IMPORTANCE SAMPLING FOR LARGE ATM-TYPE QUEUEING NETWORKS

Pierre L'Ecuyer and Yanick Champoux

Département d'Informatique et de Recherche Opérationnelle
Université de Montréal, C.P. 6128, Succ. Centre-Ville
Montréal, H3C 3J7, CANADA

ABSTRACT

We estimate, by simulation, the cell-loss rate in an ATM switch modeled as a queueing network. Cell losses are rare events, so estimating their frequency by simulation is hard. We experiment with importance sampling as a mean of improving the simulation efficiency in that context.

1. INTRODUCTION

An Asynchronous Transfer Mode (ATM) communication switch can be modeled as a network of queues with finite buffer sizes. Cells (or packets) of information join the network in a stochastic manner and some may be lost due to buffer overflow. The long-term (or steady-state) fraction of cells that are lost at a given node is called the cell-loss rate (CLR) at that node. Typical CLRs are small and the cell losses also tend to occur in bunches. They are therefore rare events, so estimating the CLRs with reasonable precision by straightforward simulation is extremely time-consuming—in some cases practically impossible.

Efficiency improvement methods have been proposed to deal with such a situation. Most of these methods improve the efficiency by reducing the variance of the estimator, and are called *variance reduction techniques*. For rare events, *importance sampling* (IS) seems the method of choice. It changes the probability laws governing the system so that the rare events of interest occur more frequently, and eventually are no longer rare events. The estimator is also changed accordingly (multiplied by a likelihood ratio) so that it remains unbiased.

For general background on efficiency improvement, consult Bratley, Fox, and Schrage (1987), Glynn (1994) and L'Ecuyer (1994). For more on IS, see Glynn and Iglehart (1989), Heidelberger (1995), Shahabuddin (1994) and the several other references given there. Application of IS to the simulation of communication systems is studied by Chang et al. (1994), Chang, Heidelberger, and Shahabuddin (1995), Fleming, Schaeffer, and Simon (1995), among several others. A somewhat related approach, which also concentrates the simulation effort on the more interesting events, is *splitting* (called "RESTART" by some authors); see, e.g., Glasserman et al. (1996).

By far the most difficult problem in the practical application of importance sampling is to find an appropriate change of measure; that is, figure out how to change the probability laws so that the variance gets reduced to an acceptable level. Theoretically, there always exists a change of measure that reduces the variance to an arbitrary small value, but it is usually much too complicated and too difficult to find.

Chang et al. (1994) proposed an approach, based on the theories of *effective bandwidth* and *large deviations*, to derive an "asymptotically optimal" change of measure for estimating the probability p that a queue length exceeds a given level x before returning to empty, given that the queue is started from empty, for a single queue with multiple independent arrival sources. Roughly, *asymptotically optimal* means that the standard error of the IS estimator converges to zero exponentially fast with the same decay rate (exponent) as the quantity to be estimated, as a function of the level x. For a more precise mathematical statement, see Chang et al. (1994) and Heidelberger (1995). An asymptotically optimal change of measure does not (generally) minimize the variance, but can reduce it by several orders of magnitude.

The probability p just described is closely related to the CLR, so this change of measure could be used as well to estimate the CLR in a single queue with finite buffer size. Chang et al. (1994) extended their method to *intree* networks of queues, which are acyclic tree networks where cells flow only towards the root of the tree. The arrival sources can feed any node. For intree networks, they did not prove that their proposed change of measure is asymptotically optimal, but conjectured that it is close and gave an upper bound on the variance. They reported numerical experiments with single-node and two-node queueing systems, and indeed observed spectacular variance reductions.

This paper reports further experimentation with this approach, for larger queueing networks of a specific type. Our findings confirm the large variance reductions observed by Chang et al. (1994) in general. We make an heuristic adaptation to a certain class of non-intree networks and observe large variance reductions as well, under certain conditions. The next section describes the queueing network model. Section 3 recalls how to compute confidence intervals for the CLR via the A-cycle method. Section 4 explains how importance sampling is applied. The numerical results are reported in Section 5.

2. THE MODEL

We consider an acyclic queueing network whose *nodes* are partitioned into four *levels*. Each node is a single-server FIFO queue with finite buffer size. Levels 2 and 3 have m_2 nodes each, while levels 1 and 4 have $m_1 m_2$ nodes each. Each level-2 node is fed by m_1 level-1 nodes, while each level-3 node feds m_1 nodes at level 4. The "customers" in the network are (indentical) *cells* containing bits of information. They arrive at level 1, then move ahead to levels 2, 3, and 4, in succession, before leaving the network. Each level-1 node is fed by m_0 independent arrival *sources*. Each source is assigned to a specific destination at level 4, so all cells from this source have a common trajectory in the network. The assignments of destinations to sources is fixed (deterministically) beforehand. In practice, we may be interested in a random assignment of destinations to sources, but in that case it is probably better to stratify the experiment over the set of possible assignments. Here, we concentrate on what to do after the assignment has been fixed.

For $\ell = 1, \ldots, 4$, all level-ℓ nodes have the same buffer size B_ℓ and the same constant service time $1/c_\ell$ (so c_ℓ is the service rate). Whenever a cell arrives at a node where the buffer is full, it is *lost* and just disappears.

The $m_0 m_1 m_2$ arrival sources are *iid* Markov modulated processes. A source is *off* for a while, then *on* for a while, then *off* for a while, and so on. During a *on* period, cells arrive at a constant rate, one cell per unit of time, whereas during a *off* period, none arrives from that source. The durations of *off* and *on* periods are independent geometric random variables with respective means κ_0 and κ_1. The parameter κ_1 is called the *average burst size*.

We want to estimate the fraction of cells lost (the CLR) at a given level of the network (among those reaching that level), in steady-state. For this, we concentrate on a selected node of the network, say node q^* at level ℓ^*, and trim down from the network all nodes at level 3 or 4 from which node q^* cannot be reached. An alternative would be to take the average CLR for all nodes at a given level as the estimator. With a straightforward simulation approach, this would yield a better estimator than concentrating on a single node. But with IS, it seems better to concentrate on a single node, and increase only the traffic to that node, to control the variance of the likelihood ratio.

Other variants of the model could be considered. For example, each arriving cell could have its destination determined randomly, or the destination could be generated randomly for each *on* period (or "burst") of each source. These models may be more difficult to handle with IS (in general) than the one we consider, because the likelihood ratio would tend to have more terms. Our fixed-assignment model is reasonable because in communication networks, a typical connection between a source and a destination would last for a period of time several orders of magnitude larger than the average time between bursts.

3. CONFIDENCE INTERVALS

To compute a confidence interval on μ, one needs to estimate the variance of $\hat{\mu}$. For this, we apply a generalization of the classical regenerative method, called the A-*cycle* method, introduced and used by Nicola et al. (1993) and Chang et al. (1994). Let A be a subset of the state space of the system. In this paper, A is taken as the set of states for which the queue at node q^* is empty. Let $t_0 = 0$ and let $t_1, t_2, \ldots$ be the successive times at which the system's state enters the set A. The system's state at those entering times t_i has a steady-state distribution π defined by:

$$\pi(\cdot) \stackrel{\text{def}}{=} \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} P\{\text{state} \in \cdot \text{ at time } t_i\}.$$

The process between times t_{i-1} and t_i is called the ith A-cycle. Let X_i denote the number of arrivals to node q^* during the ith A-cycle and Y_i be the number of lost cells among those X_i arrivals. Let E_π denote the mathematical expectation over an A-cycle when the initial state (at the beginning of the A-cycle) has distribution π. One has:

$$\mu = \frac{\mathrm{E}_\pi[Y_1]}{\mathrm{E}_\pi[X_1]}. \tag{1}$$

In the limit, as the number of A-cycles increases, the *average* distribution of the system states at the times

t_i approaches π. To reduce the initial bias, one may *warm-up* the system by discarding (say) the first n_0 A-cycles from the statistics. Then, take the averages of the Y_i and X_i over the next n A-cycles, which yields the estimator:

$$\hat{\mu} = \frac{\sum_{i=1}^{n} Y_i}{\sum_{i=1}^{n} X_i}.$$

Under mild conditions which are assumed to hold here, π is also a steady-state distribution in the pointwise sense. The A-cycles are then asymptotically identically distributed (their initial state follows π) but *not independent*. To reduce the dependence, and also improve the normality, one can *batch* them, as in the batch means method (Bratley, Fox, and Schrage 1987). One then applies the standard methodology for computing a confidence interval for a ratio of expectations, using the batch means as observations.

Suppose we take b batches of $k = n/b$ successive observations (X_i, Y_i) each. For $1 \le j \le b$, let $(\bar{X}_j, \bar{Y}_j)$ be the mean within batch j, and put

$$\bar{Z}_j \quad = \quad \bar{Y}_j - \mu \bar{X}_j.$$

The $\bar{Z}_j$ have zero expectation and variance

$$\mathrm{Var}[\bar{Z}_j] = \mathrm{Var}[\bar{Y}_j] - 2\mu \mathrm{Cov}[\bar{X}_j, \bar{Y}_j] + \mu^2 \mathrm{Var}[\bar{X}_j],$$

which can be estimated by

$$S_Z^2 = S_Y^2 - 2\hat{\mu} S_{XY} + \hat{\mu}^2 S_X^2$$

where $\bar{X}$, S_X^2, $\bar{Y}$, S_Y^2, S_{XY} denote the sample mean and variance of the $\bar{X}_j$'s, the sample mean and variance of the $\bar{Y}_j$'s, and the sample covariance between the $\bar{X}_j$'s and $\bar{Y}_j$'s, respectively. Assuming that the Z_j are iid normally distributed and denoting $\bar{Z} = \bar{Y} - \mu \bar{X}$, one obtains that

$$\frac{\sqrt{b}\bar{Z}}{S_Z} \quad = \quad \frac{\sqrt{b}\bar{X}(\hat{\mu} - \mu)}{S_Z}$$

has a Student-t distribution with $b - 1$ degrees of freedom, which is approximately standard normal for large enough b. From this, one can compute a confidence interval on μ in a standard way.

4. APPLYING IMPORTANCE SAMPLING

To estimate the CLR μ at node q^*, the straightforward approach is to simulate until n cells reach node q^*, and define $\hat{\mu}$ as the fraction of those lost due to a full buffer. This estimator turns out to have *relative error* (the standard deviation divided by the mean) $\mathrm{RE}[\hat{\mu}] = O((n\mu)^{-1/2}))$, which is unbounded as $\mu \to 0$, so one must find a better estimator.

We will use IS, with the methodology developed by Chang et al. (1994) for choosing the change of measure. We now explain how this can be applied to our model. The two quantities to estimate are the numerator and denominator in (1). The denominator $\mathrm{E}_\pi[X_1]$ is easy to estimate just by simulating several A-cycles without IS. The numerator is more difficult because it involves rare events, and IS will be used for it, as follows.

Let 0 and 1 denote the states *off* and *on* of a source. Due to the geometric assumption, each source evolves as a discrete-time Markov chain with state space $\{0, 1\}$ and transition matrix R with elements r_{ij} given by $r_{01} = 1/\kappa_0$, $r_{00} = 1 - r_{01}$, $r_{10} = 1/\kappa_1$, $r_{11} = 1 - r_{10}$. Its arrival rate is $\rho = \kappa_1/(\kappa_1 + \kappa_0)$.

Let S^* denote the set of sources feeding q^*. To increase the traffic at node q^* (and get more cell losses), we change the matrix R for all the sources in S^*, so they spend more time in the *on* state. At the beginning of an A-cycle with IS, the transition matrix R for the sources in S^* is replaced by

$$\tilde{R} = \begin{pmatrix} \tilde{r}_{00} & \tilde{r}_{01} \\ \tilde{r}_{10} & \tilde{r}_{11} \end{pmatrix} = \begin{pmatrix} r_{00}/K_0 & r_{01}e^\theta/K_0 \\ r_{10}/K_1 & r_{11}e^\theta/K_1 \end{pmatrix},$$

where $\theta \ge 0$, and K_0 and K_1 are (positive) normalizing constants ensuring that the lines sum up to 1. We will explain later on how to choose θ. Note that $\theta > 0$ increases the average input rate from the source while $\theta = 0$ leaves it (and R) unchanged. This new matrix $\tilde{R}$ is in effect until buffer q^* fills up or empties again, whichever comes first. When q^* fills up, the IS is turned off ($\tilde{R}$ is replaced by R) until the end of the A-cycle. We call this an *A-cycle with IS*.

For a given initial state, let $\tilde{\mathrm{E}}$ denote the expectation over an A-cycle with IS. Let N_{ij} be the number of times a source in S^* goes from state i to state j *while using the new probabilities* $\tilde{r}_{ij}$, during the A-cycle, for $i = 0, 1$ and $j = 0, 1$, and let $N_T = N_{00} + N_{01} + N_{10} + N_{11}$, the total number of transitions generated from $\tilde{R}$. The *likelihood ratio* associated with this change of measure is then

$$L = \left(\frac{r_{00}}{\tilde{r}_{00}}\right)^{N_{00}} \left(\frac{r_{01}}{\tilde{r}_{01}}\right)^{N_{01}} \left(\frac{r_{10}}{\tilde{r}_{10}}\right)^{N_{10}} \left(\frac{r_{11}}{\tilde{r}_{11}}\right)^{N_{11}}.$$

If V is a random variable computed during an A-cycle, then

$$\mathrm{E}[V] = \tilde{\mathrm{E}}[LV],$$

so computing LV over an A-cycle with IS yields an unbiased estimator of $\mathrm{E}[V]$.

Generally, the variance of L may increase exponentially fast with N_T, so (intuitively) one would like to keep N_T small while still making cell losses frequent enough. This is why the A-cycle methodology, with short A-cycles, is to be preferred over using IS with longer simulations.

One simulates two "versions" of each A-cycle, one with IS and the other without, both starting from the same initial state. Thus, the A-cycles come in pairs. For the ith A-cycle pair, the simulation with IS provides an estimation $W_i = L_i Y_i$ of the numerator, where L_i and Y_i are the value of the likelihood ratio and the number of cell losses for this cycle, while the no-IS one provides an estimator X_i of the denominator. The final state of the no-IS A-cycle, which obeys approximately distribution π, is taken as the initial state for the next pair of A-cycles.

For $\ell^* = 1$ or 2, one has an intree network, for which all sources feed q^*, so θ can be chosen as proposed by Chang et al. (1994). Let $\lambda(\theta)$ be the spectral radius (largest eigenvalue) of the matrix

$$\begin{pmatrix} r_{00} & r_{01}e^{\theta} \\ r_{10} & r_{11}e^{\theta} \end{pmatrix}$$

and $f(\theta)$ be the corresponding eigenvector. Define $\psi_0(\theta) = \theta^{-1}\ln(\lambda(\theta))$. Compute θ and the corresponding $\tilde{r}_{ij}$ as follows:

Algorithm 1 (For $\ell^* = 1$ or 2.)
1. If $\ell^* = 1$, then
   compute θ_1^*, a solution of $m_0\psi_0(\theta) = c_1$
 else
   compute $\tilde{\theta}_1$, solution of $d(m_0 \ln \lambda(\theta))/d\theta = c_1$;
   Let
   $$\psi_1(\theta) = \begin{cases} m_0\psi_0(\theta) & \text{if } \theta \le \tilde{\theta}_1; \\ c_1 - \tilde{\theta}_1(c_1 - m_0\psi_0(\tilde{\theta}_1)) & \text{otherwise;} \end{cases}$$
   Compute θ_2^*, a solution of $m_1\psi_1(\theta) = c_2$;
   Let $\theta_1^* = \min(\theta_2^*, \tilde{\theta}_1)$;
2. For each (i, j), define
$$\tilde{r}_{ij} = \frac{\exp(j\theta_1^*) f_j(\theta_1^*)}{\lambda(\theta_1^*) f_i(\theta_1^*)} r_{ij}.$$

For $\ell^* = 3$ or 4, the network is no longer intree, in the sense that many cells exit before reaching the root q^*. We nevertheless heuristically adapt the algorithm of Chang et al. (1994) as follows. We change r_{ij} to $\tilde{r}_{ij}$ only for the souces in S^*. When choosing θ, we neglect all the traffic not directed towards node q^*. To simplify, we also neglect the possibility that the effective bandwidth exceeds the service rate at a node other than q^*. This is reasonable because in our setup, S^* typically contains only a small fraction of the sources. This yields the following algorithm, where s^* is the cardinality of S^*, and where λ, f, and ψ_0 are defined as before.

Algorithm 2 (For $\ell^* = 3$ or 4.)
1. Compute θ^*, a solution to $s^*\psi_0(\theta) = c_{\ell^*}$;
2. Define
$$\tilde{r}_{ij} = \frac{\exp(j\theta^*) f_j(\theta^*) r_{ij}}{\lambda(\theta^*) f_i(\theta^*)}.$$

5. NUMERICAL RESULTS

5.1. The Setup

In the examples that follow, the simulation was run first using the straightforward approach without IS, for b batches of k A-cycles each, then with IS, for b batches of k pairs of A-cycles each. The tables report the value of the CLR estimator $\hat{\mu}$, its variance estimate $\hat{\sigma}^2 = S_Z^2/(b\bar{X}^2)$, the relative half-width $\hat{\Delta} = 2.57 S_Z/(\sqrt{b}\bar{X}\hat{\mu}) = 2.57 S_Z/(\sqrt{b}\bar{Y})$ of a 99% confidence interval on μ (under the normality assumption), the CPU time t (in seconds) required to perform the simulation, and the estimated *relative efficiency*, defined as $\hat{\mu}^2/(t\hat{\sigma}^2)$. These values are noisy but give a rough indication of what happens. For the cases where no cell loss was observed in all the A-cycles simulated, we put $\hat{\mu} = 0$ and the entries for the variance and efficiency estimates are left blank. The IS adds overhead: it takes more CPU time than no-IS for the same total number of simulated cells—up to 15 times more in our implementation. So, for the IS estimator to win (be more efficient) it must have approximately 15 times less variance. To compare IS with no-IS, one must look at the efficiency (eff.) columns. Beware of comparing the CPU times and efficiencies across the tables, because the experiments were run on different machines (SUN SparcStations 4, 5, and 20).

For all the examples b has been fixed to 200.

5.2. CLR Estimation at Level 1

Example 1 Take $B_1 = 512$, $m_0 = 4$, $\rho = 1/10$, $c_1 = 1$ and vary the average burst size κ_1. Let $k = 3\,000$ for IS and $k = 50\,000$ for no-IS. Table 1 gives the results. For large average burst sizes, the CLR μ is high and easy to estimate, with or without IS. But for small burst sizes (the other parameters remaining the same), μ is small and much more difficult to estimate. Then, IS is much better than no-IS. With $\kappa_1 = 50$ the no-IS estimator is no longer trustworthy, and for $\kappa_1 = 10$ and 25 not even a single cell loss was observed in the kb A-cycles of the no-IS sample.

We made several other experiments where we varied the buffer size B_1 or the number of sources m_0 (with $m_0\rho$ fixed). As B_1 increases, μ decreases exponentially fast and the no-IS estimator quickly becomes useless, whereas IS works fine. As a function of m_0, μ increases slowly, so IS produces larger gains for small m_0.

5.3. CLR Estimation at Level 2

Example 2 Let $B_1 = 256$, $B_2 = 1024$, $m_0 = 8$,

$m_1 = 4$, $c_1 = 2$, $c_2 = 4$, $\rho = 1/21$, and vary the average burst size κ_1. The average input rate at the level 2 node is thus approximatively 1.5 cells per unit of time, whereas the service rate is 4. Here, $k = 250$ for IS and $k = 7\,500$ for no-IS. Table 2 gives the results, which are similar to those of Example 1, but with smaller CLR values and more spectacular improvement for the IS over the no-IS. For any κ_1, it appears difficult to estimate μ with significant precision with no-IS.

Example 3 Same as for the previous example, except that κ_1 is now fixed at 50 and we vary the buffer size B_2. We take $k = 250$ for IS and $k = 7\,500$ for no-IS. The results are given in Table 3. As the cell-loss rate is smaller than 10^{-6} (the value around which no-IS estimation passes from difficult to near-impossible) for all buffer sizes, no cell losses was observed in any of the no-IS simulations. The CPU time used by IS increases with the buffer size while it remains constant for no-IS. This is because, when the buffer size is larger, it takes more time to fill it (and ultimately observe a cell loss) under IS. The CPU time for the no-IS is not affected by the buffer size simply because the simulation does not try to cause an overflow, and so does not work more when the buffer is larger.

5.4. CLR Estimation at Level 3

Example 4 Let $B_1 = B_2 = 512$, $B_3 = 256$, $c_1 = 1$, $c_2 = c_3 = 2$, $m_0 = 2$, $m_1 = 3$, $m_2 = 10$, $\rho = 1/21$, and we vary the average burst size κ_1. We take $k = 500$ and $9\,000$ for IS and no-IS, respectively. We assign 6 sources to the node of interest at level 3. Table 4 gives the results. While no-IS has difficulty to observe a cell loss, IS gives reasonable estimations.

Example 5 Same as for the previous example, except that κ_1 is fixed at 50 and we vary the buffer size B_3. We take $k = 500$ and $9\,000$ (for IS and no-IS). Results appears in Table 5. Again, the IS works fine while the no-IS observes no cell loss except at the lowest buffer size.

Example 6 Same as the two previous example, except that κ_1 is fixed at 50 and the buffer size B_3 at 256. We take $k = 500$ and $9\,000$ (for IS and no-IS) and we vary the number of sources directed towards the observed node. Table 6 gives the results. IS again dominates as cell losses become sufficiently rare.

5.5. CLR Estimation at Level 4

Example 7 Let $B_1 = B_2 = B_3 = B_4 = 512$, $c_1 = c_4 = 1$, $c_2 = c_3 = 4$, $m_0 = 5$, $m_1 = 10$, $m_2 = 6$, $\rho = 1/41$ and we vary the average burst size κ_1. We assign 6 sources to the node of interest at level 4. We take $k = 250$ and $4\,000$, for IS and no-IS, respectively. The results are in Table 7. IS still gives far more better performances than no-IS.

Example 8 Same as the previous example, except that κ_1 is fixed at 50 and we vary the buffer size B_4. We take $k = 250$ and $4\,000$, for IS and no-IS. The results appears in Table 8. Again, IS is more effective than no-IS.

5.6. Variants of the Algorithms

The algorithms of Chang et al. (1994), used here, provide good changes of measures in an asymptotic sense, but not the best possible values of θ. Moreover, their aim is to reduce the variance and they do not take into account the differences in computational costs. We made additional experiments where θ was varied around the value given by the algorithm, to see whether the variance and efficiency would improve. For all levels, the optimal θ was generally slightly smaller but very close to the one prescribed by the algorithm.

When estimating the CLR at level 4 with IS, when the target buffer overflows at level 4 and the IS is turned off, there should be a large number of cells already in the network at previous levels, which may produce more cell losses than necessary. So, perhaps IS could be turned off earlier; e.g., when the total number of cells in buffer q^* or at previous nodes but on their way to q^*, reaches some threshold. We experimented with this idea and (for our model) obtained no significant improvement over the basic method which turns off the IS when the buffer q^* overflows.

Another idea is to play with different definitions of the A-cycles. For example, instead of starting a new A-cycle whenever q^* is empty, start it whenever the number of cells in the buffer crosses β upward, where β is a fixed integer. We tried this but did not obtain much success in terms of efficiency improvement. When increasing β, the no-IS A-cycles tend to become excessively long.

One can also impose a lower bound, say, t_0 on the length of the A-cycles, to avoid lots of extremely short A-cycles, which tends to occur under both the IS and no-IS setup. In our experiments, values of t_0 between 50 and 100 (roughly) gave slight efficiency improvements.

ACKNOWLEDGMENTS

This work has been supported by NSERC-Canada grant # OGP0110050 and FCAR-Québec grant # 93-ER-1654 to the first author, as well as a scholarship from *Newbridge Networks Corporation* to the second author.

REFERENCES

Bratley, P., B. L. Fox., and L. E. Schrage. 1987. *A Guide to Simulation*. Second ed. New York: Springer-Verlag.

Chang, C. S., P. Heidelberger., S. Juneja., and P. Shahabuddin. 1994. Effective bandwidth and fast simulation of ATM intree networks. *Performance Evaluation*, 20:45–65.

Chang, C. S., P. Heidelberger., and P. Shahabuddin. 1995. Fast simulation of packet loss rates in a shared buffer communications switch. *ACM Transactions on Modeling and Computer Simulation*, 5(4):306–325.

Fleming, P. J., D. Schaeffer., and B. Simon. 1995. Efficient Monte Carlo simulation of a product-form model for a cellular system with dynamic resource sharing. *ACM Transactions on Modeling and Computer Simulation*, 5(1):3–21.

Glasserman, P., P. Heidelberger., P. Shahabuddin., and T. Zajic. 1996. Splitting for rare event simulation: Analysis of simple cases. In these proceedings.

Glynn, P. W. 1994. Efficiency improvement techniques. *Annals of Operations Research*, 53:175–197.

Glynn, P. W., and D. L. Iglehart. 1989. Importance sampling for stochastic simulations. *Management Science*, 35:1367–1392.

Heidelberger, P. 1995. Fast simulation of rare events in queueing and reliability models. *ACM Transactions on Modeling and Computer Simulation*, 5(1):43–85.

L'Ecuyer, P. 1994. Efficiency improvement via variance reduction. In *Proceedings of the 1994 Winter Simulation Conference*, 122–132. IEEE Press.

Nicola, V. F., P. Shahabuddin., P. Heidelberger., and P. W. Glynn. 1993. Fast simulation of steady-state availability in non-markovian highly dependable systems. In *Proceedings of the 23rd International Symposium on Fault-Tolerant Computing*, 38–47. IEEE Computer Society Press.

Shahabuddin, P. 1994. Importance sampling for the simulation of highly reliable markovian systems. *Management Science*, 40(3):333–352.

AUTHOR BIOGRAPHIES

PIERRE L'ECUYER is a professor in the "Département d'Informatique et de Recherche Opérationnelle", at the University of Montréal. He received a Ph.D. in operations research in 1983, from the University of Montréal. From 1983 to 1990, he was with the computer science department, at Laval University, Québec. He obtained the *E. W. R. Steacie* grant from NSERC-Canada for the period 1995–97. His main research interests are random number generation, sensitivity analysis and optimization of discrete-event stochastic systems, and discrete-event simulation in general. He is the Departmental Editor for the Simulation Department of *Management Science* and an Area Editor for the *ACM Transactions on Modeling and Computer Simulation*. Check his web page at `http://www.iro.umontreal.ca/~lecuyer`.

YANICK CHAMPOUX is currently M.Sc. student in the "Département d'Informatique et de Recherche Opérationnelle", at the University of Montréal. He works on the application of efficiency improvement techniques for the simulation of large queueing networks. He completed his B.Sc. in mathematics (statistics) at the same university in 1996.

Table 1: CLR Estimation at Level 1 for Different Burst Sizes

	no-IS					IS				
κ_1	$\hat{\mu}$	$\hat{\sigma}^2$	$\hat{\Delta}$	cpu	eff.	$\hat{\mu}$	$\hat{\sigma}^2$	$\hat{\Delta}$	cpu	eff.
10	0			1270		1.10E-22	1.7E-49	0.95%	2492	29
25	0			1302		7.27E-10	1.8E-23	1.5%	1425	21
50	4.42E-6	3.0E-12	101%	1314	2.0E-2	8.91E-6	4.8E-15	2.0%	934	18
100	8.71E-4	2.6E-9	15%	1323	2.4E-1	9.09E-4	1.3E-10	3.2%	661	10
150	4.38E-3	1.5E-8	7.2%	1329	9.3E-1	4.32E-3	3.8E-9	3.6%	583	8.5

Table 2: CLR Estimation at Level 2 for Different Burst Sizes

	no-IS					IS				
κ_1	$\hat{\mu}$	$\hat{\sigma}^2$	$\hat{\Delta}$	cpu	eff.	$\hat{\mu}$	$\hat{\sigma}^2$	$\hat{\Delta}$	cpu	eff.
10	0			2208		2.89E-41	8.2E-83	81%	2613	3.9E-3
25	0			2190		4.62E-21	3.9E-43	35%	2453	2.2E-2
50	0			2972		2.60E-11	4.5E-24	21%	2340	6.4E-2
100	1.54E-6	2.4E-12	256%	3161	3.7E-6	1.66E-7	3.3E-16	28%	2230	3.7E-2
150	7.34E-6	1.3E-11	123%	3269	3.7E-4	3.86E-6	2.2E-13	31%	2288	3.0E-2

Table 3: CLR Estimation at Level 2 for Different Buffer Sizes

	no-IS					IS				
B_2	$\hat{\mu}$	$\hat{\sigma}^2$	$\hat{\Delta}$	cpu	eff.	$\hat{\mu}$	$\hat{\sigma}^2$	$\hat{\Delta}$	cpu	eff.
512	0			2184		1.91E-7	1.3E-16	15%	1273	2.3E-1
764	0			2183		2.36E-9	2.8E-20	18%	1496	1.3E-1
1024	0			2184		2.60E-11	4.5E-24	21%	1719	8.7E-2
1280	0			2186		3.43E-13	1.7E-27	30%	1929	3.7E-2
1536	0			2184		5.35E-15	2.5E-30	76%	2113	5.4E-3

Table 4: CLR Estimation at Level 3 for Different Burst Sizes

	no-IS					IS				
κ_1	$\hat{\mu}$	$\hat{\sigma}^2$	$\hat{\Delta}$	cpu	eff.	$\hat{\mu}$	$\hat{\sigma}^2$	$\hat{\Delta}$	cpu	eff.
10	0			1088		5.44E-20	8.6E-42	14%	1454	2.4E-1
25	0			1050		1.29E-10	2.8E-23	11%	1213	4.9E-1
50	0			1125		6.04E-7	7.2E-16	11%	1042	4.8E-1
100	2.09E-5	2.1E-10	178%	1134	6.9E-3	4.06E-5	6.7E-12	16%	881	2.8E-1
150	1.09E-4	1.6E-9	94%	1136	1.5E-2	1.68E-4	9.1E-11	15%	813	3.8E-1

Table 5: CLR Estimation at Level 3 for Different Buffer Sizes

	no-IS					IS				
B_3	$\hat{\mu}$	$\hat{\sigma}^2$	$\hat{\Delta}$	cpu	eff.	$\hat{\mu}$	$\hat{\sigma}^2$	$\hat{\Delta}$	cpu	eff.
128	2.37E-5	1.1E-10	112%	7036	2.2E-3	4.13E-5	5.3E-12	14%	5779	5.6E-2
256	0			7024		6.04E-7	7.2E-16	11%	7316	6.9E-2
512	0			7059		3.40E-10	2.9E-22	13%	10246	4.0E-2
768	0			7037		2.51E-13	1.7E-28	13%	12930	2.9E-2
1024	0			7027		2.08E-16	1.4E-34	14%	15649	2.0E-2

Table 6: CLR Estimation at Level 3 for Different Numbers of Directed Sources

	no-IS					IS				
sources	$\hat{\mu}$	$\hat{\sigma}^2$	$\hat{\Delta}$	cpu	eff.	$\hat{\mu}$	$\hat{\sigma}^2$	$\hat{\Delta}$	cpu	eff.
4	0			2263		1.93E-8	1.2E-18	15%	2899	1.1E-1
5	0			2357		1.43E-7	4.5E-17	12%	2578	1.8E-1
6	0			2447		6.04E-7	7.2E-16	11%	2413	2.1E-1
7	0			2551		2.09E-6	9.1E-15	12%	2357	2.0E-1
8	1.10E-5	4.9E-11	163%	2651	1.8E-4	4.89E-6	2.2E-14	7.8%	2300	4.7E-1

Table 7: CLR Estimation at Level 4 for Different Burst Sizes

	no-IS					IS				
κ_1	$\hat{\mu}$	$\hat{\sigma}^2$	$\hat{\Delta}$	cpu	eff.	$\hat{\mu}$	$\hat{\sigma}^2$	$\hat{\Delta}$	cpu	eff.
10	0			3562		2.13E-28	5.9E-58	29%	9978	7.7E-3
25	0			3575		1.34E-12	1.5E-25	75%	5347	2.2E-3
50	0			3584		1.37E-7	4.4E-16	39%	3496	1.2E-2
100	2.06E-4	2.1E-8	180%	3599	6.1E-5	6.81E-5	1.1E-10	39%	2505	1.7E-2
150	1.08E-4	8.9E-9	223%	3589	6.7E-3	4.63E-4	7.8E-9	49%	2194	1.3E-2

Table 8: CLR Estimation at Level 4 for Different Buffer Sizes

	no-IS					IS				
B_4	$\hat{\mu}$	$\hat{\sigma}^2$	$\hat{\Delta}$	cpu	eff.	$\hat{\mu}$	$\hat{\sigma}^2$	$\hat{\Delta}$	cpu	eff.
128	1.59E-3	7.5E-8	44%	3580	4.3E-3	1.08E-3	4.0E-9	15%	1881	1.5E-1
256	8.27E-6	6.8E-11	255%	3593	1.3E-2	5.54E-5	1.4E-10	55%	2440	8.8E-3
516	0			3586		1.37E-7	4.4E-16	39%	3580	1.2E-2
768	0			3592		3.63E-10	2.7E-21	36%	4488	1.1E-2
1024	0			3595		1.02E-12	1.5E-26	31%	5550	1.2E-2