Accelerating Simulation of Large-Scale IP Networks: A Network Invariant Preserving Approach

Hwangnam Kim, Hyuk Lim, and Jennifer C. Hou
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801
Email: {hkim27, hyuklim, jhou}@cs.uiuc.edu

***Abstract*— In this paper, we propose a simulation framework, *TranSim*, that reduces the rate at which packet-events are generated, in order to accelerate large-scale simulation of IP networks with TCP/UDP traffic. Conceptually, we transform an IP network into an alternate network that generates a smaller number of packet events, carry out simulation in the "transformed" network, and then extrapolate simulation results for the original network from those obtained in the "transformed" network. We formally prove that if the network invariant — the *bandwidth-delay product* — is preserved, the network dynamics, such as the queue dynamics and the packet dropping probability at each link, and TCP dynamics, such as the congestion window, RTTs, and rate dynamics, remain unchanged in the process of network transformation.**

We have implemented *TranSim* in *ns-2*, and conducted an empirical study to evaluate it against packet level simulation, with respect to the capability of capturing transient, packet level network dynamics, the reduction in the execution time and the memory usage, and the discrepancy in the system throughput. The simulation results indicate maximally two orders of magnitude improvement in the execution time and the performance improvement becomes more prominent as the network size increases (in terms of the number of nodes, the number of flows, the complexity of topology, and link capacity) or as the degree of downsizing increases. The memory usage incurred in *TranSim* is comparable to that in packet level simulation. The error discrepancy between *TranSim* and packet level simulation, on the other hand, is between 1-10 % in a wide variety of network topologies, inclusive of randomly generated topologies, and traffic loads with various AQM strategies.

I. Introduction

Modern computer networks are extremely complex and do not lend well to theoretical analysis. A large number of (often heterogeneous) computer/network entities and techniques interacting and interfering with one another cannot be nicely fit into a framework of established optimization theories. As a result, it may be more feasible to carry out event-driven simulation to study the performance of network entities and protocols, and interaction among them. The major obstacle in packet level network simulation is, however, the vast number of events that have to be simulated in order to produce accurate results, especially for large-scale networks. This is because each packet will generate a number of events on its path from the source to the destination, e.g., arrival of a packet at the router, its departure, and its queuing, just to name a few. As the network size gets large and each node sends its packets (as governed by the transport layer protocol), the number of packets transported (and hence the number of events generated) easily becomes enormously large. As the CPU time required is roughly proportional to the number of events that have to be executed, packet level simulation easily becomes computationally expensive, if not infeasible, when the network size and/or the traffic amount is extremely large.

The notion of *fluid model based simulation* has been proposed to alleviate the computational overhead in packet level simulation [10], [13], [16], [22], [24]. Conceptually, a fluid model — a set of differential equations that characterize the network dynamics — is adopted and incorporated into the simulation engine. In the course of simulation, a sequence of closely-spaced packets are abstracted into a fluid chunk (usually characterized by the fluid rate) and the fluid model is used to describe the behavior of the simulation unit and to obtain the parameters of interest (e.g., the system throughput). In spite of its effectiveness in reducing the execution time, fluid model based simulation is built upon the assumption of the existence of a large number of active flows in the network, and hence is not well-suited for studying the network behavior under light and/or sporadic traffic. To address this issue, *network calculus*-based simulation was proposed by Kim and Hou [11] to characterize how TCP congestion control — additive increase and multiplicative decrease (AIMD) — interacts with AQM strategies with network calculus theory [2], [4]. In particular, they derive upper and lower bounds on the attainable TCP throughput, and then instrument the simulator to regulate TCP flows in compliance with the model. Although network calculus based simulation indeed gives encouraging results, it cannot provide packet level dynamics, such as the instantaneous queue length and the packet dropping probability, due to the use of network calculus. Note that fluid model based simulation also suffers from this shortcoming.

In this paper, different from the above theoretical-model-based approaches, we propose a simulation framework, *TranSim*, that reduces the rate at which packet events are generated by simulating a large-scale IP network with an alternate network that preserves certain *network invariants*. As mentioned above, the computational cost increases as the number of events increases with the network size and the amount of traffic. If we can *downsize* the network to one that generates a smaller number of packet events, carry out simulation in the downsized network to produce sufficient results, and extrapolate, without loss of accuracy, results for the original network, we can significantly reduce the computational cost. Note that by the *downsized* network, we mean a network in

The work reported in this paper was supported in part by NSF under Grant NSF CNS 03-05537 and MURI/AFOSR under contract F49620-00-1-0330.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the Proceedings IEEE Infocom.

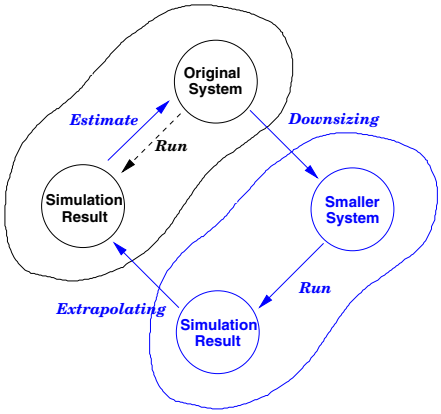


Fig. 1. The underlying concept of *TranSim*

which *a smaller number of packet events are generated*. Fig. 1 depicts the proposed simulation framework. The key issue is then what network properties should be preserved in the downsizing process so that simulation results for the original network can be correctly extrapolated from those obtained in the downsized network.

Even though many network properties can be used as network invariants during the operation of downsizing, we choose to use the *bandwidth-delay* product, as it represents the capacity of the "pipe," i.e., the amount of packets that can be transmitted and not yet acknowledged on the end-to-end path [21]. In the downsized network, we reduce the link capacity by a fraction α $(0 < \alpha \leq 1)$, and increase the link delay by $\frac{1}{\alpha}$, so as to keep the bandwidth-delay product invariant. The parameter α is termed as the downsizing parameter, and represents the degree of downsizing. In this manner, the rate at which packet-events are generated is reduced at each node in the network. Note that we change neither the number of nodes/flows nor the parameters related to the queue, e.g., the maximum buffer size or the active queue management (AQM) parameters.

We formally prove that the network capacity as perceived by each TCP connection is not changed by preserving this invariant during the downsizing process. In particular, we prove that the queue dynamics (e.g., the instantaneous queue length), the RTT dynamics, and the TCP window size dynamics, remain unchanged in the operations. In addition, we show that as far as the TCP throughput in the *steady state* is concerned, one does not have to change *external* event times (such as when a TCP connection commences and when it ends), although the downsizing operations delay, in principle, all the packet-events by $\frac{1}{\alpha}$ (because the link delay is increased by $\frac{1}{\alpha}$). Instead, one can carry out the simulation using the original set of event times given in the original network. In some sense, *TranSim* essentially simulates in the downsized network only the α-portion of an interval between two external events (specified by the user), and extrapolates the results in the entire period of time in the original network. By repeating this procedure in each interval until the end of simulation, one can obtain all the simulation results in the original network, including both queue dynamics and throughput.

We have carried out *ns-2* simulation to evaluate *TranSim* against packet level simulation, with respect to the capability of capturing the transient, packet level network dynamics, the execution time, the memory usage, and the discrepancy in simulation results. The simulation results indicate maximally two orders of magnitude in reducing the execution time, and the performance improvement becomes more prominent as the network size increases (in terms of the number of nodes or flows, the link capacity, and the complexity of network), or as the downsizing parameter decreases. The error discrepancy between *TranSim* and packet level simulation, on the other hand, is minimally 1-2 % and maximally 10 % in a wide variety of network topologies and traffic loads. The results also indicate that *TranSim* yields approximately the same amount of memory usage (as packet-level simulation) in both virtual and physical memory.

The rest of the paper is organized as follows. In Section II, we give a summary of existing work that aims to expedite network simulation. In Sections III–IV, we present the analytical model for *TranSim*, formally prove that it preserves network dynamics, and validate its correctness. Following that, we elaborate on how we implement *TranSim* in *ns-2*, and present our simulation results in Section V. The paper concludes with Section VI.

II. Related Work

In this section we summarize existing simulation techniques that aim to expedite network simulation, while retaining simulation accuracy.

Fluid model based simulation*:* Several research efforts have focused on fluid model based simulation. Liu *et al.* [13] demonstrated the fundamental performance gain in fluid model based simulation over, rather than a realistic network with detailed network protocols, simple network components. Milidrag *et al.* [16] presented various sets of differential equations that describe the behaviors of network components in the continuous time domain. They showed that as long as the behavioral characteristics in the continuous time domain can be exactly specified, fluid simulation gives results with reasonable error bounds. Wu *et al.* [24] studied the error behavior that simulation results exhibit in a simple $M/D/1$ network configuration.

Fluid models have also been used to study the throughput behavior of TCP and congestion control algorithms, together with active queue management in the steady state [18], [19], [22]. Liu *et.al.* [14] and Gu *et al.* [7] solved fluid models with the Runge-Kutta numerical method, and incorporated numerical results in the simulation of large-scale IP networks. Kim and Hou [10] investigated the feasibility of fluid model based simulation for simulating IEEE 802.11-operated wireless LANs (WLANs). They developed a throughput model to describe data transmission activities in wireless LANs and implemented fluid model based simulation in *ns-2*. Their results showed two orders of a magnitude improvement with acceptable error bounds. As indicated in Section I, fluid model based simulation may not render satisfactory performance (in terms of the discrepancy between results obtained in packet level simulation and fluid model based simulation) in the case of light and/or sporadic traffic, as it relies on the assumption of existence of a large number of flows [14], [18], [22]. Also,

it cannot fully characterize kinetic transient behaviors in the network.

Network calculus based simulation*:* Kim and Hou [11] examine the feasibility of incorporating network calculus models in simulating TCP/IP networks. By exploiting network calculus properties, they characterize how TCP congestion control — additive increase and multiplicative decrease (AIMD) — interact with the AQM strategies in the analytic model, and regulate TCP flows in a simulation engine with the derived model. They show that as compared to time stepped hybrid fluid simulation (TSHS), significant improvement can be made in expediting the simulation, while keeping the error discrepancy reasonably small. As indicated in Section I, although network calculus based simulation gives accurate steady state system throughput, it cannot give, due to the nature of network calculus, the transient behavior of the network, e.g., the instantaneous queue length and the packet dropping probability at each bottleneck link.

Simulation based on scaling the network*:* The notion of scaling down/up the network to facilitate network monitoring and performance prediction has been proposed by Pan *et al.* [20] in their *Small-scale Hi-fidelity Reproduction of Network Kinetics (SHRiNK)* work. Specifically, two SHRiNK methods have been introduced to sample, simulate, and predict the network behavior. The first method deals with TCP networks with long-lived flows and the other with a mixture of long-lived and short-lived flows. Both methods reduce the number of data packets by independently sampling real traffic in the original network, and then simulate in a down-scaled network with the sampled traffic in order to predict the behavior in the original network.

In the first method, the number of flows, the link capacity, and the maximum buffer size along with AQM parameters (e.g. the maximum/minimum thresholds in RED used at each link) are proportionally down-scaled, while *keeping the end-to-end delay unchanged*. In order to keep the end-to-end delay invariant, the queue dynamics has to be approximated (by certain linear functions), and thus the queues may respond to each TCP connection differently from what they would in the original network. As a result, the network capacity as perceived by each TCP connection during its interaction with the network is changed. In the cases where the queue dynamics cannot be adequately approximated with linear functions, e.g., when Drop Tail is used as the queue management strategy, SHRiNK cannot operate appropriately. Moreover, the degree of down-scaling is limited by the maximum buffer size or the number of flows since these two parameters cannot be less than 1, i.e., the scaling parameter should be set to a value that is larger than both $1/L$ and $1/N$, where L is the minimum of the maximum buffer sizes in the network and N is the number of flows.

The second SHRiNK method resembles *TranSim* in that the link capacity is reduced and the link delay is proportionally increased. However, the method was presented in [20] without theoretical reasoning. Both SHRiNK methods rely heavily on the assumptions that each input flow can be modeled as a Poisson process in order not to change the property of the original input traffic. Finally, SHRiNK does not deal with the case in which TCP and unresponsive (UDP) flows co-exist, but assumes that all the traffic are TCP flows.

As compared with SHRiNK, *TranSim* can be uniformly applied to both long-lived and shorted-lived flows or a mixture therein. It do not change the queue dynamics (which will be rigorously proved in Section III), and its correctness does not rely on any assumption about the input traffic. *TranSim* can also deal with both UDP and TCP traffic over IP networks.

III. Analytical Model for *TranSim*

In this section, we present the analytical model for *TranSim*. The key idea of the model is to preserve the network capacity (as perceived by TCP connections and determined by the queue dynamics) during the operation of downsizing the network. We first discuss the network invariant. Then we introduce the analytical model that serves as the theoretical base for *TranSim*.

A. Network Invariant

Our objective is to transform a large-scale network into a downsized one, carry out simulation in the downsized network, and to infer the simulation results of the original network (based on those obtained in the downsized network). As mentioned in Section I, the transformation is performed so that the number of packet-events required to be generated/processed in the downsized network is reduced, thus expediting network simulation. For example, if we reduce the link capacity at each link, we can reduce the number of sending and receiving events at the link. Similarly, if we increase the link delay at each link, we can reduce the sending rate of each TCP connection (as a result of increased round trip times).

One important issue is then how to downsize the network so that the simulation results obtained from the downsized network can be used to accurately infer the performance of the original network. We claim that the operation of downsizing should not change the network capacity as perceived by TCP connections. Note that the network capacity is determined by the network dynamics (such as the instantaneous queue length and the dropping probability) and reflects upon the throughput TCP connections can attain. Hence, we use the *bandwidth-delay* product as the network invariant to be preserved during the downsizing process. Since the bandwidth-delay product represents the number of data packets that can be in transit [21], it can be regarded as the TCP window size from the perspective of a TCP connection.

Specifically, let B and D denote, respectively, the available bandwidth and delay along a path in the original network, and P_{BDP} the bandwidth-delay product along the path. For notational convenience, we denote the corresponding variables in the downsized network by attaching an apostrophe to the variables, i.e., B', D', and P'_{BDP}. The following constraint is used in the operation of downsizing the network:

$$P_{BDP} = B \cdot D = B' \cdot D' = P'_{BDP}. \quad (1)$$

With Eq. (1), the downsizing parameter, α, is defined as

$$B' = \alpha \cdot B, \quad (2)$$

$$D' = \frac{D}{\alpha}, \quad (3)$$

where $0 < \alpha \leq 1$.

Note that we do not change the number of nodes, the number of flows, or the queue-related parameters (e.g., the maximum buffer size or the AQM parameters). The only parameters that are changed in the downsizing operation are the link capacity and the link delay. As will be formally proved in Section III-B, by keeping P_{BDP} invariant, the queue dynamics (e.g., the instantaneous queue length), the RTT dynamics, and the window size dynamics remain unchanged in the operation. As a result, the downsized network exhibits exactly the same behavior as the original network.

B. Analytical Model that Supports TranSim

We now present the analytical model for *TranSim*. Let N denote the number of flows sharing a path with the bottleneck link of capacity C. Let $q(t)$ and $p(t)$ denote, respectively, the queue length and the packet dropping probability of the bottleneck link at time t, and T the propagation delay of the path. Let $W_i(t)$ and $R_i(t)$ denote, respectively, the window size and the round trip time of flow i at time t. Then the interaction between TCP flows and the bottleneck link can be characterized with the TCP model given in [17]:

$$R_i(t) = T + \frac{q(t)}{C}, \tag{4}$$

$$\frac{dq(t)}{dt} = \sum_{i=1}^{N} \frac{W_i(t)}{R_i(t)} - C, \tag{5}$$

$$\frac{dW_i(t)}{dt} = a \cdot \frac{1}{R_i(t)} - b \cdot W_i(t) \cdot \frac{W_i(\tau_i)}{R_i(\tau_i)} \cdot p(\tau_i), \tag{6}$$

where a is the additive increase parameter, b is the multiplicative parameter, and $\tau_i = t_i - R_i(t)$.

TranSim reduces the link bandwidth and increases the link delay by the downsizing parameter, α (Eqs. (2) and (3)). Since each packet is served at the rate of $\alpha \cdot B$ (which stretches its transmission time by the factor of $\frac{1}{\alpha}$) and experiences $\frac{1}{\alpha}$ times larger delay, the time instants at which packet-events occur are also delayed by $\frac{1}{\alpha}$. Specifically, an (internal) packet-event that occurs at time t in the original network is delayed to t' in the downsized network as follows:

$$t' = \frac{1}{\alpha} \cdot t, \tag{7}$$

where $t (t \geq 0)$ and $t' (t' \geq 0)$ denote, respectively, a time instant for the original network and for the downsized network.

In what follows, we will investigate the dynamics of the queue length, the round trip time, and the TCP window size in both the downsized network and the original network. Again we denote the corresponding variables in the downsized network by attaching an apostrophe to the variables.

1) **Queue Dynamics:** We show that the queue length of the bottleneck link in the original network is the same to that in the downsized network:

$$q'(t') = q(t).$$

This implies that the queue responds to each TCP connection in the downsized network in exactly the same manner (including the packet dropping probability) as in the original network. Moreover, this is achieved without modifying or approximating any AQM parameter in the downsized network. We first look at $\frac{dq'(t')}{dt}$:

$$\begin{aligned}\frac{dq'(t')}{dt} &= \sum_{i=1}^{N} \frac{P'_{i,BDP}(t')}{D'_i(t')} - C' = \sum_{i=1}^{N} \frac{P_{i,BDP}(t)}{\frac{D_i(t)}{\alpha}} - \alpha \cdot C \\ &= \alpha \cdot \left\{ \sum_{i=1}^{N} \frac{P_{i,BDP}(t)}{D_i(t)} - C \right\} = \alpha \cdot \frac{dq(t)}{dt} = \frac{dq(t)}{\frac{1}{\alpha} \cdot dt}.\end{aligned} \tag{8}$$

Note that in Eq. (8), the change in the queue size is simply the difference between the arrival rate of all flows and the link capacity, and the arrival rate of a flow i is obtained by dividing its current bandwidth-delay product ($P_{i,BDP}(t)$) by its current delay ($D_i(t)$).

Since $t' = \frac{1}{\alpha} \cdot t$ (Eq. (7)), $dt' = \frac{1}{\alpha} dt$, and thus,

$$\frac{dq'(t')}{dt} = \frac{dq(t)}{dt'},$$

or

$$dq'(t') \cdot dt' = dq(t) \cdot dt. \tag{9}$$

Since $q'(0) = q(0) = 0$ and Eq. (9) is a separable first order equation [5], we obtain the following result by integrating both sides of Eq. (9) when $t, t' \geq 0$:

$$q'(t') = q(t). \tag{10}$$

By Eq. (10), we know as long as the downsizing operation preserves the *bandwidth-delay* product of the original network, the queue dynamics in both the original network and the downsized network are the same at each packet-event time.

2) **RTT Dynamics:** We can compute the round trip time in the time domain of the downsized network, t', as follows. Since $R_i(t) = T + \frac{q(t)}{C}$,

$$\begin{aligned}R'_i(t') &= T' + \frac{q'(t')}{C'} = \frac{T}{\alpha} + \frac{q(t)}{\alpha \cdot C} = \frac{1}{\alpha} \cdot (T + \frac{q(t)}{C}) \\ &= \frac{1}{\alpha} R_i(t).\end{aligned} \tag{11}$$

Note that we have used $q'(t') = q(t)$ in the second equality in Eq. (11). With Eq. (11), we can now re-express $\frac{dq'(t')}{dt}$ in

$$\begin{aligned}\frac{dq'(t')}{dt} &= \sum_{i=1}^{N} \frac{W'_i(t')}{R'_i(\tau'_i)} - C' = \sum_{i=1}^{N} \frac{W_i(t)}{\frac{R_i(\tau_i)}{\alpha}} - \alpha \cdot C \\ &= \alpha \cdot \left\{ \sum_{i=1}^{N} \frac{W_i(t)}{R_i(\tau_i)} - C \right\} = \alpha \cdot \frac{dq(t)}{dt} = \frac{dq(t)}{dt'},\end{aligned}$$

where the first equality results from the fact that the current TCP window size ($W_i(t)$) can be represented by the current bandwidth-delay product ($P_{i,BDP}(t)$) as perceived by each TCP connection i.

3) **Window Size Dynamics:** Eq. (11) implies that each TCP connection in the downsized network exhibits the same window dynamics but the response is $\frac{1}{\alpha}$ times slower than that in the original network (since a TCP connection in the downsized network adjusts its rate per round trip time $R'_i(t)$). To further verify this, we express the window dynamics in the downsized network based on Eq. (6) as follows:

$$\frac{dW'_i(t')}{dt'} = a \cdot \frac{1}{R'_i(t')} - b \cdot W'_i(t') \cdot \frac{W'_i(\tau'_i)}{R'_i(\tau'_i)} \cdot p(\tau'_i), \tag{12}$$

where $\tau_i' = t' - R_i'(t')$.

By plugging Eq. (11) into Eq. (12), we have

$$\begin{aligned}\frac{dW_i'(t')}{dt'} &= a \cdot \frac{1}{\frac{1}{\alpha} \cdot R_i(t)} - b \cdot W_i(t) \cdot \frac{W_i(\tau_i)}{\frac{1}{\alpha} \cdot R_i(\tau_i)} p(\tau_i) \\ &= \alpha \cdot \left\{ a \cdot \frac{1}{R_i(t)} - b \cdot W_i(t) \cdot \frac{W_i(\tau_i)}{R_i(\tau_i)} \cdot p(\tau_i) \right\} \\ &= \alpha \cdot \frac{dW_i(t)}{dt}, \end{aligned} \tag{13}$$

where $\tau_i = t - R_i(t)$. Note that in the first equality of Eq. (13), we have used the fact that the current bandwidth-delay product on the end-to-end path can be viewed as the current window size, and hence $W_i'(x') = W_i(x)$, for $\forall x' = \frac{1}{\alpha} \cdot x$. Additionally, $\tau_i' = \frac{1}{\alpha} \cdot \tau_i$ since $R_i'(t') = \frac{1}{\alpha} \cdot R_i(t)$ (Eq. (11)), and $p(t) = p'(t')$ since $q'(t') = q(t)$ (Eq. (10)), for $\forall t' = \frac{1}{\alpha} \cdot t$.

As implied in Eqs. (11) and (13), all the packet-events that occur in the original network are delayed by the factor of $\frac{1}{\alpha}$ in the downsized network. This is consistent with Eq. (7).

4) **TCP Dynamics in the Slow Start Phase:** In the TCP *slow start* phase, a TCP connection exponentially increases its congestion window until either packet loss occurs or the slow start threshold is reached, whichever occurs first. Let $W_{i,ss}(t)$ be the instantaneous window size at time t in the *slow start* phase. Then, $W_{i,ss}$ can be expressed as:

$$\begin{aligned} W_{i,ss}(0) &= 1, \text{ and} \\ W_{i,ss}(t) &= 2 \cdot W_{i,ss}(t - R_i(t)) = 2^{\left\lfloor \frac{t}{R_i(t)} \right\rfloor}, \quad \text{for } t > 0. \end{aligned}$$

We first investigate the TCP window dynamics in the slow start phase. The TCP window size at time t depends on the number of successful transmissions until time t. We appropriate $R_i(t_j) \approx R_i(t_k)$ for all $0 < t_j \leq t$ and $0 < t_k \leq t$ since in the slow start phase no packet loss is incurred, and all the packets experience approximately the same queuing time (if any). The window dynamics in the downsized network can be expressed as follows:

$$\begin{aligned}\frac{dW_{i,ss}'(t')}{dt'} &= \frac{d}{dt'} \left(2^{\left\lfloor \frac{t'}{R_i'(t')} \right\rfloor} \right) = \frac{d}{\frac{1}{\alpha} dt} \left(2^{\left\lfloor \frac{\frac{1}{\alpha} \cdot t}{\frac{1}{\alpha} \cdot R_i(t)} \right\rfloor} \right) \\ &= \alpha \cdot \frac{d}{dt} \left(2^{\left\lfloor \frac{t}{R_i(t)} \right\rfloor} \right) = \alpha \cdot \frac{dW_{i,ss}(t)}{dt}. \end{aligned} \tag{14}$$

By Eq. (14), we know that the dynamics of the TCP congestion window in the slow start phase in the downsized network is slowed down by the parameter of $\frac{1}{\alpha}$, as compared to that in the original network.

Next we prove the cumulative throughput attained by a TCP connection in the slow start phase in the downsized network is equal to that in the original network. Note that the *cumulative throughput* $Y_i(t_i, t_{i+1})$ represents the number of packets sent by the flow, i, in time interval $[t_i, t_{i+1}]$. Let $r_i(t)$ be the rate function; then $Y_i(t_i, t_{i+1})$ can be expressed as $Y_i(t_i, t_{i+1}) = \int_{t_i}^{t_{i+1}} dr_i(t)$.

Let $Y_{i,ss}(t)$ be the cumulative throughput attained until time t in the slow start phase. ($Y_{i,ss}(t) \stackrel{\triangle}{=} Y_{i,ss}(0, t)$.) Then $Y_{i,ss}'(t')$ can be expressed as

$$\begin{aligned} Y_{i,ss}'(t') &\approx 1 + 2 + 2^2 + \cdots + 2^{\left\lfloor \frac{t'}{R_i'(t')} \right\rfloor} = \sum_{k=0}^{\left\lfloor \frac{t'}{R_i'(t')} \right\rfloor} 2^k \\ &= 2^{\left\lfloor \frac{t'}{R_i'(t')} \right\rfloor} - 1 = 2^{\left\lfloor \frac{\frac{1}{\alpha} t}{\frac{1}{\alpha} R_i(t)} \right\rfloor} - 1 \\ &\approx Y_{i,ss}(t). \end{aligned} \tag{15}$$

5) **Transforming of Unresponsive Traffic:** In the case that unresponsive flows (e.g., UDP traffic) exist in the network, as no rate adaptation scheme such as the TCP AIMD mechanism (Eq. (13)) is used, we have to forcefully adjust the rate of unresponsive traffic in the downsized network, so as to handle them in a consistent manner with Eqs. (7), (11), (13), and (14).

Specifically, let $\sigma_i(t)$ denote the rate function of a unresponsive flow i used in the original network. Since we reduce the network bandwidth by a downsizing parameter of α, the rate function of flow i in the downsized network should also be reduced by the same parameter, i.e., $\alpha \cdot \sigma_i(t)$. With such an adjustment, unresponsive flows introduce the same degree of aggressiveness in the downsized network as they do in the original network.

6) **Event Times in Transformed/Original Networks:** Sometimes one would like to obtain the cumulative throughput attained by a TCP connection during a *specific* interval in the steady state, say $[t_0, t_n]$, in the original network. As all the packet events in the downsized network are delayed by a factor of $\frac{1}{\alpha}$, one would expect to carry out simulation and measure the throughput attained by the TCP connection in $[\frac{t_0}{\alpha}, \frac{t_n}{\alpha}]$. In what follows, we show this is not necessary. That is, from the perspective of obtaining the attainable throughput in the steady state, one can carry out simulation in the same interval $[t_0, t_n]$ in the downsized network and then extrapolate the results in the original network.

Let t_k for $k = 0, 1, 2, \cdots, n$ represent the n event-times at which events are externally scheduled by users, and let $Y_i(t_{k-1}, t_k)$ denote the cumulative throughput attained by a TCP connection i in the interval $[t_{k-1}, t_k]$ in the original network. Then the total cumulative throughput attained by flow i can be expressed as

$$Y_i(t_0, t_n) = \sum_{k=1}^{n} Y_i(t_{k-1}, t_k). \tag{16}$$

The cumulative throughput, $Y_i'(t_{k-1}, t_k)$ attained by a TCP flow during $[t_{k-1}, t_k]$, in the downsized network can be expressed as

$$\begin{aligned} Y_i'(t_{k-1}, t_k) &= \int_{t_{k-1}}^{t_k} \left(\frac{dW_i'(t')}{R_i'(t')} \right) \\ &= \int_{t_{k-1}}^{t_k} \left(\frac{dW_i(t)}{\frac{1}{\alpha} R_i(t)} \right) = \alpha \cdot \int_{t_{i-1}}^{t_k} \left(\frac{dW_i(t)}{R_i(t)} \right) \\ &= \alpha \cdot Y_i(t_{k-1}, t_k). \end{aligned}$$

Therefore, the total cumulative throughput attained by the TCP flow i during $[t_0, t_n]$ in the downsized network can be

expressed as

$$Y_i'(t_0,t_n) = \sum_{k=1}^{n} Y_i'(t_{k-1},t_k)$$
$$= \alpha \cdot \sum_{k=1}^{n} Y_i(t_{k-1},t_k) = \alpha \cdot Y_i(t_0,t_n). \quad (17)$$

Eq. (17) implies that from the perspective of obtaining the cumulative TCP throughput *in the steady state,* one does not have to carry out simulation for a period of time prolonged by the factor $\frac{1}{\alpha}$. Instead, one simply carries out simulation in the downsized network using the same set of time instants, $t_k, k \geq 0$, given by users, obtains simulation results in each interval $[t_{k-1}, t_k]$, and then infers (by extrapolating) simulation results for the same period in the original network.

As a matter of fact, what has been simulated in the downsized network is *the α portion of the interval between two external events* in the original network. Specifically, suppose two external (user-given) events e_1 and e_2 occur at t_1 and t_2 in the original network. The results obtained in $[t_1, t_2]$ in the downsized network are actually those in $[t_1, t_1 + \alpha \cdot (t_2 - t_1)]$ in the original network. As the results in $[t_1, t_1 + \alpha \cdot (t_2 - t_1)]$ represent the α portion of the results in $[t_1, t_2]$, the latter can be inferred accordingly. In this manner, one only needs to measure the throughput attained by TCP connections in $[t_i, t_{i+1}]$, $0 \leq i \leq n-1$, in the downsized network, infer the corresponding results in each interval in the original network, and repeat the procedure in each interval to obtain results for the entire simulation period.

C. Comparison with SHRiNK

As compared to SHRiNK [20], *TranSim* possesses several desirable features: (i) *TranSim* does not make any assumption on the input traffic, while SHRiNK relies on the Poisson assumption for the input traffic; (ii) *TranSim* preserves the queue dynamics and hence the network capacity as perceived by TCP connections. In contrast, SHRiNK approximates the queue dynamics in the downsized network in order to preserve the end-to-end queuing delay (round trip time). As a result, the queues may respond to a TCP connection differently from what they would do in the original network; (iii) *TranSim* can work together with any AQM strategy, while SHRiNK cannot be used if the modified queue dynamics cannot keep the queuing delay invariant in the downsized network. As reported in [20], if the AQM strategy is Drop Tail, SHRiNK cannot produce accurate results; (iv) There is no restriction on the downsizing parameter α in *TranSim*, while the scaling parameter in SHRiNK cannot be too small. This is because SHRiNK reduces the number of flows and the buffer size according to this parameter, and hence it has to be larger than both the inverse of the number of flows and the inverse of the minimum value of the buffer sizes in the network; and (v) the correctness of *TranSim* has been established for short-lived flows (which lasts only in the slow start phase), long-lived flows, and unresponsive flows (UDP flows), while that of SHRiNK was only established for the case with long-lived flows.

Table I gives a quantitative comparison between *TranSim* and SHRiNK. The network configuration under which the simulation is carried out is given in Fig. 2. The buffer size is set to 100 packets and RED is used as the buffer management strategy. The minimum threshold, the maximum threshold, the maximum drop probability, and the gentle option of RED are set to 30, 70, 1, and *enabled*, respectively. (The row with the downsizing parameter set to 1 corresponds to the results in packet level simulation.) As shown in Table I, the execution time incurred in *TranSim* reduces as the downsizing parameter decreases while SHRiNK cannot produce correct results (in case of parameter 0.1) and even cannot proceed as the scaling parameter becomes smaller. This is, in part, due to the fact that *TranSim* changes neither the number of flows nor the maximum buffer size at each link in its downsizing operation, while SHRiNK reduces both.

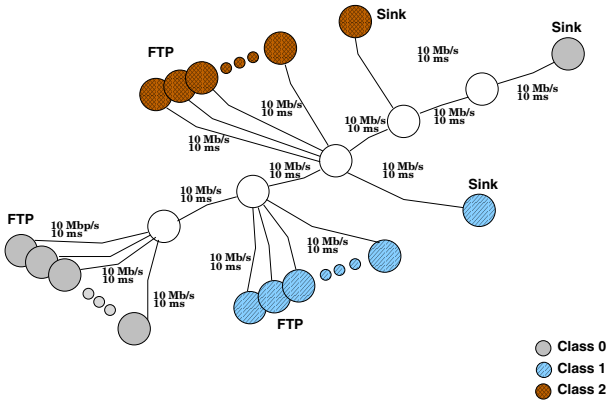


Fig. 2. The network configuration used in the comparison with SHRiNK

IV. Validation of *TranSim* with MATLAB

In this section, we validate *TranSim* with *MATLAB* [23]. Recall that all the packet events that occur at time $t (t \geq 0)$ in the original network are delayed to $t'(t' \geq 0) = \frac{1}{\alpha} \cdot t$ in the downsized network. This is attributed to the fact that the link bandwidth is reduced by the factor of α and the link delay is stretched by the factor of $\frac{1}{\alpha}$ during the downsizing operation. The dynamic network behavior that the original network exhibits at any time instant can be observed in the downsized network, except that the time instant at which the behavior occurs is delayed by the factor $\frac{1}{\alpha}$ (Eqs. (13) and (7)). In what follows we validate Eqs. (13) and (7), under the assumption that the TCP dynamics in Eqs. (4)–(6) are correct, by inspecting the trajectory of the queue length and the transient behavior of the dropping probability at a link.

The network topology in which we validate *TranSim* is a simple dumbbell network topology. The link capacity, link delay, and buffer size, of the bottleneck link are set to 1000 packets/second, 100 ms, and 100 packets, respectively. RED is used as the AQM strategy. The minimum threshold, the maximum threshold, the maximum drop probability, and the gentle option of RED are set to 50, 100, 1, and *enabled*, respectively. A total of 100 TCP connections traverse the bottleneck link.

Figs. 3 and 4 give, respectively, the kinetic dropping probability and the instantaneous queue length on the bottleneck link. Note that all the packet events presented in both figures

TABLE I

THE NUMBER OF PACKETS RECEIVED PER CLASS AND THE EXECUTION TIME (SEC.) IN A 1000-SECOND SIMULATION RUN IN *TranSim* AND SHRINK. THE NETWORK CONFIGURATION IS GIVEN IN FIG. 2, IN WHICH THE BUFFER SIZE IS SET TO 100 AND RED IS USED AS THE BUFFER MANAGEMENT STRATEGY. THE MINIMUM THRESHOLD, THE MAXIMUM THRESHOLD, THE MAXIMUM DROP PROBABILITY, AND THE GENTLE OPTION OF RED ARE SET TO 30, 70, 1, AND ENABLED, RESPECTIVELY.

# of nodes per class	parameter	*TranSim*				SHRiNK			
		class 0	class 1	class 2	time	class 0	class 1	class 2	time
20	1.0	520131	1671982	1676905	166.11	520131	1671982	1676905	166.11
	0.2	491415	1492095	1512170	32.29	506540	1570700	1604325	24.27
	0.1	490290	1537490	1529750	17.47	132890	1819520	1812290	10.24
	0.01	482000	1506000	1535100	3.80	N/A			
40	1.0	522755	1671982	1676905	189.99	522755	1671982	1676905	189.99
	0.2	473870	1596565	1620595	38.97	526885	1684530	1711410	28.16
	0.1	455530	1665950	1695290	22.75	170330	1959670	1965020	12.20
	0.01	421100	1671300	1724200	6.59	N/A			

are internal and fall in the α-portion of the interval determined by two external (user-specified) events. Fig. 3 (a) gives the trajectories of the packet dropping probability of the link in the downsized network, with each trajectory corresponding to a specific value of the downsizing parameter α[1]. As expected, packet events are delayed by the factor of $\frac{1}{\alpha}$. Fig. 3 (b) gives the trajectories of the packet dropping probability after the network is extrapolated (multiplied) with α. All the trajectories agree with one another. Fig. 4 gives the instantaneous queue length of the link in the downsized network ((a)) and after the network is extrapolated ((b)). Conclusions similar to those made in Fig. 3 can be drawn.

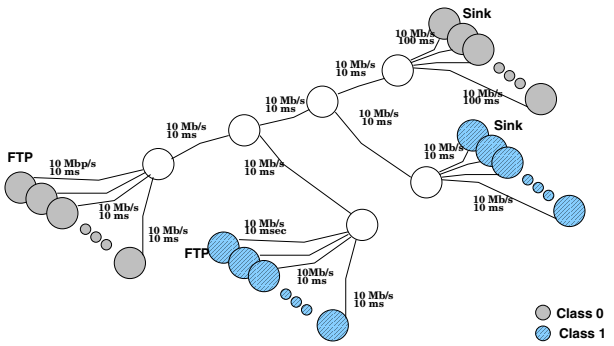


Fig. 5. The network configuration used in the first set of experiments.

V. SIMULATION STUDY

In this section, we discuss how we implement *TranSim* in *ns-2* [3] and conduct a simulation study to compare *TranSim* against packet level simulation with respect to the network dynamics (e.g., the queue dynamics) and the steady-state behavior (e.g., the TCP throughput). The performance metrics used in the simulation study are the execution time (wall time) that it takes to carry out simulation of a fixed duration and the memory usage thus incurred. The execution time consists of the loading time (the time it takes to load all the simulation modules) and the running time (the time it takes to run all the simulation tasks after the modules are loaded). The reason why we use the execution time instead of the number of events as the metrics is because each event consumes a different amount of processor time. For memory usage, we measure both the maximum virtual memory size and the maximum physical memory size (RSS) used in each simulation run.

A. Implementation

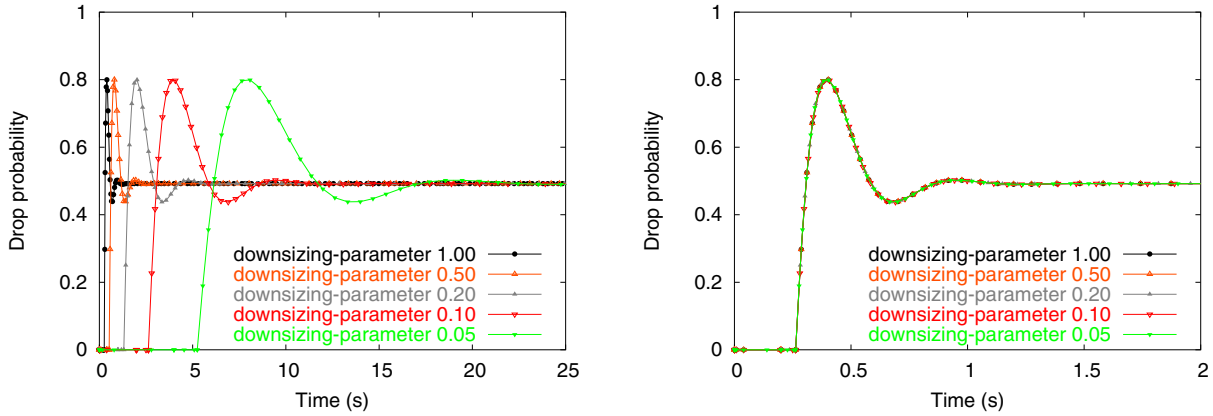
Recall that the key idea of *TranSim* is to reduce the number of packet-events per unit time in the simulation by downsizing the original network into an alternate network, while preserving the bandwidth-delay product invariant in the downsizing operation (Section III). As a result, neither the number of flows nor the queue-related parameters (e.g., the maximum buffer size and AQM parameters) need to be changed. As a matter of fact, the implementation of *TranSim* is quite simple and straightforward. The network simulator takes as input the network topology and the downsizing parameter, downsizes the original network, carries out packet level simulation in the downsized network, and then extrapolates simulation results that correspond to the original network. Only a preprocessor and a post-processor are needed to downsize the network and to extrapolate the simulation results.

[1]The curve labeled as "downsizing-parameter 1.0" denotes the results observed in the original network.

B. Simulation

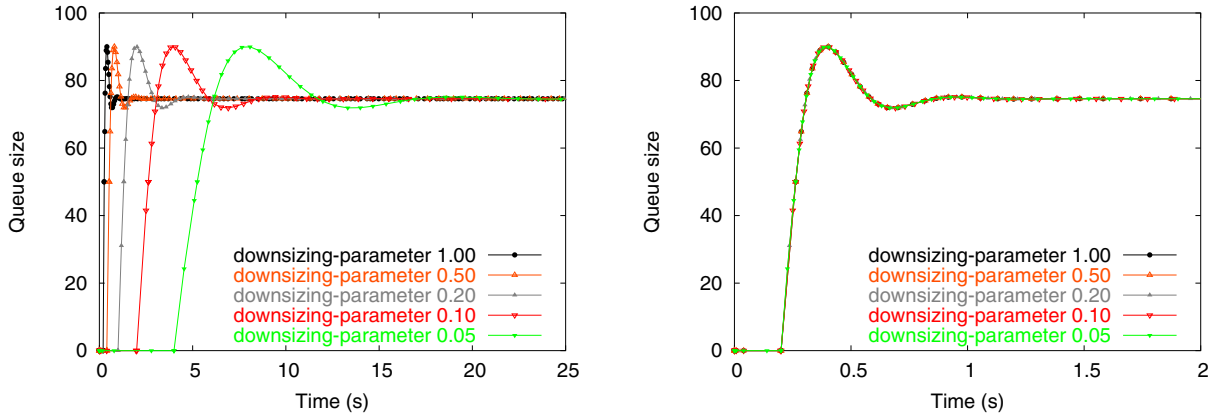
We have carried out an extensive *ns-2.1b9a* simulation study in a wide variety of network topologies and traffic loads, to assess the effectiveness of *TranSim* in improving simulation performance and in capturing transient, packet level network dynamics. Due to the space limit, in what follows we present representative simulation results obtained in the network configuration given in Fig. 5 and in Internet-like network topologies generated by *BRITE* [15]. (The complete set of simulation results can be found in [9].)

Each link in the network configuration is equipped with a buffer of size 100 packets, with the default packet size of 500 bytes. Different AQM strategies are employed at the links in different simulation runs. The various parameters in the AQM strategies are selected as follows: (i) the minimum threshold, maximum threshold, maximum drop probability, and gentle option of RED [6] are set to 30, 70, 1, and *enabled*, respectively; (ii) the update interval, reference value, and gain of REM [1] are set to 10 *ms*, 50, and 0.1, respectively; (iii) the reference queue size and sampling frequency of PI are set to 50 and 100 times per second, and the remaining parameters,



(a) In the downsized network (b) After extrapolation

Fig. 3. Trajectories of the packet dropping probability in the downsized network and after the network is extrapolated with the same downsizing parameter α.



(a) In the downsized network (b) After extrapolation

Fig. 4. Trajectories of the instantaneous queue length in the downsized network and after the network is extrapolated with the same downsizing parameter α.

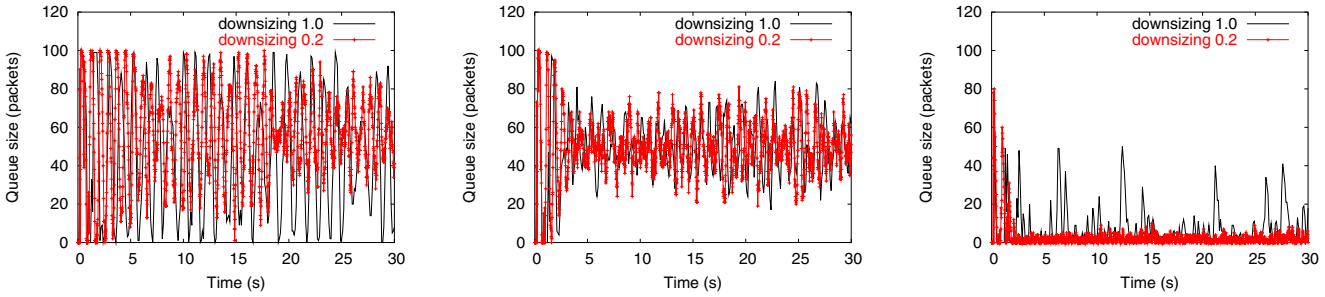
such as kp and ki (which in turn determine a and b), are determined in compliance with [8]; and (iv) the desirable utilization, γ, of AVQ [12] is set to 0.98, while the damping factor, α, is determined in compliance with Theorem 1 in [12] to ensure system stability ($\alpha = 0.15$). Parameters other than those mentioned above are set to their default values that come with the *ns-2* distribution. In addition, we have used different variations of TCP in the transport layer in each simulation run, but due to the space limit, we only present simulation results with *TCP Reno* connections. All the experiments are conducted on Linux 2.4.18 on a Pentium 4/1.9 Ghz PC with 1 $GBytes$ memory and with 2 $GBytes$ swap memory.

1) **Performance in the presence of long-lived TCP connections:** In this section, we study how effective *TranSim* is in simulating long-lived TCP flows with perspective of network dynamics, throughput, and execution time.

(a) Performance w.r.t. network dynamics: First, we examine how close the network dynamics under *TranSim* are to those under packet level simulation. Fig. 6 depicts the instantaneous queue length versus time in the network configuration given in Fig. 5. The number of nodes in each class varies from 5 to 100, and the downsizing parameter varies from 0.02 to 1.0 in these simulation runs. Also, router nodes employ a different AQM strategy (among Drop Tail, RED, REM, PI, and Virtual Queue) in each simulation run. Due to the space limit, we only present in Fig. 6 the case in which each class has 100 nodes and routers employ RED, PI, or Virtual Queue as their buffer management discipline. (Also, only the first 30 seconds are shown, although each simulation run lasts for 1000 seconds.) Regardless of the AQM strategy employed, the queue length extrapolated from the downsized network (the curves labeled with "downsizing 0.2") agrees extremely well with that observed in the original network (the curves labeled with "downsizing 1.0").

(b) Performance w.r.t. error discrepancy: We now quantitatively evaluate the discrepancy between results obtained in *TranSim* and those in packet level simulation. In both simulation modes, the TCP throughput is measured at a receiver and summed up to give the total throughput per class and the total throughput for the network.

Fig. 7 gives the total number of packets received at class 0 nodes in the network configuration given in Fig. 5. Each simulation run lasts for 1000 seconds. The error discrepancy observed between two simulation modes is at most about 10 % of the capacity of the bottleneck link as far as the downsizing parameter is appropriate. (The error discrepancy measured in



(a) RED (b) PI (c) Virtual Queue

Fig. 6. Instantaneous queue length versus time in the network configuration given in Fig. 5 when the number of nodes per class is 100. The curve labeled with "downsizing 1.0" is the instantaneous queue length in the original network and that labeled with "downsizing 0.2" is the instantaneous queue length extrapolated from the downsized network.

the total throughput as well as the aggregate throughput of all the class 1 nodes are also less than about 10 %.)

(c) Performance w.r.t. execution Time: We now evaluate the performance gain of *TranSim* (as compared to packet level simulation) in terms of the execution time required to carry out simulation. Fig. 8 depicts the execution time versus the number of nodes in a 1000-second simulation run in the network configuration given in Fig. 5. More than an order of magnitude improvement (maximally 50 times) in the execution time has been observed, and moreover, the performance improvement becomes more prominent as the network size increases (in terms of the number of nodes and the link capacity) or as the downsizing parameter decreases.

2) **Performance in the presence of long- and short-lived TCP connections:** In this section, we explore more dynamic scenarios in which long-lived and short-lived TCP connections co-exist and interfere with each other. Each short-lived connection is generated by a Pareto traffic generator in *ns-2*. Packets are sent at a fixed rate of 200 Kb/s during *on* periods and no packets are sent during *off* periods. The length of each on/off period follows a Pareto distribution with the Pareto shape parameter of 1.5 and the mean value of 100 *ms*. The application frame size is set to 210 bytes. Note that the Pareto traffic distributor is positioned on top of TCP, and that most of short-lived TCP flows only operate in their slow start phase.

(a) Performance w.r.t. network dynamics: Fig 9 depicts the instantaneous queue length versus time in the network configuration given in Fig. 5, except that the capacities of all the link, excluding the bottleneck link, have been increased to 100 Mb/s. Each class has 50 nodes (flows). The simulation run lasts for 300 seconds. Short-lived connections are only active in the interval [100, 200] seconds, and hence the entire simulation time is separated into three periods: [0, 100], [100, 200], and [200, 300].

Fig. 9 shows the dynamic changes of the queue length during the 20 % of each period when the AQM discipline is PI. (Results with other AQM disciplines are reported in [9].) As expected, the network dynamics observed in the downsized network agree very well with those in the original network.

(b) Performance w.r.t. error discrepancy: We quantitatively evaluate the discrepancy between results obtained in *TranSim* and those in packet level simulation. Fig. 10 gives the total number of packets received at class 0 nodes in the

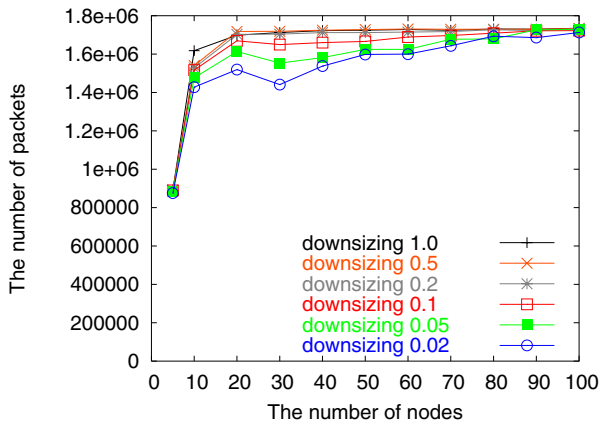
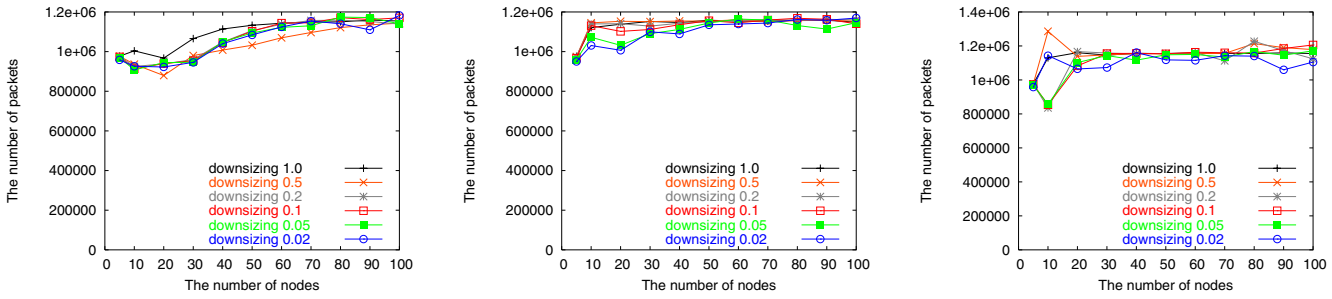


Fig. 10. The total number of packets received at class 0 nodes in the presence of both long-lived and short-lived TCP connections in a 900-second simulation run in the network configuration in Fig. 5. **PI** is employed as the AQM strategy.

network configuration given in Fig. 5. Each simulation run lasts for 900 seconds, and short-lived connections are only active in the interval [300, 600] seconds. The error discrepancy observed between two simulation modes is still within 10 % of the capacity of the bottleneck link. (Note that the error discrepancy per flow is much smaller than 10% of its attained throughput then.)

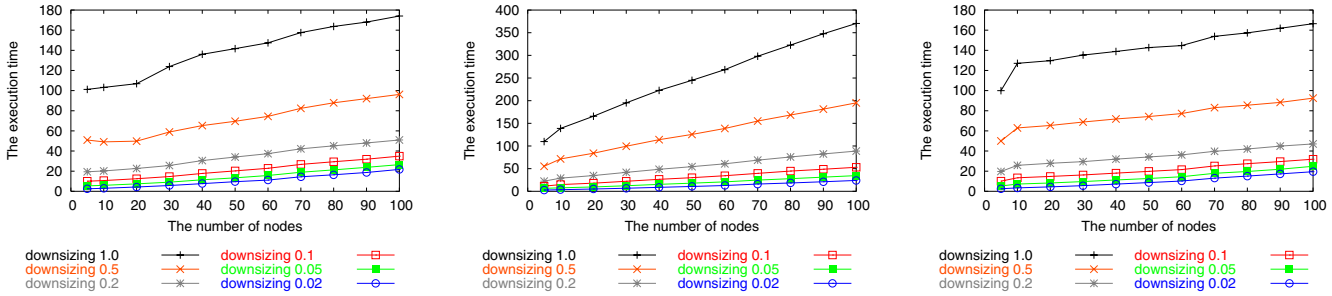
(c) Performance w.r.t. execution time: Again we evaluate the performance gain of *TranSim* (as compared to packet level simulation) in terms of the execution time required to carry out simulation. Fig. 11 depicts the execution time versus the number of nodes in a 900-second simulation run in the network configuration given in Fig. 5. The same conclusion made in the case of long-lived TCP connections can be applied here.

3) **Performance in the presence of long-lived TCP and unresponsive UDP connections:** We have also evaluated *TranSim* in scenarios in which long-lived TCP connections co-exist with unresponsive UDP connection (where both traffic have same number of connections). Each UDP connection delivers traffic generated by a CBR traffic generator with the rate set to *750 Kb/s*. The evaluation has been made with respect to network dynamics, error discrepancy, and execution time. Essentially the same conclusions drawn in the previous two sets of experiments can be applied here. Due to the space limit,



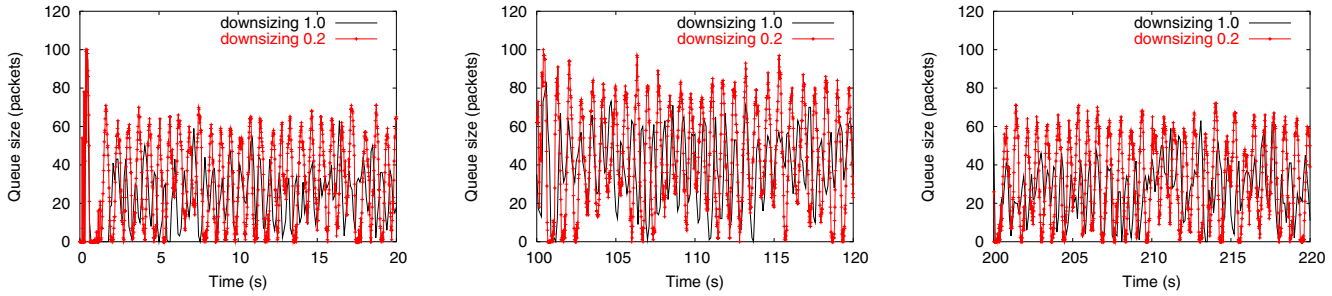
(a) Class 0 with RED (b) Class 0 with PI (c) Class 0 with Virtual Queue

Fig. 7. The total number of packets received at class 0 nodes in a 1000-second simulation run in the network configuration given in Fig. 5.



(a) RED (b) PI (c) Virtual Queue

Fig. 8. Execution time (sec.) required to carry out a 1000-second simulation run in the network configuration given in Fig. 5.



(a) From 0 to 20 (s) (b) From 100 to 120 (s) (c) From 200 to 220 (s)

Fig. 9. Instantaneous queue length versus time in the presence of both long-lived and short-lived TCP connections in the network configuration given in Fig. 5. **PI** is employed as the AQM discipline.

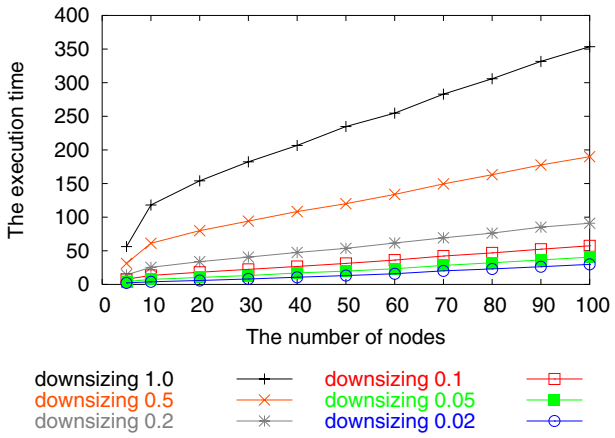


Fig. 11. Execution time (sec.) required to carry out a 900-second simulation run in the presence of both short-lived and long-lived TCP connections in the network configuration given in Fig. 5. **PI** is employed as the AQM strategy.

we do not present simulation results. The interested reader is referred to [9] for a detailed account.

4) **Performance in random, Internet-like topology:** In this section, we evaluate *TranSim* in random, Internet-like network topologies (generated by *BRITE* [15]), in perspective of throughput, execution time, and memory usage. Fig. 12 depicts one of the network topologies used in the simulation study, where the number of nodes is 200, the link capacity is determined by a heavy tail distribution between 10 and 100 Mb/s, and the link delay is automatically set by *BRITE*. We generate the same number of FTP/TCP flows as the number of nodes in the topology, and the source and destination nodes for each FTP flow are randomly chosen under the restriction that each node has only one sender and one receiver.

Tables II–III give the simulation results measured in a 200-simulation run in the network topology given in Fig. 12. Either Drop Tail (Table II) or RED (Table III) is used as the AQM discipline at each link. Several observations can

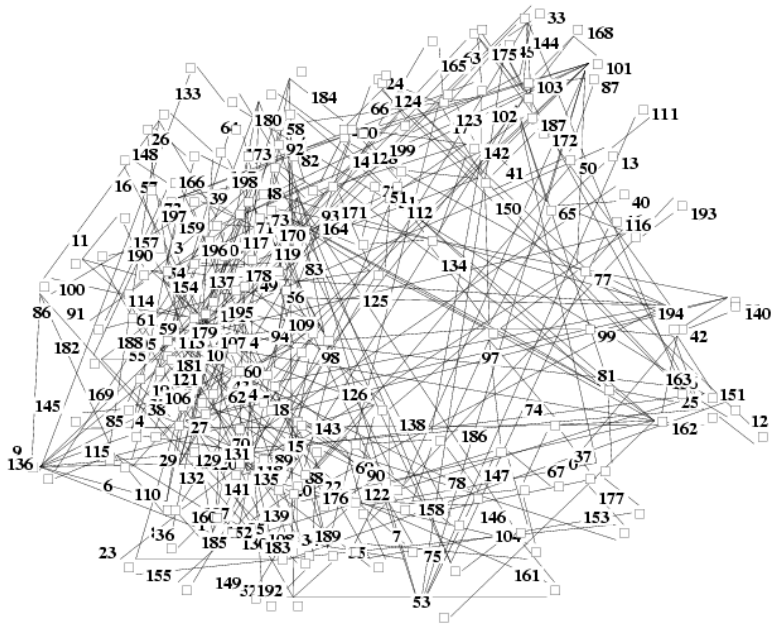


Fig. 12. A network topology with 200 nodes generated by BRITE

TABLE II

NETWORK THROUGHPUT (THE TOTAL NUMBER OF RECEIVED PACKETS), EXECUTION TIME, MEMORY USAGES MEASURED IN A 200-SECOND SIMULATION RUN IN THE NETWORK CONFIGURATION IN FIG. 12, WHERE ALL THE LINKS EMPLOY DROP TAIL AS AQM.

downsizing parameter	# of packets	execution time (s)	maximum memory usage (KB)	maximum RSS (KB)
1.0	53252216.0/1.0 = 53252216.0	3491.57	32564	27072
0.1	5317411.0/0.1 = 53174110.0	421.70	32564	27072
0.05	2654446.0/0.05 = 53088920.0	155.16	32476	26984
0.01	524435.0/0.01 = 52443500.0	38.91	32484	26996

be made from the tables: (i) the error discrepancy in the aggregate TCP throughput between the original network and any downsized network falls within 2 % after the extrapolation operation; (ii) *TranSim* achieves about two orders of magnitude of performance improvement with respect to execution time; (iii) The memory usage incurred in *TranSim* does not increase (nor does it decrease dramatically). The forth and fifth columns in Table II give, respectively, the total amount of virtual memory used for the code, data, and stack space and the total amount of physical memory used. The reason why memory saving is not as pronounced as the execution time is due to the fact that the same amount of memory is used to load all the modules in both the original network and the downsized network. Only the link bandwidth and the link delay vary.

Fig. 13 gives the per-flow throughput (the sum of which gives the aggregate TCP throughput in Tables II–III) under *TranSim* and packet level simulation. The throughput attained by each flow in the downsized network agrees well with that in the original network after the extrapolating operation.

VI. CONCLUSION

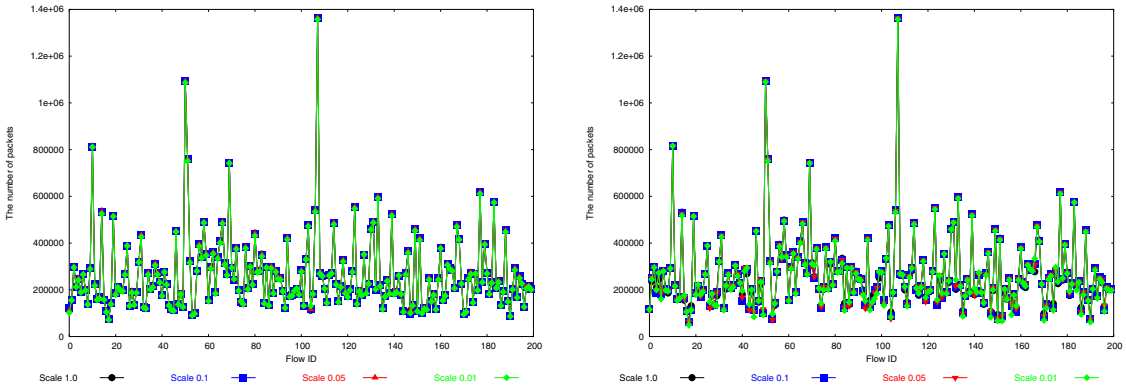
In this paper, we have presented a simulation framework, *TranSim*, that reduces the rate at which packet-events are generated, in order to accelerate large-scale simulation of IP networks with TCP/UDP traffic. Specifically, we transform the original network into an alternate network by curtailing the link capacity by a fraction α, stretching the link delay by $\frac{1}{\alpha}$, but keeping the bandwidth-delay product invariant. Neither the number of nodes/flows nor the queue parameters (the maximum buffer size or the AQM parameters) are changed. By preserving this bandwidth-delay product invariant during the downsizing operation, the network capacity as perceived by each TCP connection is preserved in the downsized network. We formally prove that the queue dynamics (e.g., the instantaneous queue length), the RTT dynamics, and the window size dynamics remain unchanged in the downsizing operation.

We have also carried out an *ns-2* simulation study comparing *TranSim* against packet level simulation, with respect to the capability of capturing transient, packet level network dynamics, the execution time reduced, the memory usage incurred, and the discrepancy in simulation results. The simulation results indicate maximally two orders of magnitude improvement in execution time and the performance improvement becomes more prominent as the network size increases (in terms of the number of nodes or flow, the network capacity, and the complexity of the network) or as the downsizing parameter decreases. The error discrepancy between *TranSim* and packet level simulation, on the other hand, is minimally 1-2 % and maximally 10 % in a wide variety of network topologies and traffic loads, together with various AQM strate-

TABLE III

AGGREGATE TCP THROUGHPUT (THE TOTAL NUMBER OF PACKETS RECEIVED), EXECUTION TIME, MEMORY USAGE MEASURED IN A 200-SECOND SIMULATION RUN IN THE NETWORK CONFIGURATION IN FIG. 12, WHERE ALL THE LINKS EMPLOY RED.

downsizing parameter	# of packets	execution time (s)	maximum memory usage (KB)	maximum RSS (KB)
1.0	54072076.0/1.0 = 54072076.0	4042.26	36128	30648
0.1	5403793.0/0.1 = 54037930.0	475.91	36084	30604
0.05	2701300.0/0.05 = 54026000.0	181.80	35968	30484
0.01	533331.0/0.01 = 53333100.0	45.87	35840	30360



(a) Drop Tail (b) RED

Fig. 13. Per-flow throughput measured in a 200-second simulation run in the network configuration in Fig. 12, when each link employs Drop Tail ((a)) and RED ((b)), respectively.

gies. The memory usage incurred in *TranSim* is comparable to that in packet level simulation. These encouraging simulation results, coupled with the fact that implementation of *TranSim* in a network simulator is simple and requires only a simple preprocessor/post-processor, suggest that *TranSim* can be used to simulate, and accurately infer, network dynamics as well as system throughput for large-scale IP networks with TCP/UDP traffic.

We have identified several research avenues for future work. First, we will analyze theoretically the relationship between the error discrepancy and the downsizing parameter. Second, we will extend *TranSim* to other network architectures, e.g., wireless LANs, by identifying the network attributes to be kept invariant and formally proving the correctness.

REFERENCES

[1] S. Athuraliya, V. H. Li, S. H. Low, and Q. Yin. REM: Active queue management. *IEEE Network*, 15(3), May/June 2001.

[2] F. Baccelli, G. Cohen, G. J. Olsder, and J.-P. Quadrat. *Synchronization and Linearity*. John Wiley & Sons, 1992.

[3] U. Berkeley, LBL, USC/ISI, and X. PARC. *The ns Manual*. http://www-mash.cs.berkeley.edu/ns/, April 2002.

[4] J.-Y. L. Boudec and P. Thiran. *Network Calculus*. Springer-Verlag, 2002.

[5] C. R. Wylie Jr. *Advanced Engineering Mathematics (second edition)*. McGraw-Hill Book Company, Inc., 1960.

[6] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1(4), 1993.

[7] Y. Gu, Y. Liu, and D. Towsley. On Integrating Fluid Models with Packet Simulation. In *Proceedings of IEEE INFOCOM 2004, (Hong Kong, China)*, March 2004.

[8] C. V. Hollot, V. Misra, D. F. Towsley, and W. Gong. On Designing Improved Controllers for AQM Routers Supporting TCP Flows. In *Proceedings of IEEE INFOCOM 2001, (Anchorage, Alaska)*, 2001.

[9] H. Kim. *Enabling Theoretical Model Based Techniques for Large Scale Network Simulation*. Ph.D. Thesis, Dept. of Computer Science, Univ. of Illinois at Urbana-Chamapaign, 2004.

[10] H. Kim and J. C. Hou. A Fast Simulation Framework for IEEE 802.11-Operated WLANs. In *Proceedings of ACM SIGMETRICS 2004, (New York, New York)*, June 2004.

[11] H. Kim and J. C. Hou. Network Calculus Based Simulation for TCP Congestion Control: Theorems, Implementation and Evaluation. In *Proceedings of IEEE INFOCOM 2004, (Hong Kong, China)*, March 2004.

[12] S. Kunniyur and R. Srikant. Analysis and Design of an Adaptive Virtual Queue (AVQ). In *Proceedings of ACM SIGCOMM 2001, (San Diego, California)*, August 2001.

[13] B. Liu, D. R. Figueiredo, 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, (Anchorage, Alaska)*, April 2001.

[14] 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, (San Diego, California)*, June 2003.

[15] A. Medina, A. Lakhina, I. Matta, and J. Byers. *BRITE: Boston University Representative Internet Topology Generator*. http://www.cs.bu.edu/brite/, 2002.

[16] N. Milidrag, G. Kesidis, and M. Devetsikiotis. An Overview of Fluid-Based Quick Simulation Techniques for Large Packet switched Communication Networks. In *Proceedings of SPIE ITCom 2001, (Denver, Colorado)*, August 2001.

[17] V. Misra, M. Gong, and D. Towsley. A Fluid-based Analysis of a Network of AQM Routers Supporting TCP Flows with an Application to RED. In *Proceedings of ACM SIGCOMM 2000, (Stockholm, Sweden)*, September 2000.

[18] V. Misra, W. Gong, and D. Towsley. Stochastic Differential Equation Modeling and Analysis of TCP Window Size Behavior. In *Proceedings of Performance 1999, (Istanbul, Turkey)*, October 1999.

[19] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP throughput: A Simple Model and its Empirical Validation. In *Proceedings of ACM SIGCOMM 1998, (Vancouver, Canada)*, September 1999.

[20] R. Pan, B. Prabhakar, K. Psounis, and D. Wischik. SHRiNK: A method for scaleable performance prediction and efficient network simulation. In *Proceedings of IEEE INFOCOM 2003, (San Francisco, California)*, March 2003.

[21] L. L. Perterson and B. S. Davie. *Computer Networks: A Systems Approach*. Morgan Kaufman, 1996.

[22] S. Shakkottai and R. Srikant. How Good are Deterministic Fluid Models of Internet Congestion Control? In *Proceedings of IEEE INFOCOM 2002, (New York, New York)*, June 2002.

[23] The Mathworks Inc. *MATLAB: The Language of Technical Computing*. The Mathworks Inc., 1999.

[24] Y. Wu and W. Gong. Time Stepped Simulation of Queuing Systems. In *Technical Report, Department of Electrical and Computer Engineering, University of Massachusetts*, 2001.