probe packets are injected into a network for measurement is a common method, and various measurement techniques for active delay measurements have been proposed in prior works. Researchers have tried to achieve accurate measurement without increasing the number of probe packets [1]–[3] since a large number of probe packets leads to communication overheads and the intrusiveness problem [4], [5]. Since a large delay (that exceeds 150 [ms] as mentioned in ITU-T Recommendation G.114 [6]) 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.

Although multiple probe flows are monitored to measure delays on multiple paths in parallel for most measurement applications, only one probe flow among those 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.

By supplementarily utilize information of the other flows, we can achieve a more accurate measurement while maintaining the load of the probe flows in the entire system. Large number of probe packets leads the increase of the load of a network. We cannot limitlessly increase the probe rate in order to improve accuracy. Therefore, it is essential to im-

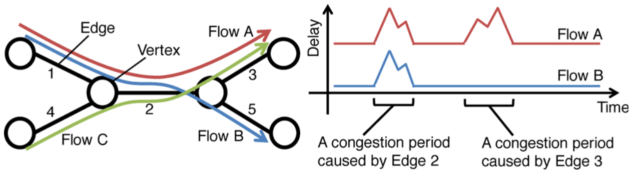

Edge
Flow A
Vertex
1
3
2
4
5
Flow C
Flow B
Delay
Flow A
Flow B
Time
A congestion period caused by Edge 2
A congestion period caused by Edge 3

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.

Manuscript received May 29, 2018.
Manuscript revised August 3, 2018.
Manuscript publicized October 3, 2018.

[†] The authors are with the Graduate School of Engineering, Nagaoka University of Technology, Nagaoka-shi, 940-2188 Japan.

[*] An earlier version of this paper was presented at the 13th International Conference on Network and Service Management (CNSM 2017).

a) E-mail: k_watabe@vos.nagaokaut.ac.jp

DOI: 10.1587/transcom.2018EBP3155

prove accuracy of delay measurement without increasing the number of probe packets since an accurate measurement is a key technology of SLA validation as we mentioned above.

We have proposed a parallel flow monitoring method in which delays on a flow are accurately measured by partially converting the observation results of other flows into the results of the flow [7]. In this method, congestion periods are taken from each observation result, and they are divided into clusters for each common edge that causes a large 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 method does not require any internal information of a measured network, including a topology, and it only uses the delay of each flow. Note that the method does not assume that all possible paths in a network are simultaneously monitored. It can appropriately perform when only a part of paths in a network are monitored. The method in Ref. [7] has as many as 5 parameters: probe packet interval δ, delay threshold x_{th}, radius parameter r, the number e of initial clusters , and edge weight parameter β. Moreover, only for a part of the parameters, the dependence of the performance on the parameters is discussed in Ref. [7].

In this paper, we modify the parallel flow monitoring method in order to reduce the number of the parameters, and provide comprehensive evaluation of it. The proposed method is independent from the number e of initial clusters and edge weight parameter β. Evaluations regarding all other parameters are provided to help parameter tuning in practical measurements. Additionally, we verify the relation between information volume and accuracy improvement. The evaluation of the proposed method is based on simulations, and we 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 2 explains a network model and several assumptions of the proposed measurement method. In Sect. 3, we summarize a conversion process for parallel flow monitoring and show the algorithm of the proposed method. Section 4 explains end-to-end metrics for delay using results of the proposed method. We evaluate the proposed method using simulations in Sect. 5. Section 6 summarizes related works. Finally, Sect. 7 concludes the paper and presents future research directions.

2. 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 [8]. 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 assumed to be 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 [9]. 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 improved by utilizing a network topology. Development of a measurement technique with a network topology is left for our future work.

3. Sample Conversion Technique Using Parallel Flow Monitoring

3.1 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 $\chi_{\text{A}}(t)$, 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 $X_{\text{A}} = \{(t^i_{\text{A}}, x^i_{\text{A}})\ ;\ i = 1, \ldots, m\}$, where t^i_{A} is the injection time of the ith probe packet of Flow A and x^i_{A} is the delay observed by the ith probe packet. Note that x^i_{A} corresponds to $\chi_{\text{A}}(t^i_{\text{A}})$. 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. Note that the proposed method does not require the satisfaction of the above three conditions for all congestion periods. If queueing delay processes within a part of congestion periods satisfy the conditions, the proposed method can work for these congestion periods. Since we assume that edges with large queueing delay are sparse among all edges, most (but not all) congestion periods approximately satisfy the conditions.

3.2 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. 3. It seems difficult to discriminate the conditions 2) and 3) without using topology and queue information, however we will design a method that uses only information from probe packets without the knowledge of topology.

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 $\bar{x}_{\mathrm{A}} \equiv \min_{1\leq i\leq m} x^i_{\mathrm{A}}$ since a delay is non-negative and we assume a propagation delay is a constant. A congestion period is observed as consecutive samples that are larger than $\bar{x}_{\mathrm{A}} + x_{\mathrm{th}}$, where the threshold x_{th} is a control parameter in the proposed method (See Fig. 2). The start time of the jth congestion period is estimated as the jth injection time among the injection times $\{t^i_{\mathrm{A}}\ ;\ x^{i-1}_{\mathrm{A}} < \bar{x}_{\mathrm{A}} + x_{\mathrm{th}}$ and $\bar{x}_{\mathrm{A}} + x_{\mathrm{th}} \leq x^i_{\mathrm{A}}\}$. The end times of the jth congestion period is also estimated as the jth injection time among the injection times $\{t^i_{\mathrm{A}}\ ;\ \bar{x}_{\mathrm{A}} + x_{\mathrm{th}} \leq x^i_{\mathrm{A}}$ and $x^{i+1}_{\mathrm{A}} < \bar{x}_{\mathrm{A}} + x_{\mathrm{th}}\}$.

In the proposed method, when congestion periods of two flows whose start and end times are respectively the same, the samples within the congestion periods of each flow are mutually converted. If we can assume that the number of congested edges 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 Sect. 3.1 (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:

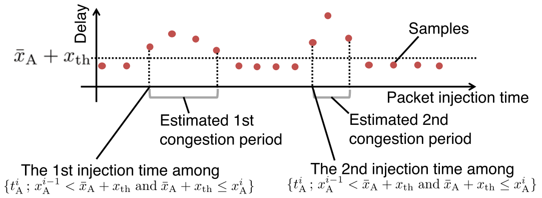


Fig. 2 Estimation of congestion periods with samples. The threshold $\bar{x}_{\mathrm{A}} + x_{\mathrm{th}}$ is used to estimate start and end times of a congestion period.

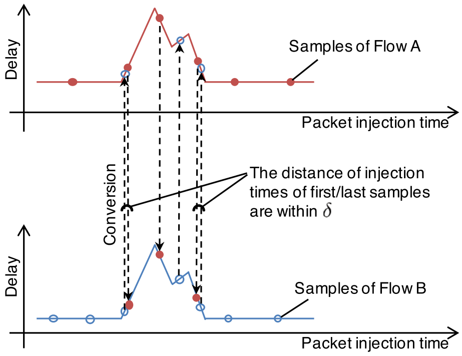


Fig. 3 A conversion process of two flows with congestion periods that started and ended at the same time.

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 (We assume that the injection intervals of all probe flows are the same). Each sample $(t^i_{\mathrm{B}}, x^i_{\mathrm{B}})$ in $X_{\mathrm{B},k}$ is converted into a sample of Flow A by $(t^i_{\mathrm{B}}, x^i_{\mathrm{B}} - \bar{x}_{\mathrm{B}} + \bar{x}_{\mathrm{A}})$, since propagation delays of Flow A and B are different even if queueing delay process are tightly overlap.

3.3 Conversion Process Based on Destination's Time

Discussions similar to Sects. 3.1 and 3.2 can be applied 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 $\hat{\psi}_{\mathrm{A}}(t) = \psi_{\mathrm{A}}(t) - \bar{x}_{\mathrm{A}}$. $\chi_{\mathrm{A}}(t)$ 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 $Y_{\mathrm{A}} = \{(u^i_{\mathrm{A}}, x^i_{\mathrm{A}}); i = 1, \ldots, m\}$, where u^i_{A} is the receive time of the ith probe packet at the destination d, and x^i_{A} 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.

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 i) – iii) are mutually converted. The converted samples of $\psi_{\mathrm{A}}(t)$ are translated into samples of $\chi_{\mathrm{A}}(t)$ by the equation $\chi_{\mathrm{A}}(t) = \psi_{\mathrm{A}}(t + \chi_{\mathrm{A}}(t))$.

3.4 Clustering Process

If multiple edges are congested at the same time, the conversion process we shown in Sects. 3.2 and 3.3 may convert inappropriate samples. The samples within congestion periods should 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 Sect. 3.2 and the latter is shown in Sect. 3.1). Therefore, even if we convert samples based on the conditions shown in Sect. 3.2, 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. 4). In this case, since the virtual queueing delay processes for Flow A and C tightly overlap, 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 not convert these inappropriate samples from samples of Flow A.

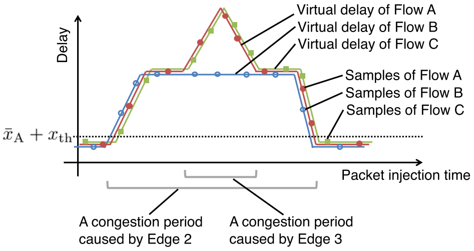


Fig. 4 A case of multiple congested edges.

To remove inappropriate samples, we utilize a clustering technique in machine learning [10]. Based on samples that are converted, we construct clusters of flows using a clustering technique. In the example of Fig. 4, 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 vectors with the same dimension to use general clustering techniques. We let $X^{\mathrm{B},k}_{\mathrm{A},j}$ 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 $F_{\mathrm{A},j}$ be the set of sample sets that are added to the jth congestion period of Flow A, i.e., $F_{\mathrm{A},j} = \{X_{\mathrm{A},j}, X^{\mathrm{B},k}_{\mathrm{A},j}, X^{\mathrm{C},l}_{\mathrm{A},j}, \ldots\}$. $\mathcal{F}_{\mathrm{A},j}$ denotes the set of all samples $\bigcup_{f \in F_{\mathrm{A},j}} f$ in $F_{\mathrm{A},j}$. First, we construct a directed graph with vertices $\{(t_{\mathrm{f}} - \delta, \bar{x}_{\mathrm{A}}), (t_{\mathrm{l}} + \delta, \bar{x}_{\mathrm{A}})\} \cup \mathcal{F}_{\mathrm{A},j}$ for each congestion period, where t_{f} and t_{l} denote the first and last injection times in $X_{\mathrm{A},j}$. In the graph, edges from a vertex $(t^i_{\mathrm{A}}, x^i_{\mathrm{A}})$ are toward all vertices with injection times that are larger than t^i_{A}. The cost of the edge from $(t^i_{\mathrm{A}}, x^i_{\mathrm{A}})$ to $(t^j_{\mathrm{B}}, x^j_{\mathrm{B}})$ is set to

$$\frac{1}{\sqrt{\frac{1}{\delta^2}(t^i_{\mathrm{A}} - t^j_{\mathrm{B}})^2 + \frac{1}{\sum_{x^i_{\mathrm{A}} \in X_{\mathrm{A},j}} |x^i_{\mathrm{A}} - x^{i-1}_{\mathrm{A}}| / |X_{\mathrm{A},j}|}(x^i_{\mathrm{A}} - x^j_{\mathrm{B}})^2}}. \quad (1)$$

In Ref. [7], the cost of the edge is defined as a function of parameter β. Parameter β balances the scale of vertical and horizontal axes in Fig. 5. It is reported that the accuracy of the end-to-end delay measurement may degrade markedly depending on a value of β. In Eq. (1), $t^i_{\mathrm{A}} - t^j_{\mathrm{B}}$ is normalized by probe packet interval δ, and $x^i_{\mathrm{A}} - x^j_{\mathrm{B}}$ is also normalized by the mean intervals of x^i_{A}, thereby it does not include parameter β. 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. 5). The path between vertices of the flow are a solution of the widest path problem (WPP) [11]. In the case of Fig. 5, by solving the problem 9 times for 9 intervals of 10 vertices of Flow A, the first vertex is connected to the last vertex via all vertices of Flow A. Next, for each flow, we transform the path into an n-dimensional vector by making the vertices evenly spaced. The lth element of the vector is a queueing delay when the injection time is $((l - 1)(t_{\mathrm{l}} - t_{\mathrm{f}}))/(n - 1)$ on the path, where n denotes the product of the number $|X_{\mathrm{A},j}|$ of original samples and multiplicity $|F_{\mathrm{A},j}|$ of flows. Through the above process, we can express the samples of each flow

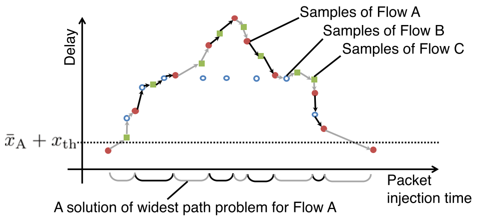


Fig. 5 Solutions of the widest path problem for Flow A.

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 [10]. 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 [12], G-Means [13], Minimum Entropy Clustering (MEC) [14], 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.

The pseudo code of the clustering process for the jth congestion period of Flow A is shown in Algorithm 1. In the pseudo code, we apply MEC clustering to our clustering process though we can apply various clustering techniques as we mentioned above. First, a path P from $(t_\mathrm{f} - \delta, \bar{x}_\mathrm{A})$ to $(t_\mathrm{l} + \delta, \bar{x}_\mathrm{A})$ via all samples in $X_{\mathrm{A},j}$ is calculated by solving WPP (Lines 3-6). Based on a path P, n-dimensional vector $\boldsymbol{v}_X$ is calculated, and $\boldsymbol{v}_X$ is added to V (Lines 7-11). In our method, we give $|X_{\mathrm{A},j}| \cdot |F_{\mathrm{A},j}|$ as n. MEC clustering divides the set of vectors V related to sample sets into clusters (Lines 12-19). In MEC clustering, samples are divided into e clusters by k-means clustering, and it reassigns samples into another cluster in order to minimize an entropy criterion. The number e of initial clusters is a parameter in MEC, and the number of clusters can be reduced by reassigning samples into another cluster. In our method, we achieve to reduce the number of parameters by dynamically setting e to $|V|$ for each congestion period. Lastly, samples that are assigned to all clusters except for the cluster of $\boldsymbol{v}_{X_{\mathrm{A},j}}$ related to the original samples are removed (Line 20).

Algorithm 1: The pseudocode of the clustering process for the jth congestion period of Flow A

```
   Input: $F_{\mathrm{A},j} = \{X_{\mathrm{A},j}, X^{\mathrm{B},k}_{\mathrm{A},j}, X^{\mathrm{C},l}_{\mathrm{A},j}, \dots\}$, $\delta$, $\bar{X}_\mathrm{A}$, $n$
1  $V \leftarrow \emptyset$
2  foreach $X \in F_{\mathrm{A},j}$ do
3      $P \leftarrow [\,]$
4      $X' \leftarrow \{(t_\mathrm{f} - \delta, \bar{x}_\mathrm{A})\} \cup X \cup \{(t_\mathrm{l} + \delta, \bar{x}_\mathrm{A})\}$
5      foreach adjacent sample pair $(s, s')$ in $X'$ do
6          push the solution of WPP from $s'$ to $s$ into $P$
7      for $l \leftarrow 1$ to $n$ do
8          $t = ((l-1)(t_\mathrm{l} - t_\mathrm{f}))/(n-1)$
9          find $(t, x)$ on a line from $P[i]$ to $P[i+1]$
10         ($l$th element of vector $\boldsymbol{v}_X$) $\leftarrow x$
11     $V \leftarrow V \cup \{\boldsymbol{v}_X\}$
12 construct initial $|F_{\mathrm{A},j}|$ clusters by $k$-means algorithm
13 repeat
14     foreach $\boldsymbol{v} \in V$ do
15         $c \leftarrow$ cluster containing most of $\boldsymbol{v}$'s neighbors
16         if $c$ is not current cluster $c_{\boldsymbol{v}}$ of $\boldsymbol{v}$ then
17             if entropy $h$ is reduced when $\boldsymbol{v}$ is in $c$ then
18                 asign $\boldsymbol{v}$ to $c$
19 until no change;
20 remove samples not related to vectors in cluster of $\boldsymbol{v}_{X_{\mathrm{A},j}}$
```

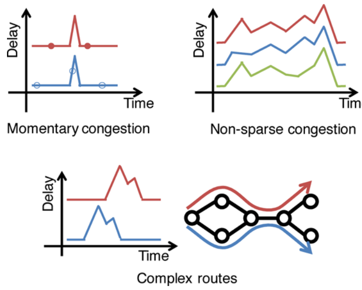
3.5 Scalability

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 denotes the number of flows and M denotes the maximum number of congestion periods of a flow. The actual converting of samples requires $O(NML)$ operations, where L denotes the maximum number of samples in a congestion period (i.e., $L = \max_{A,j} |F_{\mathrm{A},j}|$). 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 Sect. 5 is quite smaller than 800) [14].

3.6 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. 6).

- *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 Sect. 3.4.
- *Non-sparse congestion*: We have assumed sparsity of congested edges in Sect. 2. If congested edges are not sparse, i.e., queueing delays are always high on most



Momentary congestion

Non-sparse congestion

Complex routes

Fig. 6 Cases where the proposed method cannot improve accuracy by converting samples.

edges, the proposed method cannot detect start and end times of a congestion period. Even if the start and end times are detected, it is hard to divide clusters in the clustering process since queueing delay processes are different for each flow and they are not overlapped.

- *Complex routes*: The proposed method cannot convert samples between flows that are once forked and rejoined. The condition 2) in Sect. 3.1 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. Of course, a manager of a measurement may not know the existence of such flows since we do not assume the knowledge of the network topology. Even if the manager does not know the topology, the samples in such flows are not converted each other due to difference of propagation delay. Hence, the performance of the proposed method will not be worse than that of the conventional method, though accuracy improvement is not expected by conversions of samples between such flows.

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.

4. 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 with 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.

The conventional estimators [2] for end-to-end delay are defined as follows. The mean delay is defined as the arithmetic mean of delay d_s of a sample s,

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

Recall that X_{A} is the set of original samples, as we defined in Sect. 3.1. For q-quantile of end-to-end delay,

$$k = \arg\max_{j}\{j \le q|X_{\mathrm{A}}|\} = \lfloor q|X_{\mathrm{A}}|\rfloor, \tag{2}$$

is calculated, and q-quantile is estimated by the kth smallest delay among all samples X_{A}.

Our estimators are natural extensions of the conventional estimators. A weight of a sample is determined by multiplicity of flows in the congestion period that contains the sample. The weight of sample s is given as follows:

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

Recall that $\mathcal{F}_{\mathrm{A},j}$ is the set of the all samples including converted samples in jth congestion period of Flow A, and $X_{\mathrm{A},j}$ is the set of samples of Flow A in the jth congestion period (the definitions are given in Sect. 3.2 and 3.4).

w_s represents the ratio of the number of all samples in congestion period j to the number of the original samples in the period.

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 $\mathcal{F}_{\mathrm{A},*}$ denotes all samples $\bigcup_j \mathcal{F}_{\mathrm{A},j}$ in all congestion periods. For q-quantile measurement, we first calculate the following:

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

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 the kth smallest delay.

5. Experiments

5.1 Simulation Settings

We perform NS-3 [15] 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. 7 resembles Internet2 topology [16]. There are 9 nodes, and they are connected by physical links whose capacity are 15.552 [Mbps]. The numerical values written beside the links in the Fig. 7 indicate propagation delay, and we set

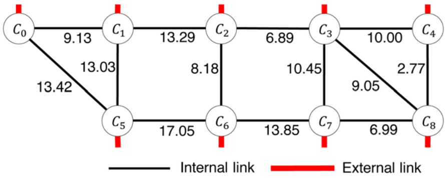


Fig. 7 A simulation topology. The numerical values beside the links indicate propagation delays in ms.

Table 1 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)
Bursty	Packet size	500 [Byte]
	Traffic pattern	On/off process with periodic arrivals in bursty periods
	Traffic intensity	8,000 [Kbps] in bursty periods
		0 [bps] in idle periods
	Bursty period	Exponential distribution with mean 0.1 [s]
	Idle period	Exponential distribution with mean 4.0 [s]
Probe	Packet size	74 [Byte]
	Traffic pattern	Periodic arrivals
	Packet intervals δ	200 [ms]

them proportional to the distance between the nodes in Internet2.

The traffic in our simulation is categorized into 3 types that are listed in Table 1. These 3 types of traffic stream between all pairs of 9 nodes (i.e. $9 \times 8 = 72$ probe flows in the entire network). Phases of packet injection are randomized while probe packets are injected periodically. The probe packets are commonly used for the proposed method and the conventional method. The estimators explained in Sect. 4 are used for each method. 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 bursty traffic are joined at the link. Though the congested links are sparse, congestions on multiple links can occur at the same time. 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} is set to 0.01 [s]. We use MEC for clustering, and its radius parameter r is set to 0.1. 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.

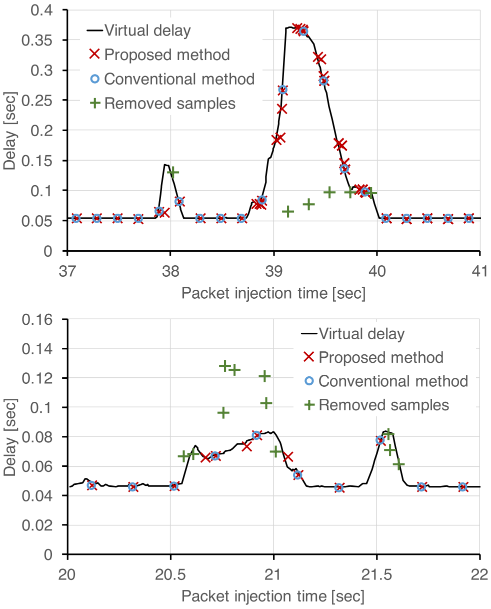


Fig. 8 Samples of the proposed method and the conventional method.

5.2 Accuracy Improvement of the Proposed Method

To confirm that the proposed method appropriately converts samples of the other flows into those of a flow, we depict examples of samples by the conventional and the proposed method in Fig. 8. The examples shown in Fig. 8 are the samples of Flow ID 5-3 and 6-4. Flow ID s-d is composed of source s and destination d of the flow. 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 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. The few samples of the conventional method can approximate the virtual delay. However, the proposed method can capture the change of the virtual delay in more detail by a large number of the samples. Therefore, the proposed method is expected that the metrics regarding the distribution of the virtual delay such as q-percentile can be accurately measured. 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 is displayed in Fig. 9 (Top), and the number of the converted samples is displayed in Fig. 9 (Middle). We display only flows whose 99th-percentile of delay exceeds 100 [ms]. The number of original samples that are obtained from probe packets is 110 samples, and they are not included in Fig. 9 (Middle). Similarly, the samples that are removed in the clustering process are not included in the figure. Up

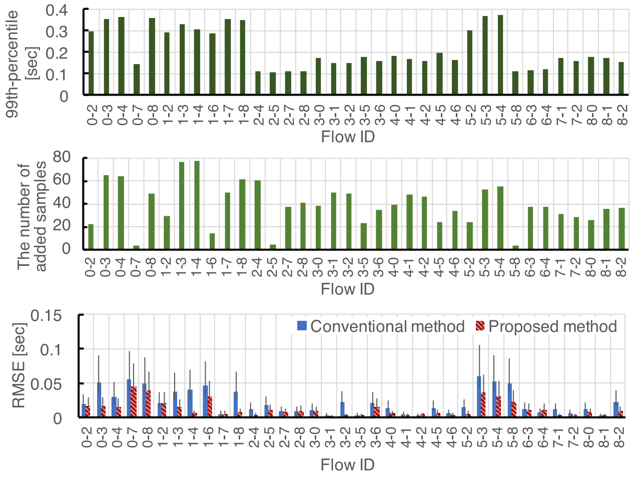


Fig. 9 (Top) The true values of 99th-percentile of delay. (Middle) The number of the converted samples. (Bottom) RMSE of the 99th-percentile measurement for each flow.

to 51 samples are converted into samples of a flow. Root Mean Squared Errors (RMSE) of 99th-percentile measurement of end-to-end delay are calculated, and the result is shown in Fig. 9 (Bottom). The error bars represent 95% confidence intervals. The proposed method provides up to 86% reduction of RMSE (the maximum reduction rate for flows is achieved at Flow ID 1-4). Additionally, the RMSE reduction rate of the worst flow is reduced by 25% (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, higher of nearly equivalent accuracy is achieved for most of the flows.

We also verified the estimators of q-quantile, for various q. For $q > 0.82$, we got similar results with the above results for $q = 0.99$, and the proposed method corresponds to the conventional method when $q \leq 0.82$. As we mentioned in the introduction, high-quantile estimation for end-to-end delay is important though it is hard to measure. From the results, the proposed method achieves good performance for a high-quantile estimation that we are interested in.

5.3 Dependency of Accuracy on a Probe Packet Interval

To verify the effect of probe packet interval δ on the performance of the proposed method, we compare RMSE of 99th-percentile of end-to-end delay by changing δ from 0.1 to 0.8 [s]. The other simulation settings except for the probe packet intervals are unchanged from the settings shown in Table 1. The results for flows whose 99th-percentile of delay exceeds 100 [ms] are shown in Fig. 10. It is confirmed that the proposed method outperforms the conventional method for most of δ values. The result of the average reduction rate of RMSE means that the proposed method achieves double accuracy without increasing the number of probe packets.

5.4 Dependency of Accuracy on a Parameter of MEC

Next, we will confirm the dependency of RMSE on the parameter of MEC. By changing the parameter r, we calculate the maximum RMSE reduction rate for flows, the RMSE reduction rate of the worst flow, and the average reduction rate of RMSE for flows whose 99th-percentile of delay exceeds 100 [ms]. The probe packet intervals δ is set to 0.2 [s], and the other parameters except for radius parameter r is not changed from the first experiment whose results are shown in Fig. 9 in this section. The results when we change r from 10^{-5} to 1.0 are shown in Fig. 11. We can confirm that the proposed method achieves good performance for 10^{-5} to 10^{-1}.

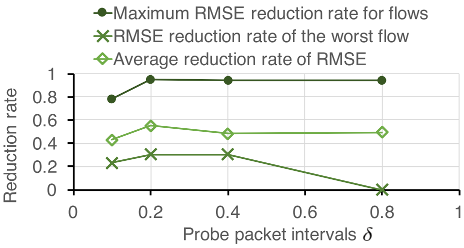


Fig. 10 Dependency of RMSE on probe packet intervals δ. The proposed method significantly reduces the values of RMSE.

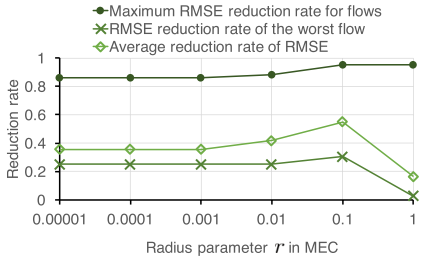


Fig. 11 Dependency of RMSE on radius parameter r of MEC. The reduction rate of RMSE is remarkable for r from 0.00001 to 0.1.

5.5 Dependency of Accuracy on a Parameter of Delay Threshold

In order to verify the dependency of the threshold x_{th} on the performance of the proposed method, we compare RMSE of 99th-percentile of end-to-end delay by changing the threshold x_{th} from 0.001 to 1.0. The other simulation settings except for the threshold are unchanged from the settings shown in Table 1. The results for flows whose 99th-percentile of delay exceeds 100 [ms] are shown in Fig. 12. We can confirm that the reduction rate converges to 0 and the proposed method corresponds to the conventional method when the threshold becomes a large value. The maximum queueing delay is 317 [ms] in our simulation, and the proposed method works effectively if the threshold x_{th} is smaller than the maximum queueing delay. Roughly, one-hundredth of the expected maximum queueing delay is recommended as x_{th} since the reduction rate for $x_{\text{th}} \leq 0.01$ is stable in Fig. 12.

5.6 Effect of a Edge Weight Parameter

We compare the proposed method that is independent from

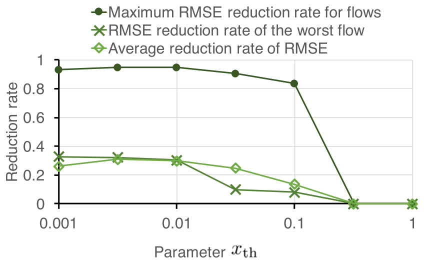


Fig. 12 Dependency of RMSE on threshold parameter $x_{\rm th}$.

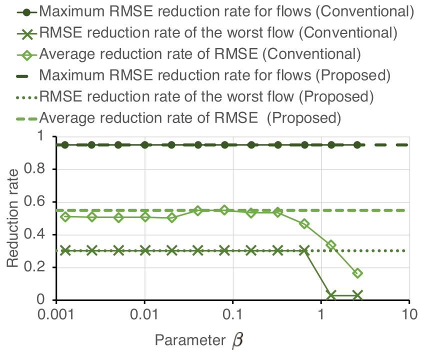


Fig. 13 Effect of edge weight parameter β for radius parameter r = 0.1.

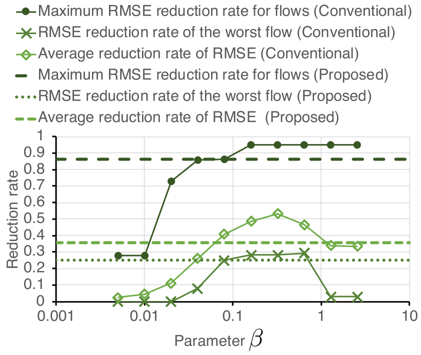


Fig. 14 Effect of edge weight parameter β for radius parameter r = 0.001.

edge weight parameter β with the method in Ref. [7]. One of the remaining issue of Ref. [7] is that the performance of the end-to-end measurement highly depends on edge weight parameter β, and it does not include how to tune the parameter. The reduction rate of 99th-percentile based on the conventional method is compared between the proposed method and the method in Ref. [7] when β is varied. The number e of initial clusters in the method in Ref. [7] is set to 10. We perform the simulation for r = 0.1 and 0.001. The other simulation settings except for the radius parameter r, e, and β are unchanged from the settings shown in Table 1. The results for r = 0.1 and r = 0.001 are shown in Figs. 13 and 14, respectively. Unfortunately, the proposed method cannot achieve equivalent performance with the best performance of the method in Ref. [7] with the best parameter (β = 0.32) since the automatic parameter tuning in the proposed method is not functioning perfectly. However, we can confirm that the proposed method can avoid the marked degradation of the performance, though the method in Ref. [7] is often close to the conventional method when β is too high/low.

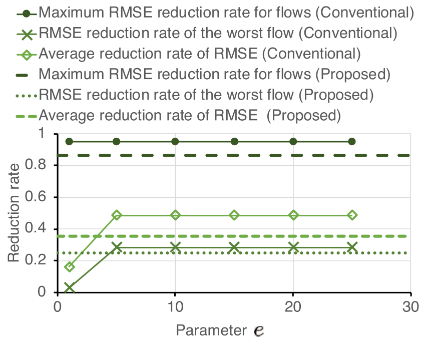


Fig. 15 Effect of the number e of initial clusters.

5.7 Effect of the Number of Initial Clusters

We compare the proposed method that is independent from the number e of initial clusters with the method in Ref. [7]. The reduction rate of 99th-percentile based on the conventional method is compared between the proposed method and the method in Ref. [7] when e is varied. Parameter β in the method in Ref. [7] is set to 0.16. The other simulation settings except for the radius parameter e and β are unchanged from the settings shown in Table 1. The results are shown in Fig. 15, and we can confirm that 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 initial number of clusters. Since the parameter β is tuned to a suitable value, the method in Ref. [7] shows better performance than the proposed method. However, we can confirm that the proposed method can avoid the marked degradation of the performance like the method in Ref. [7] with e = 1.

5.8 The Relation between Information Volume and Accuracy Improvement

Finally, we verify the relation between information volume and accuracy improvement of the proposed method. As we mentioned above, since we assume that multiple probe flows are monitored to measure delays on multiple paths in parallel, the available information for both of the conventional and proposed methods is information about the all probe flows that stream on the network (they are 72 probe flows in the experiments of this section). From the aspect of the entire measurement system, the information that can be accessed by both methods is the same. However, the proposed method fully utilizes information that is included in these flows by converting samples of other flows. In the first experiment of this section, information about 110×72 = 7,920 samples are available. The conventional method utilizes 110 samples among 7,920 samples for an estimation of a flow,

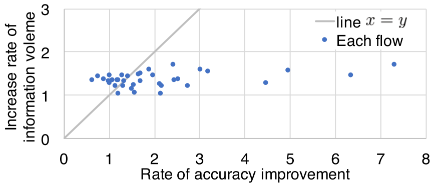


Fig. 16 The relation of increase rate of information volume and rate of accuracy improvement for each flow.

but the proposed method utilizes 110 to 188 samples. We consider the relation of the increase rate of information volume and the rate of accuracy improvement for each flow in order to confirm the efficiency of the conversion. The increase rate of information volume is defined by the ratio of the number of utilized samples in the proposed method to that of the conventional method. The rate of accuracy improvement is defined by $(1/r_{\mathrm{p}})/(1/r_{\mathrm{c}})$ where r_{p} and r_{c} are RMSE of the proposed and conventional methods, respectively.

The scatter diagram that represents the relation of them is shown in Fig. 16, and we add a line $x = y$ on the diagram. It is preferable that the points are plotted under the line. The result shows that large accuracy improvement is achieved for the many flows compared to the increase of information volumes since $2/3$ points are plotted under the line. This is because the proposed method utilizes information about samples in congestion periods that are important for estimations.

6. Related Works

There is a rich collection of literature that aims at measuring end-to-end delays [1], [2], [5], [17]–[19]. Some prior works [1], [2] have tried to estimate high quantile of end-to-end delays by active measurement. Choi et al. [1] 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. [2] 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 our 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 [4], [5], [20]–[22]. References [20], [21] 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 Refs. [4], [5], [22]. The limitation of a single flow measurement can be understood by these works.

7. 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 based on 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 achieves double accuracy comparing with the conventional method.

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.

Acknowledgments

This work was partly supported by JSPS KAKENHI Grant Number JP17K00008 and JP18K18035.

References

[1] 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, July 2007.

[2] J. Sommers, P. Barford, N. Duffield, and A. Ron, “Accurate and efficient SLA compliance monitoring,” ACM SIGCOMM Comput. Commun. Rev., vol.37, no.4, pp.109–120, Oct. 2007.

[3] F. Baccelli, S. Machiraju, D. Veitch, and J. Bolot, “On optimal probing for delay and loss measurement,” Proc. 7th ACM Conference on Internet Measurement (IMC 2007), pp.291–302, San Diego, CA, USA, Oct. 2007.

[4] M. Roughan, “Fundamental bounds on the accuracy of network performance measurements,” ACM SIGMETRICS Perform. Eval. Rev., vol.33, no.1, pp.253–264, June 2005.

[5] K. Watabe and K. Nakagawa, “Packet delay estimation that transcends a fundamental accuracy bound due to bias in active measurements,” IEICE Trans. Commun., vol.E100-B, no.8, pp.1377–1387, Aug. 2017.

[6] “One-way Transmission Time,” ITU-T Recommendation G.114, May 2003.

[7] K. Watabe, S. Hirakawa, and K. Nakagawa, “Accurate delay measurement for parallel monitoring of probe flows,” Proc. 2017 13th International Conference on Network and Service Management (CNSM 2017), Tokyo, Japan, Nov. 2017.

[8] H. Pucha, Y. Zhang, Z.M. Mao, and Y.C. Hu, “Understanding network delay changes caused by routing events,” ACM SIGMETRICS Perform. Eval. Rev., vol.35, no.1, p.73, June 2007.

[9] “CAIDA: The Cooperative Association for Internet Data Analysis,” http://www.caida.org/

[10] 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 Trans. Emerg. Topics Comput., vol.2, no.3, pp.267–279, Sept. 2014.

[11] Z. Wang and J. Crowcroft, “Bandwidth-delay based routing algorithms,” Proc. 1995 IEEE Global Telecommunications Conference (GLOBECOM 1995), pp.2129–2133, Singapore, Nov. 1995.

[12] A. Hinneburg and D. Keim, “DENCLUE: An efficient approach to clustering in large multimedia databases with noise,” Proc. 4th International Conference on Knowledge Discovery and Data Mining

(KDD 1998), pp.58–65, New York, NY, USA, Sept. 1998.
[13] G. Hamerly and C. Elkan, "Learning the k in k-means," Proc. Advances in Neural Information Processing Systems 16 (NIPS 2003), 2003.
[14] H. Li, K. Zhang, and T. Jiang, "Minimum entropy clustering and applications to gene expression analysis," Proc. 2004 IEEE Computational Systems Bioinformatics Conference (CSB 2004), pp.142–151, Stanford, CA, USA, Aug. 2004.
[15] T.R. Henderson, M. Lacage, G.F. Riley, G. Dowell, and J.B. Kopena, "Network simulations with the ns-3 simulator," Proc. ACM SIGCOMM 2008, p.527, Seattle, WA, USA, Aug. 2008.
[16] "Internet2 Network NOC," https://globalnoc.iu.edu/i2network/
[17] J.C. Bolot, "Characterizing End-to-end packet delay and loss in the Internet," J. High Speed Networks, vol.2, no.3, pp.289–298, 1993.
[18] K.P. Gummadi, S. Saroiu, and S.D. Gribble, "King: Estimating latency between arbitrary Internet end hosts," Proc. 2nd ACM SIGCOMM Workshop on Internet Measurment (IMW 2002), pp.5–18, Marseille, France, Nov. 2002.
[19] L. De Vito, S. Rapuano, and L. Tomaciello, "One-way delay measurement: State of the art," IEEE Trans. Instrum. Meas., vol.57, no.12, pp.2742–2750, Dec. 2008.
[20] F. Baccelli, S. Machiraju, D. Veitch, and J. Bolot, "The role of PASTA in network measurement," ACM SIGCOMM Comput. Commun. Rev., vol.36, no.4, pp.231–242, Oct. 2006.
[21] K. Watabe and M. Aida, "Analysis on the fluctuation magnitude in probe Interval for active measurement," Proc. 30th IEEE International Conference on Computer Communication (INFOCOM 2011) Mini-Conference, pp.161–165, Shanghai, China, April 2011.
[22] K. Watabe and K. Nakagawa, "Intrusiveness-aware estimation for high quantiles of a packet delay distribution," Proc. 2015 IEEE International Conference on Communications (ICC 2015), pp.7787–7792, London, UK, June 2015.

Kohei Watabe received his B.E. and M.E. degrees in Engineering from Tokyo Metropolitan University, Tokyo, Japan, in 2009 and 2011, respectively. He also received the Ph.D. degree from Osaka University, Japan, in 2014. He was a JSPS research fellow (DC2) from April 2012 to March 2014. He has been an Assistant Professor of Graduate School of Engineering, Nagaoka University of Technology since April 2014. He is a member of the IEEE and the IEICE.

Shintaro Hirakawa received his B.E. degree in Engineering from Nagaoka University of Technology, Niigata, Japan, in 2016. He is currently a student at Graduate School of Engineering, Nagaoka University of Technology. He is a student member of the IEICE.

Kenji Nakagawa received the B.S., M.S. and D.S. degrees from Tokyo Institute of Technology, Tokyo, Japan, in 1980, 1982 and 1986, respectively. In 1985, he joined NTT (Nippon Telegraph and Telephone Corp.). Since 1992, he has been an associate professor of Graduate School of Engineering, Nagaoka University of Technology. He is an associate editor of the IEICE Transactions on Communications. His research interests include queueing theory, performance evaluation of networks, and geometric theory of statics. Dr. Nakagawa is a member of the IEEE, SITA, and Mathematical Society of Japan.