ISSN 2047-3338

Design and Implementation of Flow-Level Simulator FSIM for Performance Evaluation of Large Scale Networks

Yusuke Sakumoto, Hiroyuki Ohsaki and Makoto Imase

***Abstract*—In this paper, we propose a flow-level simulator called FSIM (Fluid-based SIMulator) for performance evaluation of large-scale networks, and verify its effectiveness using our FSIM implementation. The notable feature of our flow-level simulator FSIM is fast simulation execution compared with a conventional flow-level simulator. For accelerating simulation execution speed, our flow-level simulator FSIM adopts an adaptive numerical computation algorithm for ordinary differential equations. Another features of our flow-level simulator FSIM are accuracy and compatibility with an existing network performance analysis tool. For improving simulation accuracy, our flow-level simulator FSIM utilizes accurate fluid-flow models. In this paper, through extensive experiments using our FSIM implementation, we evaluate the effectiveness of our flow-level simulator FSIM in terms of simulation speed, accuracy and memory consumption. Consequently, we show that our flow-level simulator FSIM outperforms a conventional flow-level simulator; i.e., it realizes approximately 200%-2,000% faster simulation with higher accuracy and less memory consumption than a conventional flow-level simulator.**

***Index Terms*—Flow-Level Simulator, Fluid-Flow Model, Large Scale Network, Performance Evaluation, Simulation**

I. Introduction

IN recent years, the scale of the Internet has been expanding rapidly. Because of widespread deployment and rapid advancement of Internet technologies, the number of hosts connected to the Internet and the capacity of the Internet has been increasing exponentially [1]–[3]. Such explosive expansions of the Internet in both size and speed make it difficult to understand behavior of the entire network [4]–[6]. Hence, performance evaluation technique for a large-scale network has been demanded by many networking researchers [7]–[9].

This work was supported by the Grant-in-Aid for Scientific Research (B) (24700072) by the Ministry of Education, Culture, Sports, Science and Technology (MEXT)).

Yusuke Sakumoto is with Graduate School of System Design, Tokyo Metropolitan University, Asahigaoka 6-6, Hino, Tokyo 191-0065, Japan (Phone: +81-42-585-8639; Email: sakumoto@ tmu.ac.jp).

Hiroyuki Ohsaki is with Graduate School of Science and Technology, Kwansei Gakuin University, 2-1-1 Sanwa, Sanda, Hyougo 669-1595, Japan (Email: ohsaki@kwansei.ac.jp).

Makoto Imase is with National Institute of Information and Communications Technology, 4-2-1 Nukui-Kitamachi, Koganei, Tokyo 184-8795, Japan (Email: imase@nict.go.jp).

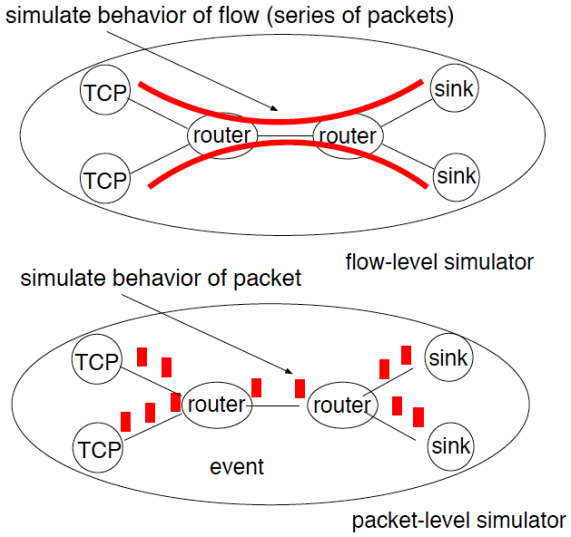

Fig. 1. Packet-level simulator and flow-level simulator

Performance evaluation techniques for communication networks are classified into three categories: mathematical analysis, simulation, and experiment [10].

Mathematical analysis is a technique for performance evaluation utilizing a mathematical model of the network under study. Mathematical analysis is generally suitable for analyzing comparatively small-scale networks. Application of mathematical analysis to performance evaluation of a large-scale network has been a hot topic among network researchers [11], [12]. However, since mathematical analysis usually requires a lot of simplifying assumptions, analytic results obtained from mathematical analysis sometimes don't meet required accuracy.

Simulation is a common technique for performance evaluation utilizing computers. In simulation, computer models of building blocks of the network under study are built, and behavior of those building blocks are simulated [10]. Compared with mathematical analysis, simulation can be applied to performance evaluation of rather complicated networks.

Experiment is a technique for performance evaluation utilizing a real system [10]. In experiment, the network under study is constructed using real devices and computers. Although experiment enables detailed performance evaluation, it

generally lacks flexibility and also requires significant amount of cost for building the real system. For performance evaluation of a large-scale network, experiment requires a number of network devices and computers, making it unrealistic to apply to a large-scale network.

Considering trade-offs between accuracy and cost, simulation is a reasonable approach for performance evaluation of large-scale networks. In the literature, there are several studies on simulation techniques for a large-scale network (see [13]–[15] and the references therein). Depending on granularity of simulation models, network simulators can be classified into two categories: packet-level simulator and flow-level simulator.

Packet-level simulator mimics behavior of every packet in 2 a network [16]. For instance, packet arrivals at a router and packet departures from a router are simulated in packet-level simulator. Packet-level simulator has been widely used by many networking researchers. Advantage of packet-level simulator includes its accuracy compared with flow-level simulator [17]. Since packet-level simulator simulates behavior of every packet, packet-level performance metrics can be measured with packet-level simulator. On the contrary, disadvantage of packet-level simulator is its inability to simulate large-scale networks. This is because computational complexity increases as the size and/or speed of a simulated network increases [17]. Several researchers try to enable packet-level simulation for a large-scale network [18], but there still remain several issues to be solved.

On the contrary, flow-level simulator mimics behavior of every flow in a network [19]. For instance, packet arrivals at a router and packet departures from a router are aggregated as a flow (i.e., a stream of packets) in flow-level simulator. Advantage of flow-level simulator includes, contrary to packet-level simulator, its fast simulation execution. Since a number of packets are statistically modeled as a single flow, flow-level simulator can simulate a large-scale network, where the number of in-flight packets is enormous [20]. Disadvantage of flow-level simulator is low granularity compared with packet-level simulator. This is because that flow-level simulator ignores packet-level behavior. Hence, by using a flow-level simulator, packet-level performance metrics cannot be measured. However, it would not be a big problem since packet-level performance metrics are not required for performance evaluation of large-scale networks; instead, flow-level and/or application-level performance metrics are required, which can be measured by flow-level simulator.

In this paper, we propose a flow-level simulator called FSIM (Fluid-based SIMulator) for performance evaluation of large-scale networks, and verify its effectiveness using our FSIM implementation. The notable feature of our flow-level simulator FSIM is fast simulation execution compared with a conventional flow-level simulator [13]. For accelerating simulation execution, our flow-level simulator FSIM adopts an adaptive numerical computation algorithm for ordinary differential equations. Another features of our flow-level simulator FSIM are accuracy and compatibility with an existing network performance analysis tool. For improving simulation accuracy, our flow-level simulator FSIM utilizes accurate fluid-flow models [21]. Also, the flow-level simulator FSIM can input and output files compatible with ns-2 [16], which is one of the most popular packet-level simulators. In this paper, through extensive experiments using our FSIM implementation, we evaluate the effectiveness of our flow-level simulator FSIM in terms of simulation speed, accuracy and memory consumption. Consequently, we show that our flow-level simulator FSIM outperforms a conventional flow-level simulator; i.e., it realizes approximately 200%-2,000% faster simulation with higher accuracy and less memory consumption than a conventional flow-level simulator.

This paper is organized as follows. In Section II, related works on fluid-flow models and flow-level simulators are summarized. Section III explains our flow-level simulator FSIM. Namely, fluid-flow models and the adaptive numerical computation algorithm utilized in our FSIM are explained. In Section IV, we evaluate the effectiveness of our flow-level simulator FSIM in terms of simulation speed, accuracy and memory consumption. Finally, Section V concludes this paper and discusses future works.

II. Related Works

There are several researches for large-scale network simulation [13], [22]–[30]. They can be classified into two categories: simulation engine acceleration (e.g., the efficient packet-event execution [22], [23] and parallel simulation [24], [25]) and simulation model abstraction (e.g., flow level simulation [13], [26], [27] and hybrid simulation [28]–[30]). Simulation engine acceleration is an approach for decreasing simulation execution time of a packet-level simulation without simulation model abstraction. On the contrary, simulation model abstraction is another approach for improving computational complexity of a packet-level simulation by using the abstract model, which represents the behavior of a large-scale network. Although methods for simulation engine acceleration are important techniques, a packet-level simulation does not scale well as the bandwidth and/or the network size increases [13]. Hence, simulation model abstraction is indispensable in practice. In simulation model abstraction, flow-level simulation is one of the key techniques for large-scale network simulation. This is because that as a flow-level simulation becomes faster, the scalability of a hybrid simulation utilizing the flow-level simulation would be improved, too.

In [13], a flow-level simulation utilizing a TCP/RED fluid-flow model was proposed for large-scale network simulation. By numerically solving ordinary differential equations (ODEs) directly derived from fluid-flow models, flow-level simulation is performed. However, the numerical computation algorithm for ODEs in [13] is quite simple; i.e., network states of fluid-level simulation are updated every fixed stepsize. Network states are updated even when network states are unchanged, causing slowdown of fluid-level simulation. In [26], [30], heuristic approaches for adaptive stepsize control have been proposed. However, accuracy of those heuristic approaches is unpredictable/unstable. It is crucial for any adaptive

TABLE I
DEFINITIONS OF SYMBOLS (CONSTANTS AND VARIABLES)

$x(t)$	input (transmission rate)
$y(t)$	output (transmission rate)
$R(t)$	round-trip time of flow
$p_{TO}(t)$	TCP timeout probability
min_{th}	minimum threshold value of RED router
max_{th}	maximum threshold value of RED router
max_p	maximum packet marking probability of RED router
α	weight of exponential moving average of RED router
c	processing speed of RED router
$q(t)$	current queue length of RED router
$r(t)$	average queue length of RED router
$p_q(t)$	packet marking probability of RED router
$p(t)$	packet loss probability of RED router
τ	propagation delay of link
F_r	set of TCP flows of link
M	number of TCP flows in the network
N	number of RED routers in the network

stepsize control to guarantee the accuracy of its solution [31]–[33]. In this paper, we take a rigorous approach for adaptive stepsize control in flow-level simulation.

III. FSIM (FLOW-LEVEL SIMLATOR)

The notable feature of our flow-level simulator FSIM is fast simulation execution compared with conventional flow-level simulators [13], [34]. For accelerating simulation execution, our flow-level simulator FSIM adopts an adaptive numerical computation algorithm for ordinary differential equations. Another features of our flow-level simulator FSIM are its accuracy and its compatibility with other network performance analysis tools. For improving simulation accuracy, our flow-level simulator FSIM utilizes accurate fluid-flow models [21]. Also, the flow-level simulator FSIM can input and output files compatible with ns-2 [16], which is one of representative packet-level simulators.

In what follows, details of our flow-level simulator FSIM − fluid-flow models, the adaptive numerical computation algorithm for ordinary differential equations, and compatibility with an existing network performance analysis tool — are explained.

A. Fluid-flow models

We explain fluid-flow models [21] utilized in FSIM. The fluid-flow model derived in [21] models the TCP timeout mechanism with high accuracy. By comparing simulation results with analytic ones, the authors of [21] show it has higher accuracy than the fluid-flow model [13]. Definitions of symbols (i.e., constants and variables) used throughout this paper are summarized in Tab. I. FSIM utilizes the fluid-flow model of the TCP congestion control mechanism derived in [21]. In the fluid-flow model of the TCP congestion control mechanism, the input $x(t)$ is the arrival rate of ACK packets and the output $y(t)$ is the transmission rate of TCP, which is given by

$$\dot{y}(t) = \frac{x(t)}{y(t)R(t)^2} - \frac{2}{3}y(t)z(t)\big(1 - p_{TO}(t)\big) - \left(\frac{4}{3}y(t) - \frac{1}{R(t)}\right)z(t)p_{TO}(t), \quad (1)$$

where $z(t) \equiv y(t - R(t)) - x(t)$. $p_{TO}(t)$ is the probability that a packet loss is detected by the TCP timeout mechanism rather than duplicate ACKs, and it can be approximated as $p_{TO}(t) \cong \min(1, 3/(y(t)R(t))$. $R(t)$ is the round-trip time of the flow, which is given by the sum of propagation delays and queuing delays on the path.

FSIM utilizes the fluid-flow model of the RED router derived in [21]. In the fluid-flow model of the RED router, the input $x(t)$ is the arrival rate of RED router and the output $y(t)$ is the departure rate of RED router, which is given by

$$y(t) = \min\Big(c, \big(1 - p(t)\big)x(t)\Big), \quad (2)$$

where p(t) is the packet loss probability. The packet loss probability $p(t)$ is given by

$$p(t) = \begin{cases} \frac{2}{3}p_q(t) & if\ Wait\ option\ on \\ \frac{2p_q(t)}{1 + p_q(t)} & \text{otherwise} \end{cases}, \quad (3)$$

where $p_q(t)$ is the packet marking probability. $p_q(t)$ is given by

$$p_q(t) = \begin{cases} 0 & \text{if } 0 \le r(t) < \min_{\text{th}} \\ \frac{\max_{\text{p}}}{\max_{\text{th}} - \min_{\text{th}}}(r(t) - \min_{\text{th}}) & \text{if } \min_{\text{th}} \le r(t) < \max_{\text{th}} \\ \frac{1 - \max_{\text{p}}}{\max_{\text{th}}}r(t) - \big(1 - 2\max_{\text{p}}\big) & \text{if } \max_{\text{th}} \le r(t) < 2\max_{\text{th}} \\ 1 & \text{if otherwise} \end{cases}, \quad (4)$$

where r(t) is the average queue length, and $q(t)$ is the current queue length , which are given by

$$\dot{r}(t) = -\alpha c\big(r(t) - q(t)\big), \quad (5)$$

$$\dot{q}(t) = \begin{cases} x(t) - c & \text{if } q(t) > 0 \\ \max\big(x(t) - c, 0\big) & \text{otherwise} \end{cases}. \quad (6)$$

In the fluid-flow model of the link, the input $x(t)$ is the incoming transmission rate and the output $y(t)$ is the outgoing transmission rate of the link, which is given by

$$y(t) = x(t - \tau), \quad (7)$$

where τ is the propagation delay of the link.

An entire network is modeled with the analysis technique proposed in [21] by connecting models of the TCP congestion control mechanism, the RED router, and the link. When the RED router has multiple input links, input to the RED router is modeled as flow convergence of incoming TCP flows. Flow convergence can be described as the sum of transmission rates of incoming TCP flows. In other words, when the transmission rate for each TCP flow is $x_i(t)$ $(i \in F_r)$ where F_r is the set of TCP flows passing through the RED router, the arrival rate $x(t)$ (i.e., the sum of transmission rates) of the RED router is given by

$$x(t) = \sum_{i \in F_r} x_i(t). \quad (8)$$

When the RED router has multiple output links, output from the RED router is modeled through distribution in outgoing TCP flows. Flow distribution can be described by distribution of outgoing traffic from the RED router to individual TCP flows. Let the departure rate from the RED router be $y(t)$, the

transmission rate $y_i(t)$ ($i \in F_r$) for each TCP flow is given by

$$y_i(t) = y(t)\frac{x_i(t)}{x(t)}. \quad (9)$$

B. Adaptive numerical computation algorithm

In fluid-flow models explained in Section III-A, network state is represented by TCP transmission rates $y(t)$, the average queue lengths $r(t)$ of RED routers, and current queue lengths $q(t)$ of RED routers. Let $z(t)$ be the state vector of a network given by

$$z(t) = \begin{pmatrix} y_1(t) \\ \vdots \\ y_M(t) \\ y_1(t - R_1(t)) \\ \vdots \\ y_M(t - R_M(t)) \\ r_1(t) \\ \vdots \\ r_N(t) \\ q_1(t) \\ \vdots \\ q_N(t) \end{pmatrix},$$

where M is the number of TCP flows in the network, and N is the number of RED routers in the network. The change of the state vector $z(t)$ is obtained from fluid-flow models (Eqs. (1), (5), and (6)). The state vector $z(t + \Delta)$ is numerically obtained from $z(t)$ and $\dot{z}(t)$ by using a numerical computation algorithm for ordinary differential equations. By repeating this process, the time evolution for the network state starting from an initial state can be numerically obtained, and flow-level simulation can be performed.

For accelerating simulation execution, FSIM uses the adaptive stepsize control, Dormand-Prince method [31], [32], which adjusts the stepsize Δ according change in ordinary differential equations. In other words, when change in the network state is large, the stepsize is decreased for decreasing error in the numerical computation. On the contrary, when change in the network state is small, the stepsize is increased for speeding up the numerical computation. With such an adaptive control, computational complexity required for flow-level simulation can be significantly reduced. Even if an adaptive stepsize control is used for speeding up flow-level simulation with reliability, the accuracy of simulation results should be maintained. Dormand-Prince method guarantees the error in a numerical computation algorithm [32] unlike adaptive stepsize controls used in [26], [30]. By using Dormand-Prince method, flow-level simulation should be accelerated while maintaining the accuracy of simulation results.

Notice that the fluid-flow model of the TCP congestion control mechanism (Eq. (1)) requires past network state (i.e., $y(t - R)$). In FSIM, past network states (up to the maximum round-trip time of all TCP flows) are recorded in the memory for enabling application of the Dormand-Prince method. Since FSIM uses the adaptive stepsize control, the timing at which the network state is updated is varied. So past network state required for calculating the next network state might not have been calculated. In FSIM, the past network state in need is approximated as an interpolation of nearby network states [32].

C. Compatibility with existing performance evaluation tools

The flow-level simulator FSIM realizes high compatibility with an existing network performance evaluation tool. Specifically, FSIM can input and output files compatible with ns-2 [16], which is one of the most popular packet-level simulators. Specifically, our flow-level simulator FSIM interprets typical ns-2 simulation file written in OTcl [35]; i.e., major ns-2 commands such as duplex-link and create-connection are parsed and translated to FSIM objects. Also, FSIM outputs its simulation logs in either ns-2 (generated with trace-all command) or nam [36] (generated with namtrace-all compatible format.

IV. Experiment

In this section, through extensive experiments using our FSIM implementation, we evaluate the effectiveness of our flow-level simulator FSIM in terms of accuracy, simulation speed, and memory consumption.

A. Experimental setup

We compare performance of three simulators: our flow-level simulator FSIM, the conventional flow-level simulator FFM [37], and packet-level simulator ns-2 [16]. FFM is one of the famous flow-level simulators. Recall the major difference between FFM and FSIM; i.e., different from FSIM, FFM has no adaptive stepsize control, and the TCP fluid-flow model in FFM does not model the TCP timeout mechanism. For purely investigating the effectiveness of the adaptive stepsize control, we compare the performance of FSIM with and without the adaptive stepsize control. We performed simulations for the same topology and parameters with those three simulators. It is essential to investigate the performance of a network simulator for several simulation scenarios since network simulators are in nature used for several purposes. We therefore use three typical simulation scenarios: dumb-bell network, random network, and hierarchical network.

- Dumb-bell network
  A dumb-bell network [38] consists of two RED routers and homogeneous TCP flows with identical propagation delays (see Fig. 2). The link between two RED routers is the bottleneck since the access link (i.e., the link between an end host and an RED router) is much faster than that between RED routers. Unless explicitly stated, the number of TCP flows is 20,000, and the bottleneck link bandwidth is 5,000 [Mbit/s]. Note that RED parameters ($\min_{th}$ and $\max_{th}$), and the buffer size of the RED router are proportional to the bottleneck link bandwidth C and then number N of TCP flows for stability of RED control. Also, note that the RED parameter is inversely proportional to the bottleneck link bandwidth C. Such a simple simulation scenario using a dumb-bell network has been widely adopted as a baseline model for many networking performance studies [38].
- Random network
  In the random network scenario (see Fig. 3), for a given

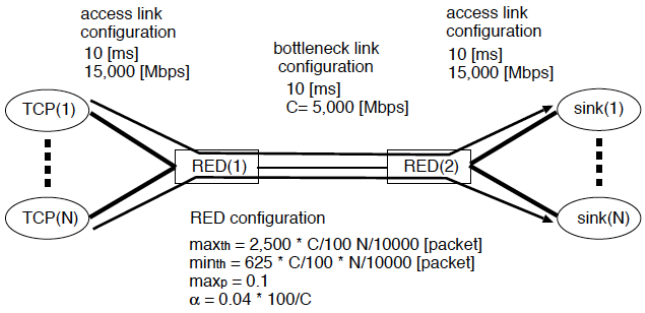

Fig. 2. Dumb-bell network

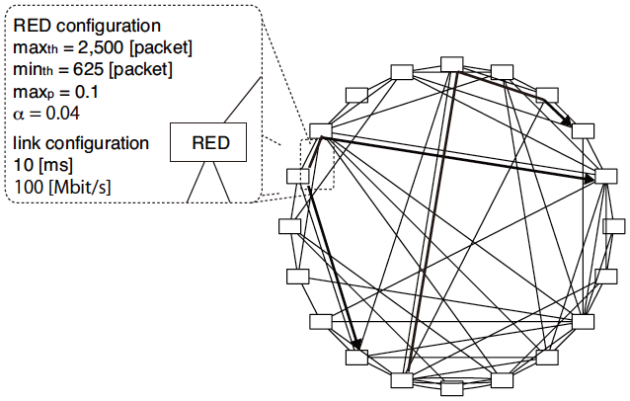

Fig. 3. A random network with 20 nodes

number of RED routers and the average degree (i.e., the average number of links connected to an RED router), a network topology is randomly generated as a random network [39]. Also, for a given number of TCP flows, TCP source hosts and sinks are attached at randomly-chosen RED routers. The random network is widely used by large-scale network researchers [40]–[42]. Unless explicitly stated, the bandwidths of all links are equally set to 100 [Mbit/s], the propagation delays of all links are equally set to 10 [ms], and the average degree is 3.

- Hierarchical network

  The hierarchical network consists of three levels, are referred to as WAN, MAN and LAN levels (see Fig. 4). The hierarchical network is an Internet-like network [43]. For a given number of RED routers, a network topology is randomly generated as a hierarchical network [43]. Then, for a given number of TCP flows, TCP source hosts and sinks are attached at randomly-chosen RED routers at LAN level. Unless explicitly stated, the bandwidths of all links are equally set to 100 [Mbit/s], and the propagation delays of all links are equally set to 10 [ms].

In all simulation scenarios, all TCP source hosts continuously transmit data to their corresponding TCP sinks.

In all experiments, a computer with Intel Core i7 CPU 960 (3.20GHz) processors with 5 [GByte] memory running Debian GNU/Linux 5.0.7 (kernel version 2.6.26) is used for executing flow-level simulators or the packet-level simulator.

In all experiments, we repeated 10 simulations, and measured the average and 95% confidence interval of measurements

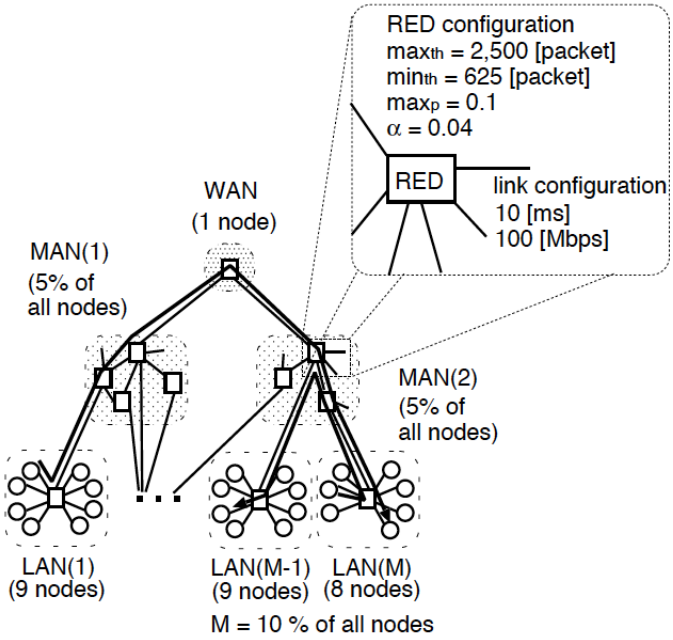

Fig. 4. Hierarchical network

(e.g., simulation execution time and maximum memory consumption). In the following figures, confidence intervals are not shown because they were sufficiently small in all experiments. Note that we optimized ns-2 configurations following the guideline in [44].

B. *Accuracy*

With three network simulators (i.e., FSIM, FFM, and ns-2), TCP packet transmission rate and the queue length of the RED router are measured (Figs. 5 through 8). Figures 6 and 8 illustrate the queue length of RED router between a flow-level simulator (FSIM or FFM) and the packet-level simulator ns-2 for different link bandwidths and numbers of TCP flows, respectively. These figures show that FSIM achieves slightly better accuracy than FFM.

C. *Simulation speed*

We investigate simulation speeds of flow-level simulators (FSIM and FFM) and the packet-level simulator ns-2 when changing the number of TCP flows, the link bandwidth, and the number of nodes. To investigate the simulation speed, we measured simulation execution times of three simulators. With three simulators, simulation execution times required for performing 50 [s] of simulation are measured.

We first investigate simulation speeds of three simulators when changing the number of TCP flows in the dumb-bell network. Figure 9 shows simulation execution times of three simulators for different numbers of TCP flows.

Figure 9 shows that the simulation execution time of FSIM is much shorter than that of FFM regardless of the number of TCP flows. This is because the adaptive stepsize control implemented in FSIM is effective regardless of the number of TCP flows. The average stepsize of FSIM with adaptive stepsize control is larger than that of FFM and FSIM without adaptive stepsize control (see Fig. 10). Figure 9 also shows that simulation execution times of FSIM and FFM do not increase even

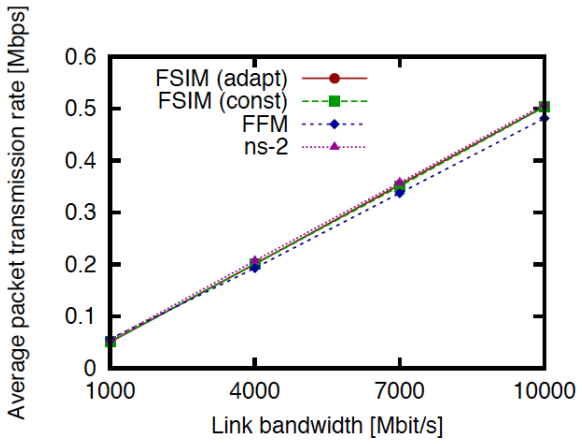

Fig. 5. Time average of TCP packet transmission rates vs. the link bandwidth

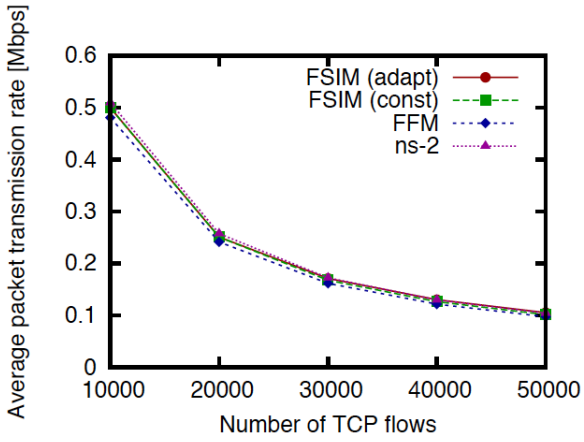

Fig. 7. Time average of TCP packet transmission rates vs. the number of TCP flows

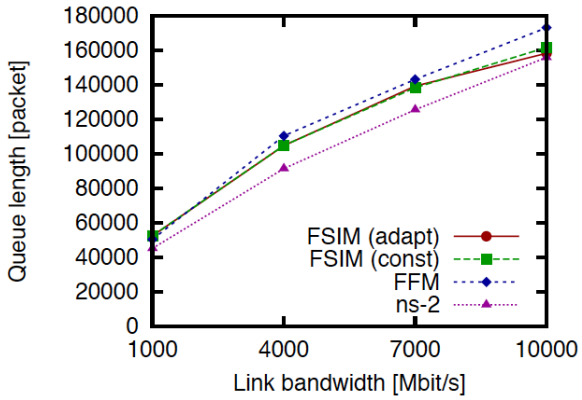

Fig. 6. Time average of the queue length vs. the link bandwidth

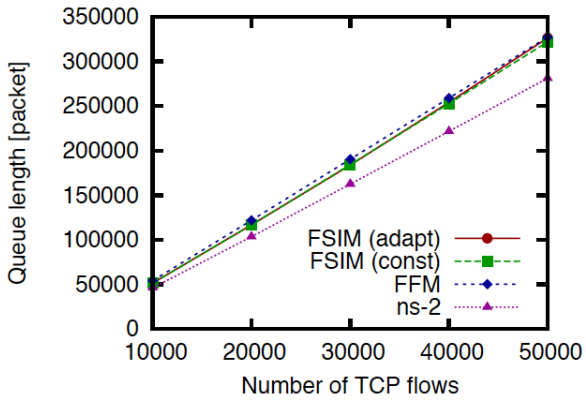

Fig. 8. Time average of the queue length vs. the number of TCP flows

when the number of TCP flows increases. This is because flow-level simulators support flow aggregation, which aggregates multiple flows with the same characteristics into a single one.

We then investigate simulation speeds of three simulators when changing the bottleneck link bandwidth in the dumbbell network. Figure 11 shows simulation execution times of three simulators for dif ferent bottleneck link bandwidths.

Figure 11 shows that the simulation execution time of FSIM is much shorter than that of FFM and ns-2 regardless of the bottleneck link bandwidth. Figure 11 shows that the simulation execution time of FSIM increases as the bottleneck link bandwidth increases. This is resulted fro m our adaptive stepsize control. Namely, as the bottleneck link bandwidth increase, the queue length of RED router is drastically changed. Thus, the stepsize must be decreased when the bottleneck link bandwidth is large (see Fig.12). From Fig. 12, the average stepsize of FSIM with the adaptive stepsize control should approach to that of FSIM without the adaptive stepsize control. Hence, for larger bottleneck link bandwidth than that shown in Figs. 11 and 12, the simulation execution time of FSIM with the adaptive stepsize control should be smaller than that of FFM.

Figure 11 shows that simulation execution times of FSIM without the adaptive stepsize control and FFM does not increase even when the bottleneck link bandwidth increases. This clearly indicates the strength of flow-level simulators; the computational complexity of flow-level simulators does not increase as the link bandwidth increases.

We then investigate simulation speeds of three simulators when changing the number of nodes. Figures 13 and 15 show simulation execution times of three simulators for different numbers of nodes in the random network and the hierarchical network, respectively. Due to memory exhaustion, we could not perform simulation of the random network with more than 3,000 nodes for FFM, the random network with more than 500 nodes for ns-2, and the hierarchical network with 5,000 nodes for ns-2.

Figures 13 and 15 show that the simulation execution time of FSIM is much shorter than that of FFM and ns-2 regardless of the number of nodes. Figures 13 and 15 also show that simulation execution times of FSIM and FFM increase when the number of nodes increases. This is because the number of TCP flows with a different path increases when the number of nodes increases. TCP flows with a different path cannot be aggregated into a single one by flow-aggregation. Thus, the computational complexity of FSIM and FFM increases when the number of nodes increases.

In summary, the simulation speed of FSIM is much faster than that of FFM regardless of the number of TCP flows, the link bandwidth, and the number of nodes. FSIM realizes 2-20 times faster simulation compared with FFM on the average.

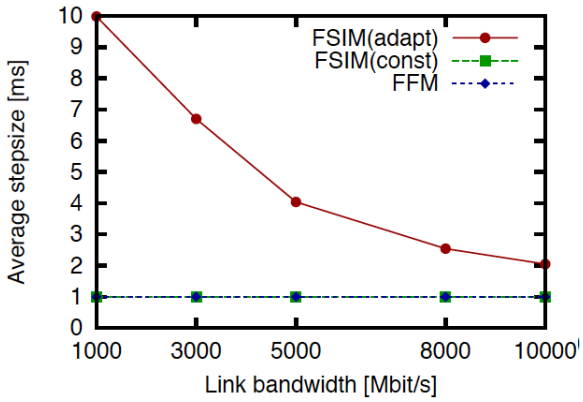

Fig. 9. Simulation execution time vs. the number of TCP flows

Fig. 11. Simulation execution time vs. the bottleneck link bandwidth

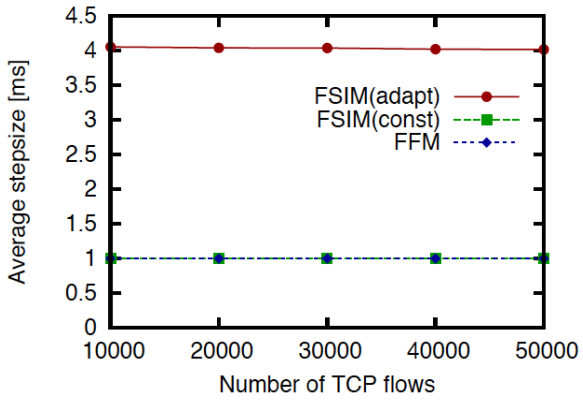

Fig. 10. Average stepsize time vs. the number of TCP flows

Fig. 12. Average of stepsize vs. the bottleneck link bandwidth

D. Memory consumption

We investigate memory consumptions of flow-level simulators (FSIM and FFM) and the packet-level simulator ns-2 when changing the number of TCP flows, the link bandwidth, and the number of nodes. Scalability of network simulators is sometimes limited by the memory size required for executing simulation [44]. To investigate the memory consumption, maximum memory consumptions (i.e., the sum of statistically and dynamically allocated memory size) during simulation run are measured for three simulators.

We first investigate memory consumptions of three simulators when changing the number of TCP flows in the dumb-bell network. Figure 17 shows that the maximum memory consumption of FSIM is much smaller than that of FFM regardless of the number of TCP flows. This can be explained as follows. FFM is embedded in ns-2, which has many functions not to be related to flow-level simulation. FFM loads such functions on a memory during execution. It is not trivial to modify FFM to load functions required only for flow-level simulation due to its high module coupling. Figure 17 also shows that maximum memory consumptions of FSIM and FFM does not increase even when the number of TCP flows increases. Similar to the phenomenon observed in Fig. 9, this is because flow-level simulators support flow-aggregation.

We then investigate memory consumptions of three simulators when changing the bottleneck link bandwidth in the dumb-bell network.

Figure 18 shows that the maximum memory consumption of FSIM is independent of much smaller than that of FFM and ns-2 regardless of the bottleneck link bandwidth. Similar to the phenomenon observed in Fig. 11, this is because flow-level simulators mimics behavior of every flow in a network.

We then investigate memory consumptions of three simulators when changing the number of nodes. Figures 19 and 20 show maximum memory consumptions of three simulators for different numbers of nodes in the random network and the hierarchical network, respectively.

Figures 19 and 20 show that the maximum memory consumption of FSIM is much smaller than that of FFM regardless of the number of nodes. This is because FSIM stores the path of a TCP flow as a list with the number of elements equal to the path length whereas FFM stores the path of a TCP flow in an array with the number of elements equal to the total number of nodes. The maximum memory consumption of FSIM when performing simulation of the random network with 10,000 nodes is approximately 1,430 [Mbyte]. Figures 19 and 20 also show that maximum memory consumptions of FSIM and FFM increase as the number of nodes increases. Similar to the phenomenon observed in Figs. 13 and 15 this is because the

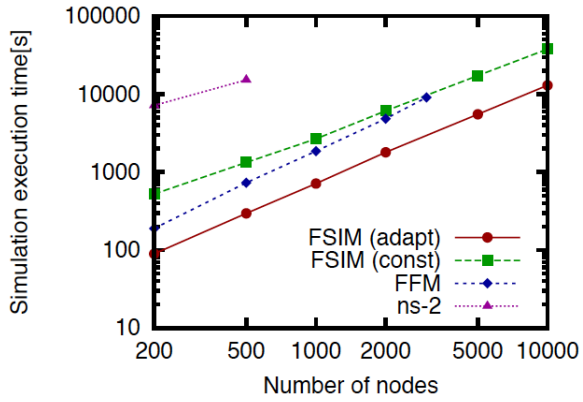

Fig. 13. Simulation execution time vs. the number of nodes in the random network

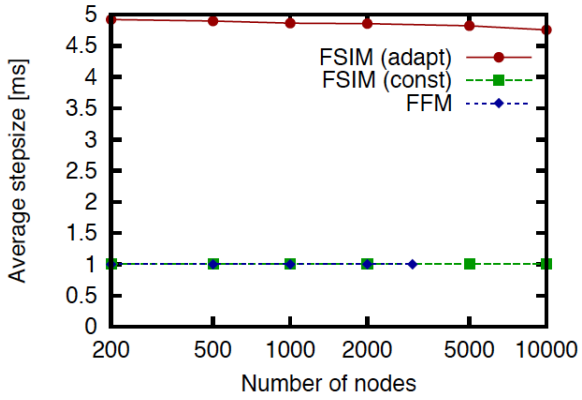

Fig. 14. Average of stepsize vs. the number of nodes in the random network

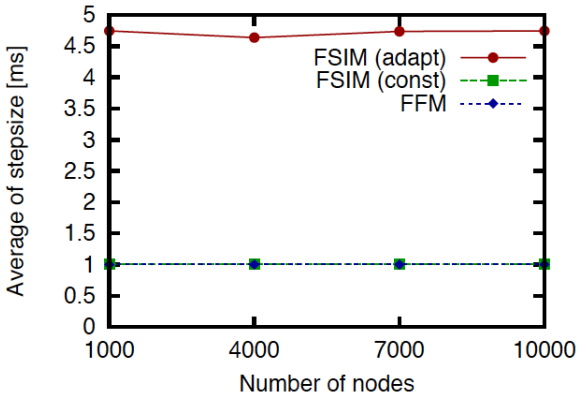

Fig. 16. Average of stepsize vs. the number of nodes in the hierarchical network

Fig. 15. Simulation execution time vs. the number of nodes in the hierarchical network

number of TCP flows with a different path increases when the number of nodes increases.

Figures 17 through 20 show that the memory consumptions of FSIM are slightly larger than that of FSIM without the adaptive stepsize control. We have confirmed that the overhead of the adaptive stepsize control would be vanishingly small.

In summary, the memory consumption of FSIM is always smaller than that of FFM regardless of t he number of TCP flows, the link bandwidth, and the number of nodes. FSIM realizes more than 3-7 times memory efficiency compared with FFM. This suggest s that for a given memory size, our flow-level simulator FSIM can simulate a larger network than FFM.

V. Conclusion and Future Works

In this paper, we have proposed a flow-level simulator called FSIM (Fluid-based SIMulator) for performance evaluation of large-scale networks, and have verified its effectiveness using our FSIM implementation. Through extensive experiments using our FSIM implementation, we have evaluated the effectiveness of our flow-level simulator FSIM in terms of simulation speed, accuracy and memory consumption. We have shown that our flow-level simulator FSIM outperforms a conventional flow-level simulator; i.e., it realizes approximately 200%-2,000% faster simulation with higher accuracy and less memory consumption than a conventional flow-level simulator. In particular, we should note that FSIM is effective for performance evaluation of a network with large link capacities and many TCP flows.

As future work, we are planning to further improve the numerical computation algorithm of differential equation. We are also planning to include support for various types of network protocols such as UDP, DCCP, high-speed TCP, and XCP utilizing fluid-flow models derived in [45]–[47].

Our FSIM implementation is available at http://www.ispl.jp/fsim/.

References

[1] Global internet geography," available at http://www.telegeography.com/.
[2] [G. Gilder, TELECOSM: How infinite bandwith will revolutionize our world. Free Press, Sep. 2000.
[3] "ISC internet domain survey," http://www.isc.org/solutions/survey.
[4] S. Floyd and V. Paxson, "Why we don't know how to simulate the Internet," Oct. 1999, available at http://www.aciri.org/floyd/papers/wsc. ps.

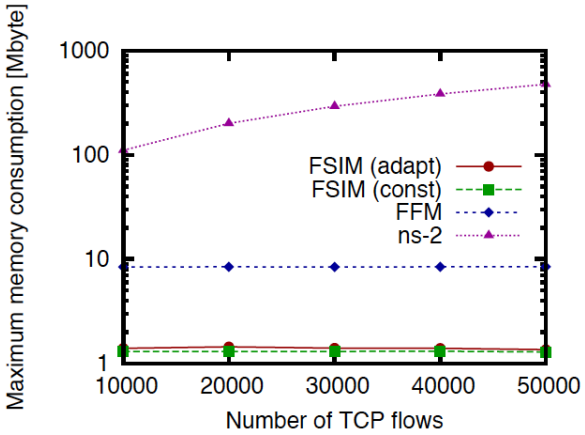

Fig. 17. Maximum memory consumption vs. the number of TCP flows

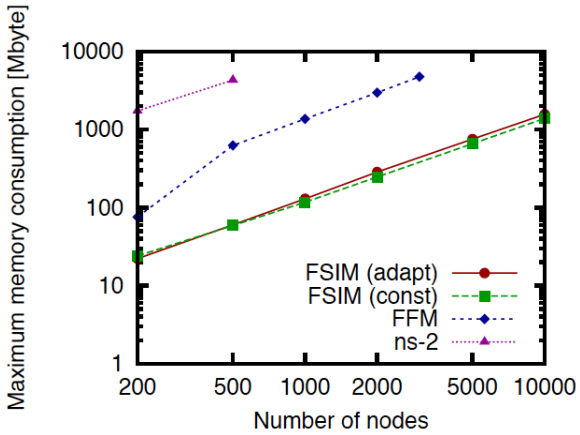

Fig. 19. Maximum memory consumption vs. the number of nodes in a random network

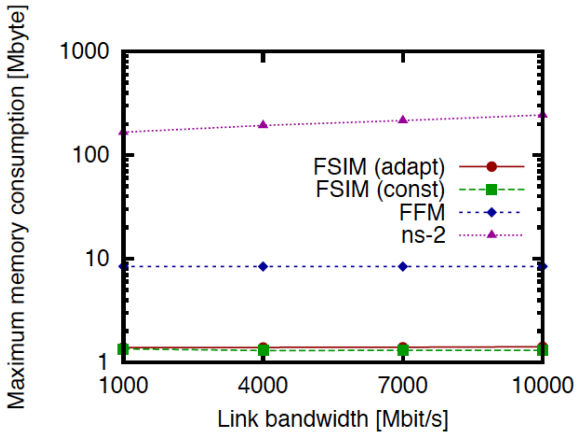

Fig. 18. Maximum memory consumption vs. the bottleneck link bandwidth

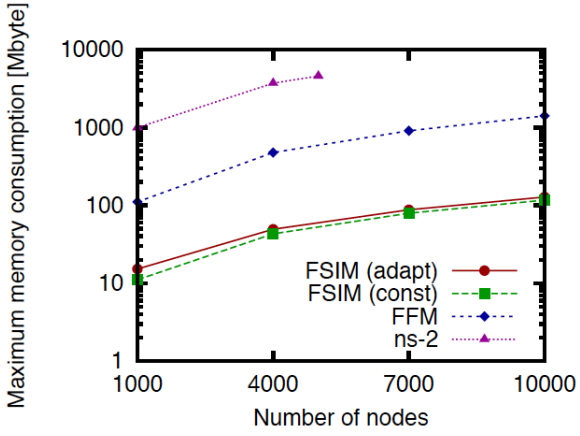

Fig. 20. Maximum memory consumption vs. the number of nodes in a hierarchical network

[5] F. Cen, T. Xing, and K.-T. Wu, "Real-time performance evaluation of line topology switched ethernet," International Journal of Automation and Computing, vol. 5, no. 4, pp. 376–380, Nov. 2008.

[6] T. Miyachi, K. ichi Chinen, and Y. Shinoda, "Starbed and springos: large-scale general purpose network testbed and supporting software," in Proceedings of the 1st international conference on Performance evaluation methodolgies and tools (VALUETOOLS 2006), Oct. 2006, pp. 2–8.

[7] Workshop on New Visions for Large-Scale Networks: Research and Applications. Large Scale Networking (LSN) Coordinating Group of the Interagency Working Group (IWG) for Information Technology Research and Development (IT R&D), Mar. 2001, available at http://www.nitrd.gov/subcommittee/lsn/lsn-workshop-12mar01/workshop-12mar01.pdf.

[8] P. Defibaugh-Chavez, S. Mukkamala, and A. H.Sung, "Efficacy of coordinated distributed multiple attacks (a proac tive approach to cyber defense)," in Proceedings of the 20 th International Conference on Advanced Information Networking and Applications-Volume 02(AINA 2006), Apr. 2006, pp. 10–14.

[9] P. Velho and A. Legrand, "Accuracy study and improvement of network simulation in the SimGrid framework," in Proceedings of the 2nd International Conference on Simulation Tools and Techniq ues, Mar. 2009, pp. 1–10.

[10] R. Jain, The Art of computer systems performance analysis. Wiley - Interscience, Apr. 1991.

[11] R. Pan and B. Prabhakar, "SHRiNK: A method for enabling scaleable performance prediction adn efficient network simulation," in Proceedings of the 48th IEEE Global Communications Conference (GLOBECOM 2005), Jun. 2005, pp. 1108–1113.

[12] A. Feketea, G. Vattaya, and L. Kocarev, "Traffic dynamics in scale-free networks," Complex Systems, vol. 3, no. 1–3, pp. 97–107, Aug. 2006.

[13] Y. Liu, F. L. Presti, V. Misra, D. Towsley, and Y. Gu, "Fluid models and solutions for large-scale IP networks," in Proceedings of the ACM Special Interest Group on Measurement and Evaluation (SIGMETRICS 2003), Jun. 2003, pp. 91–101.

[14] A. Park, R. M.Fujimoto, and K. S.Perumalla, "Conservative synchronization of large-scale network simulations," in Proceedings of 18st International Workshop on Principles of Advanced and Distributed Simulation (PADS 2004), Mar. 2004, pp. 153–161.

[15] B. K.Szymanski, Y. Liu, and R. Gupta, "Parallel network simulation under distributed Genesis," in Proceedings of 17st International Workshop on Principles of Advanced and Distributed Simulation (PADS 2003), Jun. 2003, pp. 61–68.

[16] "The network simulator – ns-2," available at http://www.isi.edu/nsnam/ns/.

[17] Y. GU, Y. Lie, and D. Towsley, "On integragin fluid model with packet simulation," in Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2004), vol. 4, Mar. 2004, pp. 2856–2866.

[18] C. Kiddle, R. Simmonds, and B. Unger, "Improving scalability of network emulation through parallelism and abstraction," in Proceedings of the 38st Annual Simulation Symposium (ANSS 2005), Apr. 2005, pp. 119–129.

[19] B. Liu, D. R. Figueired, Y. Guo, J. Kurose, and D. Towsley, "A study of networks simulation efficiency: Fluid simulation vs. packet-level simulation," in Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2001), vol. 3, Jan. 2001, pp. 22–26.

[20] J. Zhou, Z. Ji, M. Takai, and R. Bagrodia, "MAYA: a multi-paradigm network modeling framework for emulating distributed applications," in Proceedings of the 17st International Workshop on Principles of Advanced and Distributed Simulation (PADS 2003), Jun. 2003, pp. 162–170.

[21] H. Ohsaki, J. Ujiie, and M. Imase, "On scalable modeling of TCP congestion control mechanism for large-scale IP networks," in Proceedings

of the 5th IEEE International Symposium on Applications and the Internet (SAINT 2005), Feb. 2005, pp. 361–369.
[22] D. X. Wei and P. Cao, "NS-2 TCP-Linux: an NS-2 TCP implementation with congestion control algorithms from Linux," in Proceedings of the 2nd workshop on ns-2: the IP network simulator (WNS2 2006). ACM Press, Oct. 2006, pp. 1–10.
[23] H. Kim, J. C. Hou, and H. Lim, "TranSim: accelerating simulation of large-scale IP networks through preserving network invariants," Computer Networks, vol. 52, no. 15, pp. 2924–2946, Oct. 2008.
[24] S. Bhatt, R. Fujimoto, A. Ogielski, and K. Perumalla, "Paralell simulation techniques for large scale networks," IEEE Communications Magazine, vol. 38, no. 8, pp. 42–47, Aug. 1998.
[25] J. Liu, "Parallel simulation of hybrid network traffic models," in Proceedings of the 21st International Workshop on Principles of Advanced and Distributed Simulation (PADS 2007), Jun. 2007, pp. 141–151.
[26] S. Kumara, S.-J. Parka, and S. S. Iyengarb, "A loss-event driven scalable fluid simulation method for high-speed networks," Computer Networks, vol. 54, no. 1, pp. 112–132, Jan. 2010.
[27] H. Kima and J. C.Houb, "Enabling network calculus-based simulation for TCP congestion control," Computer Networks, vol. 53, no. 1, pp. 1–24, Jan. 2009.
[28] Y. GU, Y. Lie, and D. Towsley, "On integrating fluid model with packet simulation," in Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2004), vol. 4, Mar. 2004, pp. 2856–2866.
[29] K. Fujiwara and H. Casanova, "Speed and accuracy of network simulation in the simgrid framework," in Proceedings of the 2nd international conference on Performance evaluation methodologies and tools (VALUETOOLS 2007), Oct. 2007, p. 10.
[30] J. Liu and Y. Li, "On the performance of a hybrid network traffic model," Simulation Modelling Practice and Theory, vol. 16, no. 6, pp. 656–669, Jul. 2008.
[31] J. R. Dormand and P. J. Prince, "A family of embedded runge-kutta formulae," Journal of Computational and Applied Mathematics, vol. 6, pp. 19–26, Mar. 1980.
[32] E. Hairer, S. P. Norsett, and G. Wanner, Solving Ordinary Differential Equations I: Nonstiff Problems. Springer-Verlag, 1993.
[33] E. Hairer, S. P.Norsett, and G. Wanner, Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems. Springer-Verlag, Mar. 1996.
[34] B. Melamed, S. Pan, and Y. Wardi, "Hybrid discrete-continuous fluidflow simulation," The International Society for Optical Engineering, vol. 4526, pp. 263–270, Jul. 2001.
[35] "Otcl(mit object tcl)," http://otcl-tclcl.sourceforge.net/otcl/.
[36] "The network animator – NAM," available at http://www.isi.edu/nsnam/nam/.
[37] "Simulating large networks using fluid flow models (FFM)," available at http://www-net.cs.umass.edu/fluid/.
[38] L. Andrew, S. Floyd, and G. Wang, "Common tcp evaluation suite," IETF Internet Draft: draft-irtf-tmrg-tests-02.txt, Jul. 2009.
[39] B. Bloolbas, Random Graphs Second Edition. Cambridge Univ Press, Oct. 2001.
[40] O. Hiroyuki, Y. Koutaro, and I. Makoto, "On the effect of scale-free structure of network topology on t cp performance," SAINT '07:proceeding of the 2007 International Symposium on Applications and the Internet, p. 12, 2007.
[41] G. Peli and G. Papp, "Are scale free networks better?" http://arxiv.org/pdf/cond-mat/0301555, Feb. 2003.
[42] M. E. J. Newman, S. H. Strogatz, and D. J. Watts, "Random graphs with arbitrary degree distributions and their applications," Phycal Review E, vol. 64, no. 2, p. 026118, Jul. 2001.
[43] K. Calvert, M. B. Doar, A. Nexion, E. W. Zegura, G. Tech, and G. Tech, "Modeling internet topology," IEEE Communications Magazine, vol. 35, no. 6, pp. 160–163, Jun. 1997.
[44] "The network simulator – ns2: Tips and statistical data for running large simulations in ns," http://www.isi.edu/nsnam/ns/ns-largesim.html.
[45] H. Hisamatu, H. Ohsaki, and M. Murata, "Fluid-based analysis of a network with DCCP connections and RED routers," in Proceedings of the 6th International Symposium on Applications and the Internet (SAINT 2006), Jan. 2006, pp. 22–29.
[46] H. Ohsaki, H. Yamamoto, and M. Imase, "Scalable modeling and performance evaluation of dynamic RED router using fluid-flow approximation," in Proceedings of SPIE's International Symposium on the Convergence of Information Technologies and Communications (ITCom 2005), Oct. 2005.
[47] Y. Sakumoto, H. Ohsaki, and M. Imase, "On XCP stability in a heterogeneous network," in Proceedings of the 12th IEEE Symposium on Computers and Communications (ISCC 2007), Jul. 2007.

Yusuke Sakumoto received M.E. and Ph.D. degrees in the Information and Computer Sciences from Osaka University in 2008 and 2010, respectively. He is currently an assistant professor at Graduate School of System Design, Tokyo Metropolitan University, Japan. His research work is in the area of performance evaluation of congestion control protocol. He is a member of IEEE, IPSJ and IEICE.

Hiroyuki Ohsaki received the M. E. degree in the Information and Computer Sciences from Osaka University, Osaka, Japan, in 1995. He also received the Ph. D. degree from Osaka University, Osaka, Japan, in 1997. He has been an associate professor of Graduate School of Information Science and Technology at Osaka University since 2013. He is currently a professor at Department of Informatics, School of Science and Technology, Kwansei Gakuin University, Japan. His research work is in the area of traffic management in high-speed networks. He is a member of IEEE, IEICE, and IPSJ.

Makoto Imase received his B.E. and M.E. degrees in information engineering from Osaka University in 1975 and 1977, respectively. He received D.E. degree from Osaka University in 1986. From 1977 to 2001, he was engaged Nippon Telegraph and Telephone Corporation (NTT). He has been a Professor of Graduate School of Information Science and Technology at Osaka University since 2002. His research interests are in the area of information networks, distributed systems and graph theory. He is a member of IPSJ, JSIAM, and IEICE.