From *Proceedings of the 13th Symposium on Reliable Distributed Systems, Dana Point, CA, October 1994, pp. 64–73*

AN ENVIRONMENT FOR IMPORTANCE SAMPLING BASED ON STOCHASTIC ACTIVITY NETWORKS *

W. Douglas Obal II
Department of Electrical and Computer Engineering
The University of Arizona
Tucson, AZ 85721 USA
obal@ece.arizona.edu

William H. Sanders
Center for Reliable and High-Performance Computing
Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
1308 W. Main St., Urbana, IL 61801 USA
whs@crhc.uiuc.edu

Abstract

Model-based evaluation of reliable distributed and parallel systems is difficult due to the complexity of these systems and the nature of the dependability measures of interest. The complexity creates problems for analytical model solution techniques, and the fact that reliability and availability measures are based on rare events makes traditional simulation methods inefficient. Importance sampling is a well-known technique for improving the efficiency of rare event simulations. However, finding an importance sampling strategy that works well in general is a difficult problem. The best strategy for importance sampling depends on the characteristics of the system and the dependability measure of interest. This fact motivated the development of an environment for importance sampling that would support the wide variety of model characteristics and interesting measures. The environment is based on stochastic activity networks, and importance sampling strategies are specified using the new concept of the importance sampling governor. The governor supports dynamic importance sampling strategies by allowing the stochastic elements of the model to be redefined based on the evolution of the simulation. The utility of the new environment is demonstrated by evaluating the unreliability of a highly dependable fault-tolerant unit used in the well-known MARS architecture. The model is non-Markovian, with Weibull distributed failure times and uniformly distributed repair times.

1 Introduction

Evaluating the reliability, availability, and performance of distributed and parallel systems is a challenging problem due to the complexity of these systems and the nature of the questions that must be answered. The complexity of modern distributed systems is very high, often precluding the straightforward application of analytical modeling techniques. By abstracting complexity deemed unnecessary relative to a modeling study, analytical models may be constructed and solved. But the difficulty in this approach lies in deciding which features of a system may be abstracted without severely biasing the results of the study. To avoid this problem, one may turn to simulation, where the model size is usually not the constraining factor. Therefore, simulation models may be built to more accurately reflect the system under study. But simulation is not without its problems.

The main problem in evaluating reliable distributed systems via simulation is the small probability associated with important events such as system failure. One approach to solving this problem is to use importance sampling to improve the efficiency of simulation by focusing the simulation effort on sample paths that are likely to lead to such events. Importance sampling allows one to change the probability measure governing the evolution of the model so that rare events occur more frequently. To compensate, the observations taken from the altered model are weighted by a factor called the *likelihood ratio*. Informally, the likelihood ratio is the ratio of the probability of obtaining the observation under the original probability measure to the probability of obtaining the observation under the new measure.

Despite the difficulty of finding a technique that works well in general [8, 9, 13, 14, 17, 20], the potential of importance sampling to provide orders of magnitude reductions in the variance of an estimate motivated researchers to look for strategies that would work well within certain classes of models. For example, Lewis and Böhm [20] applied importance sampling to Markovian unreliability models. They developed techniques called "failure biasing" and "forced transition" and obtained significant variance reductions.

Goyal, Shahabuddin, Heidelberger, Nicola, and Glynn (see [8], and the references therein) extended these techniques to adapt to the class of Markovian dependability models representable in the language of the SAVE [1] software package. They developed methods for steady-state dependability measures. Shahabuddin [31, 34] developed the "balanced failure biasing" heuristic for SAVE models and proved that it has the property of bounded relative error, in which the error of the simulation results is independent of the rarity of the event. Furthermore, he used large deviations results [4, 27] to derive "asymptotically optimal exponential changes of measure" for systems that achieve high dependability through massive redundancy. Carrasco [3] has proposed a modification

*This work was supported in part by the Digital Equipment Corporation Faculty Program: Incentives for Excellence, IBM, Motorola Satellite Communications and the University of Arizona Advanced Telecommunications Research Project.

of the basic failure biasing approach that utilizes the "failure distance" concept to estimate dependability measures. Juneja and Shahabuddin [12] proved that straightforward application of failure biasing can lead to estimators with infinite variance in models with delayed group repair, and suggested an effective alternative strategy.

Efficient techniques for evaluating transient dependability measures of Markovian systems have been explored by Carrasco [2] and Shahabuddin [33]. In [2], Carrasco uses estimator decomposition techniques to estimate the Laplace transforms of unreliability and unavailability. Bounded relative error results for transient measures are discussed in [33].

Some work has also been done on importance sampling for more general discrete event systems. Glynn and Iglehart [7] have examined the application of importance sampling to the broad class of discrete event simulations representable by a generalized semi-Markov process (GSMP) [6], but they did not consider any specific changes of measure. Nicola, et al. [22, 23, 24] extended the failure biasing and forcing techniques to systems with more general failure and repair distributions. The systems considered in these papers have the same structure as SAVE type models, only the holding time distributions are altered. Heidelberger et al. [10] discuss bounded relative error results for transient measures of non-Markovian SAVE type systems. Van Moorsel, Haverkort and Niemegeers [35, 36] have proposed an interesting alternative to importance sampling for estimating steady-state measures influenced by rare events.

In this paper, we present work motivated by our desire to experiment with different importance sampling methods over a wide range of models. The paper concentrates on dependability evaluation, but the environment is flexible enough to support many different performance and dependability measures and the corresponding importance sampling strategies. The environment is based on stochastic activity networks (SANs) [21], a stochastic extension to Petri nets.

The first section of the body of this paper presents the "importance sampling governor," a mechanism whereby new probability measures governing the evolution of a SAN are specified as a finite state machine whose states are SAN component definitions and whose transitions depend on the evolution of the SAN. The next section describes how this approach to specifying importance sampling measures was implemented using an event rescheduling algorithm similar to that in [23]. The main point of this section is the derivation of the likelihood ratio for the simulation of a SAN under the influence of an importance sampling governor. Finally, our implementation of the described environment is used to obtain results for the unreliability of the MARS architecture over short missions. The model is non-Markovian, with Weibull failure time distributions and uniform repair time distributions. Using the new environment, we discovered that traditional failure biasing techniques performed poorly in this case. To solve the problem we introduce a natural extension to these techniques and demonstrate its effectiveness for the considered problem.

2 Importance Sampling Environment

The essence of importance sampling is the following transformation. Let Y be a random variable defined on the probability space $(\Omega, \mathcal{F}, P)$. If P' is another probability measure on $(\Omega, \mathcal{F})$, and is absolutely continuous with respect to P (i.e., $P(A) \neq 0$ implies $P'(A) \neq 0$ for all $A \in \mathcal{F}$), then

$$E_P[Y] = \int_\Omega Y\, dP = \int_\Omega Y \frac{dP}{dP'}\, dP' = E_{P'}[YL],$$

where $L = dP/dP'$ is called the likelihood ratio, and the subscript of the expectation operator refers to the probability measure with respect to which the expected value is taken. This transformation suggests that an alternative method of estimating Y under P is to estimate YL under P'.

In SAN simulation, the probability space under consideration is the space of possible sample paths. In this case, P is rarely available in an analytic form. Instead, P is induced by the stochastic components of the model. Therefore, the likelihood ratio is also rarely available in an analytic form.

The approach to importance sampling discussed in this section was designed to expedite the process of inducing P' by altering the stochastic components of the SAN, and to provide a method for calculating the likelihood ratio to compensate for the change of measure.

2.1 SAN-Based Reward Models

The primitive elements used to construct a SAN are *places*, *gates*, and *activities* [21]. Places are as in Petri nets. Gates are used to connect places to activities. Of primary importance to the discussion in this paper is the activity element, since the activity definition contains the probability distributions that determine the probability measure on the sample path space of the SAN.

There are two types of activities: *timed*, and *instantaneous*. The delay represented by a timed activity is described by a continuous probability distribution function called the "activity time distribution function." Given a timed activity a, and a marking μ, the *activity time distribution function of a in μ* is denoted by $F(\cdot, \mu; a)$. The semi-colon notation used here indicates F is parameterized by the activity. *Cases* on activities are used to represent uncertainty about the action taken upon completion of an activity. $C(\cdot, \mu; a)$ is the *case distribution of a in μ*, a discrete probability distribution over the cases.

An activity is *enabled* when all of its input gates hold. A marking in which only timed activities are enabled is called a *stable* marking. If instantaneous activities are enabled, the marking is *unstable*. We represent the set of enabled activities in a stable marking, μ, by $en(\mu)$. When an activity first becomes enabled, it is *activated*, that is, scheduled to *complete*. Activation also occurs when an activity completes, but remains enabled. The marking at activation time is known as the *activation marking*. The time between activation and scheduled completion of an activity, called the *activity time*, is sampled from the activity time distribution, $F(\cdot, \mu; a)$, where μ is the activation marking.

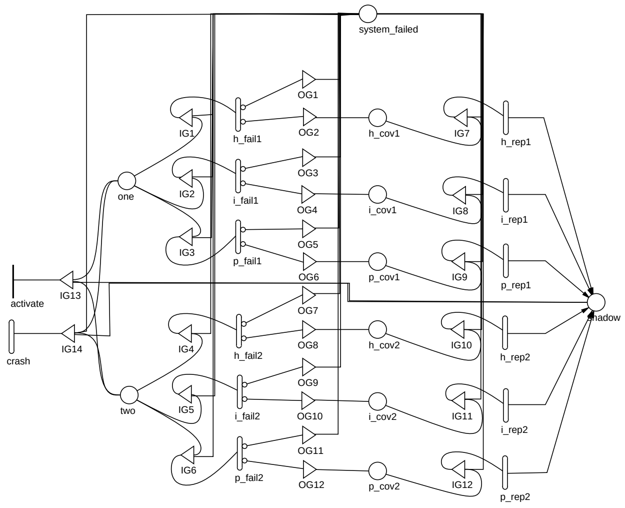

Figure 1: SAN model of a MARS FTU.

The completion of an activity in a SAN is analogous to the firing of a transition in a Petri net. Upon completion of an activity, input gate functions are executed first, followed by output gate functions. These functions operate on the current marking of the SAN to produce the next marking [21]. Activities that are activated are not required to complete. An activity is *aborted* when the SAN moves into a new stable marking in which one or more of the activity's input gates no longer hold. *Reactivation* occurs when the SAN enters a stable marking that is an element of the set of *reactivation markings of a in* μ. When an activity is reactivated, it is aborted and activated. Given a timed activity, a, and reactivation marking of a in μ, μ', the new activity time of a is sampled from $F(\cdot, \mu'; a)$.

As an example, consider a SAN model of the Fault-Tolerant Unit (FTU) used in the MAintainable Real-time System (MARS) [18], shown in Figure 1. This model is based on the definition of the MARS architecture given by Kantz [15], and Kantz and Trivedi [16] in related modeling studies. We introduce the model here to illustrate the use of SAN primitives to model systems. Results for the model are obtained in Section 4.

An FTU consists of two processors, both of which are active, while a third "shadow" processor runs in hot standby mode. The two active processors are modeled by places *one* and *two*; each place initially contains a single token. The failure time distributions of the processors are modeled through the connected activities *h_fail1, i_fail1,* etc. Each processor is susceptible to three different types of failures. For example, failures corrupting the internal state [16] of the first processor are modeled by *h_fail1* and *i_fail1*, while hardware failures are modeled via *p_fail1*. More specifically, the internal state of a processor is decomposed into the history-state and the initialization-state. *h_fail1* models the time between transient failures that corrupt the history-state of the processor, and *i_fail1* models the time between transient failures

Table 1: Activity time distributions in the MARS FTU model.

Activity	*Distribution*	*Parameter values*
h_fail1	Weibull	
	shape	*2.0*
	scale	*4231*
i_fail1	Weibull	
	shape	*2.0*
	scale	*16926*
p_fail1	Weibull	
	shape	*2.0*
	scale	*33851*
h_rep1	uniform	
	lower bound	*0.00433333*
	upper bound	*0.01233333*
i_rep1	uniform	
	lower bound	*0.0433333*
	upper bound	*0.1233333*
p_rep1	uniform	
	lower bound	*2.0 ∗ 24.0*
	upper bound	*12.0 ∗ 24.0*

that corrupt the initialization-state of the processor (stored in ROM). The activity time distributions of these activities are listed in Table 1. The failure activities associated with the second processor are identically distributed to those for the first, so only one set is shown in the table. The means of the distributions are 3750, 15000, and 30000 hours, as given in [15]. (Recall that if α is the shape parameter and m is the desired mean, the scale parameter of the Weibull distribution must be $\alpha m / \Gamma(1/\alpha)$, where $\Gamma(z) = \int_0^\infty t^{z-1} e^{-t}\, dt$ for $z > 0$ [19]. Fortunately, $\Gamma(1/2) = \sqrt{\pi}$ and $\Gamma(z+1)$ is simply $z\,!$ when z is a nonnegative integer.) Since a shadow component runs in a hot standby mode it can immediately take over the job of a failed processor. Thus an instantaneous activity is used to model the shift of the shadow processor into active duty when an active processor fails.

The MARS architecture modeled here uses self-checking hardware, a fault-tolerant operating system, and two-fold execution of tasks to achieve an estimated coverage of 0.9996 [15]. This coverage factor is modeled using cases on the failure activities. For each activity, the top case corresponds to an uncovered failure and the bottom case models a covered failure. The case distributions for each activity are listed in Table 2. When an uncovered failure occurs, the system crashes. The output gates *OG1, OG3, OG5, ...* place the SAN in an absorbing marking indicating the system has crashed.

When a processor failure is successfully covered by the fault-tolerance mechanisms repair begins immediately. Suppose the first processor suffers a hardware failure and there is at least one other working processor. The lower case (case two) of activity *p_fail1*

Table 2: Activity case distributions in the MARS FTU model.

Activity	*Case*	*Probability*
h_fail1	1	*0.0004*
	2	*0.9996*
i_fail1	1	*0.0004*
	2	*0.9996*
p_fail1	1	*0.0004*
	2	*0.9996*

is chosen (i.e., the failure is covered) with probability 0.9996, causing *OG6* to place a token in *p_cov1*. As a result, *p_rep1* is activated and repair has begun. Repairs of processors in each of the three failure modes are modeled by activities *p_rep1, i_rep1,* and *h_rep1*. The activity time distributions for the repair activities in the MARS model are listed in Table 1. The means are 1/120, 1/12, and 168 hours. If a failure occurs that exhausts the redundancy, then activity *crash* moves the model into an absorbing marking.

Upon completing the model of the system, one must specify the performance/dependability measures of interest in terms of the model. In the SAN modeling environment, performance/dependability measures are specified in terms of *reward variables* [30]. A reward variable consists of a reward structure [11] together with a variable type. The reward structure associates *rate rewards* with markings of the SAN, and *impulse rewards* with activity completions.

As an example reward variable specification, we consider the unreliability of a MARS FTU. A reward structure identifying the failed marking is

$$\begin{aligned} \mathcal{C}(a) &= 0, \forall a \in A \\ \mathcal{R}(D) &= \begin{cases} 1 & \text{if } D = \{(\mathit{system\_failed}, 1)\} \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{1}$$

In (1), $\mathcal{C}(\cdot)$ denotes the impulse reward on an activity and $\mathcal{R}(\cdot)$ is the rate reward function as defined in [30]. Using this reward structure, the unreliability during the interval $[0, t]$ is $P\{V_t = 1\}$, where V_t is a variable of type "instant-of-time" [30]. An instant-of-time variable may be used for unreliability in this case because the failed marking is an absorbing marking. It is important to note that although the reward variable used for this example is simple, the reward variable framework is capable of expressing many different and much more complicated measures of performance and dependability; it is a very important element of the environment presented here.

A pairing of a stochastic activity network with a reward structure is called a SAN-based reward model (SBRM). A technique for composing individual SBRMs into a hierarchical *composed SAN-based reward model* exists, and has been used to reduce the size of the state space considered by analytic solution methods [29], and to reduce the future event list management in simulation [28]. The methods discussed here extend to composed SBRMs.

2.2 Importance Sampling Governor

Given a SAN-based reward model, the next step in applying importance sampling is to describe the new probability measure that will govern the evolution of the sample paths of the model. The second component of our environment, the *importance sampling governor*, is a mechanism for describing the new measure in terms of the components of the SAN. The impetus behind this approach is the desire for a new probability measure that is easy to sample. If the effort required to obtain a sample from the new measure is too large, it may nullify the benefits of the variance reduction. By specifying the new measure in terms of the SAN, one can reasonably assume that the sampling effort for the new measure is comparable to that of the original measure.

Informally, the importance sampling governor, or "the governor," is similar to a finite state machine. The states of the governor correspond to biased versions of the original model. That is, each governor state describes modifications to the activity time distributions and case distributions in the original SAN. The modifications are restricted to timed activities. This restriction does not significantly impact the versatility of the governor, since the behavior due to an instantaneous activity can always be duplicated with a gate on a timed activity.

Input to the governor is a sequence of *acm-states*. An acm-state of a SAN is a triple (a, c, μ), where a is the timed activity that completed and c is the case chosen that brought the SAN into marking μ. For a given SAN, say *SAN*, the importance sampling governor is formally defined as a four-tuple, $\mathcal{M} = (\mathcal{Q}, q_0, \mathcal{Z}, \mathcal{T})$, where:

1. $\mathcal{Q}$ is a finite set of state symbols. Each $q \in \mathcal{Q}$ consists of an activity time distribution assignment and a case distribution assignment for *SAN*. In governor state q, a timed activity a in marking μ has case distribution $C(\cdot, \mu; a, q)$, and activity time distribution function $F(\cdot, \mu; a, q)$.

2. $q_0 \in \mathcal{Q}$ is the initial state of $\mathcal{M}$.

3. $\mathcal{Z}$ is a countable set of input symbols that is identical to the set of possible acm-states for *SAN*. In cases where $\mathcal{Z}$ is countably infinite, the state transition function allows only a finite subset of $\mathcal{Z}$ to cause transitions to new states.

4. $\mathcal{T} : \mathcal{Q} \times \mathcal{Z} \to \mathcal{Q}$ is the state transition function. That is, for a given state of $\mathcal{M}$, q, and *SAN* acm-state z, $\mathcal{T}(z; q)$ denotes the next $\mathcal{M}$ state. $\mathcal{T}$ is a partial function, defined only on a finite subset of $\mathcal{Q} \times \mathcal{Z}$. Elements for which $\mathcal{T}$ is undefined result in a self loop.

The formal definition of a governor is important for expressing the likelihood ratio and the nature of sample paths of a governed SAN [25], but the algebraic notation is awkward for describing real governors in terms of actual SANs. In practice, the governor is expressed in terms of a state diagram and two tables.

One table lists the the state definitions, i.e., the activity time distribution assignment and case distribution assignment that should be applied to the SAN when the governor is in each state. States are typically given descriptive names by the user. Another table is used to define the state transition function. For each governor state, this table contains a Boolean function of the SAN marking for each transition out of the governor state. An example governor for the MARS FTU is specified using this representation in Section 4.

A governed SAN evolves in a manner very similar to that of a normal SAN. In a given acm-state, an activity completes, a case is chosen, and the SAN transitions to a new marking with a probability determined, in part, by the case distribution assignment in the current governor state. Before the future events schedule is updated, the governor state transition function is executed (with the new acm-state as input) to determine if the governor changes state. Then, the future events list is updated using the activity time distribution assignment associated with the new governor state. When the governed SAN enters an acm-state that triggers a governor transition to a new governor state, newly activated activities are scheduled according to the activity time distributions specified by the new governor state. In addition, *enabled activities whose activity time distributions are altered by a governor state change are reactivated using the activity time distributions specified by the new governor state.* This execution policy provides flexibility in choosing the importance sampling strategy, at the cost of constraining the activity time distributions in the original SAN to those with closed form cumulative distribution functions.

Specifying importance sampling strategies at the SAN, rather than at the state space, level is convenient. The primary benefit is derived from the expressive power of stochastic activity networks. In particular, one can usually arrive at a compact representation for an importance sampling strategy that spans a large portion of the state space. The failure biasing heuristic [20] increases the total probability of a failure event relative to a repair event, while maintaining the conditional probabilities of each type of failure event, given a failure event occurs. Suppose one wishes to increase the total probability of a failure event to $p(\mu)$, depending on μ, the current marking of the SAN. For Markovian models, one can increase the probability of a failure event by reactivating the failure activities after scaling their rates with respect to the repair activity. The scale factor, denoted $\alpha(\mu)$, will depend on the total probability of a failure transition in μ. In general, if m and n are, respectively, the number of repair activities and the number of failure activities enabled in μ, then

$$\alpha(\mu) = \frac{p(\mu)\sum_{i=1}^{m}\rho(i;\mu)}{(1-p(\mu))\sum_{j=1}^{n}\phi(j;\mu)},$$

where $\rho(i;\mu)$ is the rate associated with the i-th repair activity enabled in marking μ and $\phi(j;\mu)$ is the rate associated with the j-th failure activity enabled in marking μ.

The balanced failure biasing technique [8, 31] is even easier to implement. For balanced failure biasing, the total probability of a failure event in μ is still $p(\mu)$ but the conditional probabilities of the enabled failure activities are made equal. That is, given a failure has occurred, the probability distribution governing which failure activity completed is a discrete uniform distribution. In this case, the rate of each failure activity is given by

$$\phi(\cdot;\mu) = \frac{p(\mu)\sum_{i=1}^{m}\rho(i;\mu)}{n(1-p(\mu))}. \tag{2}$$

In Section 4 we discuss how to apply this idea to non-Markovian SANs.

3 Importance Sampling Using Governed SANs

Let the stochastic process $(X,T) = \{X_n, T_n : n \in \mathbb{N}\}$ denote the sequence of acm-states and transition times, respectively, in the evolution of a SAN. T_n is the time of entry to X_n. Calculation of the likelihood ratio requires an expression for the probability of the event $E_{0,n} = \{X_0 = x_0, T_0 \in dt_0, X_1 = x_1, T_1 \in dt_1, \ldots, X_n = x_n, T_n \in dt_n\}$, the sample path of the SAN through the n-th state transition. The probability of this event is most easily described in terms of the conditional transition probabilities. In [25], we use an approach similar to that of Nicola et al. [23] to derive an iterative formula for the probability of a SAN sample path. Suppose activity a is activated upon entry to the i-th acm-state. Let T_a be the random variable representing the activity time of a. $P\{T_a > t\} = 1 - F(t, \mu_i; a)$. Let $\overline{F}(\cdot, \mu; a) = 1 - F(\cdot, \mu; a)$. Upon the next transition, if a has not completed and is still enabled,

$$\overline{F}_{i+1}(t, \mu_i; a) = \frac{\overline{F}_i(t + T_{i+1} - T_i, \mu_i; a)}{\overline{F}_i(T_{i+1} - T_i, \mu_i; a)}, \tag{3}$$

$$f_{i+1}(t, \mu_i; a) = \frac{f_i(t + T_{i+1} - T_i, \mu_i; a)}{\overline{F}_i(T_{i+1} - T_i, \mu_i; a)}. \tag{4}$$

Equations 3 and 4 show how to update the conditional complementary distribution, and conditional density, respectively, of the activity time, after each transition. Then

$$\begin{aligned} P(E_{0,n}) \;=\; & \prod_{i=0}^{n-1} h(\mu_{i+1}; a_{i+1}, c_{i+1}, \mu_i) \cdot \\ & C(c_{i+1}, \mu_i; a_{i+1}) \cdot \\ & f_i(T_{i+1} - T_i, \mu; a_{i+1})\, dt_{i+1} \cdot \\ & \prod_{a \in en(\mu_i) - \{a_{i+1}\}} \overline{F}_i(T_{i+1} - T_i, \mu; a). \end{aligned} \tag{5}$$

The function h represents the conditional probability that the marking μ_{i+1} is reached, given that activity a_{i+1} completes in marking μ_i and case c_{i+1} is chosen. In many cases h is trivial (zero or one), but it must be included in Equation 5 to account for intervening networks of instantaneous activities and their case distributions. However, in the likelihood ratio this function cancels, because the governor only biases timed activities.

The likelihood ratio is the ratio of the probability of a sample path in the original SAN (Equation 5) to the probability of the same path in the biased SAN. The probability of a sample path has the same form in both cases, so the likelihood ratio is

$$L(E_{0,n}) = \prod_{i=0}^{n-1} \frac{C(c_{i+1},\mu_i;a_{i+1})}{C'(c_{i+1},\mu_i;a_{i+1},q_i)} \cdot \frac{f_i(T_{i+1}-T_i,\mu;a_{i+1})}{f'_i(T_{i+1}-T_i,\mu;a_{i+1})} \cdot \prod_{a\in en(\mu_i)-\{a_{i+1}\}} \frac{\overline{F}_i(T_{i+1}-T_i,\mu;a)}{\overline{F}'_i(T_{i+1}-T_i,\mu;a)}. \quad (6)$$

The next problem to address is the calculation of the expected value of reward variables under importance sampling. Reward variables are functions of the sample path of the SAN. Suppose M is a function of the sample path of a SAN, and one wants to calculate $E_P[M(E_{0,n})]$, where the subscript P denotes the probability measure with respect to which the expected value is taken. In general, if $P'(E_{0,n}) \neq 0$ whenever $M(E_{0,n})P(E_{0,n}) \neq 0$, $E_P[M(E_{0,n})] = E_{P'}[M(E_{0,n})L(E_{0,n})]$, as long as the expected values exist and are finite. We have presented the likelihood ratio expression for sample paths containing a fixed number of transitions.

It is often the case in transient simulation that one wishes to estimate $E_P[M_t]$, where M_t is the value of M when the simulation clock time is t. Because the stopping criterion is based on the simulation clock, the number of transitions in a sample path is a random variable. Let the random variable $N = min\{n : T_n > t\}$. The N-th transition advances the simulation clock past t, the observation point of the function M. Since $\{N = n\} \in \mathcal{F}_\backslash$, N is a stopping time. If N is finite with probability one under P and P', $E_P[|M_t|] < \infty$, and $P'(E_{0,n}) \neq 0$ whenever $M_tP(E_{0,n}) \neq 0$, then $E_P[M_t] = E_{P'}[M_tL(E_{0,N})]$, where $L(E_{0,N})$ is calculated as in Equation 6, with N substituted for n. For more details on the extension to stopping times, see [7].

For example, the expected value of V_t, with the reward structure given in Equation 1, over the sample path space of the delayed repair SAN, is

$$E_P[V_t] = \sum_{D\in\wp(P\times I\!N)} \mathcal{R}(D)\cdot E_P[I_t^D] + \sum_{a\in A}\mathcal{C}(a)\cdot E_P[I_t^a].$$

The function I_t^D is an indicator function that will equal one if the marking at time t contains D. The indicator function I_t^a will equal one if a was the most recent activity to complete at time t. Therefore, under importance sampling one has

$$E_P[V_t] = \sum_{D\in\wp(P\times I\!N)} \mathcal{R}(D)\cdot E_{P'}[I_t^D L(E_{0,N})] + \sum_{a\in A}\mathcal{C}(a)\cdot E_{P'}[I_t^a L(E_{0,N})].$$

To estimate $E_P[V_t]$ using importance sampling, one generates sample paths under P' and forms the sample mean

$$\xi = \sum_{i=1}^{K} V_t^i L(E_{0,N_i}),$$

where K is the number of generated paths, N_i is the number of the transition that causes the simulation clock to first exceed t during the i-th path generation, V_t^i is the i-th observation of the variable, and $L(E_{0,N_i})$ is the likelihood ratio corresponding to the i-th sample path. As indicated earlier, the expected value of any performance/dependability measure expressed in terms of the sample path of a SAN can be estimated using importance sampling. Included in this category are reward variables collected over an interval of time, an instant of time t as $t \to \infty$, and an interval of time $[t, t+l]$, as $l \to \infty$. The last two are steady state measures, so of course there must exist a steady state measure on the sample path space of the SAN for these variable types to make sense. Furthermore, one can look at the time average of reward accumulated over a given interval.

We have implemented a prototype of our environment for importance sampling in the *UltraSAN* [5] modeling package. The implementation includes a simulation engine for governed SANs and a user interface for specifying the importance sampling governor [26]. The next section describes some results obtained using the *UltraSAN* implementation of our environment.

4 Application and Results

In this section we present some results on the unreliability of a MARS FTU. The MTTF for various MARS configurations was evaluated by Kantz and Trivedi in [16] using analytical methods, but the reliability for a given mission interval was not considered. The paper by Kantz [15] examines unreliability and several other dependability measures using a hierarchical approximation approach. He notes that for high reliability applications with mission times on the order of ten hours, coverage failure is the dominant system failure mode. Neither study looked at a non-Markovian model of the system. In this section we evaluate the unreliability of a MARS FTU over an interval of ten hours using the non-Markovian model of a MARS FTU introduced in Section 2. The example serves to illustrate the utility of the new environment. In particular, we show how to implement a state-of-the-art strategy for fast simulation of dependable systems using our environment. However, because of the nature of the MARS model, this technique did not perform as well as expected. We show that a natural extension to the method provides a dramatic performance improvement and that with this extension the method performs very well.

4.1 Choosing a Strategy

The importance sampling strategy we chose for this study is balanced failure biasing [8, 31]. This strategy has been proven effective [31, 33] for a large class of Markovian systems and related methods have been suggested [22] and proven effective [10] for a large class of non-Markovian systems. As shown in Equation 2 (see Section 2), when applied to Markovian systems,

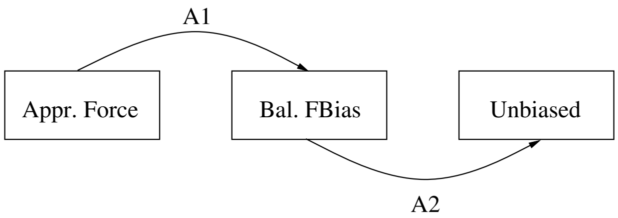

Figure 2: Governor state diagram for balanced failure biasing with exponential transformation.

balanced failure biasing increases the probability that a failure event occurs before the next repair event. Furthermore, the conditional probabilities of all possible events, given a failure occurred, are made equal. In terms of a SAN model, this is equivalent to balancing all activity rates and case distributions. In a Markovian model all hazard rates are constant and combine easily. In the non-Markovian case, one is faced with calculating hazard rates for the distribution of, for example, the minimum of m random variables representing next occurrence times for the various component failures. For this reason, an exact implementation of balanced failure biasing on a non-Markovian model is difficult. A strategy developed by Heidelberger, Shahabuddin, and Nicola [10, 22] called "balanced failure biasing with exponential transformation" has been demonstrated as an effective heuristic for non-Markovian models.

In balanced failure biasing with exponential transformation, repairs are sampled from their original distributions while failures are sampled from an exponential distribution. Given that the next failure is earlier than the next repair, the failing component is chosen according to a discrete uniform distribution over the operational components. The failure mode and components affected are sampled from their original distributions. The failure mode and components affected constructs model events of the type usually modeled by cases on activities in SANs. In SAN terms, the exponential transformation technique described in [10] balances the conditional probabilities of the failure activities, *but does not alter the case distributions.* The results we obtained for the MARS FTU indicate that in models where some case probabilities are very small, it is important to balance the case distributions as well. Although it has been mentioned in [10] that small probabilities can be handled using balanced failure biasing with exponential transformation, the exact method was not given. Balancing the case distributions is a natural approach, suggested by the balanced failure biasing heuristic used for Markovian models [31]. With this extension, the balanced failure biasing with exponential transformation technique performed very well.

4.2 Creating the Governor

Figure 2 shows the state diagram of an importance sampling governor that implements balanced failure biasing with exponential transformation. Example activity biases for each state are listed in Table 3 and the governor state transition function is given in Table 4. The first state of the governor implements approximate forcing, where the probability of a failure occurring within the ten hour time horizon is increased to 0.8. In the spirit of exponential transformation and balanced failure biasing, the bias applied to the SAN in this governor state consists of changing all the failure activity time distributions to exponential with rates equal to $-0.1\ln(0.2)/6$. Upon the first failure event, the governor transitions to the second governor state, which implements balanced failure biasing with exponential transformation. As shown in Table 4, the predicate $A1$ holds if the SAN enters a marking in which the marking of place *shadow* is zero. This will always be true right after the first failure, since the instantaneous activity *activate* immediately shifts the token from *shadow* into the place corresponding to the failed processor.

Table 3: Example bias on activity *h_fail1* for each governor state.

State	**Bias Dist. and Param. Values**
Appr. Force	exponential with rate 0.02682
Bal. FBias	exponential with rate

	int sixfails = MARK(*two*) > 0;
	double rate = 0.0;
	if (MARK(*h_cov1*) +
	  MARK(*h_cov2*) > 0)
	  rate = 120.0 / (3.0 + 3.0 * sixfails);
	else if (MARK(*i_cov1*) +
	  MARK(*i_cov2*) > 0)
	  rate = 12.0 / (3.0 + 3.0 * sixfails);
	else if (MARK(*p_cov1*) +
	  MARK(*p_cov2*) > 0)
	  rate = 1.0 / (168.0 * (3.0 +
	    3.0 * sixfails));
	return rate;

Unbiased	unbiased

As shown in Table 3, the implementation of balanced failure biasing with exponential transformation is accomplished by converting the failure activity time distributions to exponential and using marking dependent rates. The marking dependency is needed because the proper bias for the failure activities depends on which repair activities are enabled. The sum of the rates of enabled failure activities is made equal to the inverse of the expected time to the next repair [10]. This heuristic approximates the balanced failure biasing heuristic for Markovian models, as defined in [31, 33], since the distribution of the next repair time is the minimum of the activity time distributions of the enabled repair activities. The technique reduces to an exact implementation of balanced failure biasing if the repair activities are exponentially distributed. This approach also neglects the fact that a repair activity may remain enabled after a marking change. In that situation the correct distribution

Table 4: Governor state transition function.

Cur. State	Predicate	Next State
Appr. Force	MARK(*shadow*) == 0	Bal. FBias
Bal. FBias	MARK(*shadow*) == 1	Unbiased

to consider in the calculation of the minimum is the remaining time to completion of the repair activity.

In the FTU model the minimum of the enabled repairs is fairly easy to calculate since the associated uniform distributions do not overlap. Thus the expected value of the minimum of two different repair activities is simply the mean time of the faster repair. Given the already approximate nature of the implementation it seems reasonable to treat the case of two simultaneously enabled repairs of the same type as if there were only one. Therefore, the failure activities are assigned rates according to the following algorithm. Let n be the number of enabled failure activities, and let $en_f(\mu)$ be the set of failure activities enabled in μ. If at least one *h_rep* activity is enabled in μ, then the rate for $a \in en_f(\mu)$ is $120/n$. Otherwise, if at least one *i_rep* activity is enabled, then the rate assigned to $a \in en_f(\mu)$ is $12/n$. Otherwise, if at least one *p_rep* activity is enabled in μ then the rate is $1/168n$. It is important to note that despite the use of a heuristic that approximates balanced failure biasing, if observations are weighted by the likelihood ratio calculated using Equation 6, the resulting estimator is unbiased.

As an example implementation of balanced failure biasing with exponential transformation, consider the definition of activity *h_fail1*, shown in Table 3. Activity *h_fail1* is enabled when there is a token in place *one* and no token in place *system_failed*. To determine the biased rate of *h_fail1* we need to know which repair activities are enabled (to find the total repair rate), and we need to know which failure activities are enabled (so we know how to balance their rates). The flag `sixfails` is used to determine whether the second processor is working or failed. If the second processor is working, then its failure activities are enabled. As shown in Table 3, `sixfails` is used to determine whether the total repair rate is split three ways or six ways. The markings of the places *h_cov1*, *h_cov2*, *i_cov1*, etc. are used to determine the approximation of the total repair rate, which we have based on the fastest enabled repair activity.

The governor remains in the balanced failure biasing state unless the SAN enters a marking in which all processors in the FTU have been returned to working order. At this point, a path leading to system failure before the time horizon is reached is so unlikely that we turn off importance sampling so that the replication ends as quickly as possible. The predicate $A2$ holds if the SAN enters a marking where the marking of *shadow* is one. This means that a repair activity completed and deposited a token in *shadow* while both active processors were still operational. (Otherwise *activate* would have removed the token immediately, and the governor does not pay attention to unstable markings.) In this marking, all processors are operational. When $A2$ holds, the governor transitions to a third state in which no bias is applied to the SAN. This causes the failure activities to be rescheduled according to their original distributions and makes it very unlikely that another event will occur before the simulation reaches the time horizon, thus ending the replication quickly.

4.3 Unreliability of a MARS FTU

Using the governor for balanced failure biasing with exponential transformation we ran a simulation of 100,000 replications to estimate the unreliability of the MARS FTU modeled in Figure 1 using the reward structure in Equation 1. The results are shown in Table 5. The numbers in the columns labeled "Result" are the point estimates and the numbers in the "Error" columns are estimated relative half-widths of the confidence intervals at a confidence level of 99%. To test the bounded relative error property, we ran the model with four different scale factors applied to the reciprocals of the means of the failure time distributions and the coverage failure probability. For example, at a scale factor of 0.1, the mean of the activity time distribution for *h_fail1* was changed from 3,750 to 37,500 hours, and the probability of an uncovered failure was decreased from 0.0004 to 0.00004.

As can be seen from the first column of Table 5 the balanced failure biasing strategy did not perform as well as expected. As the scale factor decreases the estimator for balanced failure biasing increasingly underestimates the unreliability. For a scale factor of 0.01, balanced failure biasing with exponential transformation yields an estimate where the corresponding 99% confidence interval fails to cover the result given by the augmented technique. This phenomenon occurs when the simulation underestimates the sample variance because of the rarity of system failure events. In this case the balanced failure biasing technique is neglecting the most likely path to system failure, which is a coverage failure, and is instead focusing the simulation effort toward paths leading to the exhaustion of redundancy. When the sample variance is underestimated, the confidence interval calculated using the estimate is optimistic and hence too small. Thus for models like the MARS FTU, the coverage failure probability must be treated as a rare event in the same way as processor failures. Otherwise too much of the simulation effort is spent on unlikely paths to system failure.

The second column of the table shows results after we modified the strategy to balance the case distributions on the failure activities. This modification is a natural extension for models such as the MARS FTU, and the empirical data support the bounded relative error property for this case. Furthermore, a simple modification of the proof given in [10] shows that balancing the case distributions is sufficient to recover the bounded relative error property for models like the MARS FTU [32].

5 Summary and Conclusions

Importance sampling is a well known technique for improving the efficiency of rare event simulations.

Table 5: Unreliability of a MARS FTU in the interval $[0, 10]$ hours.

	Balanced Failure Biasing		Augmented Balanced Failure Biasing	
Scale Factor	Result	Error	Result	Error
10.0	4.96×10^{-6}	38%	4.87×10^{-6}	3%
1.0	5.55×10^{-9}	79%	4.83×10^{-9}	3%
0.1	2.15×10^{-12}	250%	4.83×10^{-12}	3%
0.01	5.55×10^{-16}	260%	4.84×10^{-15}	3%

However, finding a strategy that works well in general is widely recognized as a difficult problem. Thus research has focused on finding good strategies that work well within certain classes of systems, such as highly dependable computer systems. However, even within the class of highly dependable computer systems there are a wide variety of system characteristics and measures of dependability. From the surveyed literature it is clear that no single technique has been discovered that works well for all combinations of systems and measures. This situation motivated us develop a flexible environment for experimenting with importance sampling strategies for a wide variety of systems.

In this paper we have presented an environment for applying importance sampling to the broad class of discrete event systems representable as SAN-based reward models. The central feature of this new environment is the concept of the importance sampling governor. The governor is similar to a finite state machine, where the states of the machine redefine the stochastic parameters of the activities in the SAN, and the transitions are dependent on the evolution of the SAN. This design facilitates the use of dynamic importance sampling strategies that have proven important in the area of dependable computer system simulation. Moreover, the environment supports importance sampling for non-Markovian as well as Markovian models. This is important since many systems are more accurately modeled using non-exponential delays between events.

The utility of the new environment was demonstrated through the evaluation of a non-Markovian model of a MARS FTU for the unreliability within an interval of ten hours. First, a non-Markovian SAN-based reward model of the FTU was introduced. Then an importance sampling strategy was chosen from the literature and the construction of a governor to implement the strategy was discussed. Finally, a prototype implementation of the environment was used to obtain results.

There are many avenues for future research. The environment for important sampling presented here is applicable to a wide variety of systems and measures. It is our hope that the speed with which importance sampling strategies can be implemented, tested, and changed in this environment will aid us in the search for good strategies for highly dependable computer systems exhibiting behavior that is troublesome for known heuristics. Specifically, massive redundancy, long time horizons, and repair disciplines such as delayed group repair have yet to be rigorously addressed for all transient measures of dependability.

Acknowledgement

The authors thank the anonymous reviewers and Perwez Shahabuddin for their helpful comments.

References

[1] A. M. Blum, P. Heidelberger, S. S. Lavenberg, M. Nakayama, and P. Shahabuddin, "System availability estimator (SAVE) language reference and user's manual version 4.0," Technical Report RA 219 S, IBM Thomas J. Watson Research Center, June 1993.

[2] J. A. Carrasco, "Efficient transient simulation of failure/repair Markovian models," in *Proceedings of the 10th Symposium on Reliable Distributed Systems*, pp. 152–161, Pisa, Italy, September 1991.

[3] J. A. Carrasco, "Failure distance-based simulation of repairable fault-tolerant systems," in *Proceedings of the Fifth International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, pp. 337–351, Torino, Italy, February 1991.

[4] M. Cottrell, J.-C. Fort, and G. Malgouyres, "Large deviations and rare events in the study of stochastic algorithms," *IEEE Transactions on Automatic Control*, vol. AC-28, no. 9, pp. 907–920, September 1983.

[5] J. Couvillion, R. Freire, R. Johnson, W. D. Obal II, M. A. Qureshi, M. Rai, W. H. Sanders, and J. Tvedt, "Performability modeling with *UltraSAN*," *IEEE Software*, vol. 8, no. 5, pp. 69–80, September 1991.

[6] P. W. Glynn, "A GSMP formalism for discrete event systems," *Proceedings of the IEEE*, vol. 77, no. 1, pp. 14–23, January 1989.

[7] P. W. Glynn and D. L. Iglehart, "Importance sampling for stochastic simulations," *Management Science*, vol. 35, no. 11, pp. 1367–1392, November 1989.

[8] A. Goyal, P. Shahabuddin, P. Heidelberger, V. F. Nicola, and P. W. Glynn, "A unified framework for simulating Markovian models of highly dependable systems," *IEEE Transactions on Computers*, vol. 41, no. 1, pp. 36–51, January 1992.

[9] J. M. Hammersley and D. C. Handscomb, *Monte Carlo Methods*, Methuen & Co., Ltd., London, 1964.

[10] P. Heidelberger, P. Shahabuddin, and V. Nicola, "Bounded relative error in estimating transient measures of highly dependable non-Markovian systems,"

ACM Transactions on Modeling and Computer Simulation, vol. 4, no. 2, pp. 137–164, April 1994.

[11] R. A. Howard, *Dynamic Probabilistic Systems. Vol II: Semi-Markov and Decision Processes*, Wiley, New York, 1971.

[12] S. Juneja and P. Shahabuddin, "Fast simulation of Markovian reliability/availability models with general repair policies," in *Proceedings of the Twenty-Second Annual International Symposium on Fault-Tolerant Computing*, pp. 150–159, Boston, Massachusetts, July 1992.

[13] H. Kahn, "Random sampling (Monte Carlo) techniques in neutron attenuation problems," *Nucleonics*, vol. 6, pp. 27–33,36,60–65, May 1950.

[14] H. Kahn and M. Marshal, "Methods of reducing sample size in Monte Carlo computations," *Journal of the Operations Research Society of America*, vol. 1, pp. 263–278, November 1953.

[15] H. Kantz, "Flexible handling of diverse dependability requirements in MARS," in *Proceedings of the Tenth Symposium on Reliable Distributed Systems*, pp. 142–151, Pisa, Italy, September-October 1991.

[16] H. Kantz and K. Trivedi, "Reliability modeling of the MARS system: A case study in the use of different tools and techniques," in *4th Int. Workshop on Petri Nets and Performance Models*, pp. 268–277, Melbourne, Australia, December 1991.

[17] J. P. C. Kleijnen, *Statistical Techniques in Simulation*, volume 1, Marcel-Dekker, Inc., New York, 1974.

[18] H. Kopetz, A. Damm, C. Koza, M. Mulazzani, W. Schwabl, C. Senft, and R. Zainlinger, "Distributed fault-tolerant real-time systems: The MARS approach," *IEEE Micro*, vol. 9, no. 1, pp. 25–40, February 1989.

[19] A. M. Law and W. D. Kelton, *Simulation Modeling and Analysis*, McGraw-Hill Book Company, New York, 1982.

[20] E. E. Lewis and F. Böhm, "Monte Carlo simulation of Markov unreliability models," *Nuclear Engineering and Design*, vol. 77, pp. 49–62, January 1984.

[21] J. F. Meyer, A. Movaghar, and W. H. Sanders, "Stochastic activity networks: Structure, behavior, and application," in *Proceedings of International Workshop on Timed Petri Nets*, pp. 106–115, Torino, Italy, July 1985.

[22] V. F. Nicola, P. Heidelberger, and P. Shahabuddin, "Uniformization and exponential transformation: Techniques for fast simulation of highly dependable non-Markovian systems," in *Proceedings of the Twenty-Second Annual International Symposium on Fault-Tolerant Computing*, pp. 130–139, Boston, Massachusetts, 1992.

[23] V. F. Nicola, M. K. Nakayama, P. Heidelberger, and A. Goyal, "Fast simulation of dependability models with general failure, repair and maintenance processes," in *Proceedings of the Twentieth Annual International Symposium on Fault-Tolerant Computing*, pp. 491–498, Newcastle upon Tyne, United Kingdom, June 1990.

[24] V. F. Nicola, P. Shahabuddin, P. Heidelberger, and P. W. Glynn, "Fast simulation of steady-state availability in non-Markovian highly dependable systems," in *Proceedings of the Twenty-Third Annual International Symposium on Fault-Tolerant Computing*, pp. 38–47, Tolouse, France, June 1993.

[25] W. D. Obal II, *Importance Sampling Simulation of SAN-Based Reward Models*, Master's thesis, The University of Arizona, July 1993.

[26] W. D. Obal II and W. H. Sanders, "Importance sampling simulation in *UltraSAN*," *Simulation*, vol. 62, no. 2, pp. 98–111, February 1994.

[27] S. Parekh and J. Walrand, "A quick simulation method for excessive backlogs in networks of queues," *IEEE Transactions on Automatic Control*, vol. 34, no. 1, , January 1989.

[28] W. H. Sanders and R. S. Freire, "Efficient simulation of hierarchical stochastic activity network models," To appear in *Discrete Event Dynamic Systems: Theory and Applications*.

[29] W. H. Sanders and J. F. Meyer, "Reduced base model construction methods for stochastic activity networks," *IEEE Journal on Selected Areas in Communications*, vol. 9, pp. 25–36, January 1991.

[30] W. H. Sanders and J. F. Meyer, "A unified approach for specifying measures of performance, dependability, and performability," in *Dependable Computing for Critical Applications, Vol 4: of Dependable Computing and Fault-Tolerant Systems*, A. Avizienis and J. C. Laprie, editors, pp. 215–238, Springer Verlag, Vienna, 1991.

[31] P. Shahabuddin, *Simulation and Analysis of Highly Reliable Systems*, PhD thesis, Stanford University, 1990.

[32] P. Shahabuddin, December 1993. Private communication.

[33] P. Shahabuddin, "Fast transient simulation of Markovian models of highly dependable systems," in *Proceedings of the PERFORMANCE '93 Conference*, pp. 274–289, Rome, Italy, September 1993.

[34] P. Shahabuddin, "Importance sampling for the simulation of highly reliable Markovian systems," *Management Science*, vol. 41, no. 3, pp. 333–352, March 1994.

[35] A. P. A. van Moorsel, *Performability Evaluation Concepts and Techniques*, PhD thesis, Universiteit Twente, 1993.

[36] A. P. A. van Moorsel, B. R. Haverkort, and I. G. Niemegeers, "Fault injection simulation: A variance reduction technique for systems with rare events," in *Second International Working Conference on Dependable Computing for Critical Applications*, pp. 57–64, Tucson, Arizona, February 1991.