

Network Simulation via Hybrid System Modeling: A Time-Stepped Approach

Amogh Kavimandan[†], Wonsuck Lee[‡], Marina Thottan[‡], Anirudha Gokhale[†], and Ramesh Viswanathan[‡]

[†]Department of EECS, Vanderbilt University, Nashville, TN 37235

[‡]Bell Laboratories, Lucent Technologies, Murray Hill, NJ 07920

Abstract—The ever increasing complexity of networks dramatically increases the challenges faced by service providers to analyze network behavior and (re)provision resources to support multiple complex distributed applications. Accurate and scalable simulation tools are pivotal to this cause. The recently proposed *hybrid systems model* for data communication networks shows promise in achieving performance characteristics comparable to fluid models while retaining the accuracy of discrete models. Using the hybrid systems paradigm, this paper provides contributions to the modeling of TCP behavior and the analysis/simulation of data communication networks based on these models. An important distinguishing feature of our simulation framework is a faithful accounting of *link propagation delays* which has been ignored in previous work for the sake of simplicity. Other salient aspects of our work include a new finite state machine model for a drop-tail queue, a new model for fast recovery/fast retransmit mode, a revised sending rate model, and an embedded time-out mode transition mechanism all of which employ a time-stepped solution method to solve the hybrid system network models. Our simulation results are consistent with well-known packet based simulators such as ns-2, thus demonstrating the accuracy of our hybrid model. Our future efforts will be directed towards studying and improving the computational performance of hybrid model based simulations.

I. INTRODUCTION

Communication service providers use a variety of tools including visualization, analysis and simulations to track the performance of their networks and (re)provision existing and new applications. Efficient and scalable network analysis and simulation techniques are required due to the following technical needs:

- *Heterogeneous, multilayered networks* – where networks are made up of several layers starting with the physical layer on which are overlaid data link layers like ATM and MPLS followed by the Internet protocols like TCP and IP. Existing techniques to analyze network behavior are typically restricted to individual layers. For example, simulators like ns-2 are particularly useful to analyze a single layer, such as IP traffic. Thus, it is tedious and in many cases infeasible, with existing techniques, to determine the effect of disruptions at one layer across multiple layers of the network.
- *Networks of networks* – where the ever expanding size of the Internet has given rise to a large collection of interconnected but independently administered networks that include both public and private networks. Analyzing the behavior of these networks requires accurately modeling

the structure, policies and behavior of the networks while also scaling to the large size of the networks. Current state of the art in network analysis do not provide these capabilities.

Addressing the challenges outlined above requires a new approach to analyzing network behavior where the computational requirements must be kept manageable. We describe a technique using hybrid systems as the means for scalable and accurate multilayer network simulations. Hybrid system models combine the strengths of continuous and discrete models. A system is modeled as a discrete event system at a coarse granularity where a discrete state transition occurs when certain guard conditions are met. Within each state, however, a system is modeled to evolve according to some continuous dynamics.

II. SURVEY OF WORK ON NETWORK MODELING & SIMULATION

Network modeling and simulation paradigms can be categorized into three mainstream approaches: packet-level models, flow-level models, and hybrid models.

Packet-level simulation has been the most widely used simulation methodology by the network research community. Packet-level simulators are based on an event-logic driven simulation paradigm. For example, ns-2 [1], SSFNet [2] and JSim-INET [3] fall into this category. Packet-level simulators keep track of all the individual packets in the network. Therefore, they offer highly accurate and detailed results; however, the computational complexity increases rapidly as the size of the network becomes large.

During the past decade fluid-level simulations have been studied extensively and continue to be researched actively. In the framework of fluid models, a chunk of packets is modeled as a single continuous artifact, namely, a fluid. The dynamics of the network is then represented by a system of ordinary differential equations (ODEs), which describe fluid rate changes inside the network. Most of the research efforts in this area [4], [5], [6] have focused on modeling TCP protocols [7]. In this framework, the arrival and the loss of packets are often modeled as stochastic processes [4], [6]. The traffic source and queue dynamics constitute a system of equations which governs the TCP controlled data transfer over an IP network.

Recently, hybridization of the packet and the flow modeling approaches have been studied. Guo *et. al.* proposed the Time-stepped Hybrid Simulation (TSHS) [8] method to deal with

the scalability issue faced by traditional packet-level discrete-event simulation methods. TSHS considers a chunk, grouped from the packets that are in the same time-step, as a unit entity. The packets in a chunk are assumed to be evenly spaced within the time-step. In order to identify any event, every node in the network is checked at every time-step. TSHS is capable of offering flexible choice in simulation fidelity based on the simulation abstraction level.

A newly proposed hybrid system modeling framework [10], which is different from the aforementioned hybridization approaches, describes continuous dynamic behavior of a network within an organized finite state machine (FSM) formalism. Bohacek *et. al.* proposed such a framework for data communication networks and studied various communication protocols using hybrid systems [11]. Bohacek *et. al.*'s work forms the basis for what we report in this paper. We note that this paper shares some common ground with Bohacek *et. al.*'s work but differs since we model explicitly the propagation delay and provide a more mathematically accurate hybrid system model of the TCP protocol.

III. HYBRID SYSTEMS MODELING OF TCP NETWORKS

In this section, we describe a hybrid system TCP model that we have developed. We focus on a TCP New-Reno implementation for the source, a drop-tail queuing policy, and we model their interactions using FSM models. The aspects of our model that differs from Bohacek *et. al.*'s work will be highlighted. Our solution method differs from the previous work in that we use the time-stepped solution method to solve the constructed hybrid system. This approach resembles the solution techniques used in TSHS [8].

A. Mathematical Model of a Network

Mathematically a network is a directed graph $(\mathcal{N}, \mathcal{L})$ where a set of vertices \mathcal{N} denotes nodes and a set of edges \mathcal{L} denotes a collection of links connecting two nodes. For instance, two nodes $u \in \mathcal{N}$ and $v \in \mathcal{N}$ are connected by a link $l = l_{u,v} \in \mathcal{L}$. Three real positive numbers are assigned to each link $l \in \mathcal{L}$, these are: bandwidth (link-rate), propagation delay, and maximum queue size. Let B^l , τ^l , and q_{\max}^l denote the bandwidth, the propagation delay, and the maximum queue size of a link l , respectively. These three parameters characterize each link in a network. We remark that the *propagation delay* has not been included in the previous study [10], hence only simplified topological settings were studied.

Let \mathcal{F} represent a set of *flows*. Each flow $f \in \mathcal{F}$ is associated with a source node $u_s \in \mathcal{N}$ and a destination node $u_d \in \mathcal{N}$ where f flows from u_s to u_d . We assume that flow f is generated and enters node u_s with an incoming rate a_f at a prescribed time $t = t_f^0$. A flow f on a link l can be quantified by a link-in-rate (a sending rate), s_f^l , and a link-out-rate (an arrival rate), a_f^l .

Bandwidth B^l imposes an upper bound on flow sending rate on a link l .as follows:

$$\sum_{f \in \mathcal{F}} s_f^l \leq B^l, \quad \forall l \in \mathcal{L}. \quad (1)$$

Here $s_f^l = s_f^l(t)$ is a dynamically evolving quantity.

In our model, a queue is associated with a link $l \in \mathcal{L}$. q_f^l denotes the size of the flow f in the queue of a link l . Therefore, given the maximum capacity of the queue q_{\max}^l of the link l , the following inequality should hold for all times.

$$\sum_{f \in \mathcal{F}} q_f^l \leq q_{\max}^l, \quad \forall l \in \mathcal{L}. \quad (2)$$

Here q_f^l is also a function of time, *i.e.*, $q_f^l = q_f^l(t)$.

In general the transmission delay is composed of link propagation delay, processing delay, and queuing delay. Associated with each link l , we only consider a fixed propagation delay τ^l and a queuing delay q^l/B^l where $q^l = \sum_{f \in \mathcal{F}} q_f^l$. For the sake of simplicity, we assume the processing delay is minimal compared to the other two terms. Round-trip-time (RTT) is a measure of the current delay on a network or time elapsed to receive an acknowledgment packet (ACK) from the destination node.

B. Hybrid Model of TCP

In order to study transient behavior of a network, modeling TCP protocols has been our first goal since not only is a significant portion of Internet traffic controlled by the TCP protocol but also it potentially possesses rich dynamics due to the concept of varying the sending rate in response to the network conditions. Widely implemented TCP-New Reno policy include slow-start, fast-recovery/fast-retransmit (fr/fr), congestion-avoidance and time-out. The TCP-New Reno algorithm chooses a certain mode based on the limited information it holds and has received from the destination host.

In *slow-start* mode of TCP-New Reno, the congestion window size w_f of a given flow $f \in \mathcal{F}$ is governed by the following ordinary differential equation [10], [12]:

$$\frac{d}{dt} w_f(t) = \frac{\log m_{ss}}{RTT_f} w_f(t), \quad (3)$$

where m_{ss} is a multiplicative constant such that w_f is being multiplied by m_{ss} every RTT. w_f is initialized to 1 whenever the source enters slow-start mode. Hence, in slow-start continuous time (CT) domain we solve the initial value ODE problem.

In TCP flow control w_f is used to limit the number of packets to be sent out from the sender. Because the sender is able to send more packets only when the ACKs arrive, effectively, only w_f number of packets are sent in RTT_f time window. With the assumption that the packets (or flow f) are sent over RTT_f period with evenly spaced pattern, the instantaneous sending rate s_f should be

$$s_f(t) = \frac{w_f(t)}{RTT_f}. \quad (4)$$

Here we should note that the evenly spaced packets over RTT_f time period is a rather strong assumption. It may be a good model when traffic flow is up and running for some time so that ACKs become spread over time due to a certain random process. However, when traffic is at its early stage, like the very first slow-start mode, TCP tends to burst out traffic. In such cases the sending rate model, Eqn. (4) might be poor. We revisit this question in the discussion section.

Bohacek *et. al.* used a coefficient β to adjust the sending rate formula as follows:

$$s_f(t) = \frac{\beta w_f(t)}{RTT_f}, \quad (5)$$

where $\beta = 1.45$ is obtained from the trace comparison against **ns-2** simulation. However, we argue that it might be an artifact from the hybrid simulation paradigm employed in Bohacek *et. al.*'s paper [10].

During the *congestion avoidance* mode every ACK increases congestion window size by a small portion of the current window size,

$$w_f^{\text{new}} = w_f^{\text{old}} + \frac{L}{w_f^{\text{old}}}, \quad (6)$$

where L is a real positive number (normally $L = 1$). The ODE model that governs the congestion window size evolution in the congestion avoidance mode is

$$\frac{d}{dt}w_f(t) = \frac{L}{RTT_f}. \quad (7)$$

The model shows a linear increase of congestion window size in congestion avoidance mode as long as no drop is detected. Occurrence of dupACKs or no ACK ends the congestion avoidance mode and is followed by fr/fr or time-outs.

In TCP-New Reno policy fr/fr mode is initiated whenever the sender receives triple dupACKs. These dupACKs do not increase the congestion window size in real TCP flow control. Therefore, it is not accurate if the state remains at either slow-start or congestion avoidance mode since they are governed by their own dynamic discipline which evolves the congestion window size in time. In order to model congestion window evolution correctly over the state transition, the source should enter fr/fr once a drop is detected. However the actual transition should be delayed by the right amount of time to account for elapsed time between the drop event and the arrival of dupACKs in the real system. Due to lack of space, we redirect readers to [12] for the details on the hybrid model of fr/fr mode.

In our implementation, whenever a drop is detected, the source enters fr/fr. In fr/fr, if ACKs cease to return, the congestion window stays at a constant value and the number of outstanding packets is not decreased resulting in no new packets being sent for some time period.

For the *time-out* mode, a parameter t_{out} is introduced. It is set to 0 when the source enters fr/fr. t_{out} keeps on increasing with the simulation clock. If t_{out} exceeds the given time-out period, say, RTO , then the source enters slow-start mode with $w_f = 1$. Note that if time-out occurs in the simulation, the fr/fr state entered exhibits no effect on congestion window dynamics because no new packets were sent and no ACKs were received by the sender.

1) *Queue Dynamics*: Suppose ω and ν are in-bound (west) and out-bound (east) links, respectively, in the traffic path of a flow f . Let \mathcal{F}_α be a set of such flows sharing the links ω and ν . We have the following equation describing the queue changing rate

$$\frac{d}{dt}q_f^\omega = a_f^\omega - s_f^\nu - d_f^\omega, \quad (8)$$

where q_f^ω is the size of flow f buffered at q^ω . a_f^ω is flow arrival rate from the link ω . s_f^ν denote flow sending rate into the link ν .

Suppose there is no packet loss during transmission over the links in a network, then incoming flow into a link ω must all exit from that link, that is, $a_f^\omega(t) = s_f^\omega(t - \tau_\omega)$ where τ_ω is the link propagation delay. Here $s_f^\omega(t - \tau_\omega)$ is readily available at time t .

Let q_{max}^ω represent maximum queue size associated with the in-bound link ω . B^ν (B^ω) is a given fixed bandwidth of the link ν (ω).

Integration of Eqn. (8) without the drop term gives

$$q_f^\omega(t) = q_f^\omega(t_0) + \int_{t_0}^t (a_f^\omega(t') - s_f^\nu(t')) dt'. \quad (9)$$

Here s_f^ν is the sending rate into the link ν , *i.e.*, the sending rate from the q^ω .

Unlike the queue model by Bohacek *et. al.*'s, the states are not determined by the queue size q_f^ω . Rather, the bandwidth of the out-bound link, B^ν , and the size of the arriving and departing packet determine the state of a queue. For the flow sending s_f^ν , the sending rate becomes,

$$s_f^\nu = \begin{cases} \frac{q_f^\omega}{\sum_{\bar{f} \in \mathcal{F}_\alpha} q_{\bar{f}}^\omega}, & \text{if } \sum_{\bar{f} \in \mathcal{F}_\alpha} q_{\bar{f}}^\omega \geq B^\omega \\ a_f^\omega + q_f^\omega, & \text{if } \sum_{\bar{f} \in \mathcal{F}_\alpha} (a_{\bar{f}}^\omega + q_{\bar{f}}^\omega) \leq B^\omega \\ q_f^\omega + \frac{a_f^\omega}{\sum_{\bar{f} \in \mathcal{F}_\alpha} a_{\bar{f}}^\omega} (B^\omega - \sum_{\bar{f} \in \mathcal{F}_\alpha} q_{\bar{f}}^\omega), & \text{o.w.} \end{cases} \quad (10)$$

Eqns. (9) and (10) completely describe the queue size and the flow sending rate changes in time. However, the sum of solutions $\sum q_f^\omega$ may be greater than q_{max}^ω which should be understood as buffer overflow or flow drop.

C. Solution Techniques

We have created the simulator without relying on any existing simulation frameworks, such as, Stateflow/Simulink [14], Modelica [15], Ptolemy [16] etc. Therefore we have full control of the solution method of the hybrid system, numerical scheme to be used in CT domain, and scope of component model. Furthermore, it is easy to identify any computational bottleneck and the sources of inaccuracy in the model specific to the implementation.

In general FSM is not a timed model, however, its composition with CT domain requires detection of any events that may be triggered on the time axis. Each state in FSM is being refined during the continuous evolution of the sources and the queues in our model. At each time abscissa, which may be a priori chosen or dynamically updated, active states are evaluated. If the state transition condition is enabled, then the FSM makes corresponding transitions according to the condition and the system continues to evolve.

Assume that CT-FSM composition refers to a master clock for their time. Consider one tick of that master clock to be the time step Δt of our hybrid system simulation. It is trivial that a smaller Δt introduces small error in state transition. In our simulation, we do not allow FSM state transitions

during an epoch of length Δt . Now, CT domain simulation can be performed as accurately as possible, say, over an interval $[n\Delta t, (n+1)\Delta t]$ where $n = 0, 1, \dots$. At each time point, $n\Delta t$, current states are evaluated, checked against transition enablers, and changes are made to the states, as necessary.

The tasks described above can be summarized in the following algorithm.

Algorithm for the Hybrid Simulation

Input:

- A network $(\mathcal{N}, \mathcal{L})$,
- Bandwidth of all the links $l \in \mathcal{L}$,
- Maximum queue size at all the queues,
- Set of source nodes \mathcal{S} ,
- Set of destination nodes \mathcal{D} ,
- Source wake up time and interval,
- Time-step h ,
- End of simulation time T_{end} .

Construct \mathcal{L}_f for each $f \in \mathcal{F}$

Set $t = 0$

while $t \leq T_{end}$ **do**

$t = t + h$

Wake-up sources as scheduled

Solve equations for all $s \in \mathcal{S}$ and $q \in \mathcal{N}$

for all $s \in \mathcal{S}$ **do**

switch $mode(s)$:

case ($mode(s)=slow\ start$):

if $cwnd(s) \geq ssthresh$ **then**

$mode(s) \xrightarrow{t+\text{delay}} c.a.$

else if flow f from s suffer drop **then**

$mode(s) \xrightarrow{t+\text{delay}} fr/fr$

else

remain in slow start mode

case ($mode(s)=fr/fr$):

if $n_f^{drop} = 0$ **then**

$mode(s) \xrightarrow{t} c.a.$

else

remain in fr/fr

case ($mode(s)=c.a.$):

if flow f from s suffer drop **then**

$mode(s) \xrightarrow{t+\text{delay}} fr/fr$

else

remain in $c.a.$

end for

for all queue **do**

evaluate and set queue state for the next iteration

end for

end while

Finally, we remark that the proposed solution approach is somewhat similar to the Time-stepped Hybrid Simulation (TSHS) [8] and shares the idea of packet smoothing, however, the concept of hybrid system of CT-FSM composition is fundamentally different from TSHS.

IV. HYBRID MODEL & SIMULATION VALIDATION

A. Simulation Set Up & Environment

Our simulation studies comprise multiple experiments using the packet-based **ns-2** simulator and the hybrid model-driven simulator that we have developed. Since we are interested in comparing the two simulators we had to use the same traffic characteristics for both simulators. Furthermore the

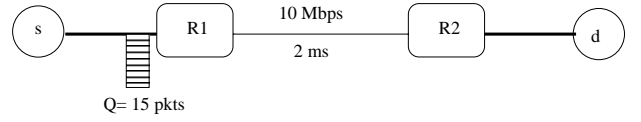


Fig. 1. A Simple Topology with 1 TCP New-Reno Source.

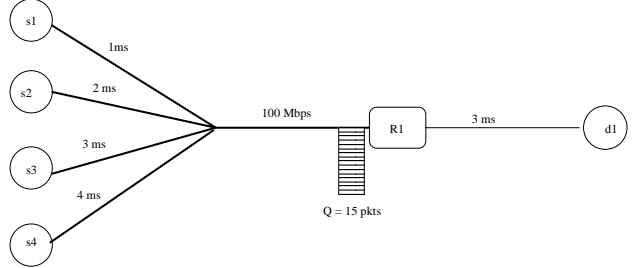


Fig. 2. A Dumbbell topology with 4 TCP New-Reno Sources.

specific choice of the traffic pattern is dictated by how well it models network traffic. Experimental studies have shown that individual sources of traffic can be approximated by ON/OFF processes, where the On and Off periods have sub-exponential distributions. Within the ON state, a source transmits at peak rate. We use this model for our experiments with both the **ns-2** and the hybrid simulations. Packets are sent at a fixed rate during ON state while no packets are sent during OFF state. Both on and off periods are taken from a Pareto distribution with constant size packets. These traffic sources can be used to generate aggregate network traffic that exhibits the long range dependency seen in the IP network traffic.

B. Simulation Validation & Results

The first goal of our experimental set up was to baseline the performance of our hybrid model using well known packet level simulators such as **ns-2**. For this purpose we used two topologies. A simple topology as shown in Figure 1 with a single TCP New-Reno source and a dumbbell topology as shown in Figure 2 with 4 TCP New-Reno sources.

For the **ns-2** simulations we used an identical TCP source model with no background traffic. The simple topology is used to evaluate the behavior of the queue dynamics as well as TCP source behavior. It consists of a single edge and two network nodes. The link propagation delay was 2ms. In the dumbbell topology, we have 4 TCP sources that are aggregated at the queue for router 1 and each of these sources have link delays of 1, 2, 3, and 4ms each. All four TCP sources are destined for the same destination node **d**. The queue size is set to 15 packets with each packet consisting of 512 bytes. The run time for each of the simulations was 1000ms. The TCP flows were triggered to start at 5ms. The metrics used for comparing the hybrid simulator with the **ns-2** simulator are, the queue sizes and the congestion window sizes of the TCP source.

Figure 3 shows the queue size changes of the simple topology case 1 using different time steps in the hybrid simulator as well as in **ns-2**. The queue sizes computed using the finer time steps in the hybrid model are much closer to the

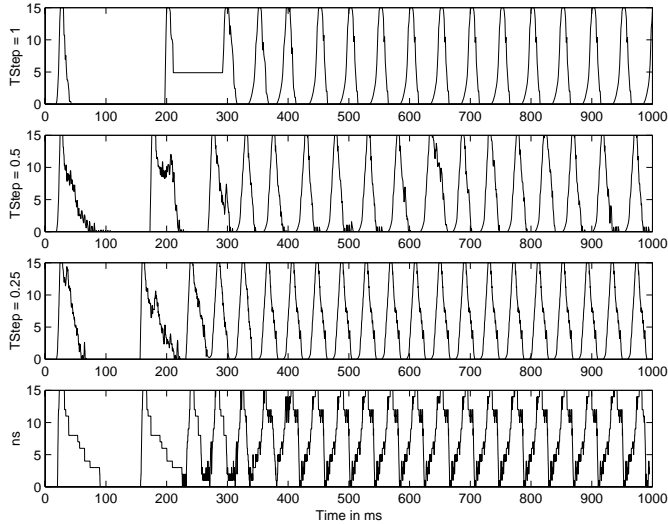


Fig. 3. Comparison of queue size changes using the hybrid model and **ns-2** of the Simple Topology case 1. Queue size changes are evaluated using different time step values (0.25, 0.5, 1.0) in the hybrid simulator.

actual behavior. However using a finer time step increases the computational complexity. The queue size changes obtained by the hybrid simulator using a time step of 0.25 provides the closest match to the queue size change observed in **ns-2**.

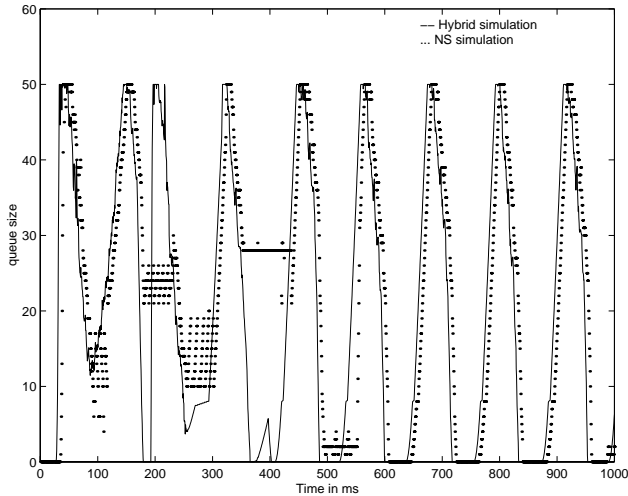


Fig. 4. Comparison of the queue size changes: the hybrid model vs. **ns-2**.

Figure 4 compares the queue size changes between the hybrid and the **ns-2** simulations for the Dumbbell topology case 2. As the simulation time progresses and reaches a steady state, we obtain a close match between the queue behavior of the **ns-2** and the hybrid models.

We obtained excellent agreement in congestion window size evolution for the both test cases. All the results are not shown in this short version of paper except the comparison of the average congestion window from hybrid system simulation and **ns-2** of the Dumbbell topology case, Table I.

	source1	source2	source3	source4
hybrid	26.92	19.92	15.89	13.84
ns-2	26.92	19.82	16.02	13.94

TABLE I
DUMBBELL TOPOLOGY, STEADY STATE ($t \geq 500$ ms) AVERAGE
CONGESTION WINDOW SIZE.

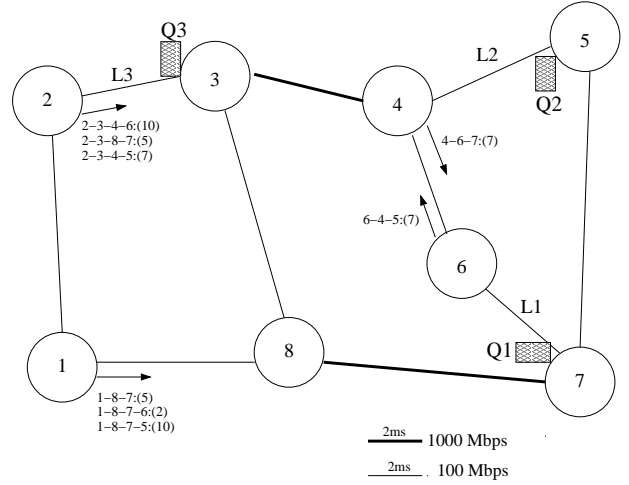


Fig. 5. US Topology loaded with 53 sources: A dashed-numbers (e.g. 1-8-7-6) denotes the path of a flow. The parenthesized number represents how many flows are in that path.

V. SIMULATION OF LARGE NETWORK TOPOLOGIES

In order to study the effectiveness of the hybrid simulator we evaluated its performance using a larger topology. We compare the behavior of queue sizes under different TCP source configurations for both the hybrid simulator and **ns-2**.

A. Network Topology

Figure 5, called to be *US Topology*, shows a network of 8 nodes connected by duplex-links of 100 Mbps bandwidth and 2ms link-delay. Links 2-3 and 8-7 have higher, 1000 Mbps, bandwidth with 2ms link-delay also. For all the nodes, maximum queue size is set to 50 packets. Queue sizes are traced for the link 6-7 (L1), the link 4-5 (L2), and the link 2-3 (L3). We have used 53, 86, and 129 TCP sources where destination nodes are spread over the network.

B. Results

Figure 6 shows the change in queue (Q2) size under three different source configurations (53, 86 and 129 sources each) for both the **ns-2** as well as the hybrid simulator. In our simulation scenarios, of all the queues in the network this particular queue was the one that had the most discrepancy with the **ns-2** simulation. However note that even this worst case deviation from **ns-2** is not very significant as can be seen from Table II.

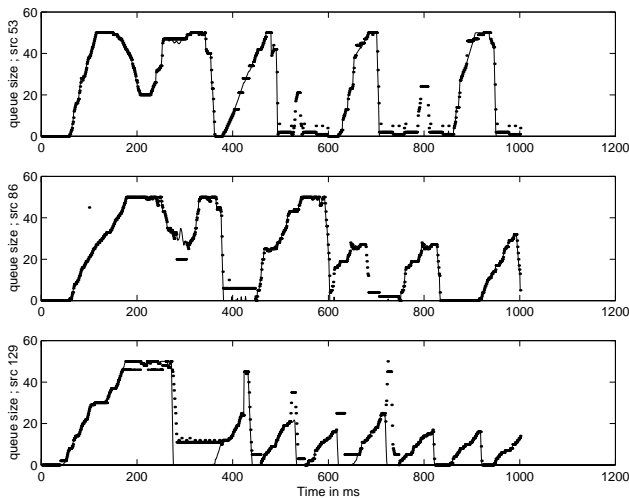


Fig. 6. Comparison of queue size changes using the hybrid model and **ns-2** for US topology. Queue size changes are evaluated using different number of sources. Legend: \cdots **ns-2** and $-$ hybrid

	Q ₁	Q ₂	Q ₃
53 connections	(23.7:24.8)	(21.0:22.3)	(8.7:10.5)
86 connections	(24.4:23.8)	(20.5:22.5)	(11.8:13.0)
129 connections	(28.5:28.5)	(13.9:16.4)	(21.0:20.6)

TABLE II

US TOPOLOGY AVERAGE QUEUE LENGTH FOR Q₁, Q₂, Q₃ WRITTEN FOR (HYBRID:NS-2)

VI. DISCUSSION & CONCLUSION

In this section we discuss the sending rate model (4) and the hybrid model implementation of TCP and complexity issues. We conclude the section with a brief summary of the paper.

A. Transport Layer & Application Layer

In Sec. III-B we pointed out that the sending rate model (4) might be inappropriate due to the bursty nature of the TCP protocol and the dependency on packet availability from applications.

The model in equation (4) sends out packets (flow) over a time interval of length RTT. More specifically, if congestion window size is 30 and RTT is 6 ms then 5 packets are sent per 1 ms in our model. In real situation, it is more likely to send all the 30 packets almost instantaneously. Hence if the bandwidth of the out-bound link is not capable of processing all the incoming traffic, the queue overflows. However, with the model in equation 4, the queue has enough time to process the flows that are arriving spread out over time. Thus, the model (4) is more appropriate at the asymptotic stage, when the packets are more evenly spread out.

B. Implementation Issues

Our current hybrid system simulator employs a fixed time-step approach. More specifically, we use the notion of a global clock and advance time by a fixed time-step till the end of simulation time. After each time-step, all the sources

and queues are recomputed and checked against the transition enablers. This approach allows us to accurately incorporate propagation delays without further modeling effort. On the other hand, because the checks are carried out regardless of the states and the events at any time interval, its performance is not optimal. In future work, we plan to implement more intelligent time marching techniques — the resulting simulator would provide a better basis for studying the performance characteristics of hybrid based simulations.

C. Conclusions

In this paper, we have demonstrated the strength of the hybrid system modeling and simulation method for TCP network. New models, enhancements, and revisions were proposed and studied. We have used the time-stepped solution method to solve the hybrid system model of a TCP network. The extensive experiments showed good agreements with the results from Network Simulator **ns-2**. However, we have not yet studied the stability and the convergence issues of the numerical methods we have employed. The robust simulation tool which we have developed can be used to analyze complex network behavior so that service providers are better equipped to support the required quality of service (QoS) needs of their applications.

REFERENCES

- [1] The VINT Project on NS2, “The Network Simulator”, ns2: <http://www.isi.edu/nsnam/ns>
- [2] SSFNet, “The Scalable Simulation Framework”, <http://www.ssfnet.org/homePage.html>
- [3] J-Sim, “A component-based simulation environment”, <http://www.j-sim.org>
- [4] V. Misra, W-B. Gong, and D. Towsley, “Stochastic Differential Equation Modeling and Analysis of TCP Window Size Behavior”, Performance ’99 (Istanbul, Turkey, October 1999)
- [5] V. Misra, W-B. Gong, and D. Towsley, “A Fluid-based Analysis of a Network of AQM Routers Supporting TCP Flows with an Application to RED”, Proceedings of ACM SIGCOMM’00, (Stockholm, Sweden, September 2000)
- [6] E. Altman, K. Avrachenkov, and C. Barakat, “A Stochastic Model of TCP/IP with Stationary Random Losses”, ACM SIGCOMM Computer Communication Review, 30: 231-242, 2000
- [7] V. Jacobson, “Congestion Avoidance and Control”, Computer Communication Review, vol. 18, 1988
- [8] Y. Guo, W-B. Gong, and D. Towsley, “Time-stepped Hybrid Simulation (TSHS) for Large Scale Networks”, Proceedings of IEEE Infocom 2000, 441-450, 2000
- [9] Y. Gu, Y. Li, and D. Towsley, “On Integrating Fluid Models with Packet Simulation”, Proceedings of IEEE Infocom 2004, 2004
- [10] S. Bohacek, J.P. Hespanha, J. Lee, and K. Obraczka, “A Hybrid Systems Modeling Framework for Fast and Accurate Simulation of Data Communication Networks”, ACM SIGMETRICS’03, June 2003
- [11] A.J. Van Der Schaft and J.M. Schumacher, An Introduction to Hybrid Dynamical Systems (Lecture Notes in Control and Information Sciences, 251) Springer-Verlag Telos, 1999
- [12] W. Lee, M. Thottan, R. Viswanathan, A. Gokhale, and A. Kavimandan “Network Simulation via Hybrid System Modeling: A Time-Stepped Approach”, Lucent Technologies Technical Memorandum, ITD-05-46094C, March 2005.
- [13] S.D. Conte and C. Boor, “Elementary Numerical Analysis (An Algorithmic Approach)” McGraw-Hill, 1981
- [14] The MathWorks Inc., “Simulink”, <http://www.mathworks.com/products/simulink>
- [15] The Modelica Association, “Modelica: ”, <http://www.modelica.org>
- [16] Dept. of EECS, University of California at Berkeley, “The Ptolemy Project”, <http://ptolemy.eecs.berkeley.edu>