Proceedings of the 2004 Winter Simulation Conference
R .G. Ingalls, M. D. Rossetti, J. S. Smith, and B. A. Peters, eds.

NEW EVENT-DRIVEN SAMPLING TECHNIQUES FOR NETWORK RELIABILITY ESTIMATION

Abdullah Konak

Information Sciences and Technology
Penn State Berks-Lehigh Valley
Reading, PA 19610, U.S.A.

Alice E. Smith

Department of
Industrial and Systems Engineering
Auburn University, AL 36849, U.S.A.

Sadan Kulturel-Konak

Management Information Systems
Penn State Berks-Lehigh Valley
Reading, PA 19610, U.S.A.

ABSTRACT

Exactly computing network reliability measures is an NP-hard problem. Therefore, Monte Carlo simulation has been frequently used by network designers to obtain accurate estimates. This paper focuses on simulation estimation of network reliability. Using a heap data structure, efficient implementation of a previous approach, dagger sampling, is proposed. Two new techniques, geometric sampling and block sampling, are developed to efficiently sample states of a network. These techniques are event-driven rather than time-driven, and are thus efficient for highly reliable networks. To test relative performance, computational experiments are carried out on various types of networks using the new procedures.

1 INTRODUCTION

In network reliability analysis, a telecommunication network with unreliable components is usually modeled as an undirected network $G(V, E)$ with node set $V=\{v_1,\ldots,v_n\}$ and arc set $E=\{e_1,\ldots,e_m\}$ under the following assumptions:

- Nodes are perfectly reliable; however, arcs randomly fail.
- Each arc $e_i \in E$, independently from each other, can be in either of two states, operative or failed, with respective probabilities p_i and q_i=1-p_i.
- No repair is allowed.

Let $\mathbf{X} = \{x_1, x_2, \ldots, x_m\}$ denote the state vector of $G(V,E)$ such that $x_i = 1$ if arc e_i is in the operative state and $x_i = 0$ if arc e_i is in the failed state. Hence, the probability of observing a particular state $\mathbf{X}$ is given by

$$\Pr\{\mathbf{X}\} = \prod_{e_i \in E} [q_i + x_i(2p_i - 1)]. \quad (1)$$

The main function of a telecommunication network is to provide connectivity service. Let $T \subset V$ be a set of some specified nodes of $G(V, E)$. Network reliability analysis is concerned with the following question: "What is the probability that all nodes in T are connected to each other?" With respect to connectivity, a network can be in either of two states: connected or not connected. Therefore, the structure function is defined as:

$$\Phi(\mathbf{X}) = \begin{cases} 1 & \text{if all nodes in } T \text{ are connected,} \\ 0 & \text{otherwise.} \end{cases}$$

An important problem in network reliability analysis is to calculate the expected value of the structure function $\Phi(\mathbf{X})$, i.e.,

$$R = E[\Phi(\mathbf{X})] = \sum_{\mathbf{X} \in S} \Phi(\mathbf{X}) \Pr\{\mathbf{X}\} \quad (2)$$

where S is the state space of the all possible network states. If T=V, then Equation (2) refers to *all-terminal* reliability and the exact calculation of R for any T is known to be NP-hard in general networks (Ball 1980).

In general, exact calculation of Equation (2) is extremely difficult due to the enormous number of possible states (a network with m arcs has 2^m possible states). Therefore, Monte Carlo (MC) simulation has been a common alternative to estimate Equation (2). In the naive implementation of MC simulation, K state vectors are sampled from S with the probability distribution given in Equation (1), then the estimator for Equation (2) is calculated as

$$\hat{E}[\Phi(\mathbf{X})] = \frac{1}{K}\sum_{k=1}^{K}\Phi(\mathbf{X}^{(k)})$$

where $\mathbf{X}^{(k)}$ is the state of the network in the k^{th} sample.

The naive implementation of MC simulation has been criticized for being inefficient, especially when individual arc reliabilities are high; therefore, several alternatives have been proposed. Alternative approaches can be considered in three main groups. The first group is called state based techniques. They aim to generate state vector more efficiently (e.g., Markov Model of Mazumdar, Coit, and McBride (1999)) or to introduce negative correlation between sampled state vectors (e.g., Dagger sampling by Kumamoto, Tanaka, Inoue, and Henley (1980)). The second group uses the sample space of arc permutations instead of the original sample space S (e.g., the Sequential Construction/Destruction of Easton and Wong (1980) and Destruction, Creation, and Merging Processes of Elperin, Gertsbakh, and Lomonosov (1991)). These approaches are particularly efficient if all arcs have identical reliability.

The third group is called bounding techniques, which can be considered as stratified and importance sampling variance reduction techniques (Van Slyke and Frank 1972; Kumamoto, Tanaka, and Inoue 1977; Fishman 1986). This group requires prior information such as the path and cut sets of the network studied. Bounding techniques cannot directly be applied in the case of a non-binary structure function since they are based on cut and path sets assuming a binary structure function.

In this paper, we introduce event-driven approaches for efficiently generating network states as alternatives to the usual time-driven implementations of state based simulation techniques. Event-driven implementation of dagger sampling (DS) is devised using a heap data structure and two new techniques, block sampling (BS) and geometric sampling (GS), are proposed to efficiently generate state vectors. Since these are based on directly generating network states, both new techniques are easy to implement and can be used for very general networks (e.g., for binary or continuous $\Phi(\mathbf{X})$, or for distinct failure reliabilities) without requiring prior information.

A thorough computational study is conducted to compare the performance of these simulation techniques over various types of networks in terms of density (ratio of m to n), size, and individual arc reliabilities.

2 EXISTING STATE BASED APPROACHES AND PROPOSED EVENT-DRIVEN IMPLEMENTATIONS

The most basic simulation technique to generate state vectors is Crude Monte Carlo Sampling (CMCS). In CMCS, a state vector $\mathbf{X}$ is sampled by individually simulating the state of each arc $e_i \in E$ using a Bernoulli random variable (r.v.) with mean p_i. To estimate reliability, K state vectors are sampled as defined above and connectivity is checked for each state sampled. In turn, the ratio of the number of connected network states to sample size K yields an unbiased estimator of reliability.

PROCEDURE CMCS
1 up=0
2 **FOR** k=1,…,K **DO** {
3 **FOR** each $e_i \in E$ **DO** {
4 **IF** Uni[0,1] ≤ p_i **THEN** x_i=1 **ELSE** x_i=0 }
5 $\mathbf{X}$={x_i: $e_i \in E$}
6 up=up+$\Phi(\mathbf{X})$
7 }

The estimator of R in CMCS is given by

$$\hat{R} = \frac{1}{K}\sum_{k=1}^{K}\Phi(\mathbf{X}^{(k)}) = \frac{up}{K}.$$

Since the outcome of $\Phi(\cdot)$ is a Bernoulli trial with the expected value of R, the variance of estimator $\hat{R}$ is given by

$$\mathrm{Var}(\hat{R}) = \frac{1}{K^2}\sum_{k=1}^{K}\mathrm{Var}(\Phi(\mathbf{X}^{(k)})) = \frac{R(1-R)}{K} \qquad (3)$$

and the coefficient of variation (cv) on 1-R is equal to

$$cv = \frac{\sqrt{\mathrm{Var}(\hat{R})}}{1-R} = \sqrt{\frac{R}{(1-R)K}}.$$

Estimating reliability for highly reliable networks is daunting. CMCS is inherently inefficient, requiring a total of $m \times K$ random numbers, most of which are used to sample operational states of arcs. For example, consider a highly reliable arc (e.g., p_i = 0.999). In this case, K random numbers are used to simulate a rare event (x_i=0), which will be observed only once in 1,000 samples, on average.

2.1 Event-Driven Simulation for Network Reliability Estimation

CMCS is a *discrete time-driven* approach in which the time is advanced in increments of time units (replications) and events (arc failures) for each unit time increment are simulated. On the contrary, in the *event-driven* approach, time points at which future events occur are generated, and then the simulation time is advanced to the time point of the most imminent of events and the state of the system is evaluated. An event-driven simulation progresses by simulating the next event time for the most imminent of the event(s) and updating a timetable accordingly. In some cases, the event-driven approach can be more efficient than the time-driven approach to estimate reliability and performability.

In this section we develop the event-driven version of DS. In the event-driven simulation approach to network reliability, events are arc failures or repairs. Implementation of this approach requires: (*i*) an *event list* containing the next time when each arc failure will occur, (*ii*) a *timing routine* to determine the time of the most imminent of arc failures, and (*iii*) an *event routine* updating the network state and generating the next failure times whenever arc failures occur. We use a heap data structure for the implementation of all methods. A heap is a priority queue allowing efficient removal and insertion of elements to and from a collection of elements stored in a binary tree structure and organized according to their *keys*, which identify weights or priorities of elements. Here, arcs correspond to heap elements and the key of an arc e_i is the next failure time (event time $et(i)$) of the arc. The following heap operators are used:

- insertElement(PQ, e_i, $et(i)$): insert arc e_i with key $et(i)$ into heap PQ.
- removeMin(PQ): return and remove the element with the smallest key from PQ.
- minElement(PQ): return (without removing) the smallest key of heap PQ.
- initializeHeap(PQ): initialize an empty heap PQ.

O(m) memory space is required to store the heap data structure. The heap operators, removeMin(), insertElement(), and minKey(), have time complexity of O(log(m)), O(log(m)), and O(1), respectively.

2.2 Event-Driven Approach for Dagger Sampling

Kumamoto, Tanaka, Inoue, and Henley (1980) proposed DS to improve upon the sampling inefficiency of CMCS. In DS, only a single random number is used to sample L_i consecutive states of arc e_i. Here, L_i is called a sub-block of arc e_i and equals $\lfloor 1/q_i \rfloor$. Using a single random number U, a sub-block of L_i consecutive states of arc e_i is generated as follows. If $U > q_i L_i$, then x_i=1 for all samples within the sub-block; otherwise, x_i=0 in the $(\lfloor U/q_i \rfloor+1)^{th}$ sample, and x_i=1 in all other samples. DS is superior to CMCS in terms of not only using fewer random numbers but also having a smaller sampling variance. Using a single random number to generate a sub-block of samples induces a negative correlation between the state vectors of a sub-block. Therefore, DS can be considered a multi-dimensional *antithetic variable technique* for variance reduction. The major drawback of DS appears when individual component reliabilities are very high, but relatively different. In this case, each arc e_i will have a very different sub-block size L_i and assembling these sub-blocks to obtain state vectors is difficult and memory intensive. For example, the time-driven DS procedure given by Fishman (1986) has a memory requirement of O(Lm), where L is the least common multiple of the L_i's and can be quite large when arc reliabilities are not identical. To overcome this drawback, we present an efficient event-driven implementation of DS requiring only O(m) memory space. The following procedure is the event-driven implementation of DS using a heap data structure.

PROCEDURE DS

L_i sub-block size of arc e_i
NB_i number of sub-blocks to be simulated for arc e_i
$Bt(i)$ number of sub-blocks simulated for arc e_i

```
1  initializeHeap(PQ)
2  set Li = ⌊1/qi⌋ for each ei ∈ E
3  L = the least common multiples of Li's
4  NBi = L/Li for each ei ∈ E
5  up=0, np=0;
6  FOR each ei ∈ E DO {
7    Bt(i)=1, failed=no
8    DO {
9      U=Uni[0,1]
10     IF (U<qi×NBi) THEN
           {et(i)=(Bt(i)-1)×Li+⌊U/qi⌋+1, failed=yes}
11     Bt(i)=Bt(i)+1
12   }WHILE(failed=no & Bt(i) <NBi)
13   IF (failed=yes) THEN insertElement(PQ, ei, et(i))
14 }
15 DO {
16   etmin=MinElement(PQ)
17   set xi=1 for each ei ∈ E
18   DO {
19     ei=removeMin(PQ), xi=0, and failed=no
20     DO {
21       U=Uni[0,1]
22       IF (U < qi×NBi ) THEN
         {et(i)=(Bt(i)-1)×Li+⌊U/qi⌋+1,  failed=yes}
23       Bt(i)=Bt(i)+1
24     } WHILE(failed=no &Bt(i)<NBi)
25     IF (failed=yes) THEN insertElement(PQ, ei, et(i))
26   } WHILE (etmin=MinElement(PQ))
27 up=up+ Φ(X)
28 } WHILE (PQ is not empty)
29 R=up/L
```

The procedure DS has two parts. In the first part (between lines 6 and 14), the initial failure times of arcs are generated and inserted into heap *PQ*. In the second part (between lines 15 and 28), arcs with the most imminent failure times are removed from the top of the heap, the network state is updated, the next failure times are generated, and the removed arcs are inserted into the heap again. In DS, arcs are allowed to fail at most once within a sub-block size of L_i replications. This is controlled by variable $Bt(i)$, which counts the number of sub-blocks that arc e_i has been simulated. A single random number U is used to determine the failure time of arc e_i within the $Bt(i)^{\text{th}}$ sub-block as follows (see lines 10 and 22). If $U < q_i \times NB_i$, then the failure time of arc e_i is calculated as $et(i)=(Bt(i)-1)\times L_i+\lfloor U/q_i \rfloor+1$, and this event is inserted into heap *PQ*. Otherwise, no failure occurs within the sub-block. In both cases, the simulation time moves to the next sub-block by simply increasing $Bt(i)$ by one. These steps are either repeated until a failure is observed or the maximum number of sub-blocks is reached (**DO-WHILE** loop between lines 20 and 24). As a result, only arc failures are stored in the heap by skipping sub-blocks without a failure. The proposed procedure is an efficient event-driven approach to DS with a memory requirement of only $O(m)$, which is much less than $O(Lm)$ of the procedure given in Fishman (1986) and Kumamoto, Tanaka, Inoue, and Henley (1980).

3 NEW EVENT-DRIVEN SAMPLING TECHNIQUES FOR NETWORK RELIABILITY

In this section, we devise two new methods to estimate network reliability. These are event-driven approaches and called BS and GS. The BS approach is an efficient alteration of the event-driven version of DS.

3.1 Block Sampling

Using a heap data structure overcomes the large computer memory requirement of DS. However, the inability to control L due to different sizes of sub-blocks is another shortcoming of DS. In this section, we develop a new sampling method to remedy this problem by using sub-blocks of equal sizes. Consider a block of L consecutive state vectors, $\mathbf{X}^{(1)},\ldots,\mathbf{X}^{(L)}$, which are generated by using CMCS. Let Y_i denote the number of times that arc e_i is in the failed state in L consecutive replications. It is clear that Y_i follows a binomial distribution with the following probability mass function (p.m.f.):

$$\Pr\{Y_i=y\} = \binom{L}{y} q_i^{\,y} p_i^{\,L-y} \qquad y = 0,\ldots,L.$$

The main idea behind BS is to construct state vectors directly from r.v. Y_i instead of individually sampling its elements. In BS, failures of arc e_i within a block of L replications are generated in two steps: (*i*) sample Y_i from the binomial distribution with parameters L and q_i; (*ii*) randomly distribute Y_i failures over L replications to determine the replications with $x_i=0$. Once individual arc failures are generated, the state vector of the network can be constructed. The procedure is given as follows:

PROCEDURE BS

```
1   for each e_i ∈ E do {
2     Generate Y_i from Binomial(L, q_i)
3     FOR l =1,…,Y_i DO {
4       Choose integer k from [1, L] without replacement
5       insertElement(PQ, e_i, k)
6       }
7     }
8   DO {
9     set x_i = 1 for each e_i ∈ E, up=0, and np=0
10    et_min=MinElement(PQ)
11    DO {
12      e_i=removeMin(PQ)
13      x_i=0
14    } WHILE (et_min=MinElement(PQ))
15    up=up+Φ(X)
16  } WHILE (PQ is not empty)
17  R=up/L
```

The procedures DS and BS are given for a block of simulation, if B independent replications of blocks are performed, the estimation of network reliability is given as

$$\hat{R} = \frac{1}{B}\sum_{i=1}^{B} \hat{R}_i.$$

In DS and BS, state vectors within the same block are not *s*-independent since the same random numbers are used to generate them. Therefore, Equation (3) cannot be used to obtain an unbiased estimate of $Var(\hat{R})$. To obtain an unbiased esti.mate of the variance, estimates from B independent replications of blocks can be used as follows:

$$Var(\hat{R}) = \frac{1}{B(B-1)}\sum_{i=1}^{B} (\hat{R}-\hat{R}_i)^2.$$

In BS, since state vectors within the same block are not *s*-independent, it is necessary to show that BS is unbiased (i.e., a state $\mathbf{X}$ is sampled with the probability of $\Pr\{\mathbf{X}\}$). Although the state vectors within the same block are not *s*-independent, the individual elements of a state vector $\mathbf{X}$ are *s*-independent. Therefore, if $\Pr\{x_i = 0\}$ is equal to q_i in BS, then a state vector $\mathbf{X}$ can be sampled with the probability given in Equation (1). In the following, it is shown that the BS procedure (i.e., first sampling the number of failures, then distributing them randomly over repli-

cations) produces unbiased samples from S. $\Pr\{x_i = 0\}$ can be calculated by conditioning on Y_i as follows:

$$\Pr\{x_i = 0\} = \sum_{y=0}^{L} \Pr\{x_i = 0 \mid Y_i = y\} \cdot \Pr\{Y_i = y\}.$$

Since failures are randomly distributed over L samples, for a given y, $\Pr\{x_i = 0 \mid Y_i = y\}$ is equal to y/L. Thus,

$$\begin{aligned} \Pr\{x_i = 0\} &= \sum_{y=0}^{L} \left(\frac{y}{L} \binom{L}{y} q_i^{\,y} p_i^{\,L-y} \right) \\ &= q_i \sum_{y=1}^{L} \left(\binom{L-1}{y-1} q_i^{\,y-1} p_i^{\,L-y} \right). \end{aligned} \tag{4}$$

The summation term in Equation (4) is equal to 1 since it is the sum of the p.m.f of the binomial distribution with parameters L-1 and q_i for all possible outcomes. Hence,

$$\Pr\{x_i = 0\} = q_i.$$

The result above shows that BS produces unbiased samples. Next, the variance of estimator $\hat{R}$ is analyzed for BS. The variance of estimator $\hat{R}$ within a block size of L is given by

$$\mathrm{Var}(\hat{R}) = \frac{1}{L^2} \left[\begin{array}{l} \sum_{i=1}^{L} \mathrm{Var}(\Phi(\mathbf{X}^{(i)})) \\ +2 \sum_{i=1}^{L} \sum_{j=i+1}^{L} \mathrm{Cov}(\Phi(\mathbf{X}^{(i)}), \Phi(\mathbf{X}^{(j)})) \end{array} \right]. \tag{5}$$

Calculating $\mathrm{Cov}(\Phi(\mathbf{X}^{(i)}), \Phi(\mathbf{X}^{(j)}))$ is intractable since $\Phi(\cdot)$ is a very complex function. Nonetheless, the covariance between the corresponding elements of two state vectors can be calculated, and then used to make an assessment about the covariance between state vectors. For a given Y_i, the covariance between $x_i^{(j)}$ and $x_i^{(k)}$ of state vectors $\mathbf{X}^{(j)}$ and $\mathbf{X}^{(k)}$ in the same block, $j \neq k$, is

$$\begin{aligned} \mathrm{Cov}(x_i^{(j)}, x_i^{(k)} \mid Y_i = y) &= \mathrm{E}\{x_i^{(j)} x_i^{(k)} \mid Y_i = y\} \\ &\quad - \mathrm{E}\{x_i^{(j)} \mid Y_i = y\}\, \mathrm{E}\{x_i^{(k)} \mid Y_i = y\} \\ &= \frac{(L-y)(L-y-1)}{L(L-1)} - \left(\frac{(L-y)}{L} \right)^2 \\ &= \frac{-y(L-y)}{L^2(L-1)}. \end{aligned}$$

Therefore,

$$\begin{aligned} \mathrm{Cov}(x_i^{(j)}, x_i^{(k)}) &= \sum_{y=0}^{L} \mathrm{Cov}(x_i^{(j)}, x_i^{(k)} \mid Y_i = y) \Pr\{Y_i = y\} \\ &= \frac{q_i(q_i - 1)}{L}. \end{aligned} \tag{6}$$

If $\Phi(\cdot)$ is a coherent structure function and, if for each $x_i^{(j)} \in \mathbf{X}^{(j)}$ and $x_i^{(k)} \in \mathbf{X}^{(k)}$, $\mathrm{Cov}(x_i^{(j)}, x_i^{(k)}) < 0$, then $\mathrm{Cov}\{\Phi(\mathbf{X}^{(i)}), \Phi(\mathbf{X}^{(j)})\} < 0$ (Kumamoto, Tanaka, and Inoue 1980). Therefore, based on Equation (6), BS has less sampling variance than CMCS. Notice that the first part of Equation (5) is the variance of the CMCS estimator for K=L and the second part (the summation of covariance terms) is a negative number due to the fact that $\Phi(\cdot)$ is a coherent structure function and the corresponding elements of state vectors with the same block are negatively correlated.

In addition to a smaller sampling variance compared to CMCS, one obvious advantage of BS is that it uses fewer random numbers for the same number of replications since it generates only failures. To generate L states of arc e_i, BS requires one binomial r.v. for Y_i and on average $L \times q_i$ random numbers to distribute Y_i failures over L replications. (See page 562 of Ross (1997) for how to choose k numbers out of n numbers ($k \leq n$) without replacement using only k random numbers.) On the other hand, CMCS uses L Bernoulli r.v.s for the same sample size. As q_i approaches 0, BS requires fewer and fewer random numbers, making BS computationally less expensive for networks with highly reliable arcs. However, as arc reliabilities approach 1, the negative correlation between the state vectors diminishes. Compared to DS where sub-block size L_i depends on $1/q_i$, BS uses a single block size L for each arc. Therefore, BS can be used without difficulty or alteration for networks with different component failure probabilities.

3.2 Geometric Sampling

Consider K replications of CMCS where x_i (the state of arc e_i) is sampled in K discrete time points. Let Y_i be a r.v. corresponding to the number of states of arc e_i sampled until the first failure is observed. Since x_i is a Bernoulli r.v. with mean p_i, Y_i is a geometric r.v. with the following p.m.f.:

$$\Pr\{Y_i = y\} = \begin{cases} (1-p_i)\, p_i^{\,y-1} & y = 1, 2, \ldots \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

This observation is key to our new technique called GS. The GS procedure is an event-driven simulation where events are arc failures randomly occurring at dis-

crete time intervals given by Equation (7). In order to implement GS, a heap data structure is used as follows.

PROCEDURE GS:

```
1   initializeHeap(PQ)
2   FOR each arc e_i ∈ E DO {
3       a_i = 1/ln(p_i)
4       et(i) = ⌊a_i ln(U)⌋+1
5       insertElement(PQ, e_i, et(i))
6   }
7   up=0, np=0;
8   FOR k=1,…,K DO {
9       set x_i = 1 for each e_i ∈ E
10      et_min = minElement(PQ)
11          DO {
12              e_i=removeMin(PQ)
13              set x_i = 0
14              et(i) = et(i)+⌊a_i ln(U)⌋+1
15              insertElement(PQ, e_i, et(i))
16          } WHILE (et_min=minElement(PQ))
17      up=up+Φ(X)
    }
```

In the procedure above, geometric r.v.s are generated by the inverse-transformation method from the cumulative mass function of the geometric distribution (lines 4 and 14). It can be shown that $\lfloor \ln(U)/\ln(p_i) \rfloor$ is a geometric r.v. with mean $1/q_i$ (Law and Kelton 2000).

In each replication within the **DO-WHILE** loop between lines 11 and 16, arcs with the same most imminent failure time ($et_{\min}$) are removed from the top of the heap, the state of the network is updated by setting their state to failed, the next failure times are generated, and these arcs are inserted into the heap again. Note that the procedure generates network states with at least one failed arc. For state $\mathbf{X}_o$ with all arcs operational, $\Phi(\mathbf{X}_o)=1$ and $\Omega(\mathbf{X}_0)=1$. Hence, estimation of R is given as:

$$\hat{R} = \Pr\{\mathbf{X}_0\} + \frac{up}{K}(1-\Pr\{\mathbf{X}_0\}).$$

The correctness of the GS procedure is based on the memoryless property of the geometric distribution which implies $\mathrm{Cov}(x_i^{(k)}, x_i^{(l)}) = 0$ for any two distinct k and l replications for each $e_i \in E$; hence, $\mathrm{Cov}(\Phi(\mathbf{X}^{(k)}), \Phi(\mathbf{X}^{(l)})) = 0$. Therefore, GS does not provide variance reduction compared to CMCS, however, its efficiency stems from the fact that random numbers are only generated to simulate arc failures, and state $\mathbf{X}_o$ is skipped without being actually generated. For the same number of replications, GS implicitly considers $1/(1-\Pr\{\mathbf{X}_o\})$ times more states than CMCS. Therefore, as $\Pr\{\mathbf{X}_o\}$ approaches one, the efficiency of GS significantly increases. That is, for highly reliable networks, GS becomes comparatively more efficient than CMCS.

4 COMPUTATIONAL EXPERIMENTS AND DISCUSSIONS

In this section, the performances of the new techniques that we have developed in this paper are tested for all-terminal reliability. Since Markov Model (MV) (Mazumdar, Coit, and McBride 2000) is also an alternative event-driven approach, it was also coded and included in comparisons. Two performance criteria are most often used in the literature: the variance of the estimator for a given number of replications and the CPU time required. If V_M and T_M denote the variance achieved and the time taken by using technique M, the relative efficiency is given by $\Delta V_M = (V_{CMC}/V_M) \times (T_{CMC}/T_M)$. The higher ΔV_M is, the better technique M is. The CPU time requirement of CMCS depends on the data structure used to represent the network within a computation (mainly due to memory requirements and updating procedures) and the connectivity check algorithm. We use an arc list representation and the graph merging connectivity algorithm, which favors this representation, for CMCS. In addition, if no arc failure occurs, the connectivity check is skipped.

To investigate the performance of the techniques with respect to network density (m/n), two types of grid networks with 16 (4×4) and 64 (8×8), nodes were considered, as shown in Figure 1. To test the performance of the techniques with respect to arc reliabilities, two different sets of arc reliabilities were used for (p_h, p_v, p_s, p_w): *L*-type with (0.97, 0.99, 0.995, 0.90) and *H*-type with (0.997, 0.995, 0.993, 0.991). In addition, four network density types, (a)-type being the least dense and (d)-type being the most dense, were tested. Density (a)-type networks have $(\sqrt{n}-1)(\sqrt{n}+2)$ arcs and (b), (c), and (d)-type networks are obtained by adding new arcs to the preceding type networks as shown in Figure 1. To identify the networks in Tables 2, the number of nodes, density, and reliability type are used. For example, network 16*aL* corresponds to the 16-node, 4×4 grid network, with density (a)-type and arc reliability *L*-type. In addition to these networks, the dodecahedron network (Fishman 1986) with 20 nodes and 30 arcs were used. For all networks, $K=10^7$ replications were used for CMCS, DS, BS, and GS unless otherwise is expressed. Since MV uses a continuous simulation time, the CPU time taken by CMCS to perform the specified number of replications for a given network was used as the termination criteria for this technique.

Table 1 summarizes the simulation results for the dodecahedron network with different levels of arc reliabilities. DS and BS did not provide a significant level of variance reduction for the dodecahedron network. However, the relative efficiency of both methods increased with increased arc reliability. This result can be explained with the increased sampling efficiency of these methods in case

of highly reliable arcs. GS did not provide variance reduction for p=0.80 and p=0.90, however, variance reduction

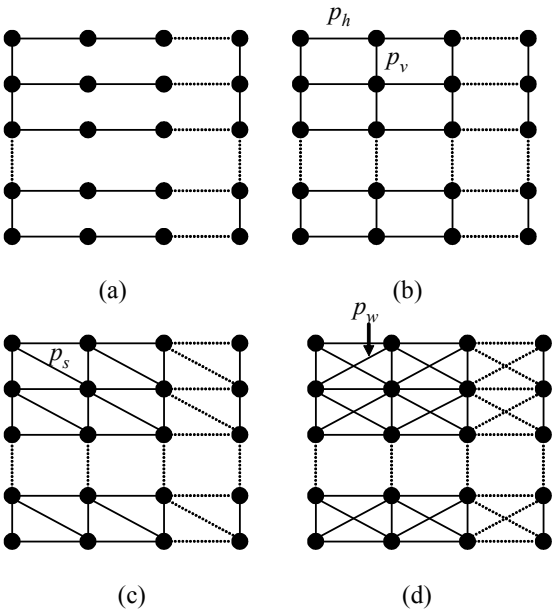

Figure 1: Grid Networks Used in Experiments

was achieved for higher arc reliabilities (3.8 times for p=0.99). It should be noted that the variance reduction obtained by GS is due to generating only network states with at least one arc failed. For example, for p=0.99, the probability that all arcs are in the operative state is 0.739, which means that on the average, 74% of the time CMCS samples the network state with all arcs operative. In the process of event-driven simulation, GS skips this state and samples 3.8 (≈1/(1-0.739)) times more network states with arc failures than CMCS, which in turn provides the variance reduction. MV performed poorly for the dodecahedron network with p=0.80, p=0.90, and p=0.95. The performance of MV improved with increased arc and network reliability.

Table 1: All-Terminal Reliability Simulation Results for the Dodecahedron Network (Fishman, 1986)

p	R*	Metric	DS	BS	GS	MV
0.80	0.8108435855	V/V_M	1.2	1.0	1.0	0.2
		ΔV_M	1.1	0.9	0.8	0.2
0.90	0.9771308359	V/V_M	1.0	1.0	1.0	0.4
		ΔV_M	1.1	1.1	0.9	0.5
0.95	0.9973118634	V/V_M	1.0	1.0	1.2	0.8
		ΔV_M	1.4	1.4	1.1	0.8
0.98	0.9998351727	V/V_M	1.0	1.0	2.2	1.7
		ΔV_M	2.0	2.0	1.6	1.7
0.99	0.9999796990	V/V_M	1.0	1.0	3.8	3.0
		ΔV_M	3.1	3.2	2.4	3.1

Table 2 summarizes the simulation results for all-terminal reliability for the grid networks with different size, density and arc reliability. DS, BS, and GS perform best for sparse networks with high arc reliabilities, and they perform relatively poorly for the 64-node dense networks. Being event-driven approaches, their performance mainly depends on the probability that all arcs are operative. This probability diminishes as networks get larger and denser. The same is true for MV.

Table 2: All-Terminal Reliability Simulation Results for the 16-Node and 64-Node Grid Networks

Network	R	Metric	DS	BS	GS	MV
16*aL*	0.985430398	V/V_M	1.1	1.0	3.0	1.1
		ΔV_M	2.7	2.4	2.0	1.1
16*bL*	0.998697153	V/V_M	1.0	1.0	2.7	1.8
		ΔV_M	2.5	2.5	2.0	1.8
16cL	0.999411033	V/V_M	1.1	2.2	5.3	3.2
		ΔV_M	2.8	4.8	3.7	2.6
16*dL*	0.999941743	V/V_M	1.0	1.1	1.3	1.8
		ΔV_M	1.8	1.9	1.4	1.8
16*aH*	0.999635710	V/V_M	1.0	1.0	15.6	7.0
		ΔV_M	9.5	9.1	7.5	5.8
16*bH*	0.999938880	V/V_M	1.1	1.0	11.2	5.7
		ΔV_M	8.8	7.5	6.3	5.6
16c*H*	0.999969813	V/V_M	1.0	1.0	6.7	4.6
		ΔV_M	5.9	5.8	4.6	4.6
16*dH*	0.999999509	V/V_M	1.1	0.9	4.3	5.1
		ΔV_M	5.2	4.5	3.6	5.1
64*aL*	0.862765235	V/V_M	1.3	1.0	1.2	0.4
		ΔV_M	1.4	1.0	1.1	0.4
64*bL*	0.998606424	V/V_M	1.1	1.0	1.1	0.6
		ΔV_M	1.1	1.1	1.1	0.6
64*cL*	0.999397343	V/V_M	1.0	1.0	1.1	0.5
		ΔV_M	1.1	1.1	1.1	0.5
64*dL*	0.999984011	V/V_M	1.0	1.2	1.3	0.9
		ΔV_M	1.2	1.5	1.4	0.9
64*aH*	0.997897200	V/V_M	1.0	1.0	4.8	1.5
		ΔV_M	1.7	1.7	1.6	1.4
64*bH*	0.999938151	V/V_M	1.1	0.9	2.7	1.4
		ΔV_M	1.6	1.4	1.4	1.4

DS = Dagger Sampling (new implementation from this paper of Kumamoto, Tanaka, Inoue, and Henley (1980)); BS = Block Sampling (this paper); GS = Geometric Sampling (this paper); MV = Markov Model (new implementation from this paper of Mazumdar, Coit, and McBride (2000)); *Exactly calculated by using a factoring algorithm and network reductions similar to given by Page and Perry (1988).

5 CONCLUSIONS

This paper presented event-driven simulation approaches to network reliability analysis as an alternative to time-driven simulation. Using the event-driven approach and a

heap-data structure, several improvements were proposed for dagger sampling. In addition, two new sampling techniques were proposed, geometric sampling and block sampling. These new event-based simulations provided variance reduction with minimum overhead; however, they are most effective for highly reliable sparse networks. These techniques can be used for very general structure functions and cases where other advance variance reduction techniques are not applicable.

REFERENCES

Ball, M. O. 1980. Complexity of Network Reliability Calculation. *Networks* 10**:** 153-165.

Colbourn, C. J. 1987. *The Combinatorics of Network Reliability*. New York: Oxford University Press.

Easton, M. C., Wong, C. K. 1980. Sequential Destruction Method for Monte Carlo Evaluation of System Reliability. *IEEE Transactions on Reliability* R-29: 27-32.

Elperin, T., Gertsbakh, I., Lomonosov, M. 1991. Estimation of Network Reliability Using Graph Evolution Models. *IEEE Transactions on Reliability* 40**:** 572-581.

Fishman, G. S. 1986. A Comparison of Four Monte Carlo Methods for Estimating the Probability of s-t Connectedness. *IEEE Transactions on Reliability* R-35: 145-154.

Kumamoto, H., Tanaka, K., Inoue, K. 1977. Efficient Evaluation of System Reliability by Monte Carlo Method. *IEEE Transactions on Reliability* R-26: 311-315.

Kumamoto, H., Tanaka, K., Inoue, K., Henley, E. J. 1980. Dagger-Sampling Monte Carlo for System Unavailability Evaluation. *IEEE Transactions on Reliability* R-29: 122-125.

Law, M. A., Kelton, W. D. 2000. *Simulation Modeling and Analysis*. 3rd ed. Boston: McGraw-Hill.

Mazumdar M., Coit, D. W., McBride, K. 2000. A Highly Efficient Monte Carlo Method for Assessment of System Reliability Based on a Markov Model. *American Journal of Mathematical and Management Sciences* 19**:** 115-133.

Page, L. B., Perry, J. E. 1988. A Practical Implementation of the Factoring Theorem for Network Reliability. *IEEE Transactions on Reliability* 37**:** 259-267.

Ross, S. M. 1997. *Introduction to Probability Models*. 6th ed. San Diego: Academic Press.

Van Slyke, R. V., Frank, H. 1972. Network Reliability Analysis: Part I. *Networks* 1: 279-290.

AUTHOR BIOGRAPHIES

ABDULLAH KONAK, Ph.D. is an Assistant Professor of Information Sciences and Technology at Penn State Berks-Lehigh Valley College. He received his B.S. degree in Industrial Engineering from Yildiz Technical University, Istanbul, Turkey, M.S. in Industrial Engineering from Bradley University, and Ph.D. in Industrial Engineering from University of Pittsburgh. Previous to this position, he was an instructor in the Department of Systems and Industrial Engineering at Auburn University for two years. His current research interest is in the application of Operations Research techniques to complex engineering problems, including such topics as telecommunications network design, network reliability analysis/optimization, facilities design, and data mining. He is a member of IIE and INFORMS. His email address is <konak@psu.edu>.

ALICE E. SMITH, Ph.D., P.E. is Philpott-WestPoint Stevens Professor and Chair of the Industrial and Systems Engineering Department at Auburn University. Previous to this position, she was on the faculty of the Department of Industrial Engineering at the University of Pittsburgh, which she joined in 1991 after ten years of industrial experience with Southwestern Bell Corporation. Dr. Smith has degrees in engineering and business from Rice University, Saint Louis University and University of Missouri - Rolla. Her research in analysis, modeling and optimization of manufacturing processes and engineering design has been funded by NASA, the National Institute of Standards (NIST), Lockheed Martin, Adtranz (now Bombardier Transportation), the Ben Franklin Technology Center of Western Pennsylvania and the National Science Foundation (NSF), from which she was awarded a CAREER grant in 1995 and an ADVANCE Leadership grant in 2001. Dr. Smith holds editorial positions on *INFORMS Journal on Computing*, *IEEE Transactions on Evolutionary Computation* and *IIE Transactions*. Dr. Smith is a fellow of IIE, a senior member of IEEE and SWE, a member of Tau Beta Pi, INFORMS and ASEE, and a Registered Professional Engineer in Industrial Engineering in Alabama and Pennsylvania. She serves on the Educational Foundation Board of the Institute of Industrial Engineers. Her email address is <aesmith@eng.auburn.edu>.

SADAN KULTUREL-KONAK, Ph. D. is an Assistant Professor of Management Information Systems at Penn State Berks-Lehigh Valley College. She received her degrees in Industrial Engineering: B.S. from Gazi University, Turkey in 1993, M.S. from the Middle East Technical University, Turkey in 1996 and from the University of Pittsburgh in 1999,and Ph.D. from Auburn University in 2002.Her research interests are in modeling and optimization of complex systems and robustness under uncertainty with applications to facility layout. She is a member of INFORMS, IIE, the Operational Research Society of Turkey, Alpha Pi Mu, and Phi Kappa Phi.. Her email address is <sadan@psu.edu>.