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

Yusuke Sakumoto, Ryouta Asai, Hiroyuki Ohsaki, and Makoto Imase
Graduate School of Information Science and Technology
Osaka University, Suita, Osaka 565-0871, Japan
{y-sakumt,r-asai,oosaki,imase}@ist.osaka-u.ac.jp

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 features of our flow-level simulator FSIM are its accuracy and fast simulation execution compared with conventional flow-level simulators. For improving simulation accuracy, our flow-level simulator FSIM utilizes accurate fluid-flow models. For accelerating simulation execution speed, our flow-level simulator FSIM adopts an adaptive numerical computation algorithm for ordinary differential equations. Another notable feature of our flow-level simulator FSIM is its compatibility with the existing network performance analysis tool. In this paper, through several 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 100% faster simulation with higher accuracy and less memory consumption than a conventional flow-level simulator.

I. Introduction

In recent years, the scale of the Internet has been expanding rapidly. Because of widespread deployment of Internet technologies, the number of hosts connected to the Internet has been increasing exponentially [1]. Such explosive expansion of the Internet makes it difficult to understand behavior of the entire network. Hence, performance evaluation technique for a large-scale network has been demanded by many networking researchers [2].

Performance evaluation techniques for communication networks are classified into three categories: mathematical analysis, simulation, and experiment [3]. Simulation is a common technique for performance evaluation utilizing computers. In simulation, computer models of building blocks of the target network are built, and behavior of those building blocks are simulated [3]. Compared with mathematical analysis, simulation can be applied to performance evaluation of more complicated networks.

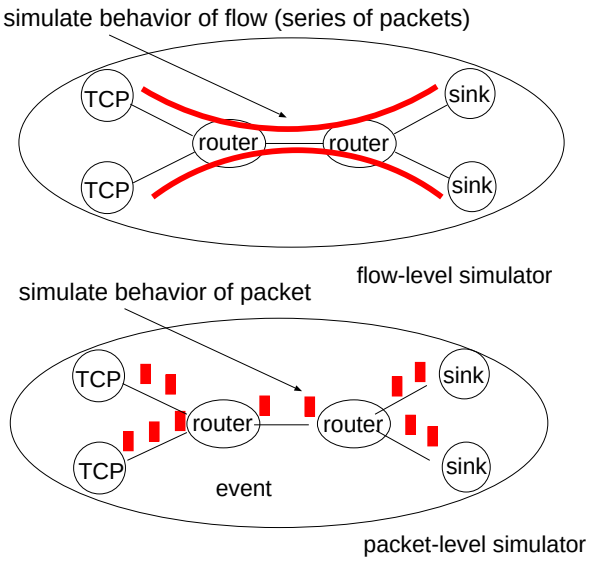

Fig. 1: Packet-level simulator and flow-level simulator

Considering trade-offs between accuracy and cost, simulation is the promising technique for performance evaluation of a large-scale networks. In the literature, there are several studies on simulation techniques for a large-scale network (see [4]–[6] 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 a network [7]. 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 [8]. 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 scale of a simulated network becomes large [8]. Several researchers try to enable packet-level simulation for a large-scale network [9], but there still remains several issues to be solved.

On the contrary, flow-level simulator mimics behavior of every flow in a network [10]. 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 approximately modeled as a single flow, flow-level simulator can simulate a large-scale network or high-speed network, where the number of in-flight packets is enormous [11]. Disadvantage of flow-level simulator is its inaccuracy compared with packet-level simulator. Since flow-level simulator ignores packet-level behavior, packet-level performance metrics cannot be measured. However, it would not be a big problem since packet-level performance metrics are not necessary required for large-scale network simulation; 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 features of our flow-level simulator FSIM are its accuracy and fast simulation execution compared with conventional flow-level simulators [4], [12]. For improving simulation accuracy, our flow-level simulator FSIM utilizes accurate fluid-flow models [13]. For accelerating simulation execution, our flow-level simulator FSIM adopts an adaptive numerical computation algorithm for ordinary differential equations. Another notable feature of our flow-level simulator FSIM is its compatibility with the existing network performance analysis tool. Specifically, the flow-level simulator FSIM can input and output files compatible with ns-2 [7], which is one of the most popular packet-level simulators. In this paper, through several 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 100% 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 presents our flow-level simulator FSIM: overview, fluid-flow models, and the adaptive numerical computation algorithm for ordinary differential equations. 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

In [4], an approach for large-scale network simulation utilizing a TCP/RED fluid-flow model is proposed. Ordinary differential equations directly derived from fluid-flow models are numerically solved for performing flow-level simulation. However, the numerical computation algorithm for ordinary differential equations in [4] is quite simple; network states of fluid-level simulation are updated every fixed step time. Network states are updated even when network states are unchanged, causing slowdown of fluid-level simulation. Also, the TCP fluid-model developed in [4] does not model the TCP timeout mechanism, so that accuracy of the fluid-flow model is not satisfactory.

In [13], another TCP/RED fluid-flow model is derived, which models the TCP timeout mechanism. By comparing simulation results with analytic ones, the authors of [13] show it has higher accuracy than the fluid-flow model derived in [4]. Both input and output of fluid-flow models derived are uniformly defined as packet transmission rate. So, those fluid-flow models can be easily interconnected for building the fluid-flow model of a large-scale network. In this paper, we utilize those fluid-flow models and the modeling approach for [13] for realizing an accurate flow-level simulator for a large-scale network.

In [14], a hybrid system model is proposed. The hybrid system model switches multiple fluid-flow models according to the TCP operation phase (i.e., slow-start phase and congestion avoidance phase) for improving the modeling accuracy. The hybrid system is numerically solved for performing flow-level simulation. However, similar to [4], the numerical computation algorithm in [14] is quite simple; network states of fluid-level simulation are updated every fixed step time. Network states are updated even when network states are unchanged, causing slowdown of fluid-level simulation.

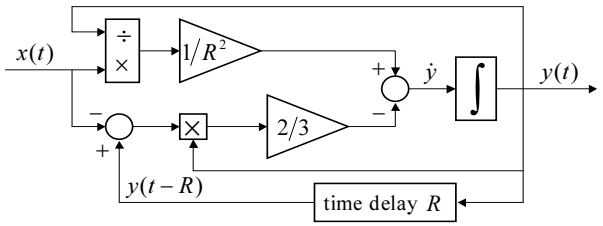

Fig. 2: Block diagram of TCP congestion control mechanism

III. FSIM (Flow-level SIMmulator)

The notable features of our flow-level simulator FSIM are its accuracy and fast simulation execution compared with conventional flow-level simulators [4], [12]. For improving simulation accuracy, our flow-level simulator FSIM utilizes accurate fluid-flow models derived in [13]. For accelerating simulation execution, our flow-level simulator FSIM adopts an adaptive numerical computation algorithm for ordinary differential equations. Another notable feature of our flow-level simulator FSIM is its compatibility with other network performance analysis tools. Specifically, the flow-level simulator FSIM can input and output files compatible with ns-2 [7], 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 the existing network performance analysis tool — are explained.

A. Fluid-flow models

FSIM utilizes the fluid-flow model of the TCP congestion control mechanism derived in [13] (Fig. 2). Definition of symbols (constants and variables) used throughout this paper are summarized in Tab. I.

In the fluid-flow model of the TCP congestion congestion mechanism, the input $x(t)$ is the arrival rate of ACK packets and the output $y(t)$ is the transmission rate of data packets.

$$\begin{aligned} \dot{y}(t) &= g(t, x(t), y(t), R) \\ &= \frac{x(t)}{y(t)\,R^2} - \frac{2}{3} y(t)\, z(t)\, (1 - p_{TO}(t)) \\ &\quad - \left(\frac{4}{3}\, y(t) - \frac{1}{R} \right) z(t)\, p_{TO}(t), \end{aligned} \tag{1}$$

where $z(t) \equiv y(t - R) - x(t)$. $p_{TO}(t)$ is the probability that a packet loss is detected by the timeout mechanism rather than duplicate ACKs, and can be approximated as $p_{TO}(t) \simeq min(1, 3/w(t))$.

TABLE I. Definitions of symbols (constants and variables)

$x(t)$	input (packet transmission rate)
$y(t)$	output (packet transmission rate)
R	TCP round-trip time
$w(t)$	TCP window size
$p_{TO}(t)$	TCP timeout probability
min_{th}	minimum threshold value
max_{th}	maximum threshold value
max_p	maximum packet marking probability
α	weight of exponential moving average
$c(t)$	processing speed of RED router
$b(t)$	current queue length of RED router
$r(t)$	average queue length of RED router
$p_b(t)$	packet marking probability
$p(t)$	packet loss probability

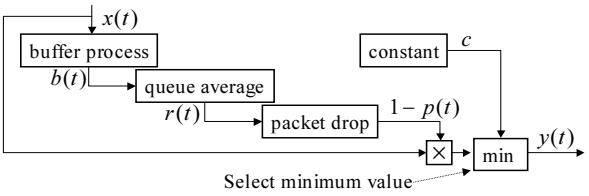

Fig. 3: Block diagram of RED router

FSIM utilizes the fluid-flow model of the RED router derived in [13] (Fig. 3). In the fluid-flow model of the RED router, the input $x(t)$ is the packet arrival rate and the output $y(t)$ is the packet transmission rate.

$$\begin{aligned} y(t) &= h(t, x(t), p(t), c(t)) && (2) \\ &= \min(c(t), (1 - p(t))\, x(t)), && (3) \end{aligned}$$

where $p(t)$ is the packet loss probability. $p(t)$ is given by the packet marking probability $p_b(t)$, the current queue length $b(t)$, and the average queue length $r(t)$ as

$$p(t) = \frac{2 p_b(t)}{1 + p_b(t)} \tag{4}$$

$$p_b(t) = \begin{cases} 0 \text{ if } r(t) < min_{th} \\ \dfrac{max_p}{max_{th} - min_{th}} (r(t) - min_{th}) \\ \quad \text{if } min_{th} \le r(t) < max_{th} \\ \dfrac{1 - max_p}{max_{th}} r(t) - (1 - 2max_p) \\ \quad \text{if } max_{th} \le r(t) < 2max_{th} \\ 1 \text{ if } r(t) \ge 2max_{th} \end{cases} \tag{5}$$

$$\dot{r}(t) = -\alpha\, c(t)(r(t) - b(t)) \tag{6}$$

$$\dot{b}(t) = \begin{cases} x(t) - c(t) & \text{if } b(t) > 0 \\ max(x(t) - c(t), 0) & \text{otherwise} \end{cases} \tag{7}$$

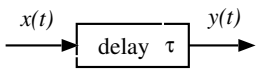

Fig. 4: Block diagram of link propagation delay

The link propagation delay is modeled where input $x(t)$ is the packet transmission rate and output $y(t)$ is the packet transmission rate (Fig. 4).

$$y(t) = x(t - \tau), \tag{8}$$

where τ is the propagation delay for the link.

An entire network is modeled with the analysis technique proposed in [13] by connection of the model of the TCP congestion control mechanism, the model of the RED router, and the model of the link propagation delay. When the RED router has multiple input links, this is modeled as flow convergence from individual input links. Flow convergence can be described as the sum of packet transmission rates for individual links. In other words, when the transmission rate for each flow is $x_i(t)(1 \leq i \leq N)$ and the sum of transmission rates is $y(t)$, the following equation is established.

$$y(t) \quad = \quad \sum_{i=1}^{N} x_i(t) \tag{9}$$

When the RED router has multiple output links, output from the RED router is modeled through distribution in multiple flows. Flow distribution can be described by distribution of 1 packet transmission rate N times. Here, N is the number of output links. When the flow before distribution is $x(t)$, each flow after distribution is $y_i(t)(1 \geq i \geq N)$, and each flow distribution ratio is $f_i(t)(1 \geq i \geq N)$, the following equation is established.

$$y_i(t) \quad = \quad f_i(t)x(t) \tag{10}$$

B. Adaptive numerical computation algorithm

In fluid-flow models explained in Section III-A, network state is represented by the TCP packet transfer rate $y(t)$, the current queue length $b(t)$ of RED router, and the average queue length $r(t)$ of RED router. Let $\mathbf{z}(t)$ be the state vector of a network given by

$$\mathbf{z}(t) \quad = \quad \begin{pmatrix} y_1(t) \\ \vdots \\ y_M(t) \\ b_1(t) \\ \vdots \\ b_N(t) \\ r_1(t) \\ \vdots \\ r_N(t) \end{pmatrix}, \tag{11}$$

where M is the number of TCP flows in the network, and N is the number of RED routers in the network. The state vector at $t + \Delta$ is given by

$$\mathbf{z}(t + \Delta) \simeq \mathbf{z}(t) + \dot{\mathbf{f}}(t, \mathbf{z}(t))\, \Delta, \tag{12}$$

where $\dot{\mathbf{f}}$ is obtained from fluid-flow models (Eqs. (1)–(6)).

Using a numerical computation algorithm for ordinary differential equations, evolution of the network state starting from an initial state can be numerically obtained. As a numerical solution for Eq. (12), FSIM uses a well-known numerical computation algorithm for ordinary differential equations, the Runge-Kutta method [15].

For accelerating simulation execution, FSIM uses the adaptive stepsize control for the Runge-Kutta method [15], which adjusts the stepsize according change in ordinary differential equations. Namely, when change in the network state is large, the stepsize is decreases for minimizing 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, while maintaining the accuracy of simulation results.

Notice that the fluid-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 into the memory for enabling application of the Runge-Kutta method. Since FSIM uses the adaptive stepsize control for the Runge-Kutta method, 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.

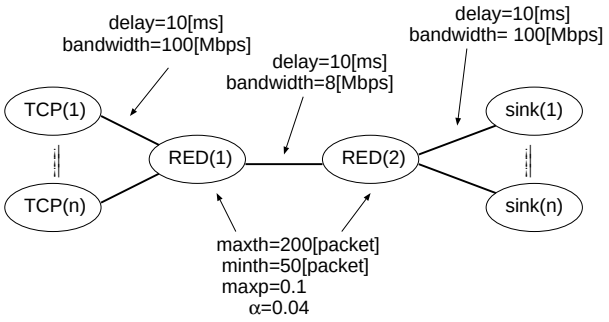

Fig. 5: Network topology used in simulations

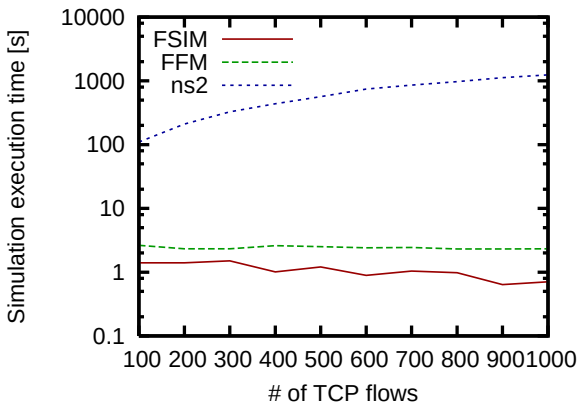

Fig. 6: Simulation execution time vs. the number of TCP flows

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 [7], which is one of the most popular packet-level simulators.

Advantages of compatibility with the popular packet-level simulator is countless. For instance, a simulation scenario file can be easily developed using existing GUI-based scenario editor [16].

IV. Experiments

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

We compare performance of three simulators: our flow-level simulator FSIM, the conventional flow-level simulator FFM [17], and packet-level simulator ns-2 [7]. We performed simulations for the same topology and parameter configuration with those three simulators. The network topology used in all simulations is shown in Fig. 5. All TCP source hosts continuously transmit data to their corresponding TCP sinks. In all experiments, a computer with two Intel Pentium 4 3.06 [GHz] processors and 512 [MB] memory running Debian GNU/Linux 3.1 (kernel version 2.4.32) 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 (i.e., simulation execution time and maximum memory consumption). In the following figures, confidence intervals are not shown because they were sufficiently small in all experiments.

We first investigate simulation speeds of flow-level simulators and the packet-level simulator. With flow-level simulators (FSIM and FFM) and the packet-level simulator, simulation execution times required for performing 50 [s] simulation are measured. Figure 6 shows simulation execution times of three simulators as a function of the number of TCP flows. This figure shows that speeds of flow-level simulators are dramatically faster than that of the packet-level simulator. One can find that the simulation execution time of the packet-level simulator increases as the number of TCP flows increase. On the contrary, simulation execution times of flow-level simulators hardly increases even 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 flow. It should be noted that a dumbbell topology like Fig. 5 is quite advantageous to flow-level simulators supporting flow-aggregation. With complex network topologies, performance benefit of flow-aggregation is not so significant, so that the simulation execution time should be affected by the number of TCP flows.

For clarifying the efficiency of the adaptive stepsize control used in FSIM, simulation execution times of flow-level simulators, FSIM and FFM, are shown in Fig. 7, where the number of TCP flows is fixed at 1,000 and the duration of simulation is ranged from 100 to 1,000 [s]. This figure shows that our flow-level simulator FSIM outperforms the conventional flow-level simulator FFM; i.e., the simulation execution time of FSIM realizes approximately 100% faster simulation than FFM.

We then evaluate simulation accuracy of our flow-level simulator FSIM. The network topology shown in Fig. 5 with 20 TCP flows is used. With three network simulators (i.e., FSIM, FFM, and ns2), TCP window sizes, the queue length of the RED router, and the packet loss probability at the RED router are measured (Figs. 8 through 10). Figure 8

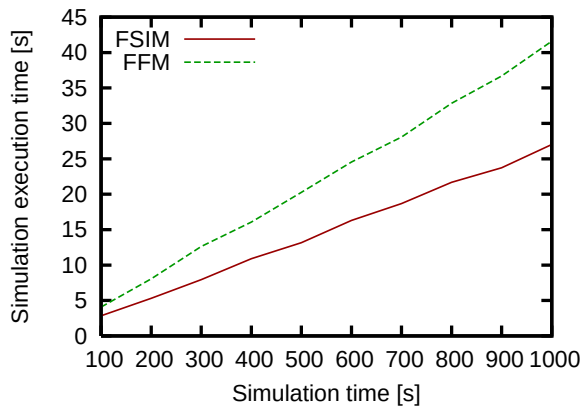

Fig. 7: Simulation execution time vs. simulation time

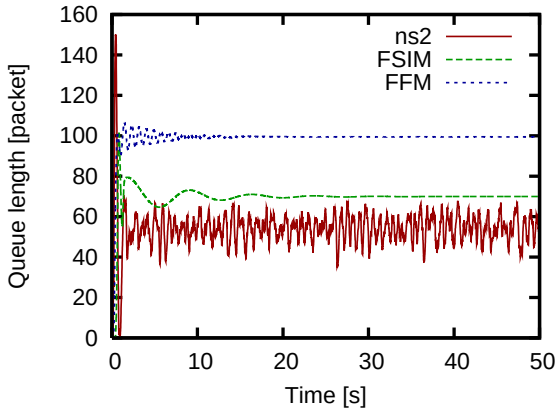

Fig. 9: Evolution of queue length of RED router

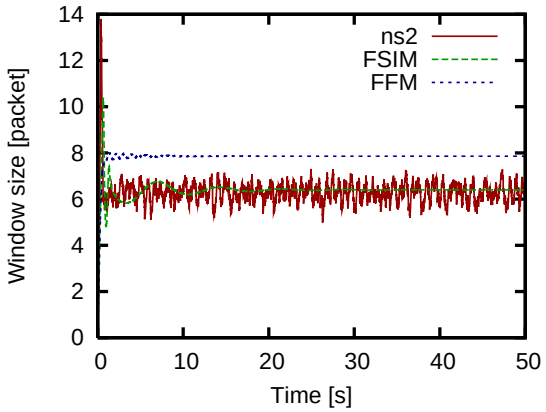

Fig. 8: Evolution of TCP window size

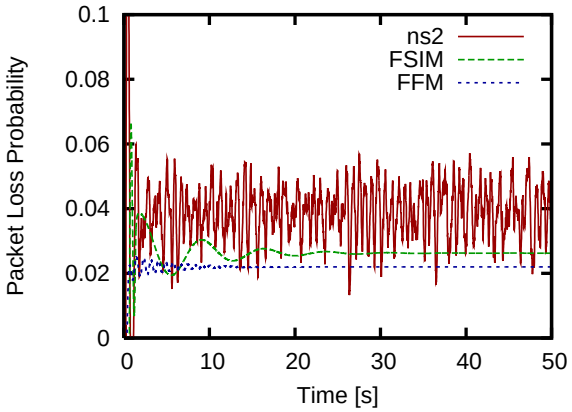

Fig. 10: Evolution of packet loss probability at RED router

illustrates evolution of the window size of a TCP flow. One can find from this figure that simulation result obtained with FSIM almost coincides with that of the packet-level simulator ns2. One can also find that simulation result obtained with FFM is approximately 30% larger than that of the packet-level simulator ns2. Similar tendency can be observed in other figures: the queue length (Fig. 9) and the packet loss probability (Fig. 10). This clearly indicates high accuracy of our flow-level simulator FSIM. Note that as discussed above, our flow-level simulator is significantly faster than FFM (see Fig. 7). Namely, our flow-level simulator FSIM is faster *and* more accurate than FFM; i.e., our flow-level simulator FSIM succeeds to realize high accuracy without compromising simulation speed.

Finally, we investigate the memory consumption of flow-level simulators and the packet-level simulator. Scalability of network simulators is sometimes limited by the memory size required for executing simulation [18]. The network topology shown in Fig. 5 with varying number of TCP flows is used. Maximum memory consumptions (i.e., the maximum of statistically and dynamically allocated memory size) during simulation run are measured for flow-level simulators, FSIM and FFM, and the packet-level simulator ns2. Figure 11 shows flow-level simulators, FSIM and FFM, consume much less memory than the packet-level simulator. Note that the maximum memory consumption does not increase even when the number of TCP flows increases. Similar to the phenomenon observed in Fig. 11, this is because flow-level simulators support flow-aggregation. Note that the maximum memory consumption of FSIM is approximately 20% of FFM's maximum memory consumption. This suggests that for a given memory size, our flow-level simulator FSIM can simulate a larger network than FFM.

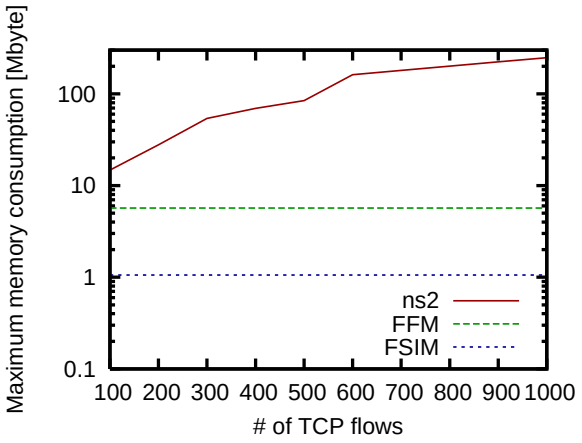

Fig. 11: Maximum memory consumption vs. the number of TCP flows

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 several 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 100% faster simulation with higher accuracy and less memory consumption than a conventional flow-level simulator.

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, HighSpeed TCP, and XCP utilizing fluid-flow models derived in [19]–[21].

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

Acknowledgements

The authors would like to thank Prof. Masayuki Murata for his insightful suggestions.

References

[1] "Hobbes' Internet timeline v6.1," available at http://www.zakon.org/robert/internet/timeline/.

[2] *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/iwg/lsn/lsn-workshop-12mar01/index.html.

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

[4] Y. Liu, F. L. Presti, V. Misra, D. Towsley, and Y. Gu, "Fluid models and solutions for large-scale IP networks," in *Proceedings of ACM/SIGMETRICS 2003*, June 2003, pp. 91–101.

[5] P. A., F. RM, and P. KS, "Conservative synchronization of large-scale network simulations," in *Proceedings of Parallel and Distributed Simulation 2004(PADS 2004)*, Mar. 2004, pp. 153–161.

[6] S. BK, L. Y, and G. R, "Parallel network simulation under distributed Genesis," in *Proceedings of Parallel and Distributed Simulation 2003(PADS 2003)*, June 2003, pp. 61–68.

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

[8] Y. GU, Y. Lie, and D. Towsley, "On integragin fluid model with packet simulation," in *Proceedings of IEEE INFOCOM 2004*, vol. 4, Mar. 2004, pp. 2856–2866.

[9] C. Kiddle, R. Simmonds, and B. Unger, "Improving scalability of network emulation through parallelism and abstraction," in *Proceedings of IEEE ANSS 2005*, Apr. 2005, pp. 119–129.

[10] 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 IEEE INFOCOM 2001*, vol. 3, Jan. 2001, pp. 22–26.

[11] J. Zhou, Z. Ji, M. Takai, and R. Bagrodia, "Maya: a multi-paradigm network modeling framework for emulating disributed applications," in *Proceedings of IEEE PADS 2003*, June 2003, pp. 162–170.

[12] B. Melamed, S. Pan, and Y. Wardi, "Hybrid discrete-continuous fluid-flow simulation," in *Proceedings of IEEE SPIE 2001*, vol. 4526, July 2001, pp. 263–270.

[13] H. Ohsaki, J. Ujiie, and M. Imase, "On scalable modeling of TCP congestion control mechanism for large-scale IP networks," in *Proceedings of IEEE SAINT 2005*, Feb. 2005, pp. 361–369.

[14] A. Kavimandan, W. Lee, M. T. A. Gokhale, and R. Viswanathan, "Netowrk simulation via hybrid system modeling: A time-stepped approach," in *Proceedings of ICCCN 2005*, Oct. 2005, pp. 531–536.

[15] W. H.Press, W. H.Vetterling, S. A.Teukolsky, and B. P.Flannery, *Numerical Recipes in C*. Cambridge University Press Cambridge, May 1993.

[16] "The network animator – NAM," available at http://www.isi.edu/nsnam/nam/.

[17] "Simulating large networks using fluid flow models (FFM)," available at http://www-net.cs.umass.edu/fluid/fluid.html.

[18] "The network simulator – ns2: Tips and statistical data for running large simulations in ns," http://www.isi.edu/nsnam/ns/ns-largesim.html.

[19] H. Hisamatu, H. Ohsaki, and M. Murata, "Fluid-based analysis of a network with DCCP connections and RED routers," in *Proceedings of the 2006 International Symposium on Applications and the Internet (SAINT 2006)*, Jan. 2006, pp. 22–29.

[20] H. Ohsaki, H. Yamamoto, and M. Imase, "Scalable modeling and performance evaluation of dynamic RED router using fluid-flow approximation," in *Proceedings of OpticsEast/ITCom 2005*, Oct. 2005.

[21] Y. Sakumoto, H. Ohsaki, and M. Imase, "On XCP stability in a heterogeneous network," in *Proceedings of IEEE Symposium on Computers and Communications(ISCC 2007)*, July 2007.