

4. SUMMARY AND A FUTURE WORK

We reviewed here several different approaches for the efficient simulation of large data networks, mainly based on *fluid* approximations and hybrid variations. In our paper we have considered techniques that use simplified models of the real systems in order to provide speed-up. We have also described fluid versions of models for components commonly found in today networks.

A crucial aspect of time-driven techniques is their ability to vary the level of abstraction. Simulation at a high-level of abstraction can point out critical parts in a network design, at a low modeling and simulation cost, while detailed event-driven simulation can provide a more accurate picture but at a higher cost. The techniques presented, combined with supporting parallelism, efficient implementation, and fast hardware could allow simulations of large networks at a reasonable overall cost.

Future work should attempt to ascertain which of the discussed techniques performs best under a given traffic mix and network topology, design and dimension. Finally, future studies need to assess the magnitude of fluid network model error for more complex and realistic network designs involving hierarchical schedulers, switch fabrics, etc.

REFERENCES

1. J.S. Ahn and P.B. Danzig. Packet Network Simulation: Speedup and Accuracy Versus Timing Granularity. *IEEE/ACM Transactions on Networking*, Vol. 4, No. 5, pp. 743-757, Oct. 1996.
2. L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, H. Yu. Advances in network simulation. *IEEE Computer Mag.*, Vol. 33, No. 5, pg. 55-67, May 2000.
3. Y. Guo, W. Gong and D. Towsley. Time-Stepped Hybrid Simulation (TSHS) for Large Scale Networks. In *Proc. of INFOCOM 2000, Tel Aviv, Israel*, Mar. 2000.
4. D. Jagerman, B. Melamed and W. Willinger. Stochastic modeling of traffic processes. In *Frontiers in Queuing: Models, Methods and Problems*, J. Dshalalow, Ed., CRC Press, 1996.
5. G. Kesidis and A. Singh. An overview on Cell-level ATM network simulation. In *High Performance Computing Systems Conf. Montreal, Canada*, 1995.
6. G. Kesidis, A. Singh, D. Cheung and W.W. Kwok. Feasibility of Fluid-Driven Simulation for ATM Networks. In *Proc. IEEE GLOBECOM, London*, pp. 2013-2017, Nov. 1996.
7. T. Konstantopoulos and G. Last. On the dynamics and performance of stochastic fluid systems. *to appear in Applied Probability*.
8. B. Liu, D.R. Figueiredo, Y. Guo, J. Kurose, D. Towsley. A Study of Networks Simulation Efficiency: Fluid Simulation vs. Packet-level Simulation. In *Proc. IEEE Infocom, Alaska*, Apr. 2001.
9. D. Nichol, M. Goldsbay and M. Johnson. Fluid-based simulation of communication networks using SSF. In *Proc. 1999 European Simulation Symposium, Erlangen-Nuremberg, Germany*, Oct. 1999.
10. A.K. Parekh and R.G. Gallager. A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single Node Case. *IEEE/ACM Trans. Networking*, Vol. 1, No. 3, June 1993.
11. P.A. Skelly, M. Schwartz and S. Dixit. A histogram-based model for video traffic behavior in an ATM multiplexer. *IEEE/ACM Trans. Networking*, Vol. 1, No. 4, Aug. 1993.
12. A. Yan and W.B. Gong. Fluid Simulation for High Speed Networks. *IEEE Trans. On Information Theory*, June, 1999.

arrival rate is smaller than the service rate is greater than zero.”⁸ In particular, the event rate does not go to infinity and it can also be lower than event rate in a packet level simulator. In conclusion, they propose a flow aggregation method to deal with ripple effect: the component sources of the flow of interest are not aggregated while those of other flows are. In addition, they show the fact GPS will have less ripple effect than the FIFO scheduling policy (which is intuitive since the flows under GPS interact with each other less than they would in a fully shared FIFO queue).

3.2. Fluid Network Model Error

Nichol et al⁹ argued that the model error between a event-driven fluid network simulator and the corresponding packet-by-packet simulator is small. The statistics gathered with a packet simulator and a corresponding fluid simulator were compared. Simulation results indicated that error generated by fluid approximation is relatively low, so fluid simulation gives valid representation of the state of the network.

A time-driven fluid simulator of Yan et al¹² considered a single FIFO queue for which two bounds were defined, “forward” and “backward,” for the departure process ($D^{f,b}$) and the queue occupancy ($X^{f,b}$). The forward bound is reached if we assume that the traffic for specific time-interval arrives at the beginning of the interval. The backward bound results when traffic arrives at the end of the interval. The “actual” packet queue, X , of the packet-by-packet simulation may have packet arrivals at any time in the interval. For a simulation time-step (interval size) of h , $D^b(t) \leq D(t) \leq D^f(t)$ and $X^f(nt) \leq X(nh) \leq X^b$ where D is the departure process of the packet queue.

Their analysis assumes that servers are work-conserving and are fed by a class (priority) of fluid. For a feedforward network, it is assumed that propagation delay between queues should be a multiple of step size h (a negligible assumption). A topologically sorted feedforward network is analyzed with $p_{i,j}$ being a fraction of fluid going from router i to router j . Beginning with the single-queue forward and backward bounds, the error of the entire fluid simulator can be directly derived for such a network.

Error bounds are also obtained for a network with feedback (they assumed any queue can feed any other queue). One advantage of time-driven simulation is that it mitigates the ripple effect. The model error depends on the time-step size h . When this time-step size is very small, the fluid network has small error but offers little speed-up over the packet-by-packet simulator. Larger step sizes will speed-up the simulation but the resulting statistics will have error proportional to the step size. Note that, with time-driven simulation, the error does not grow as the size of the network increase, i.e., time-driven simulation scales well in this direction.

3.3. Hybrid Simulation Approaches

The time-stepped hybrid simulator (TSHS) of Guo et al³ is also time-driven. In order to decrease number of events that simulator has to process, TSHS samples a traffic source and creates “chunks” of data. A chunk is created only if in the corresponding time interval the source has generated one or more packets. All packets generated in an interval will be pushed into a chunk’s packet queue. This way the granularity of traffic (abstraction level) is coarser than in packet level simulation which can considerably decrease event rate of simulator. Consequently, if the simulation step is small enough so that each chunk contains at most times one packet, the simulation accuracy be high.

On the other hand, a larger time-step will speed-up the simulation at the expense of higher model error. Packets generated by the source contain information regarding their time of creation, sequence number, as well as other information needed for implementation of different protocols. A chunk must contain information about the amount of traffic it carries so that the service time can be calculated at each output queue. Also it contains other information such as creation time, time of the last event and time of the next event. We therefore see that the time-driven/“data-chunk” approach is somewhat of a necessary compromise between the extremes of a full-blown fluid network model and a packet-by-packet simulator.

In this approach we find two very good properties: First, compared with event-driven simulation it does not suffer from a *ripple effect*. Second, besides providing a state of the network, due to its hybrid nature it enables simulation of different packet-based protocols. On the other side worst-case “analytical” error bounds are increasing as the number of hops increases. In particular, this means that TSHS method does not provide accurate results for large networks. However, the reported results show error values well below analytical error bounds.

2.6. Priority Scheduler

Consider again a FIFO queue shared by two flows. The first flow has priority over the second. The queueing dynamics of the first flow are independent of the second flow and can be immediately obtained from the equations of Section 2.1. The *aggregate* queue content is the same as that of the shared (by equal priority flows) queue covered in the previous section. The departure process of the second (lower priority) flow is simply the difference between that of the aggregate and that of the first flow. Thus, we can obtain the queue occupancy process of the second flow by simply subtracting its cumulative arrivals from its cumulative departures. In a similar way, we can obtain the dynamics of a priority scheduler handling more than two flows.

2.7. End-to-end Delay Computations

Consider a connection passing through H queues. Let $L^h(t)$ be the total byte loss experienced by this connection at node $h \in \{1, 2, \dots, H\}$ in $[0, t]$. Define $L(t) = \sum_{h=1}^H L^h(t)$. Also for this connection let $R(t)$ be the amount of fluid received at the destination and $T(t)$ be the amount of fluid transmitted by the source in $[0, t]$.

Clearly, the average end-to-end byte loss rate over $[0, t]$ for this connection is $L(t)/T(t)$. Define $\hat{T}(t) = T(t) - L(t)$. The average end-to-end delay of the connection over $[0, t]$ is given by

$$\frac{1}{R(t)} \int_0^{R(t)} \left(R^{-1}(s) - \hat{T}^{-1}(s) \right) ds$$

3. PERFORMANCE OF FLUID-TYPE AND HYBRID METHODS

The papers briefly discussed in this section consider simulation strategies that are hybrids of fluid model-based and packet-by-packet. Also, they are hybrids of event-driven and time-driven. The goal is to identify which hybrid approach results in maximal simulation speed-up subject to some kind of bound on (fluid) network modeling error. The optimal “blend” depends on the number of flows, volume of traffic and the degree of iteration of the traffic at the points of bandwidth and buffer contention, i.e., the points in the network where queues are present.

3.1. The ripple-effect of an event-driven simulator

Previously,⁶ a fluid queueing network was considered and simulated in an event-driven manner. As described in Section 2.1 above, a change in an arrival rate to a FIFO queue \mathbf{F} may cause its departure rate to change which may, in turn, cause some of the departure rates of FIFO queues downstream to change as well. Under an idling bandwidth scheduler, only those downstream nodes for which \mathbf{F} is a tributary are affected by changes in the departure rate of \mathbf{F} . Thus, a change in arrival rate of a source will cause a “ripple effect” in the network. Under a work-conserving scheduler (GPS), when \mathbf{F} becomes empty (or its departure rate changes when $d < \phi c$), the departure rates of all nonidle FIFO queues *at the node containing \mathbf{F}* will change.

Consequently, all upstream FIFO queues which handle connections that use the node containing \mathbf{F} may be affected by changes in the arrival rate to \mathbf{F} . So there will be a greater ripple effect in a network of work-conserving nodes than in one consisting of idling nodes. Consequently, a network of idling nodes will have faster execution time. It has been previously observed⁶ that as the considered network dimensions grew, the ripple effect caused the fluid-based simulator to require more execution time than the corresponding packet-by-packet simulator.

This study has been expanded⁸: the event-rate of a fluid simulator was compared to that of the corresponding packet-level simulator for a single queue, a feedforward network and a feedback network. Both GPS and flow aggregation are analyzed.

For a single node, the fluid simulation is always faster. For a network of tandem queues, they assumed that the same number of sources is feeding each queue. One flow traverses all queues while the others interact with this main flow for only one hop. Three experiments are conducted. The first shows that the fluid simulation speed-up decreases as the number of nodes increase so that eventually the packet simulator becomes more efficient. The second experiment shows that if the *number of sources* (constituting each flow) increases, the fluid simulator’s speed-up decreases. Finally, in the third experiment, they show that as the *rate* of the sources increases, the fluid simulator speed-up increases too.

For a feedback network it was shown that: “Even for a closed-loop queueing system a flow rate change can only cause a finite number of new rate changes at downstream queues, if the probability that queue is empty and aggregate

2.4. Idling Schedulers

Consider a non-work-conserving or “idling” bandwidth scheduler acting on N queues for a bandwidth resource of size c as in the previous section. The fluid version of such a scheduler is a bank of FIFO queues with *dedicated* service rates ϕc (whose individual dynamics were described in Section 2.1 above) where the departure processes are simply aggregated. That is, equation (4) continues to hold but the “idle bandwidth” (i.e., the bandwidth due to FIFO queues being temporarily idle) is not distributed to nonidle FIFO queues.

2.5. FIFO Queue Scheduler (Shared FIFO Queue)

Consider a fluid queue shared by two fluid flows. For now, assume a constant service rate of c . Let $A(s, t)$ represent the total amount of fluid arriving to the queue over the interval of time (s, t) ; similarly let $A_n(s, t)$ be the total amount of fluid arriving *from the n^{th} source* to the queue over (s, t) . Thus, $A = A_1 + A_2$. Also let $D(s, t)$ be the total amount of fluid departing the queue over $D(s, t)$ and similarly define D_n . Finally, let $X(t)$ be the contents of the queue (work to be done) at time t :

$$X(t) = \sup_{s \leq t} A(s, t) - c(t - s).$$

We are primarily interested in the distribution of the delay experienced by the first flow’s fluid in this shared queue. To this end, define the *virtual delay process*

$$S(t) = t + X(t)$$

and note that, clearly, $S(t) \geq t$. The generalized inverse of S is

$$S^{-1}(z) = \inf\{u \leq z \mid S(u) = z\}$$

The time derivative of the total departures at time t is given by

$$\begin{aligned} \dot{D}_n(t) &= \begin{cases} c\dot{A}_n(S^{-1}(t))/\dot{A}(S^{-1}(t)) & \text{if } X(t) > 0 \text{ and } \dot{A}(S^{-1}(t)) > 0 \\ \dot{A}_n(S^{-1}(t)) & \text{else} \end{cases} \\ &= \dot{A}_n(S^{-1}(t)) - [1 - c/\dot{A}(S^{-1}(t))]\dot{A}_n(S^{-1}(t))\mathbf{1}\{X(t)\dot{A}(S^{-1}(t)) > 0\} \end{aligned}$$

Note that we have assumed that the A_n are differentiable which implies that the $\dot{A}_n \geq 0$ are piecewise continuous.

Thus, the component queue occupancy X_n of the n^{th} fluid flow satisfies:

$$\begin{aligned} \dot{X}_n(t) &= \dot{A}_n(t) - \dot{D}_n(t) \\ &= \dot{A}_n(t) - \dot{A}_n(S^{-1}(t)) + \left[1 - \frac{c}{\dot{A}(S^{-1}(t))}\right] \dot{A}_n(S^{-1}(t))\mathbf{1}\{X(t)\dot{A}(S^{-1}(t)) > 0\} \end{aligned}$$

and, therefore,

$$X_n(t) = A_n(S^{-1}(t), t) + \int_{-\infty}^t [1 - c/\dot{A}(S^{-1}(u))]\dot{A}_n(S^{-1}(u))\mathbf{1}\{X(u)\dot{A}(S^{-1}(u)) > 0\}du.$$

The aggregate queue occupancy is:

$$\begin{aligned} \dot{X}(t) &= \begin{cases} c & \text{if } X(t) > 0 \text{ and } \dot{A}(S^{-1}(t)) > 0 \\ \dot{A}(S^{-1}(t)) & \text{else} \end{cases} \\ X(t) &= \int_{-\infty}^t \mathbf{1}\{X(t) > c(t - u)\}A(du). \end{aligned}$$

We direct the reader to our references for discussions of the subtleties of fluid queue dynamics.⁷

Finally, suppose the total arrival rate a is piecewise constant (as it would be for an MMF source) and that at time τ the arrival rate of the j^{th} flow changes. Also, suppose that $X(s) > 0$ for all $s \in [\tau, t := \tau + X(\tau)(c)^{-1}]$. Note that the change in the arrival rate at the input will take $X(\tau)(c)^{-1}$ seconds to propagate to the output of the FIFO queue:

$$\frac{\dot{D}_j(t)}{\dot{D}(t)} = \frac{\dot{A}_j(\tau)}{\dot{A}(\tau)}.$$

Note that the last term in the previous equation is not present in discrete queueing systems, i.e., there are no departures from an empty queue.⁷

Finally note that the loss rate L of fluid at this queue satisfies

$$\dot{L} = a\mathbf{1}\{X = B\}.$$

2.2. Fluid Leaky Bucket

There are three parameters associated with a fluid leaky bucket (FLB): the packet buffer capacity of B_P bytes, the token buffer capacity of B_T tokens (one token corresponds to one byte), and the token arrival rate r tokens/second*. Note that a packet marker or policer (dropper) will have $B_P = 0$ and a flow shaper will have $B_P > 0$. At time t , let $a(t)$ be the arrival rate of fluid to the packet buffer, $d(t)$ be the departure rate of fluid from the leaky bucket, $X(t)$ be the occupancy of the packet buffer, and $T(t)$ be the occupancy of the token buffer.

In order for fluid to leave the packet buffer, it must “consume” an equal amount of fluid from the token buffer. Thus, fluid accumulates in the token buffer only if the arrival rate to the packet buffer is smaller than r . Also, fluid accumulates in the packet buffer only if the arrival rate to the packet buffer is greater than r . The FLB operates so that $X(t)T(t) = 0$ for all t .

More precisely, the dynamics of the FLB are given by the following equations:

$$\begin{aligned} \dot{X} &= (a - r)\mathbf{1}\{0 < X < B_P\} + (a - r)^+\mathbf{1}\{X = 0\} \\ &\quad + (a - r)^-\mathbf{1}\{X = B_P\} \end{aligned} \tag{1}$$

$$\begin{aligned} \dot{T} &= (r - a)\mathbf{1}\{0 < T < B_T\} + (r - a)^+\mathbf{1}\{T = 0\} \\ &\quad + (r - a)^-\mathbf{1}\{T = B_T\} \end{aligned} \tag{2}$$

$$d = a\mathbf{1}\{X = 0\} + r\mathbf{1}\{X > 0\} \tag{3}$$

where the dependence on time of the quantities involved is not shown, $(y)^+ := \max(0, y)$, $(y)^- := \min(0, y)$, and $\mathbf{1}$ is the indicator function.

2.3. FIFO Queues of a Generalized Processor Sharing Node

We now consider a *work-conserving* (nonidling) multiplexer of queues called Generalized Processor Sharing (GPS).¹⁰ GPS divides the output link bandwidth (of c bytes/s) among a set of N FIFO queues according to positive bandwidth partitioning parameters $\{\phi^1, \phi^2, \dots, \phi^N\}$ satisfying $\phi^1 + \phi^2 + \dots + \phi^N \leq 1$. Each FIFO queue typically handles a group of connections. GPS is work-conserving and operates so that the bandwidth allotted to FIFO queues that are currently idle is distributed among nonidle queues *in proportion to their bandwidth partitioning parameters*.

More precisely, the dynamics of a GPS node are describe by the following equations. For all $i, j \in \{1, 2, \dots, N\}$, consider one FIFO queue of a GPS node and let $a^i(t)$ be its arrival rate, $d^i(t)$ be its departure rate, $X^i(t)$ be its contents in bytes, and B^i be its capacity in bytes.

$$\begin{aligned} \dot{X}^i &= (a^i - d^i)\mathbf{1}\{0 < X^i < B^i\} + a^i\mathbf{1}\{X^i = 0\} \\ &\quad + (a^i - d^i)^-\mathbf{1}\{X^i = B^i\} \end{aligned} \tag{4}$$

$$\frac{d^i}{d^j} = \frac{\phi^i}{\phi^j} \text{ if } X^i > 0 \text{ and } X^j > 0 \tag{5}$$

A statement equivalent to Equation (5) is: for all $i \in \{1, 2, \dots, N\}$,

$$d^i = \frac{\phi^i \mathbf{1}\{X^i > 0\}}{\sum_{j=1}^N \phi^j \mathbf{1}\{X^j > 0\}} c \tag{6}$$

taking $\frac{0}{0} = 0$.

*In the leaky bucket literature, the variables σ and ρ are commonly used instead of B_T and r respectively.

One can attempt to make simulation faster by ignoring the precise time of the packet arrival during an *on* period by assuming that the source generates traffic at a constant rate in a *continuous* fashion. This approach is called *fluid-flow modeling* and the corresponding source model is called Markov-modulated fluid (MMF). Three issues arise in the transition from packet-level model to fluid modeling:

- Accuracy: What is the effect of increasing the level of abstraction on the fidelity of the results?
- Speed-up: Does fluid modeling reduce execution time of the simulator and, if so, under what circumstances?
- Portability: Is there an accurate fluid-compatible model of the *network* through which the fluid traffic models can flow?

A typical sequential discrete-event simulation maintains a clock and the event list data structure. These two quantities provide synchronization for the execution of different events. Every event scheduled for future execution at some point in time has to be entered into event list with its associated time of execution. The clock variable holds the time up to which all events have been simulated. At each step, the event with the smallest associated time execution is removed from the event list, its effect on a real system is simulated, and the clock is advanced to the event’s associated (completion) time. How the event list is manipulated significantly affects the efficiency of the simulator. The simulator can spend from 25% to 50% of its time inserting and deleting events.¹ Different algorithms have been proposed and all of them can be equally used in a packet-level simulator or in a fluid simulator. Intuitively, the main advantage of fluid simulation is supposed to be a smaller event rate compared to that of a corresponding packet-level simulator. Therefore, a smaller number of events has to be executed by the simulator, and since the event list contains smaller number of events inserting and deleting from the list takes less time. In order to reduce the event rate, the traffic can be simulated at coarser level than that of the packet. These ideas address some problems but may also lead to new ones: the accuracy of the simulation may be reduced, the execution time per event may increase, and the execution of one event may spawn more events that must be inserted in the event list. This causes an “ripple effect” of events⁶ that dramatically degrades the performance of the simulator, see Section 3 below.

The main building blocks of a fluid network simulator structure are source and multiplexer models. One of many fluid source models has been already presented above. The rest of the paper is organized as follows. In Section 2, we briefly overview fluid network dynamics. In Section 3, we briefly review the current literature on model (specifically fluid model) based techniques for quick simulation of packet-switched communication networks. The paper concludes in section 4 with a discussion future work.

2. FLUID NETWORK MODEL DYNAMICS

In this section, we give the nuts and bolts equations describing the dynamics of fluid-based models of networking components. We first describe the mechanics of a single queue and leaky bucket. The fluid version of four commonly found schedulers is then considered: GPS (work-conserving), idling (non-work-conserving), first-in-first-out (FIFO) (simply a FIFO queue shared by more than one flow), and priority. In practice, all scheduling mechanisms in Internet routers are combinations of these four. In the following, the dimension of all rate quantities of fluid models of (variable-length) packet flows is assumed to be bytes/second.

2.1. FIFO Queue

Consider a FIFO queue with occupancy X bytes, capacity B bytes and constant service rate c . Also let a be the fluid arrival rate and d be the fluid departure rate. We simply have that:

$$\begin{aligned} \dot{X} &= (a - d)\mathbf{1}\{0 < X < B\} + (a - d)^+\mathbf{1}\{X = 0\} \\ &\quad - d\mathbf{1}\{X = B\} \end{aligned}$$

and

$$d = c\mathbf{1}\{X > 0\} + a\mathbf{1}\{X = 0\}.$$

An Overview of Fluid-Based Quick Simulation Techniques for Large Packet-switched Communication Networks

Nenad Milidrag^a, George Kesidis^b and Michael Devetsikiotis^c

^a E&CE Dept, University of Waterloo, Waterloo, ON N2L 3G1, Canada

^b EE and CS&E Depts, Pennsylvania State University, University Park, PA 16802

^c ECE Dept, NC State University, Raleigh, NC 27695-7911

ABSTRACT

We consider a large packet-switched communication network. Traffic in such networks is heavily aggregated especially in the network core. Fluid traffic models have been used for this reason and because the individual packets are very small compared to the volume of aggregated traffic. Fluid models have also been considered for the network components themselves in order to explore the possibility of simulation speed-up. In the event-driven simulation of Kesidis et al⁶ of such a “fluid” network, a “ripple effect” was described to explain the substantial degradation in simulation speed-up as the network size grew, especially when work-conserving bandwidth schedulers were present. Thereafter, studies attempted to identify under what network dimensions and designs and under what traffic conditions the ripple effect is minimized. Hybrids of packet/fluid and event/time-driven simulation strategies were considered. This paper gives an overview of the fluid network modeling approach and surveys recent work on such hybrid approaches.

Keywords: broadband packet-switched networks, simulation, fluid models

1. INTRODUCTION

Modern data communication networks are extremely complex and do not lend well to theoretical analysis. To better analyze the performance of proposed protocols, algorithms, hardware, topologies, etc., simulation studies are typically employed. Because of the lack of accurate tractable models, simulation is a most reliable tool for network design and analysis.

The main obstacle in traditional packet-level discrete-event approach in simulating networks is the vast number of packets that have to be simulated for accurate results. Each packet will generate a number of events on its path from source to destination and each event has to be executed by the simulator at specified point in time (an event can be arrival of packet to the switch, its departure, that a queue is empty, etc.). Consider, for example, a 16x16 router where every link can carry about 370,000 50-byte packets per second. A packet-level simulator of the router has to process at least 5,900,000 events to simulate one second of the router. The simulator of Ahn and Danzig¹ can process 28,400 events per second; so, simulation of 15 minutes of real time of the aforementioned router would require 52.5 hours. Running a simulation for 15 minutes of real time, however, might be insufficient for accurate estimation of rare events such as packet loss in a well-provisioned buffer. The number of events that have to be executed is roughly proportional to the CPU time required. Therefore, the computational cost can be measured by the event-rate $N(t)/t$ where t is the time up to which the simulator has advanced and $N(t)$ is the number of events that have been executed up to t .

Markov-modulated traffic models typically involve a Markov chain X operating over a finite state-space. At time t , the “intensity” of the packet transmission rate is a function of $X(t)$, $f(X(t))$, where f is not necessarily one-to-one. The number of states used to model the source depends on the nature of the source. The intensity $f(X)$ of arrivals could simply be the exact transition rate of packets by the source model. Another commonly used model is the Markov-modulated Poisson process (MMPP) where $f(X(t))$ is the intensity of a Poisson process at time t and each point of the (doubly stochastic) Poisson process is a packet arrival time. For example, voice and certain bursty data sources can be successfully modeled with a two-state Markov-modulated process (called an “on-off” model), while a more complex video source can be represented with a multiple-state process.¹¹

This work was funded by new faculty start-up grants at Penn State and NC State, respectively. Send correspondence to kesidis@engr.psu.edu or mdevets@eos.ncsu.edu