

An Overview of Cell-Level ATM Network Simulation

G. Kesidis and A. Singh

E.& C.E. Dept, University of Waterloo

Waterloo, ON, Canada, N2L 3G1

g.kesidis@eandce.uwaterloo.ca asingh@etude.uwaterloo.ca

Abstract

An ATM simulator is often faced with the challenging task of estimating rare events in a very large network without the availability of a large amount of simulation time. This paper provides an overview of cell-level ATM network simulation, and concentrates on strategies that can be employed to overcome these challenges. We first give some possible simulation objectives and describe the general software structure of the simulator. Parametric and nonparametric traffic modeling methods are then defined. In the context of traffic modeling for the purposes of simulation, we describe preliminary calculations, traffic aggregation, importance sampling, and extrapolation methods (the last two methods are used for estimating rare events). Finally, the role of parallelism is explored including the use of parallel independent experiments and distributed simulation using multiple workstations or tightly-coupled multiprocessors.

1 Introduction

ATM-based B-ISDN is mandated to provide individual quality of service (QoS) guarantees to a wide variety of traffic types (e.g., video, voice and data). The precise QoS received by a particular connection is not presently calculatable even if Markovian models of the traffic sources are given. Consequently, ATM network designers are forced to use simulation to evaluate, for example, the precise traffic capacity of a particular network design.

This paper gives an introduction to and overview of cell-level simulation of ATM Networks. Whenever possible, good survey papers on the subtopics covered are cited. As there is presently a lack of realistic benchmarks to evaluate the performance of each approach to the simulation problem, we have focused on describing promising approaches in an organized fashion.

There are two main challenges for an ATM network simulator. The first is that ATM networks often have very large dimensions and ATM cells are quite small. So, a cell-level simulation may require a large amount of simulation time. The second is that the simulator is often asked to estimate a very small quantity such as a cell loss probability of the order 10^{-9} . A large amount of simulation time would be required by a direct Monte-Carlo approach to obtain an accurate estimate of the likelihood of such rare events.

We assume that the ATM network we are simulating functions in discrete-time and that all devices are synchronized. That is, all cell transmission times and propagation and queuing delays are multiples of some constant c^{-1} (for a 155Mbps network, $c^{-1} = 53 \times 8 / (155 \times 10^6) s = 2.7 \mu s$). In this way, integer operations are used instead of floating point operations when, for example, making compares for insertions into event lists (cf. §3). Also, since c^{-1} is so small, the lack of synchronization among the devices of a real network has negligible performance consequences.

Another approach to ATM network simulation is based on “fluid” source models (see, e.g., [16]). Instead of keeping track of a potentially enormous number of cells in the network, the simulator keeps track of the *rate* at which each source is currently emitting cells. The advantage of the fluid approach is that a reduction in complexity results in a speed up over the cell tracking approach (source rates typically change over a longer time scale than a cell transmission time). However, keeping track of how changes in source rates propagate

through the network is complicated for large networks; work-conserving bandwidth schedules cause similar “ripple effects” in the network because bandwidth allotment temporarily increases to some FIFOs when another FIFO becomes idle. Another disadvantage of the fluid approach is that fluid models must be found for both traffic sources and the bandwidth schedulers used at the switches (like the GPS fluid analogue of the PGPS scheduler [50]) thereby reducing the variety of ATM networks that can be simulated. Also, cell delay is more complicated to measure for a lossy connection. In [33], a specification of a fluid ATM network simulator is given.

This paper is organized as follows. In §2, we list possible simulation objectives. The software framework of our paper is described in §3. The role of traffic modeling and related issues like preliminary calculations, importance sampling and traffic source aggregation are discussed in §4. Computationally inexpensive extrapolation methods that are used to quickly estimate rare events are described in §5. The role of parallelism is discussed in §6. Finally, we summarize in §7.

2 Simulation Objectives

There are basically two categories of simulation objectives for a given ATM network topology (in the context of any of the simulation goals given below, a simulation study could compare different network topologies and designs). The first is customer-oriented or cell-level simulation goals (ATM layer). For example, the problem could be to find the network’s capacity to handle traffic subject to individual QoS requirements (fixed network “dimensions”) or to find suitable network dimensions (i.e., buffer memory and/or link bandwidth) to handle a particular traffic load.

The second category is network-oriented simulation goals (network layer). For example, the problem could be: to study how, where and with what frequency congestion arises in the network, to evaluate the performance of call-set up and routing strategies (with respect to throughput or call set-up times), or to evaluate call blocking probabilities (which is also a measure of network capacity). In this paper, we will focus on cell-level simulation goals.

3 Event Lists and Program Structure

In this section we will describe the program structure of an ATM network simulator based on event lists. An event list is simply a stack of jobs that the simulator must execute. Examples of jobs are: “cell is transmitted by source to node”, “cell joins buffer”, and “buffer transmits a HOL cell onto link”. The jobs have an associated time at which they must be executed; in the event list(s), jobs are stacked in increasing order of their time of execution. The program functions by simply popping jobs from the event list and executing them; note that the execution of a job may spawn more jobs that must then be inserted into the event list.

An event-driven approach is recommended over a time-driven approach because the unit of time c^{-1} is very small. Consider a voice source which transmits one cell into the network on every 15000 units of time on average when $c^{-1} = 2.7\mu s$. The time-driven simulation will check the source for a transmitted cell every unit of time whereas an event-driven approach will deal with the source only when it transmits a cell (the only “event” related to the source). Mandatory simulation of each time unit for each source, switch, and link of the network (whether or not an event is present in the object at that time) drives up the book-keeping costs for a time driven simulation. This results in higher computational requirements and lower efficiency.

Because of the enormous number and variety of devices and sources involved in a large ATM network simulation, a programming language that facilitates code-reuse should be employed. Since ATM devices, such as switches, have several functional components (e.g., a switch fabric, buffers, schedulers, flow-control devices), hierarchical data structures should

be used. For these reasons, an object-oriented programming language, such as C++ [53], is recommended so that the resulting code is compact and easy to debug.

The program structure presented in this section describes a sequential simulator. In §6 we discuss the issue of parallelization of simulation of an ATM network.

4 The Role of Modeling

There is a vast literature on ATM traffic modeling. We now discuss the role that traffic source modeling plays in the context of cell-level simulation. The modeling process begins with raw “cell-transmission-time” data. For example, traffic data of the entire Star Wars movie under MPEG-1 compression can be obtained by anonymous ftp from thumper.bellcore.com. Traffic source modeling is the process by which the statistics of this traffic data is represented in a much more compact form; for example, voice is commonly modeled as a simple on-off Markov chain [45]. A good model therefore eliminates the need to maintain a collection of actual source traces. However, highly nonstationary sources (such as compressed, full-length motion pictures) are difficult to model and, consequently, actual traces or models of individual traces [43] are often used in network simulation.

Traffic models can also allow for preliminary calculations that can give approximate solutions to the stated simulation objectives. Furthermore, certain models can allow for significant importance sampling simulation speed ups over direct Monte-Carlo approaches. These two topics will now be briefly discussed followed by a discussion of the role of modeling aggregate traffic.

4.1 Parametric and Nonparametric Traffic Source Modeling

We could choose, for example, to “fit” a discrete-time, on-off Markov chain¹ to the data by matching the first three moments of the (marginal) interarrival time distribution of the data, m_1 , m_2 and m_3 , with that of the model. This is a kind of “parametric” modeling (see [19] for a recent survey). In the context of ATM network simulation, the “goodness” of the model is determined by the degree to which it accurately represents the relevant statistical *queueing properties* of the data. So, in parametric modeling, the queueing properties of the data are approximated by those of the model.

On the other hand, some modeling methods propose to *directly* consider the queueing properties of the data. For example, instead of calculating the m_i , a queue is simulated with the arrival process given by the data and a deterministic service rate c cells/s. Assume c is chosen between m_1^{-1} and p where p^{-1} is smallest interarrival time of the data (p is the data’s peak rate). This queue’s occupancy is monitored and the probability that an arriving customer sees β cells in the queue is calculated²; let $p_\beta(c)$ represent this measured probability. The *function* $p_\beta(\cdot)$ is a nonparametric model of the data. Under certain conditions and for large β , we can define $I(c) := -\beta^{-1} \log p_\beta(c)$ where the function $\alpha := I^{-1}$ (i.e., $\alpha(I(c)) \equiv c$) is called an *effective bandwidth* of the data; see, e.g., [3, 35, 16, 11, 24, 15].

Of course, the effective bandwidth is also a nonparametric model of the traffic. Basically, a nonparametric directly model regards the relevant queueing of the connection instead of via some “intermediate” (parametric) modeling. Examples of nonparametric modeling approaches are found in [42] (the burstiness curve that is related to $\sigma - \rho$ constraints) and [8, 1, 14, 27]. Nonparametric modeling is the basis of the “extrapolative” quick estimation methods of §5 below.

4.2 Preliminary Calculations

Under certain idealizations, preliminary calculations are possible to get rough estimates of the desired quantities. For example, consider a suitably decoupled [9] ATM network wherein

¹A model with 3 parameters: the two transition probabilities and the “on” cell transmission rate.

² β may be a parameter related to the required QoS of the traffic.

effective bandwidth traffic models are used. Conservative estimates of the QoS received by each connection are calculatable from these models. We can use the loss network framework [31] to compute call blocking probabilities. Also, we can phrase the buffer dimensioning problem (i.e., what amount of buffer memory is required at each network node to handle a particular traffic load) as an optimization problem [42]. In both cases, the ATM network is approximated by a circuit-switched network (i.e., a network that does not take advantage of statistical multiplexing gains to admit more connections).

Currently, “end-to-end” performance evaluation (see, e.g., [9, 49, 4]) and statistical multiplexing (see, e.g., [40, 2, ?]) are very active areas of study. Whenever possible, the most recent results on performance evaluation should, of course, be applied to the problems described in §2 *before* a simulation study is initiated.

4.3 Importance Sampling for Estimating Rare Events

The problem of estimating rare events can be dealt with by using statistical quick simulation methods such as importance sampling (see [25] for a survey) and extrapolation (cf. §5). Importance sampling methods typically require Markovian (parameterized) models of the traffic sources. The traffic source models are not simulated exclusively with their original parameters; “tilted” parameters are also used. These tilted parameters can be obtained from the effective bandwidths of the traffic sources [34, 5]. Unbiased estimates of the desired quantities for the original parameters are obtained from the tilted simulation via likelihood ratios. For a single queue, the resulting speed ups over direct Monte-Carlo simulation using the original parameters are enormous (often several orders of magnitude) even taking into account the computation of the tilted parameters, etc., at initialization and of the likelihood ratios during the simulation run. Importance sampling for networks is discussed in [5, 12, 18]. Apart from the modeling requirements, a disadvantage of these methods is that the management of the likelihood ratios for a large ATM network is complex.

4.4 Source Aggregation

Source aggregation is method by which a group of traffic sources is modeled as a single source. For example, assume that a group of N i.i.d. Markov-modulated sources (each having K states) sharing the same virtual path through an ATM network is modeled as a single Markov-modulated source with $n \ll N \times K$ states where $N \times K \gg 1$. To see the simulation speed up afforded by aggregation, consider a program that creates an event list of cell transmission times for the N sources. The aggregate model will generate the transition times of the single Markov chain; between consecutive transition times, cells will be generated at a constant rate. The aggregate source will simply place a number of transmission times at the end of the event list for every transition of the Markov chain (the number depends on the time between transitions of the Markov chain and the current cell rate). On the other hand, if the N sources are simulated as an N -dimensional Markov chain, the Markov transitions will take longer to calculate because there will be a greater number of possible transitions. If the N sources are simulated separately, then their transition times must be ordered to determine consecutive intervals of time over which the aggregate cell rate is constant. Another benefit of aggregation is that preliminary calculations can be significantly simplified by the reduction in the size of the state-space afforded by aggregation (see, e.g., [28, 29]) and cf. §6.2.

5 Extrapolative Methods for Estimating Rare Events

Using the following methods, a wide variety of received qualities of service can be estimated in a reasonable amount of simulation time. Models of the traffic sources are *not* required³.

³For these reasons, the approaches described in §5 can also be used for real-time traffic policing of stated traffic descriptors [32, 10].

Some of these methods are, however, biased estimators in general. We now consider the simulation objective of evaluating the QoS received by a particular traffic pattern given all details of a particular ATM network design.

5.1 Extrapolative Methods for Cell Delay

In this section, we will focus a single connection through the network that requires a bound on the likelihood of excessive total cell queuing delay (i.e., the sum of the cell queuing delays in each buffer along its virtual circuit). Each connection would be equipped with an extrapolative estimator (described below) that would be physically located at the connection's destination.

Thus, we want to estimate $\mathbf{P}\{X \geq B\}$ where $X \times c^{-1}$ is distributed as the steady-state total queuing delay of cells through the network and B is large ($X \in \mathbf{Z}$). A reasonable amount of simulation time may not be sufficient to witness a single excursion by X to B . The record of these total queuing delays will be "extrapolated" to B by the methods described below. In preparation, let π_t be the empirical distribution of the total queuing delay divided by c^{-1} over the interval of simulated time $[0, t]$. We now describe three extrapolative approaches that use π_t to estimate $\mathbf{P}\{X \geq B\}$.

5.1.1 Chebyshev Inequality

By Chebyshev's inequality,

$$\mathbf{P}\{X \geq B\} \leq B^{-n} \mathbf{E}X^n \quad (1)$$

$$\approx \sum_{k=0}^{\infty} (k/B)^n \pi_t(k) =: f(n). \quad (2)$$

Let B_t be the smallest integer $n \geq 0$ such that $\pi_t(k) = 0$ for all $k > n$ (i.e., $[0, B_t]$ is the support of π_t). If $B_t < B$ then minimizing f is a decreasing function of n with $\lim_{n \rightarrow \infty} f(n) = 0$. So, although the inequality (1) holds for all $n \geq 0$, the approximate inequality (2) in general does not. Typically, this approach and its many variants give poor estimates.

5.1.2 The Kullback-Leibler Method

This method assumes that there is an integer B_1 such that $0 \leq B_1 \ll B$ and, for all $b \geq B_1$,

$$\mathbf{P}\{X \geq b\} \approx A e^{-bI}$$

where A and I are two unknown parameters. For this method to work, there must be enough simulation time to accurately estimate $\mathbf{P}\{X = B_1\}$ (i.e., t is large enough so that $\pi_t(B_1) \approx \mathbf{P}\{X = B_1\}$).

Consider the following distribution p defined on integers $\geq B_1 - 1$: $p(b) = A \exp(-bI) - A \exp(-(b+1)I)$ for $b \geq B_1$, and $p(B_1 - 1) = 1 - A \exp(-bI)$. The two unknowns A and I are *fitted* to π_t by minimizing the Kullback-Leibler distance between p and π_t over A and I on $[B_1, \infty)$. The values of A and I that achieve the minimum are [8]:

$$\hat{I} = \log \left(1 + \frac{1 - \Pi(B_1 - 1)}{\sum_{b=B_1}^{\infty} b \pi(b) - B_1 (1 - \Pi(B_1 - 1))} \right)$$

and

$$\hat{A} = (1 - \Pi(B_1 - 1)) \exp(B_1 \hat{I})$$

where $\Pi(B_1 - 1) := \sum_{b=0}^{B_1-1} \pi(b)$. These expressions for \hat{A} and \hat{I} can be easily updated by the simulator. The two-parameter Kullback-Leibler method estimates

$$\mathbf{P}\{X \geq B\} \approx \hat{A} \exp(-B\hat{I}). \quad (3)$$

A function p with more than two unknowns can be chosen for greater accuracy (e.g., $p(b) = A \exp(-b^\nu I)$ [2]), but we found no closed form expression for the minimizing values of the unknowns. An alternative “direct” approach to estimating the parameter I is described [14], equation (10); the expression for \hat{A} above can also be used with this method to obtain the estimate in equation (3).

5.1.3 Using Generalized Extreme Value Theory

An interesting approach to estimating $\mathbf{P}\{X \geq B\}$ based on generalized extreme value theory (GEVT) is described in [1] and [13]. This approach can quickly estimate $\mathbf{P}\{X \geq B\}$ if $\lim_{B \rightarrow \infty} B^{-\nu} \log \mathbf{P}\{X \geq B\} = I < 0$ for some constant I even when $\nu \neq 1$ (see Equation (15) and Figure 1 of [1]). GEVT is more computationally intensive than the Kullback-Leibler approach, however.

5.2 Extrapolative Methods for Cell Loss Probability

In this section, we consider connections through the network that require a bound on cell loss probability. If we assume that cell loss events at each buffer are independent, then the end-to-end cell loss probability of a connection is simply the sum of the cell loss probabilities at each buffer of the connection’s virtual circuit [17]. Each buffer will be equipped with the following extrapolative estimator of cell loss probability. The cell loss probability for a buffer of size b cells is denoted by $F(b)$. We wish to estimate $F(b)$ when b is the (large) capacity of the buffer we are considering.

In [14], the following approach was proposed. Let Y be distributed as the steady-state number of cells in the buffer. Let $M = \mathbf{P}\{Y > 0\}$. The cell loss probability is then approximated by

$$F(b) \approx \hat{M} \exp(-b\hat{I})$$

where \hat{I} is estimated as in §5.1.2 and the estimate \hat{M} is also based on the empirical distribution of Y .

6 The Role of Parallelism

Even with the methods suggested in §5 above, estimation of rare events for large networks would require enormous amount of simulation time. A typical telecom switch would process about a million of packets per second [36], whereas the number of ATM cells that a typical simulator on a workstation (such as a SPARCstation) can process is of the order of several thousand cells per second [26]. Therefore, even simulating a single switch could result in a slow down by a factor ranging from a 100 to a 1000. Simulating an ATM network of a modest size and complexity for an hour of traffic could take days or even weeks. Given this limitation of a uniprocessor simulation, one might turn to the idea of employing a parallel or distributed solution to the problem. Depending on the size and complexity of the simulation, its intended goals, and the hardware and software resources at hand, the simulation can be done in several different ways. In this section, we explore various types of parallel or distributed solutions to the problem and specify the necessary hardware requirements for implementing these solutions.

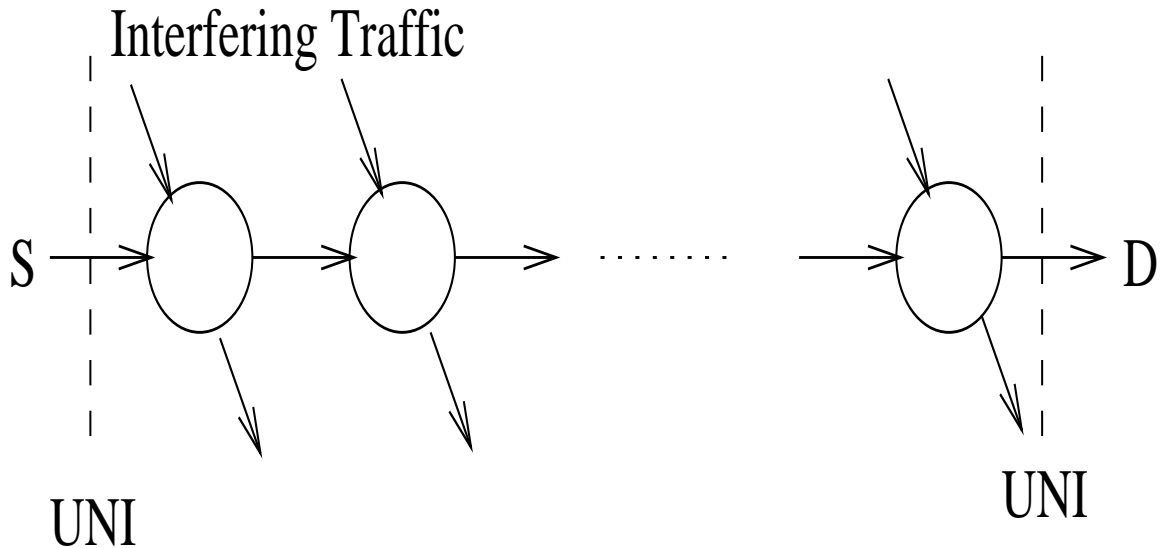


Figure 1: A Virtual Circuit Simulation Schematic

6.1 ATM Network Simulation versus Generic Discrete-Event Simulation

In this subsection we will briefly describe two aspects of ATM networks that can be exploited to obtain simulation speed up using some form of parallelism. The first aspect is that ATM networks are connection-oriented; i.e., all cells of a connection use the same path (called a virtual circuit) through the network. This preserves cell ordering and simplifies connection admission control and cell routing in the network. One can exploit the connection-oriented nature of ATM to decompose the problem of simulating a large ATM network to that of simulating the individual virtual circuits separately, see §6.2 below. The second aspect is that many cell operations (e.g., a cell transmission from a buffer) occur simultaneously. Such simultaneous events can occur in the same switch or in switches separated by great distances. Thus, an ATM network is intrinsically a parallel, distributed machine. We explore distributed simulation of ATM networks in §6.3 below.

6.2 Parallel Independent Experiments

Often, experiments involving simulation of an ATM network are quite repetitive in nature. First, several simulations of the same network may be done using different sets of parameters. Second, experiments may be replicated to get average values of desired quantities.

Alternatively, one could simulate virtual circuits separately using different workstations. Figure 1 is a schematic of one such virtual circuit. The quality of service received by the connections using the entire virtual circuit (from source “S” to destination “D”) are to be estimated by simulation. These independent simulation tasks would require approximations of the interfering traffic; since traffic streams are typically “smoothened” as they traverse an ATM network, using the source models/data for the interfering traffic could result in conservative estimates of performance. If the diameter of the network (i.e., the maximum number of buffers visited by a connection through the network) is not too large, then any of these individual virtual circuit experiments can be handled by a single CPU simulation. Also, network importance sampling approaches cited in §4.3 can be effectively used. Note that source aggregation on the interfering traffic can also be used to speed up each virtual circuit simulation.

Since the tasks executing in parallel hardly interact with each other, it may not be difficult to obtain a speed up factor of N using N workstations. Although, several experiments can be executed in parallel, a significant disadvantage of this approach is that it does not shorten the turn-around time for an individual experiment.

Such independent simulations can be profitably executed on tightly as well as loosely coupled processor networks. One environment that is particularly suited to such experiments is the workstation environment where a number of workstations are connected via a local area network. Workstations computing environments are ubiquitous today. Studies have revealed that in a typical workstation environment even during the busiest hours of the day about 50% of the workstations are idle [54]. Several researchers have shown the feasibility of using such idle computing power for the purpose of parallel processing [52, 51, 39]. Although, issues of effective load balancing and task migration when a workstation becomes occupied remain to be addressed in a satisfactory manner, high-level tools available today allow nonexperts in parallel computing to exploit idle computing power for such experiments [51, 39].

6.3 Distributed or Parallel Simulation

The nature of parallelism that is exploited by a particular application can be classified as coarse-grain to fine-grain depending on the level of interaction that is required among the parallel tasks. Fine-grain parallel applications require much more interaction among parallel tasks to achieve proper synchronization and communication. From this viewpoint, the strategy of parallelizing the simulation of an ATM network by running independent parallel simulations, as discussed in the previous subsection, falls into the category of coarse-grain parallelism as there is almost no interaction among the different parallel experiments. A finer-grain method of parallelizing an ATM network simulation would be to partition a single simulation experiment onto more than one processor. This can be done in at least two ways: First, the ATM network may be partitioned into several parts and each of the parts may be simulated on a separate processor; we call this topological partitioning. Second, several book-keeping and monitoring related functions can be simulated on separate processors; we call this functional partitioning. The following subsections discuss each of these two approaches.

6.3.1 Functional Partitioning

Functions such as simulating the traffic sources, statistics collection and storage, monitoring of simulation, etc. can be decoupled from the simulation of the network. These auxiliary functions can be done on a separate processor. Since such tasks are relatively independent of the main simulation task, it is easier to develop a parallel solution. However, the expected speed up factor with this approach will be limited if these auxiliary tasks represent only a relatively small fraction of the overall computing that is required. Previous research results indicate speed up factors ranging from 1.3 to 1.6 depending on the distributed processor architecture used for implementation and details of partitioned functions [6, 7]. Up to a maximum of 4 processors were used for these experiments and it was not possible to achieve higher speed ups by employing more processors.

6.3.2 Topological Partitioning

Conceptually, it is possible to partition an ATM network into several parts and the simulation of each of the parts may be done on a separate processor. Each partition typically handles the task of simulating the nodes belonging to it in an asynchronous manner. In an event based simulation of the network, each partition receives events from the traffic sources or other partitions. It maintains an event-list(s) where events that require processing are kept in a sorted manner (§3). If the next node that should process a cell belongs to a different partition, a message containing this cell is sent to the destination partition. In

this manner, a cell finally reaches its target node where it may be discarded after collecting results. Insuring correctness in an asynchronous event driven simulation requires a certain amount of periodic synchronization. The basic ideas for this are related to Lamport's work on logical clocks in distributed systems [37]. Two techniques -the conservative approach and the optimistic approach- are commonly employed for synchronization while using topological partitioning. Details of these approaches can be found in [44, 30, 21, 47].

Since ATM networks will have strong coupling among the partitions, and consequent higher communication and synchronization requirements, speed up factors that can be attained by this method would vary significantly depending on the cost of synchronization/communication relative to the speed of processors used for implementation. On a loosely coupled processor network, high speed ups are not possible due to higher communication costs. Indeed, many of the experiments that have been reported in the past have used slow communication LANs. It may be possible to achieve higher speed ups on loosely coupled processor networks that employ high speed LANs such as FDDI, ATM [38, 46], etc.

Of course, it is also possible to employ tightly-coupled multiprocessors to exploit parallelism via topological partitioning. In theory, each partition could have a single switch. In practice, however, the size of each partition would be determined by the grain-size of the computation that can be effectively exploited on a given parallel machine and number of processors available for the simulation. Both conservative as well as optimistic methods for asynchronous event based simulation are amenable to such medium to fine grain parallel implementations. Grain size can be adjusted by selecting the size of each partition. Some experiments with simulation of other similar communication networks have reported encouraging speed ups on multiprocessors [20, 23, 48].

6.3.3 Combined Approach

It is also possible to combine aspects of functional and topological partitioning. Most likely, this is the approach with maximum potential for speed up from parallelism. As mentioned before, auxiliary tasks such as traffic source simulation, file I/O, graphics interface for monitoring etc. may be separated from the network simulation task. The simulation of the ATM network itself then can be divided into several partitions all of which would be simulated in parallel.

6.3.4 Automatic or Semi-automatic Parallelization

Since various switches of an ATM network basically perform the same operations in parallel to different cells, there is a high degree of data parallelism inherent in the simulation of different nodes of the ATM network. Therefore, the task of automatically parallelizing the simulation of an ATM network using a high level parallel language is worth considering. The principal source of difficulty in this approach is the mapping of nodes to the processors. The general mapping problem has been proven to be NP-complete [22]. Also, the selection of appropriate simulation parameters for the purpose of achieving maximum parallelism is largely dependent on the nature of application. For example, the amount of look-ahead in the conservative approach, or how often global virtual time is calculated in the optimistic approach, are difficult to ascertain in an automatic manner. As is the case with other types of data parallel applications [41], a semi-automatic approach where the user supplies the required mapping and other simulation specific parameters whereas the system automatically generates parallel tasks based on this knowledge, seems more practical.

7 Future Directions

We are currently building an ATM simulation test-bed that would incorporate several ideas presented in this paper. A preliminary sequential version of the simulator is now functional [26]. It is currently being used to gain insights into various ATM related issues such as the

appropriate cell scheduling mechanisms as well as simulation specific issues such as suitable synchronization techniques for parallel simulation.

8 Summary

Like any other packet switched network, ATM networks can be simulated using well known methods of discrete event simulation. However, in this paper we have attempted to focus on two major challenges that distinguish the problem of ATM simulation: estimating rare events and large network dimensions. It is suggested that extrapolative methods can be used to quickly estimate a wide variety of received qualities of service. Simulation of networks with large dimensions would still require large simulation time. We outline several methods of parallelizing the simulation. Wherever applicable, conducting parallel independent experiments is the most efficient and simplest method of speeding up a network simulation. Indeed, parallel simulation of independent virtual circuit experiments, each employing extrapolative methods to estimate received quality of service, promises to be an acceptably accurate strategy requiring a reasonable amount simulation time.

A small amount of speed up may be possible by functionally or topologically partitioning the simulation task on a network of workstations. Fine grain parallelism would require tightly-coupled multiprocessors. Completely automatic parallelization of the simulation task is currently not a feasible proposition. However, a semi-automatic approach where a programmer supplies the application specific knowledge and the system then completes the rest of the parallelization task is promising.

Acknowledgement

We would like to acknowledge the helpful comments made by our reviewers and by Dr. P. Heidelberger about an earlier draft of this paper.

References

- [1] F. Bernabei, R. Ferretti, M. Listanti, and G. Zingrillo. ATM system buffer design under very low cell loss probability constraints. *IEEE INFOCOM Proc.*, pages 8c.3.1–8c.3.10, 1991.
- [2] D.D. Botvich and N.G. Duffield. Large deviations, the shape of the loss curve, and economies of scale in large multiplexers. Technical Report No. DIAS-APG-94-12, Dublin Institute for Advanced Studies, 1994.
- [3] C.-S. Chang. Stability, queue length and delay of deterministic and stochastic queueing networks. *IEEE Trans. Auto. Control*, Vol. 39:pp. 913–931, 1994.
- [4] C.-S. Chang and J. Cheng. Computable exponential bounds for intree networks with routing. *to appear in Proc. IEEE INFOCOM'95*, 1995.
- [5] C.S. Chang, P. Heidelberger, S. Juneja, and P. Shahabuddin. Effective bandwidth and fast simulations of atm intree networks. *Performance Evaluation*, Vol. 20:pp. 45–65, 1994.
- [6] J.C. Comfort. The simulation of a master-slave event set processor. *Simulation*, 42(3):117–124, March 1984.
- [7] J.C. Comfort and R.R. Gopal. Environment partitioned distributed simulation with transputers. In *Proceedings SCS MultiConf. on Dist. Simulation*, pages 103–108, 1988.

- [8] C. Courcoubetis, G. Kesidis, A. Ridder, J. Walrand, and R. Weber. Call acceptance and routing using inferences from measured buffer occupancy. *to appear in IEEE Trans. Comm.*, 1993.
- [9] G. de Veciana, C. Courcoubetis, and J. Walrand. Decoupling bandwidths for networks: A decomposition approach to resource management for networks. In *IEEE INFOCOM Proc.*, volume 2, pages 466–474, 1994.
- [10] G. de Veciana, G. Kesidis, and J. Walrand. Resource management in wide-area ATM networks using effective bandwidths. *preprint*, 1994.
- [11] G. de Veciana and J. Walrand. Effective bandwidths: Call admission, traffic policing and filtering for ATM networks. *To appear in Queueing Systems*, 1995.
- [12] M. Devetsikiotis and J. K. Townsend. Statistical optimization of dynamic important sampling parameters for efficient simulation of communication networks. *IEEE/ACM Trans. Networking*, Vol. 1, No. 3:pp. 293–305, Jan. 1993.
- [13] V. Dijk, E. Aanen, and H. van den Berg. Extrapolating ATM-simulation results using extreme value theory. *Queueing, Performance and Control in ATM (ITC-13)*, Elsevier Science Publishers B.V. (North-Holland):pp. 97–104, 1991.
- [14] N. G. Duffield, J.T. Lewis, N. O’Connell R. Russell, and F. Toomey. The entropy of an arrivals process: A tool for estimating QoS parameters of ATM traffic. In *Proceedings of the 11th IEE Teletraffic Symposium*, March 1994.
- [15] N.G. Duffield and N. O’Connell. Large deviations and overflow probabilities for the general single-server queue, with applications. Technical Report No. DIAS-APG-93-30, Dublin Institute for Advanced Studies, 1993.
- [16] A.I. Elwalid and D. Mitra. Effective bandwidth of general Markovian traffic sources and admission control of high speed networks. *IEEE/ACM Trans. Networking*, Vol. 1, No. 3:pp. 329–343, June 1993.
- [17] D. Ferrari and D.C. Verma. A scheme for real-time channel establishment in wide-area networks. *IEEE JSAC*, Vol. 8, No. 3:pp. 368–379, 1990.
- [18] J.A. Freebersyser, M. Devetsikiotis, and J. K. Townsend. Efficient simulation of high speed tandem networks using importance sampling and stochastic gradient techniques. In *Proc. Globecom’94*, pages 1095–1099, 1994.
- [19] V.S. Frost and B. Melamed. Simulating telecommunications networks with traffic modeling. *IEEE Communications Magazine*, Vol. 32, No. 3:pp. 70–81, Mar. 1994.
- [20] R.M. Fujimoto. Performance measurement of distributed simulation strategies. In *Proceedings SCS Conf. on Dist. Simulation*, pages 14–20, 1988.
- [21] R.M. Fujimoto. Parallel discrete event simulation. *Communications of the ACM*, 33(10):30–53, October 1990.
- [22] M.R. Garey and D.S. Johnson. Computers and intractability: A guide to the theory of NP-completeness. *Freeman*, 1979.
- [23] J.B. Gilmer Jr. An assessment of time warp parallel discrete event simulation. In *Proceedings of SCS Multiconf. on Distributed Simulation*, pages 45–49, 1988.
- [24] P.W. Glynn and W. Whitt. Logarithmic asymptotics for steady-state tail probabilities in a single-server queue. *J. Appl. Prob.*, 31, 1994.

- [25] P. Heidelberger. Fast simulation of rare events in queueing and reliability models. Technical Report No. RC 19028 (83112), IBM, 1993.
- [26] A. Hung and G. Kesidis. SimATM: A cell-level ATM network simulator. Technical Report (draft), 1993.
- [27] A. Hung and G. Kesidis. Resource management of prerecorded VBR video sources in ATM networks. *in preparation*, 1995.
- [28] C.-H. Hwang and S.-Q. Li. On input state reduction of buffer noneffective region. In *Proc. IEEE INFOCOM'94*, pages 1018–1028, 1994.
- [29] C.-H. Hwang and S.-Q. Li. On the convergence of traffic measurements and queueing analysis: a Statistical-MATCH Queueing (SMAQ) tool. *to appear in Proc. IEEE INFOCOM'95*, 1995.
- [30] D.R. Jefferson. Virtual time. *ACM Trans. on Prog. Lang. and Sys.*, 7(3):404–425, 1985.
- [31] F.P. Kelly. Effective bandwidths of multi-class queues. *Queueing Systems*, Vol. 9, No. 1:pp. 5–16, 1991.
- [32] G. Kesidis. A traffic regulator for effective bandwidth usage parameter control in ATM networks. Technical Report No. 93-03, E&CE Dept, Univ. of Waterloo, 1993.
- [33] G. Kesidis. Specification of a fluid ATM network simulator. Technical Report (in preparation), 1995.
- [34] G. Kesidis and J. Walrand. Quick simulation of ATM buffers with multiclass on-off Markov fluid sources. *ACM TOMACS*, Vol. 3, No. 3:pp. 269–276, July 1993.
- [35] G. Kesidis, J. Walrand, and C.-S. Chang. Effective bandwidths for multiclass Markov fluids and other ATM sources. *IEEE/ACM Trans. Networking*, Vol. 1, No. 4:pp. 424–428, Aug. 1993.
- [36] J.F. Kurose and H.T. Mouftah. Computer-aided modeling, analysis, and design of communication networks. *IEEE JSAC*, 6(1):130–145, January 1988.
- [37] L. Lamport. Time, clock, and ordering of events in a distributed system. *Comm. of the ACM*, 21(7):558–565, 1978.
- [38] M. Lin et al. Distributed network computing over local atm networks. In *Proceedings of IEEE Supercomputing'94 Conference*, pages 154–163, August 1994.
- [39] M.J. Litzkow, Livny M., and M.W. Mutka. Condor—a hunter of idle workstations. In *Proceedings of the Conf. on Distributed Computing Systems*, pages 104–111, June 1988.
- [40] Z. Liu, P. Nain, and D. Towsley. Exponential bounds for a class of stochastic processes with applications to call admission control in networks. In *IEEE CDC Proc.*, pages 156–161, 1994.
- [41] D.B. Loveman. High performance Fortran. *IEEE Parallel and Distributed Technology*, 1(1):25–42, February 1993.
- [42] S. Low and P. Varaiya. A new approach to service provisioning in ATM networks. *IEEE/ACM Trans. Networking*, Vol. 1, No. 5:pp. 547–553, Oct. 1993.
- [43] B. Melamed and D. Pendarakis. A Markov-renewal-modulated TES model for VBR Star Wars video. *preprint*, 1995.
- [44] J. Misra. Distributed discrete-event simulation. *Computing Surveys*, 18(1):39–65, 1986.

- [45] R. Nagarajan. *Quality-of-Service issues in high-speed networks*. PhD thesis, CS Dept, University of Massachusetts at Amherst, 1993.
- [46] P. Newman. ATM local area network. *IEEE Communication Magazine*, 32:86–98, March 1994.
- [47] D. Nicol and R. Fujimoto. Parallel simulation today. *ORSA Journal of Computing*, 94.
- [48] D. Nicol and P. Heidelberger. On extending parallelism to serial simulators. *Technical report, IBM Research Report RC 19863*, August 1994.
- [49] N. O’Connell. Large deviations in queueing networks. Technical Report No. DIAS-APG-94-13, Dublin Institute for Advanced Studies, 1994.
- [50] 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:pp. 344–357, Jun. 1993.
- [51] J. Schaeffer, D. Szafron, G. Lobe, and I. Parsons. The enterprise model for developing distributed applications. *IEEE Parallel and Distributed Technology*, 1(3):85–96, 1993.
- [52] A. Singh, J. Schaeffer, and M. Green. A template-based approach to the generation of distributed applications using a network of workstations. *IEEE Transactions of Parallel and Distributed Systems*, 2(1):52–67, January 1991.
- [53] B. Stroustrup. C++ programming language, 2nd edition. *Addition-Wesley*, 1991.
- [54] M. Theimer, K. Lantz, and D. Cheriton. Preemptable remote execution facility for the V system. In *Proceedings of 10th ACM Symposium on Oper. Sys. Principles*, pages 2–12, December 1985.