Approximating the end-to-end delay using local measurements: a preliminary study based on conditional expectation

Huu-Nghi Nguyen, Thomas Begin, Anthony Busson, Isabelle Guérin Lassous

To cite this version:

Huu-Nghi Nguyen, Thomas Begin, Anthony Busson, Isabelle Guérin Lassous. Approximating the end-to-end delay using local measurements: a preliminary study based on conditional expectation. IEEE 3rd International Symposium on Networks, Computers and Communications, ISNCC, May 2016, Hammamet, Tunisia. hal-01387732

HAL Id: hal-01387732
https://hal.archives-ouvertes.fr/hal-01387732
Submitted on 26 Oct 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Approximating the end-to-end delay using local measurements: a preliminary study based on conditional expectation

Huu-Nghi Nguyen, Thomas Begin, Anthony Busson and Isabelle Guérin Lassous
Université Lyon, UCB Lyon 1, CNRS, ENS de Lyon, Inria, LIP UMR 5668
15 parvis René Descartes, F-69342, Lyon, FRANCE

Abstract—With Software-Defined Networking (SDN), computer networks will gain a better control and management on their physical resources and on flows requiring a specific type of QoS. To fulfil these objectives, networks must be able to characterize the end-to-end performance of flows, which are usually unknown and not directly measurable. Instead, networks have typically at their disposal only local measurements, collected at each node.

In this paper, we propose a new method to evaluate the variance of the end-to-end delay based only on measurements collected on nodes. The core of our solution is to link samples of waiting times at different nodes in order to approximate the corresponding covariance terms, needed to refine the approximation value of the sought variance. We evaluate the accuracy of our solution using two different scenarios. The obtained results show that the method is generally good, with a relative error on the estimated standard deviation of the end-to-end delay usually close to 5%, though it may exceed 10% when the network is facing high levels of load and experiencing packet losses.

I. INTRODUCTION

In recent years, Software-Defined Networking (SDN) has emerged as a promising paradigm for computer networks. In SDN, the network comprise switches and controllers. SDN switches are mostly dedicated to the forwarding of packets, and to statistic computation on the conveyed traffic. On the other hand, one or several controllers are in charge of managing the network. This includes tasks such as route computations as well as more elaborated tasks (e.g., load sharing, resource allocation, QoS provisioning, admission control). Therefore, within the SDN paradigm, the control plane is centralized, and all decisions are taken by the controllers. This approach starkly differs from the classical distributed approach of the Internet. SDN networks are expected to be more flexible, manageable, and prone to efficient optimization policies than their classical IP counterparts. This new technological framework also offers the opportunity to implement services, typically QoS, using a global knowledge of the network performance.

Thus, SDN controllers will be able to make their decisions (e.g., routing, resource allocation, admission control) based on end-to-end performances of flows. This ability appears as a huge potential asset for guaranteeing an end-to-end quality of service to certain flows. However, end-to-end performance measurements are typically hard to collect. A crucial question in this context is then: is it possible to infer end-to-end performance based only on local measurements collected on the SDN switches? In this paper, we tackle this issue for the end-to-end delay. Although each SDN switch can measure the delays spent by packets kept waiting before transmission, the end-to-end delays of packets cannot be directly derived from these local measurements. Indeed, correlations occur between these local delays, and the SDN controllers must handle them properly to obtain relevant indicators of the performance: variance, or distribution of the end-to-end delay. We propose a method to estimate the variance of the end-to-end delay experienced by packets within SDN networks.

This paper is organized as follows. In the next section, we present related work and the motivation of this work. In section III, we present the considered scenario, and the notation. We also decompose the different terms involved in the variance of the end-to-end delay. Most of these terms are measurable or computable locally by the SDN switches. We show that the correlation mainly lies in the successive waiting times (time spent by a packet in a buffer). Section IV describes our proposed method to evaluate this correlation. Numerical results are shown in section V. Section VI concludes this paper.

II. PROBLEM STATEMENT AND RELATED WORK

In this paper, we focus our efforts on the one-way end-to-end delay of packets. Note that this quantity differs from the round-trip delay measurement, also known as RTT [1], [2]. Note that the former is not simply half of the latter because network paths are known to be asymmetric [3]. Also, depending on the application or type of traffic (e.g. FTP or VoIP/Video), the relevance of each type of delays differs.

Besides, measurement schemes may either be intrusive [4] or non-intrusive [5]. Intrusive schemes rely on probe packets to assess the latency between two arbitrary nodes of the network. Because they operate without any knowledge on the network (high level of privacy ensured), many intrusive schemes were proposed [6]–[10]. Despite considerable efforts, intrusive measurement schemes still suffer from a somewhat lack of understanding (e.g., effect of probe packet size and rate), and furthermore, the probe traffic may itself cause buffer...
overflows thereby representing a bias in the outcome [11–
[13].

Note that the vast majority, if not all, of existing measure-
ment schemes (intrusive and non-intrusive altogether) heavily
relies on clock synchronization between the nodes [14–[16].
This can be viewed as a major drawback since this typically
requires to equip nodes with a highly-specialized and costly
GPS device.

Instead, we propose to infer the empirical distribution of
the one-way end-to-end delay of packets, and a fortiori its
two first moments, without any need of clock synchronization.
The proposed method belongs to non-intrusive measurement
schemes. It only resorts to measurements that can be collected
independently by each node (e.g. packet sizes, inter-arrival
times, source and destination nodes). In a previous work [17],
we introduced a method to approximate the variance of the
two first moments, without any need of clock synchronization.

The proposed method belongs to non-intrusive measurement
schemes. It only resorts to measurements that can be collected
independently by each node (e.g. packet sizes, inter-arrival
times, source and destination nodes). In a previous work [17],
we introduced a method to approximate the variance of the
two first moments, without any need of clock synchronization.

B. Assumptions

Throughout our work, we assume that nodes can measure for
each packet they process:
- the size,
- the departure time (time at which a packet leaves the node
  based on a non-synchronized local clock),
- the destination address,
- the source address,
- the waiting time (time spent in the buffer waiting for
  transmission).

Thanks to the combination of the source and destination
addresses, nodes are able to distinguish the different flows.
Finally, we assume that each node $k$ knows the propagation
delay and the capacity of subsequent links on the primary path,
i.e., $R_{i,l+1}$ and $C_{i,l+1}$ for $l \geq k$.

C. End-to-end delay decomposition

The time spent by any packet of a given flow $f$ to go through
a node $k$ can be decomposed as a sum of random variables as
follows:

(i) the processing delay, which is the necessary time to
handle the packet header and to commutate the packet
  to the proper network interface of the node - it is
considered as negligible (it is also close to constant and
thus its variance is null);

(ii) the transmission delay $S_{k}^{f}$ that corresponds to the time
  to transmit physically the packet on the link - it depends
  on the packet size and the link capacity;

(iii) the waiting time $W_{k}^{f}$ that amounts to the time spent by
  the packet kept in the buffer before its transmission;

(iv) the propagation delay $R_{k,k+1}$ which is the time to
  propagate one bit from node $k$ to node $k+1$ - it is
  supposed constant and its variance is consequently null.

It follows that the local delay for a packet of flow $f_1$ at node
$k$, i.e. the time between its arrival at node $k$ and its arrival at
the next node $k+1$, can be expressed as:

$$ D_{k}^{f_1} = (W_{k}^{f_1} + S_{k}^{f_1} + R_{k}). $$

Note that this local delay, $D_{k}^{f_1}$, can be measured at node
$k$. Therefore, we can formulate the end-to-end delay of the
primary flow $D_1$ as:

$$ D_1 = \sum_{k=1}^{N-1} \left( W_{k}^{f_1} + S_{k}^{f_1} + R_{k} \right). $$

From (2), the mean of the end-to-end delay can be straight-
forwardly derived. It suffices that each node along the path
sends an estimate of the mean local delay $D_{k}^{f_1}$ to the SDN
controller. Then, the SDN controller simply computes the sum.
However, computing the variance requires a more complex
algorithm as the correlations between the local delays have to
be taken into account. Using the properties of the variance on
(2), and noting that $R_{k,k+1}$ is a constant, we obtain:

$$ \mathbb{V}[D_{f_1}] = \mathbb{V}\left[ \sum_{k=1}^{N-1} W_{k}^{f_1} \right] + \mathbb{V}\left[ \sum_{k=1}^{N-1} S_{k}^{f_1} \right] $$
$$ + 2 \times \sum_{1 \leq k < l \leq N-1} \text{Cov}(W_{k}^{f_1}, S_{l}^{f_1}). $$

$S_{k}^{f_1}$ is a linear function of $P_{f_1}$, namely $S_{k}^{f_1} = \frac{p_{f_1}}{C_{k,k+1}}$. It
follows that $\mathbb{V}\left[ \sum_{k=1}^{N-1} S_{k}^{f_1} \right]$ can be easily computed given the
packet size distribution. Typically, this computation can be
modeled scenario. The rationale of the proposed method is to allow the SDN controller to link samples of the waiting times on the two nodes and infer the waiting time at node \(k+1\) from local measures at node \(k\). Nodes \(k\) and \(k+1\) computes conditional expectation of their local waiting times according to this function. The controller is then able to map the conditional expectations according to the value of the recurrent function. To help the reader to understand which quantity is measured and which one is inferred, all quantities/samples that are inferred have a hat (\(\hat{\cdot}\) for instance), and the ones that are measured by the nodes do not have it.

Our method is based on the following conditional expectation of \(\mathbb{E}[W^f_k W^f_{k+1}]\):

\[
\mathbb{E} \left[ W^f_k W^f_{k+1} \right] = \mathbb{E} \left[ \mathbb{E} \left[ W^f_k \big| W^f_{k+1} \right] W^f_{k+1} \right].
\] (5)

To obtain an estimation of Equation 5 we propose the following approach. It is possible for a node to infer the waiting time at the next node. Indeed, it can measure the departure time of the leaving packets. These times correspond to the arrival time at the next node. Let us denote \(p^m\) the packet size of the \(m\)-th packet (\(m \geq 1\)). We also consider the variable \(s^m_{k+1}\) that describes the inter-arrival time between packets \(m-1\) and \(m\) at node \(k+1\) and \(m\) at node \(k\), \(s^m_k\) is the transmission time of packet \(m\) at node \(k\) (it is locally measured or directly deduced from the packet length). Assuming that the server discipline is FIFO (First In-First Out), the waiting time and inter-arrival time at the next queue can be inferred through the following recurrent function:

\[
\begin{align*}
\hat{w}^m_{k,k+1} &= \max \left\{ 0, \hat{w}^{m-1}_{k,k+1} + s^{m-1}_k - \hat{w}^{m-1}_{k,k+1} \right\} \\
\hat{\delta}^m_{k,k+1} &= s^m_k + \max \left\{ 0, \hat{\delta}^{m-1}_{k,k+1} - \hat{w}^{m-1}_{k,k+1} - s^{m-1}_k \right\}, m > 1.
\end{align*}
\] (6)

where \(\hat{w}^m_{k,k+1}\) is thus the waiting time of the packet \(m\) at node \(k+1\) inferred by node \(k\). We denote \(\hat{W}^f_{k,k+1}\) the random variable which corresponds to these samples. Node \(k+1\) will use the same recurrent function to obtain the values inferred by node \(k\). The corresponding random variable is denoted \(\hat{W}^f_{k+1,k+1}\). Local waiting times measured on the two nodes \(k\) and \(k+1\) can then be mapped through the inferred values of \(\hat{w}^m_{k,k+1}\). Equation 5 is then approximated as:

\[
\mathbb{E} \left[ \mathbb{E} \left[ W^f_k \big| W^f_{k+1} \right] W^f_{k+1} \right] \approx \mathbb{E} \left[ \mathbb{E} \left[ W^f_k \big| \hat{W}^f_{k,k+1} \right] \hat{W}^f_{k,k+1} \right].
\] (7)

The right hand side of Equation (7) is computed by the use of two histograms evaluated at node \(k\). Node \(k\) divides the range of the computed values of \(\hat{W}^f_{k,k+1}\) into \(L\) disjoint
The variance (\(\sigma^2\)) of a given random variable is statistically equivalent to the mean of the squared differences from the mean. This can be expressed as:

\[
\sigma^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i - \mu)^2
\]

where \(x_i\) are the individual observations, \(\mu\) is the mean of the observations, and \(n\) is the total number of observations.

In the context of the transmission delays, the standard deviation is statistically equivalent to the mean of the squared differences from the mean. This can be used to approximate the statistical behavior of the delays. To simulate the real behavior of a network, and evaluate the statistical behavior of each solution, we use the discrete-event simulator NS-3 [18].

Note that, under these conditions, we ensure that the incoming traffic is able to apply exactly the same recurrent function as in the real trace. The trivial method neglects the covariance terms.

We consider two scenarios. Our first scenario, scenario A, deals with a network composed of 4 nodes and a total of 3 levels of sending rate for the primary flow, namely 0.5, 1 and 1.5 Mbps.

The results for the scenario A are presented in Figures 3a, 3b, 3c. It displays the cumulative distribution function of the relative error (in percents) committed by our solution and by the trivial solution. When the rate of the primary flow \(f_1\) is set to 0.5 Mbps, Figure 3a shows that, in more than 95\% of replications, our solution leads to a relative error for the variance of the end-to-end delay less than 10\%. Using the trivial solution in these same conditions, only around 15\% of replications have an error below 10\%. Figures 3b and 3c report the corresponding results when the primary flow \(f_1\) is increased to 1 and 1.5 Mbps, respectively. Although the values slightly differ, the overall behavior of the proposed solution and the trivial remains roughly the same.

We now turn to scenario B, illustrated by Figure 2b. Here, the network comprises 6 nodes and a larger number of flows. The transmission capacities of the links are as follows: \(C_{1,2}=6\), \(C_{2,3}=4\), \(C_{3,4}=7\), \(C_{4,5}=3\), \(C_{5,6}=5\) Mbps. As for the multiple secondary flows, the secondary flows \(f_{2,4}, f_{2,5}, f_{3,4}, f_{4,6}\) have sending rate of 1, 0.8, 2 and 0.2 Mbps, respectively. Note that, under these conditions, we ensure that the incoming rate at each link remains below 70\% of its capacity. Again, we consider three levels of sending rate for the primary flow, namely 0.5, 1 and 1.5 Mbps.

Figures 3d, 3e, 3f show the results found for scenario B. Broadly speaking, the results are generally poorer than in scenario A. For example, with a primary flow of rate 1 Mbps, 80\% (compared to 95\%) of replications lead to a relative error on the variance of the end-to-end delay less than 10\%. Nonetheless, the proposed solution still significantly outperforms the trivial solution. Also, we notice that as the intensity of the sending rate of \(f_1\) increases, the accuracy of our solution decreases. We believe that stems from the
Table I: Overall distribution of the relative error on the standard deviation of the end-to-end delay.

<table>
<thead>
<tr>
<th>Error</th>
<th>≤ 10%</th>
<th>≤ 20%</th>
<th>&gt; 30%</th>
<th>Mean</th>
<th>≤ 10%</th>
<th>≤ 20%</th>
<th>&gt; 30%</th>
<th>Mean</th>
<th>≤ 10%</th>
<th>≤ 20%</th>
<th>&gt; 30%</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\sigma_{sol_acenc-A})</td>
<td>96.11</td>
<td>99.31</td>
<td>6.07</td>
<td>1.84</td>
<td>93.26</td>
<td>99.17</td>
<td>0.07</td>
<td>3.05</td>
<td>94.01</td>
<td>98.96</td>
<td>0.21</td>
<td>3.19</td>
</tr>
<tr>
<td>(\sigma_{sol_acenc-B})</td>
<td>93.06</td>
<td>99.58</td>
<td>0.14</td>
<td>3.80</td>
<td>79.40</td>
<td>97.77</td>
<td>0.07</td>
<td>6.02</td>
<td>63.37</td>
<td>93.75</td>
<td>0.07</td>
<td>8.47</td>
</tr>
</tbody>
</table>

Figure 2: The primary flow \(f_1\) workload is set to 0.5, 1.0 and 1.5 (Mbps). In Figure 2a, the utilization rates at the first bottleneck are 50%, 60% and 70%. In Figure 2b, the utilization rates at the first bottleneck are 37%, 50% and 63%, at the second bottleneck are 32%, 50% and 66%.

Figure 3: Cumulative distribution function of the relative errors for the standard deviation of the end-to-end delay for different workloads of the primary flow \(f_1\).

Increasing number of packet losses that disrupt our method. Interestingly, the trivial methods tends to enhance as the rate of \(f_1\) grows, but stays far less accurate than the proposed solution. We believe that this improvement occurs because at higher level of loads, the queuing delays become less variable and so their correlation tends to decrease.

Finally, Table I summarizes the results obtained in these two scenarios by our solution. We observe that in the many studied examples, regardless of the actual value of sending rate for \(f_1\), the relative errors on the estimation of the variance of the end-to-end delay remains below 10% in over 90% of the replications explored. The mean relative error of our method tends to peak when the traffic load is high (with a corresponding relative error around 10%), and in general, delivers accurate results with a mean relative error less than or close to 5% in most cases.

VI. CONCLUSION

With the development of SDN, computer networks aim at gaining a better control on the management of their resources. This may lead to cost savings and to a better handling of flows requiring a specific type of QoS. However, to fulfil these objectives, networks must be able to characterize the end-to-end performance of flows. Unfortunately, computer networks are usually not equipped with tools that allow direct end-to-end measurements. Instead, networks have typically at their
disposal only local measurements, collected at each node. Although evaluating the mean value of a path metric is usually rather simple, things are becoming much more complex for higher-order moments.

In this paper, we propose a new method to evaluate the variance of the end-to-end delay based only on measurements collected on nodes. The core of our solution is to link samples of waiting times at different nodes in order to approximate the corresponding covariance terms, needed to refine the approximate value of the sought variance. We evaluate the accuracy of our solution using two different scenarios. The obtained results show that the method is generally good, with a relative error on the estimated standard deviation of the end-to-end delay usually close to 5%, though it may exceed 10% when the network is facing high levels of load and experiencing packet losses.

ACKNOWLEDGMENT

This work is funded by the French National Research Agency ANR INFRA DISCO under the “ANR-13-INFR-013” project.

REFERENCES