Adaptive Importance Sampling Technique for Markov Chains Using Stochastic Approximation

T. P. Imthias Ahamed
Department of Electrical Engineering,
T. K. M. College of Engineering,
Kollam 691005, India
imthi4@rediffmail.com

V. S. Borkar*
School of Technology and Computer Science,
Tata Institute of Fundamental Research,
Homi Bhabha Rd.,
Mumbai 400005, India
borkar@tifr.res.in

S. Juneja
School of Technology and Computer Science,
Tata Institute of Fundamental Research,
Homi Bhabha Rd.,
Mumbai 400005, India
juneja@tifr.res.in

Abstract For a discrete-time finite-state Markov chain, we develop an adaptive importance sampling scheme to estimate the expected total cost before hitting a set of terminal states. This scheme updates the change of measure at every transition using constant or decreasing step-size stochastic approximation. The updates are shown to concentrate asymptotically in a neighborhood of the desired zero variance estimator. Through simulation experiments on simple Markovian queues, we observe that the proposed technique performs very well in estimating performance measures related to rare events associated with queue lengths exceeding prescribed thresholds. We include performance comparisons of the proposed algorithm with existing adaptive importance sampling algorithms on a small example. We also discuss the extension of the technique to estimate the infinite horizon expected discounted cost and the expected average cost.

*Work supported in part by grant III.5(157)/99-ET from Dept. of Science and Technology, Government of India

1 Introduction

In this paper we develop adaptive importance sampling techniques to efficiently estimate via simulation certain important and commonly used performance measures for finite state space Markov chains with one step transition costs. We are primarily concerned with the performance measure corresponding to the expected total cost till a specified terminal set of states is hit, starting from any state. We call this the 'expected cost till termination' problem. Applications include estimation of many commonly encountered and important rare event probabilities associated with Markov chains. These find wide applications in communications networks, insurance, reliability systems etc. (see, e.g., Heidelberger 1995, Juneja 2003). Another application is the estimation of mean time to failure in reliability systems modelled as Markov chains. The proposed techniques may also be useful in efficient approximate policy evaluation via simulation for a given policy associated with a Markov decision process (see, e.g., Bertsekas and Tsitsiklis 1996).

In the last fifteen years there has been a significant interest in developing efficient importance sampling-based simulation techniques for estimating probabilities associated with rare events (see, e.g., Glynn and Iglehart 1989, Heidelberger 1995 for an introduction and an overview). As is well known, importance sampling involves simulating the system under a new probability measure and then suitably unbiasing the resultant simulation output. Research in this area has largely focused on identifying a 'static' change of measure for efficiently estimating rare events associated with simple stochastic models such as single queues and random walks, and proving the effectiveness of the proposed change of measure. In more complex settings such as queuing networks or general finite state space Markov chains, such attempts have met with a limited success (see, e.g., Andradottir, Heyman and Ott 1996, Heidelberger 1995, Juneja and Nicola 2003).

For the expected cost till termination problem in a finite state Markov chain setting, the expression for the zero variance change of measure is well known, but involves knowing *a priori* the quantities to be estimated (see Booth 1985, Kollman et al. 1999, Desai and Glynn 2001; Kuruganti and Strickland 1995 develop this in a specialized setting). A number of adaptive importance sampling techniques have been proposed in the literature that iteratively attempt to learn this zero variance change of measure. These usually involve generating many independent identically distributed sample paths till termination using a fixed change of measure in an iteration. Each path typically involves a large number of transitions of the Markov chain. One then updates the change of measure for the next iteration based on these observations (see De Boer et al. 2000, De Boer 2001, Kollman et al. 1999, Desai and Glynn 2001).

In this paper we approach the problem of learning the zero variance change of measure (and hence the expected cost till termination) using a stochastic approximation framework, in which the measure is updated after every transition of a single simulated sample path of the Markov chain. As is typical in stochastic approximation, we may use either a small constant step-size or decreasing step-sizes to smoothen the updates. We concurrently update the estimates of the expected total cost till termination and those of the transition probabilities associated with the change of measure. In this general setting, we prove that the proposed iterative scheme concentrates with high probability in a neighborhood of the zero variance change of measure. Empirically, we find that the proposed scheme performs very well. Compared to other existing schemes, it offers the advantages of simple implementation, good

performance (speed / accuracy) and robustness. This will be discussed in greater detail in Section 4.

The key differences between our approach and the existing ones may be summarized as follows: The existing methods rely on Monte Carlo-type estimation of expectations by arithmetic means of independent samples based on the law of large numbers. Often, many paths need to be generated to get a reasonably accurate estimator and these generated paths may comprise of a large number of transitions. These paths are generated under a change of measure, and for each path the likelihood ratio (ratio of the original measure and the change of measure of the generated path) needs to be used in the computation of an unbiased estimator. In the Markov chain framework, determining the likelihood ratio for a generated path entails collecting the product of successive single transition likelihood ratios along the path. The resultant output can have high variance unless the selection of the change of measure is carefully managed.

Our approach takes an altogether different viewpoint. It attempts to solve a linear system of equations satisfied by the desired averages, using stochastic approximation to do the *conditional* averaging implicit in these equations. Thus, in particular, our method involves using likelihood ratios of only single transitions at a time. This is a much simpler object analytically and computationally. The conditional averaging of the stochastic approximation scheme is also on a per transition basis. The scheme thus combines aspects of Monte Carlo simulation with deterministic numerical schemes (such as fixed point iterations) that one may employ to solve the linear system of equations in question. In fact, methodologically our approach sits between pure simulation schemes and pure deterministic numerical schemes and captures the trade-off between the two. While it may have lower variance than the former, it may require less computational time compared to the latter for getting ball-park estimates in large problems.

Note that even if our interest may be limited to estimating the expected cost till termination starting from a specified state, the proposed scheme estimates such expectations for all states in the state space as initial states. We maintain that this is unavoidable in any *adaptive* importance sampling scheme that tries to learn the zero variance change of measure. This is because the definition of the latter involves knowing such expectations for all initial states. Furthermore, estimating all such expectations has explicit utility in some contexts, as will be noted in Section 4.

In Precup et al. (2000) and Randhawa and Juneja (2004) importance sampling is used in conjunction with stochastic approximation-based learning techniques to estimate the expected cost till termination. These works, however, use a fixed or static change of measure, whereas our focus is on adaptively learning the best zero variance change of measure. Stochastic approximation approaches to importance sampling in the specific context of option pricing have been developed in Vázquez-Abad and Dufresne (1998), Su and Fu (2000) and Su and Fu (2002). These take a 'stochastic gradient'-based approach using an approximate gradient search. This is distinct from our approach which is closer to the 'stochastic fixed point iterations' of the reinforcement learning literature.

The paper is organized as follows: Section 2 develops notation and gives an overview of some existing relevant literature before stating our main adaptive algorithm. Convergence of the proposed algorithm to the zero variance change of measure is shown in Section 3. Section 4 discusses some related theoretical and practical issues. In particular, we discuss the extension of the methodology to estimate the total discounted cost and the average cost in

a finite state Markov chain setting. In Section 5, the simulation experiments are described. Section 6 concludes with some pointers to future directions. A word on terminology: we shall throughout use abbreviations '*a.s.*' for '*almost sure*' (meaning 'with probability one'), '*w.r.t.*' for '*with respect to*', and '*r.h.s.*' (resp., '*l.h.s.*') for '*right* (resp., *left*) *hand side*'.

2 Background and the Proposed Algorithm

2.1 Notation

Consider a Markov chain $(X_n : n \geq 0)$ where each X_n takes values in a finite state space S. Let $P = (p_{xy} : x, y \in S)$ denote its transition matrix and $G = (g(x, y) \geq 0 : x, y \in S)$ denote the one-step costs associated with each transition. Let $T \subset S$ denote the set of terminal states and $I = S - T$ denote the set of *interior* states. We assume that T is reachable from all states in I. Thus $\tau \stackrel{\Delta}{=} \inf\{n : X_n \in T\}$ is an a.s. finite stopping time for all initial values of X_0. Let $E_x[\ \cdot\ ]$ denote the expectation operator corresponding to the transition matrix P and initial state x. We consider the problem of estimating the 'value function' $J^* = (J^*(x) : x \in S)$, where each $J^*(x)$ denotes the total expected cost incurred along the Markov chain starting from state x till the chain hits the terminal set. Specifically,

$$J^*(x) = E_x[\sum_{k=0}^{\tau-1} g(X_k, X_{k+1})]. \tag{1}$$

Set $J^*(x) = 0$ for $x \in T$. This quantity is a typical 'cost' of interest in Markov decision processes. Other similar costs will be discussed in section 4. This framework subsumes usual objectives such as rare event probabilities if g is taken to be a suitable indicator random variable. That is, if $T = A \cup R$ with $A \cap R = \phi$ and $g(x, y) = I_{\{y \in R\}}$, then $J^*(x)$ for $x \in I$ denotes the probability of hitting set R before hitting set A, starting from the state x. Here A may denote an 'attractor set', i.e., a set visited frequently by the Markov chain, while R may denote a 'rare set'. Similarly, in case of a discrete time Markov chain arising out of uniformization of a continuous time Markov chain, if $g(x, y)$ denotes the mean time spent in state x (this will not depend on y) for each $x \in I$, then $J^*(x)$ denotes the mean time to hit set T starting from x.

2.2 Existing adaptive techniques

We first briefly describe the adaptive importance sampling methodology proposed in Booth (1985) and applied in Kollman et al. (1999) and Desai and Glynn (2001). For ease of presentation, we refer to it as the Adaptive Monte Carlo (AMC) algorithm. We then describe the Cross-Entropy (CE) algorithm proposed in De Boer et al. (2000) and De Boer (2001). These descriptions are useful as the AMC and the CE algorithms represent substantially different approaches to adaptive importance sampling compared to the one we propose. Later in Section 5, we compare via simulation the proposed algorithm with the AMC and the CE algorithms on a small example.

The AMC algorithm proceeds iteratively as follows: First we select an initial importance sampling transition matrix $P^{(0)} = (p^{(0)}_{xy} : x \in I, y \in S)$ (e.g., this may correspond to the original transition matrix). Then, with a state $x \in I$ as the initial state, the Markov chain

is simulated up till time τ using the transition matrix $P^{(0)}$, and the simulation output is adjusted by using the appropriate likelihood ratio. The average of many such independent samples gives an estimate $J^{(1)}(x)$. This is repeated independently for all $x \in I$, and $(J^{(1)}(x) : x \in I)$ is an estimator of the solution $(J^*(x) : x \in I)$.

Suppose that $J^{(n)} = (J^{(n)}(x) : x \in S)$ denotes the best guess of the solution $(J^*(x) : x \in S)$ at the initiation of iteration $n \geq 1$ (since $J^*(x) = 0$ for $x \in T$, we set $J^{(n)}(x) = 0$ for such x, for each $n \geq 1$). This $J^{(n)}$ is used to construct a new importance sampling change of measure that will then drive the sampling in the iteration n. The transition probabilities $P^{(n)} = (p_{xy}^{(n)} : x \in I, y \in S)$ associated with $J^{(n)}$ are given as:

$$p_{xy}^{(n)} = \frac{p_{xy}(g(x,y) + J^{(n)}(y))}{\sum_{z \in S} p_{xz}(g(x,z) + J^{(n)}(z))}. \tag{2}$$

Then, as with $n = 0$, with a state $x \in I$ as the initial state, the Markov chain is simulated up till time τ using the transition matrix $P^{(n)}$, the simulation output is adjusted by using the appropriate likelihood ratio, and the average of many such independent samples gives a new estimate $J^{(n+1)}(x)$. This is repeated independently for all $x \in I$, and the resultant estimators $(J^{(n+1)}(x) : x \in I)$ determine the transition matrix $P^{(n+1)}$ used in the next iteration. This process is repeated over a number of iterations, and it is shown that under certain conditions the rate of convergence is exponential as a function of the number of iterations.

An important point to note is that the transition matrix corresponding to J^* is the zero variance estimator of J^*. We refer to this transition matrix as $P^* = (p_{xy}^* : x, y \in S)$. Note that for $x \in I$ and $y \in S$,

$$p_{xy}^* = \frac{p_{xy}(g(x,y) + J^*(y))}{\sum_{z \in S} p_{xz}(g(x,z) + J^*(z))} = \frac{p_{xy}(g(x,y) + J^*(y))}{J^*(x)}, \tag{3}$$

where the last equation follows as J^* satisfies the equations resulting from the one-step analysis of the Markov chain. In our implementation in Section 5.1.2, $J^*(x)$ denotes a certain probability associated with an open Jackson network that is known to be positive for $x \in I$ (see Section 5.1.2, for detailed description). Thus, in our experiments, we keep each $J^{(n)}(x) > 0$ for $n \geq 1$ and $x \in I$. This is achieved as follows: We initially assign reasonable positive values to $J^{(0)}(x)$ for $x \in I$. If, for any $n \geq 1$, $x \in I$, $J^{(n+1)}(x)$ equals zero, we set it equal to $J^{(n)}(x)$.

De Boer et al. (2000) and De Boer (2001) consider a more specialized Markov chain where $T = A \cup R$, $A \cap R = \phi$ and $g(x,y) = I_{\{y \in R\}}$, so that $J^*(x)$ equals the probability that starting from state x, R is visited before A. Specifically, they consider a stable open Jackson queuing network. The set A corresponds to the single state where all the queues are empty, and R corresponds to the set of states where a prescribed threshold is crossed. The probability of interest is the probability that starting from a single arrival to an empty network, the aforementioned threshold is reached before the network re-empties.

The CE algorithm is also iterative: As in the AMC algorithm, first an initial importance sampling transition matrix $P^{(0)} = (p_{xy}^{(0)} : x \in I, y \in S)$ is selected. Suppose that at iteration $n \geq 0$, the importance sampling transition matrix is $P^{(n)} = (p_{xy}^{(n)} : x \in I, y \in S)$. Using this transition matrix, a large number of paths are generated that originate from the attractor set of states and terminate when either the attractor or the rare set is hit. Let k denote the number of such paths generated. Let $I_i(RE)$ denote the indicator function of path i that

takes value one if the rare set is hit and zero otherwise. The estimator for the rare event probability at iteration n equals

$$\frac{1}{k}\sum_{i=1}^{k} L_i * I_i(RE),$$

where L_i denotes the likelihood ratio of path i, i.e., the ratio of the original probability of the path (corresponding to transition matrix P) and the new probability of the path (corresponding to transition matrix $P^{(n)}$). The new $p_{xy}^{(n+1)}$ corresponds to the ratio

$$\frac{\sum_{i=1}^{k} L_i * N_{xy}(i) I_i(RE)}{\sum_{i=1}^{k} L_i * N_x(i) I_i(RE)},$$

where:

1. $N_{xy}(i)$ denotes the number of transitions from state x to state y along the path i,
2. $N_x(i)$ denotes the total number of transitions from state x along the path i.

It is easy to see (e.g., De Boer 2001) that as $k \to \infty$, $p_{xy}^{(n)} \to p_{xy}^{*}$. Note that when $p_{xy}^{*} > 0$ for some $x \in I$, $y \in I \cup R$, we do not want $p_{xy}^{(n)}$ to be zero. This is achieved in our implementation of this algorithm in Section 5.1.2 as follows: if the numerator above is found to be zero for such an (x, y) transition, we set $p_{xy}^{(n+1)} = p_{xy}^{(n)}$ and then suitably normalize the resulting probabilities from state x (we select $p_{xy}^{(0)} > 0$ for such transitions). If the denominator itself is zero, all probabilities out of x are left unchanged.

When the state space is large, the above references also propose a number of elegant modifications to their method. We, however, do not implement these. This is discussed further in Section 5.

2.3 The proposed algorithm

Let $a > 0$. This will serve as the 'step-size' or 'learning parameter'. Let μ denote a distribution with support I such that for any state in I there is a positive probability that the Markov chain with μ as the initial distribution and transition matrix P visits it before hitting the set T. Since T is reachable from any state in I, such a μ always exists, e.g., if μ assigns equal probability to each state in I. (In some cases, we may be able to choose a μ concentrated at a single point in I.) Such a μ is used to reset the Markov chain in our algorithm every time the set T is hit. Let d denote the cardinality of S, that is, $d = |S|$.

Our algorithm involves generating a path via simulation as follows:

- Select a state $x_0 \in I$ with distribution μ and select $(J^{(0)}(x) > 0 : x \in I)$. Intuitively, one expects that closer $(J^{(0)}(x) > 0 : x \in I)$ is to $(J^{*}(x) : x \in I)$, the better it is. Since, $(J^{*}(x) : x \in I)$ is not known, any reasonable choice for $(J^{(0)}(x) > 0 : x \in I)$ may be made. Similarly, the initial transition probabilities $(p_{xy}^{(0)} : x \in I, y \in S)$ are selected. For example, these may equal the original transition probabilities. These probabilities are used to generate the next state x_1 in the simulation.

- *Case 1:* Suppose $x_n \in I$. At transition n, state x_{n+1} is generated using $(p_{xy}^{(n)} : x \in I, y \in S)$. The updated values $(J^{(n+1)}(x) : x \in I)$ and $(p_{xy}^{(n+1)} : x \in I, y \in S)$ are determined as follows:

$$J^{(n+1)}(x_n) = (1-a)J^{(n)}(x_n) + a(g(x_n, x_{n+1}) + J^{(n)}(x_{n+1})) \left(\frac{p_{x_n x_{n+1}}}{p_{x_n x_{n+1}}^{(n)}} \right), \qquad (4)$$

and $J^{(n+1)}(x) = J^{(n)}(x)$ for $x \neq x_n$. Also,

$$\tilde{p}_{x_n x_{n+1}}^{(n+1)} = \max(p_{x_n x_{n+1}} \left(\frac{g(x_n, x_{n+1}) + J^{(n+1)}(x_{n+1})}{J^{(n+1)}(x_n)} \right), \delta), \qquad (5)$$

where $\delta > 0$ is a small scalar threshold which ensures that the likelihood ratio $(p_{x_n x_{n+1}}/p_{x_n x_{n+1}}^{(n)})$ remains bounded from above by a constant (say) $M > 0$. We pick a δ much smaller than $\min(p_{xy} : x \in I, y \in S, p_{xy} > 0)$ as a good guess. Let $\tilde{p}_{x_n y}^{(n+1)} = \tilde{p}_{x_n y}^{(n)}$ for $y \neq x_{n+1}$. Then, the probabilities $(p_{x_n y}^{(n+1)} : y \in S)$ are obtained through the normalization, i.e.,

$$p_{x_n y}^{(n+1)} = \frac{\tilde{p}_{x_n y}^{(n+1)}}{\sum_{z \in S} \tilde{p}_{x_n z}^{(n+1)}}.$$

Again for $x \neq x_n$, $p_{xy}^{(n+1)} = p_{xy}^{(n)}$ for all y. Finally, n is incremented by 1.

Case 2: If $x_n \in T$, we select x_{n+1} in I according to μ, and the simulation is resumed. $(J^{(n+1)}(x) : x \in I)$ and $(p_{xy}^{(n+1)} : x \in I, y \in S)$ are then set to $(J^{(n)}(x) : x \in I)$, and $(p_{xy}^{(n)} : x \in I, y \in S)$ and n is incremented by 1.

The computational effort in generating state x_{n+1} from x_n is $O(d)$ (it may be smaller if the transition matrix of the Markov chain is sparse, as for the Jackson networks). Also, the computational effort for the normalization of probabilities at each transition (x_n, x_{n+1}) may be $O(d)$, as the probability corresponding to each transition (x_n, y) for $y \in S$ needs to be adjusted. This may be reduced to $O(1)$ using the following approach: Suppose that at transition n we have stored a positive constant $NC_n(x_n)$ and non-negative numbers $(q_{x_n y} : y \in S)$ such that $p_{x_n y}^{(n)} = q_{x_n y}/NC_n(x_n)$ for $y \in S$. Then, using these stored numbers, it is straightforward to generate in $O(d)$ time a transition (x_n, x_{n+1}), with x_{n+1} having the distribution $(p_{x_n y}^{(n)} : y \in S)$. Now by setting

$$NC_{n+1}(x_n) = NC_n(x_n)(1 + \tilde{p}_{x_n x_{n+1}}^{(n+1)} - q_{x_n x_{n+1}}/NC_n(x_n)),$$

resetting $q_{x_n, x_{n+1}} = NC_n(x_n)\tilde{p}_{x_n x_{n+1}}^{(n+1)}$, and noting that

$$p_{x_n y}^{(n+1)} = \frac{p_{x_n y}^{(n)}}{1 + \tilde{p}_{x_n x_{n+1}}^{(n+1)} - p_{x_n x_{n+1}}^{(n)}}$$

for $y \neq x_{n+1}$ and

$$p_{x_n x_{n+1}}^{(n+1)} = \frac{\tilde{p}_{x_n x_{n+1}}^{(n+1)}}{1 + \tilde{p}_{x_n x_{n+1}}^{(n+1)} - p_{x_n x_{n+1}}^{(n)}},$$

it follows that $p_{x_n y}^{(n+1)} = q_{x_n y}/NC_{n+1}(x_n)$ for $y \in S$. This is an $O(1)$ computation. The algorithm may then be appropriately adjusted. For example, $p_{x_n x_{n+1}}^{(n)}$ is replaced by $q_{x_n x_{n+1}}/NC_n(x_n)$ in (4) in this case.

Issues related to the stopping rule for the algorithm are discussed later in Section 4. When decreasing step-sizes $(a_n : n \geq 1)$ are used, a_n replaces a in (4). The usual assumptions from stochastic approximation theory apply: $\sum_n a_n = \infty, \sum_n a_n^2 < \infty$. For notational convenience, we refer to the proposed algorithm as the Adaptive Stochastic Approximation (ASA) algorithm.

Typically, our interest is in estimating $J^*(x)$ for a particular state x, or more generally, estimating a single function $\tilde{J}$ of $(J^*(x) : x \in I)$. For example, in rare event settings we may be interested in estimating the probability that R is hit before A where $T = A \cup R$ with $A \cap R = \phi$ and $g(x, y) = I_{\{y \in R\}}$, starting from a specified initial distribution on the states I. To generate confidence intervals, each run of the proposed algorithm may be run for a large number of transitions m and $\tilde{n}$ such independent runs may be generated. Let $\tilde{J}_i(m)$ denote the estimate based on the final reading of the algorithm in the run $i \leq \tilde{n}$. Note that $(\tilde{J}_i(m) : i \leq \tilde{n})$ are i.i.d. Then, $\sum_{i \leq \tilde{n}} \tilde{J}_i(m)/\tilde{n}$ denotes the sample mean of $\tilde{J}$. Similarly, the sample variance s^2 equals

$$\frac{1}{\tilde{n}-1} \sum_{i \leq \tilde{n}} \left(\tilde{J}_i(m) - (\frac{1}{\tilde{n}} \sum_{j \leq \tilde{n}} \tilde{J}_j(m)) \right)^2 .$$

With these estimates, for large $\tilde{n}$, the approximate 95% confidence interval for $\tilde{J}$ may be constructed by using standard central limit theorem based normal approximations. When $\tilde{n}$ is small, say < 30, then approximations involving Student-t distribution may be used (see any standard simulation text, e.g., Fishman 1996).

In our algorithm $(J^{(0)}(x) : x \in I)$ may differ significantly from $(J^*(x) : x \in I)$ and hence for each i, $\tilde{J}_i(0)$ may differ significantly from $\tilde{J}$. Therefore, in each independent run, the simulation output may have a significant initialization bias (this bias becomes negligible asymptotically at an exponential rate; see the discussion in Section 4). Hence, generating independent runs is wasteful in that the initialization bias is incurred in all the runs. Alternatively, using our algorithm, a single long run $(\tilde{J}(m) : m \geq 0)$ may be generated, and a method of batch means (see Bratley et al. 1987) may be used to construct confidence intervals. Here some threshold value k_0 is chosen so that the bias is small for iterates $\tilde{J}(k)$ for $k \geq k_0$. Thereafter, the iterates from the algorithm are divided into contiguous batches of size m_0 each. Then the sample average over $k_0 \leq m < k_0 + m_0$, i.e.,

$$\frac{1}{m_0} \sum_{m=k_0}^{k_0+m_0-1} \tilde{J}(m)$$

denotes the average of the first batch, the sample average over $k_0 + m_0 \leq m < k_0 + 2m_0$ denotes the average of the second batch, and so on. For large k_0 and m_0, these averages may be assumed to be approximately i.i.d. by invoking a suitable 'mixing' property and may be used to construct confidence intervals. In our experiments, however, we generate independent samples to avoid the added approximation that the batch means technique introduces. Note that this way our estimates of computational effort are conservative. Furthermore, we note that in many of our experiments, the iterates generated are very close to each other, so that even a single sample provides a reliable estimate.

In certain applications, we may be interested in estimating $J^*(x)$ for all $x \in I$. In particular, we may be interested in generating confidence intervals that hold uniformly for all estimates of $(J^*(x) : x \in I)$. This may happen in the Markov decision process setting, where the above algorithm may be used to approximately evaluate a policy via simulation. In the existing literature, techniques relying on Bonferroni inequality exist for developing simultaneous confidence intervals for all the estimates. The drawback of these techniques is that the computational effort required becomes enormous as the state space I becomes large (see, e.g., Fishman 1996 for a discussion on related issues). An alternative approach (which we do not follow here, but shall discuss later) is to use the functional central limit theorems associated with stochastic approximation schemes to come up with simultaneous confidence intervals.

3 Convergence Analysis

This section briefly sketches the theoretical underpinnings of our scheme. It draws upon the theory of stochastic approximation algorithms, notably the analytic techniques therein that have been developed for reinforcement learning algorithms. Our main results are Theorems 1 and 2 below. The expectation throughout what follows is w.r.t. the actual law of the non-stationary chain $\{x_n\}$ generated by the simulation. We first take up the decreasing step-sizes case for which the theory is readily available. Let

$$\hat{p}_{xy} = \frac{\max(p^*_{xy}, \delta)}{\sum_z \max(p^*_{xz}, \delta)}.$$

This equals p^*_{xy} if indeed $\delta < p^*_{xy}$. Let $\hat{P}$ denote the corresponding transition matrix. In the remaining discussion, $||\cdot||$ denotes the Euclidean norm. Later we introduce two other norms, $||\cdot||_w, ||\cdot||_\infty$. These are all equivalent in the sense that any one of them can be bounded from above by a constant times the other.

Theorem 1 For step-sizes $(a_n : n \geq 1)$ satisfying the conditions $\sum_n a_n = \infty$ and $\sum_n a_n^2 < \infty$, the iterates satisfy a.s.,

$$\begin{aligned} \sup_n ||J^{(n)}||^2 &< \infty, \\ J^{(n)} &\to J^*, \\ P^{(n)} &\to \hat{P}. \end{aligned}$$

Proof Rewrite (4) as

$$\begin{aligned} J^{(n+1)}(x_n) &= J^{(n)}(x_n) + a_n(\sum_j p_{x_n j}(g(x_n, j) + J^{(n)}(j)) - J^{(n)}(x_n)) \\ &\quad + a_n[(g(x_n, x_{n+1}) + J^{(n)}(x_{n+1}))\left(\frac{p_{x_n x_{n+1}}}{p^{(n)}_{x_n x_{n+1}}}\right) \\ &\quad - \sum_j p_{x_n j}(g(x_n, j) + J^{(n)}(j))]. \end{aligned} \tag{6}$$

This can be rewritten as

$$J^{(n+1)}(i) = J^{(n)}(i) + a_n I(x_n = i)[(F_i(J^{(n)}) - J^{(n)}(i)) + M_i(n+1)] \tag{7}$$

for all $i \in I$, where:

1. $F = [F_1, \cdots, F_{|I|}]^T : \mathcal{R}^{|I|} \to \mathcal{R}^{|I|}$ is defined by

$$F_i([x_1, \cdots, x_{|I|}]) = \sum_{j \in I} p_{ij} x_j + \sum_{j \in S} p_{ij} g(i,j), \ i \in I.$$

2. $\{M(n)\}$, $M(n) = [M_1(n), M_2(n), \cdots, M_{|I|}(n)]$, is a sequence of random vectors in $\mathcal{R}^{|I|}$ satisfying

$$\begin{aligned} E[M(n+1)|x_m, m \leq n] &= 0, \\ ||M(n+1)|| &\leq K(1 + ||J^n||), \end{aligned}$$

for some constant $K > 0$. Specifically, $M(n+1)$ will be the vector whose all but the x_n-th elements are zero and the x_n-th element is given by the expression in square brackets in (6).

The matrix $\bar{P} \stackrel{\Delta}{=} (p_{xy} : x, y \in I)$ is sub-stochastic. Thus by the Perron-Frobenius theorem, it has an eigenvalue $\alpha \in (0,1)$ and the associated positive eigenvector $\bar{w} = [w_1, \cdots, w_{|I|}]$. Define the associated weighted sup-norm $||x||_w \stackrel{\Delta}{=} \max_{i \in I} |\frac{x_i}{w_i}|$. It is easily verified (see, e.g., the proof of Lemma 9, Tsitsiklis 1994) that

$$||F(x) - F(y)||_w \leq \alpha ||x - y||_w. \tag{8}$$

That is, F is a contraction w.r.t. this norm. In particular, it has a unique fixed point, viz., J^*. The a.s. boundedness of the iterations now follows by applying Theorem 1 of Tsitsiklis (1994) (see also Prop. 4.7, pp. 159-160, of Bertsekas and Tsitsiklis 1996). Note also that if $||x||_\infty \stackrel{\Delta}{=} \max_i |x_i|$, then F is *non-expansive* w.r.t. this norm in the sense that

$$||F(x) - F(y)||_\infty \leq ||x - y||_\infty.$$

The first two claims now follow from Theorem 3 of Tsitsiklis (1994). The last claim then follows by our definition of $\{p^*_{xy}\}$ and our choice of δ. □

For the constant step-size case, we have as the counterpart Theorem 2 below. As the results of Tsitsiklis (1994) are explicitly for the decreasing step-size case, we adapt instead the arguments of Borkar and Meyn (2000) based on the 'o.d.e.' (for '*ordinary differential equations*') approach to stochastic approximation. This approach, which goes back to Derevitskii and Fradkov (1974), treats the stochastic approximation iterations as a noisy discretization of a limiting o.d.e. and establishes their limiting behavior by proving that they track the o.d.e. trajectories in an appropriate sense.

Theorem 2 Suppose $\delta \approx O(a)$. Then, under a sufficiently small constant step-size $a > 0$, the iterates satisfy

$$\sup_n E[||J^{(n)}||^2] < \infty, \tag{9}$$

$$\limsup_{n\to\infty} E[||J^{(n)} - J^*||^2] = O(a), \tag{10}$$

$$\limsup_{n\to\infty} E[||P^{(n)} - \hat{P}||^2] = O(a). \tag{11}$$

Proof We adapt Theorem 2.1(ii) and Theorem 2.3(ii) of Borkar and Meyn (2000) for our purposes. The arguments used by Borkar and Meyn (2000) to prove these require the o.d.e. limit of the stochastic approximation. In our case, this is

$$\dot{x}(t) = \Pi(t)(F(x(t)) - x(t)),$$

where $\Pi(t)$ is a diagonal matrix for each t whose diagonal elements are nonnegative and add to one (see, e.g., Lemma 3.1 - Theorem 3.1, pp. 845-848, in Section 3 of Borkar 1998 where such a result is proved for *decreasing step-sizes*). Note that this differs from the o.d.e. limit for the 'synchronous' iterations, in which all components of $J^{(n)}$ are updated concurrently. In that case, the matrix $\Pi(t)$ on the r.h.s. is missing. The latter arises here due to averaging of the indicator $I(x_n = i)$ on the r.h.s. of (7). That is, one views $\{x_n\}$ as 'Markov' noise for the stochastic approximation scheme and applies the standard theory of averaging w.r.t. such noise (Section 8.4, Kushner and Yin 1997). The vector form of the iteration has the diagonal matrix with $I(x_n = i)$ as the $i-$th diagonal element. These get replaced by their average, leading to the $\Pi(t)$ term above. Because the underlying chain is non-stationary with slowly varying transition probabilities, there will be explicit time-dependence. Intuitively, the $i-$th diagonal element of $\Pi(t)$ will reflect the relative frequency with which the $i-$th component is being updated. This o.d.e. may also be written as

$$\dot{x}(t) = \tilde{F}_t(x(t)) - x(t), \tag{12}$$

where $\tilde{F}_t(x) \stackrel{\Delta}{=} (I - \Pi(t))x + \Pi(t)F(x)$, I being the identity matrix. The only difference between this and the framework of the Borkar and Meyn (2000) referred to above is the explicit time dependence of the r.h.s. of this o.d.e. Nevertheless, from the proofs of the aforementioned theorems in Borkar and Meyn (2000), it will clearly suffice to show:

1. J^* is the unique globally (with reference to *both* the initial state and the initial time) asymptotically stable equilibrium of (12) (*which ensures (10) if the second moment of the iterates remain bounded*) and,

2. the origin is the globally (in the above sense) asymptotically stable equilibrium of the o.d.e.

$$\dot{x}(t) = \bar{F}_t(x(t)) - x(t), \tag{13}$$

where, $\bar{F}_t(x) \stackrel{\Delta}{=} (I - \Pi(t))x + \Pi(t)\hat{F}(x)$, for

$$\hat{F}_i([x_1, \cdots, x_{|I|}]) = \sum_{j\in I} p_{ij}x_j, \ i \in I.$$

This in turn ensures that the the second moment of the iterates do indeed remain bounded. ($\hat{F}$ corresponds to replacing g identically by zero in the definition of F.)

That is, we need the properties of F used in the proofs of the aforementioned theorems of Borkar and Meyn (2000) to hold also for $\tilde{F}_t$ uniformly in t. In view of (8), these follow exactly as in Theorem 3.1 of Borkar and Soumyanath (1997) if we show that the diagonal elements of $\Pi(t)$ remain bounded away from zero. This is because the latter condition ensures that

$$||\tilde{F}_t(x) - \tilde{F}_t(y)||_w \leq \bar{\beta}||x - y||_w$$

for a *common* $\bar{\beta} \in (0, 1)$, and likewise for $\{\bar{F}_t\}$. In our case, $\bar{\beta} = 1 - a + a\alpha$ will suffice. Arguing as in Borkar and Soumyanath (1997), one can then show that $||x - J^*||_w$, resp., $||x||_w$ serve as the 'Liapunov functions' for the two o.d.e.s in question (the development in Borkar and Soumyanath (1997) is for the w_i's identically equal to one, but it takes a minor modification of the proofs therein to accommodate more general weights). Also see Theorem 3.1(b) of Borkar 1998 for analogous arguments.

To verify that the diagonal elements of $\Pi(t)$ are indeed bounded away from zero, apply the standard theory of 'averaging w.r.t. Markov noise' for stochastic approximation (see, e.g., Section 8.4 of Kushner and Yin 1997) to conclude that the $x-$th diagonal element, denoted by $\pi_x(t)$, will in fact be the stationary probability of x under transition probabilities of the form

$$q_{xy} = \max(p_{xy}\left(\frac{g(x,y) + \tilde{J}(y)}{\tilde{J}(x)}\right), \delta)$$

for some vector $\tilde{J} > 0$, suitably normalized. (Intuitively, this is because $\pi_x(t)$ will be tracking the stationary average of the indicator $I(x_n = i)$ in (7) w.r.t. the current, slowly varying transition mechanism being used, which has the above form by virtue of (5).) It is easy to see that under our irreducibility assumptions on the original chain, this suffices to ensure that the above $\pi_x(t)$ remains bounded away from zero for all x, t (see, e.g., Lemma 2.2, p. 843, of Borkar 1998).

This allows us to invoke the aforementioned results of Borkar and Meyn (2000) to conclude the first two claims. The third follows easily from the second by our definition of $\{p^*_{xy}\}$ and our choice of δ. □

Note that if we drop the requirement $\delta \approx O(a)$, we need to replace $O(a)$ by $O(a + \delta)$ in the third claim above. The convergence/near-convergence of $\{J^{(n)}\}$ to J^* 'independent of the underlying transition mechanism' may appear surprising. This is because the averaging property of stochastic approximation ensures that the algorithm sees the one step conditional average w.r.t. the underlying law of the Markov chain. This average turns out to be *exactly the same* as that under the original law after the correction due to the likelihood ratio is accounted for. This is clear from the fact that the definition of the function F above does not involve the $\{p^{(n)}_{xy}\}$, though that of $M(n + 1)$ does. The latter 'noise variables' are 'averaged out' by the algorithm. Hence they do not affect the limiting o.d.e. and therefore the asymptotic limit, though they certainly affect the transients and the rate of approach to the limit.

4 Discussion and Extensions

1. **Advantages:** The main advantages of the ASA algorithm are:

- *Simple implementation:* The scheme has a single simple iteration with low demand for per iterate computation and memory.
- *Good performance:* Our simulations show very good performance both in terms of speed and accuracy. This is hardly surprising, as the iterations use only the single transition likelihood ratio $p_{x_n x_{n+1}}/p^{(n)}_{x_n x_{n+1}}$ at time n unlike in other approaches, where many independent paths are generated, and for each path the likelihood ratio is the product of the single transition likelihood ratios accumulated along its transitions. Both computationally and analytically, single transition likelihood ratio is a much more manageable object.
- *Robustness:* We found the performance of the algorithm quite robust to parameters such as step-size over a broad range of values, initial guesses, etc. In particular there were no 'special choices' involved. For example, the method of De Boer (2001) calls for an initial change of measure under which the network is unstable. Our scheme does not require any such special choice. While good initial guesses, when available, will no doubt help to reduce the initial transients (see Section 5.2), they were not crucial for the successful working of the algorithm.
- *Extensibility:* Our scheme extends well to criteria other than the expected total cost till termination. This being an important issue, we deal with it in greater detail later.

2. **Convergence rate:** For stochastic approximation, the time to converge will obviously depend on the initial condition. For decreasing step-size schemes that are synchronous (i.e., all components are updated simultaneously), estimates for sample complexity, i.e., number of iterates (equivalently, transitions) needed to be within a given accuracy of the goal (and remain there) with prescribed probability, are given in Borkar (2002). For constant step-size, observe that our iteration has the form

$$z_{n+1} = f(z_n) + a\xi_{n+1},$$

where, for $\gamma = 1 - a + a\alpha \in (0, 1)$,

$$||f(y) - f(x)||_w \leq \gamma ||y - x||_w \ \forall y, x, \tag{14}$$

and $||\xi_m||_w \leq K \ \forall n$, for some constant $K > 0$ (the norm $||\cdot||_w$ was defined in Section 3). (Take $f(x) = (1 - a)x + aF(x)$.) Also, J^* is the unique fixed point of f. A simple induction using (14) shows that the $n-$step mean square error can be bounded by a sum of two terms,: a term exponentially decaying in n with exponent γ and an $O(a)$ term independent of n. The former decay is slower for smaller values of a. The second term arises from the 'stationary error probability': recall that a constant step-size stochastic approximation does not converge to the intended limit in general, but concentrates around it with high probability in the limit (see Theorem 2 above).

The exponential decay of the first term means that the initial bias decays exponentially. In particular, this justifies the use of asymptotic formulae for variance for confidence intervals. These confidence intervals may be derived from the functional central limit theorems for suitably normalized fluctuations around the average behavior depicted by the o.d.e. trajectory (see, e.g., Kushner and Yin 1997). The variance therein

will be dictated by the conditional variance of the martingale terms $\{M_{n+1}\}$. In our experiments, however, we avoid this complex analysis and simply generate many i.i.d. runs of our algorithm. Each has a large and equal number of transitions. From these one can develop confidence intervals using classical statistical methods (also see Hsieh and Glynn 2002 for a similar approach).

3. **Extensions:** We can extend the foregoing quite easily to other important performance measures. For example, suppose that the Markov chain $(X_n : n \geq 0)$ is irreducible over the finite state space S. Consider the 'expected discounted cost'

$$E[\sum_m \beta^m g(X_m, X_{m+1})],$$

where $\beta \in (0,1)$ is the discount factor. One can consider the augmented Markov chain $(X_n, Y_n, n \geq 0)$ on $S \times \{0,1\}$ with $\{Y_n\}$ independent of $\{X_n\}$, such that $Y_0 = 0$, $P(Y_{n+1} = 1|Y_n = 0) = 1 - \beta = 1 - P(Y_{n+1} = 0|Y_n = 0)$, $P(Y_{n+1} = 0|Y_n = 1) = 1$. Replace the above by the equivalent cost

$$E[\sum_{m=0}^{T-1} g(X_m, X_{m+1})],$$

where $T = \inf\{n \geq 0 : Y_n = 1\}$. This is in the desired format. Another interesting performance measure in 'expected average cost' setting is

$$\gamma = \sum_{x \in S} \pi(x) \sum_{y \in S} p_{xy} g(x,y),$$

where $\pi(\cdot)$ is the stationary probability distribution for the Markov chain. Noting that the process regenerates every time some prescribed state $x_0 \in S$ is visited, this can be written as the ratio of $G(x_0) \stackrel{\Delta}{=} E_{x_0}[\sum_{m=0}^{T-1} g(X_m, X_{m+1})]$ and $\bar{T}(x_0) \stackrel{\Delta}{=} E_{x_0}[T]$ where $T = \inf\{n > 0 : X_n = x_0\}$. Both these quantities can be estimated as above by separate simulations. This procedure will also calculate $G(x)$ and $\bar{T}(x)$ for $x \neq x_0$, which seems an unnecessary overhead. However, the quantities $G(x) - \gamma \bar{T}(x), x \in S$, are useful for computing the gradient (sensitivity) of γ w.r.t. an underlying parameter - see, e.g., Cao and Chen (1997). An important example in the average cost setting is $g(x,y) = I\{x \in B\}$ for a rare set B, where we estimate the steady state probability of being in set B.

Another important possibility is to use this scheme to learn an optimal or near-optimal policy for a Markov decision process. While this is an area rich in possibilities, we shall very briefly outline only one such scheme by way of illustration. Replace p_{xy} above by $p_{xy}(u), u \in U$, where U is a finite 'action' set and the problem is to choose a good $v : S \to U$ such that the transition probabilities $\{p_{xy}(v(x))\}$ minimize a given performance measure (say, the total expected cost till termination as above) over the choice of all such $v(\cdot)$. One can mimic the 'actor-critic' paradigm of Konda and Borkar (1999) to run (4) and (5) above in conjunction with a 'search algorithm' for the optimal policy on a slower time scale. Konda and Borkar (1999) describe three paradigms for such a scheme for the discounted cost, all of which can easily be adapted to total cost till termination. We refer the reader to the above reference for details. It should be noted,

however, that these search schemes perform the search in the larger set of randomized policies. These amount to performing at each time an independent random experiment to choose an element of U, according to a distribution which, however, depends on the current state. Thus the search space is the set of probabilities on U rather than U itself.

We note *en passant* that the related $TD(\lambda)$ algorithm of reinforcement learning for evaluating the value of a fixed policy can be viewed as stochastic approximation with additional 'approximate control variates' included to reduce variance (see, e.g., Bertsekas and Tsitsiklis 1996, Randhawa and Juneja 2004). Such features could be built into our scheme, but we did not do so because the variance was well contained even without incurring such additional computational overhead.

5 Simulation Results

This section presents some sample simulation results to support the foregoing analysis. It is organized as follows: To begin with, we consider the problem of estimating the probability that the total queue length hits a large threshold N in a busy cycle of a simple open Jackson network comprising two single-server queues (busy cycle is the time between the two consecutive instants when an arrival to the network finds it empty). Recall that this fits the framework used to describe the CE method in Section 2.2. In Section 5.1.1, we first empirically analyze how the choice of step-size affects the rate of convergence of the proposed algorithm. Then in Section 5.1.2, we compare the performance of the ASA algorithm with the AMC and the CE algorithms for this example. We then compare the performance of the three algorithms for an example with a larger state space corresponding to a 5-queue open Jackson network (with a single server at each queue), in Section 5.1.3. In Section 5.1.4, we briefly discuss the performance of the ASA algorithm vis-a-vis deterministic iterative procedures such as the value iteration method. Finally, in Section 5.2 we compare the performance of the proposed ASA algorithm and the AMC algorithm for more general performance measures by estimating the mean time for the total queue length to hit a specified threshold N in an open Jackson network. We do not consider the CE method in this setting as in the literature its methodology is restricted to estimating the probability of certain rare events (while it may be feasible to extend the methodology to more general measures, we do not attempt that here). All our experiments are performed on a Pentium-IV, 2 GHz, 512 MB RAM machine.

5.1 Probability of Hitting a Rare Set

Consider a two queue tandem open Jackson network with a single server at each queue, where the external arrival rates to the two queues are $(\lambda_1, \lambda_2) = (0.04, 0.0)$ (in particular, the second queue serves only the output stream of the first) and the service rates are $(\mu_1, \mu_2) = (0.48, 0.48)$. As mentioned earlier, we are interested in estimating the probability of the total queue length reaching a large threshold N in a busy cycle, denoted by $J^*([1, 0])$, where, for $y_1 + y_2 < N$, $J^*([y_1, y_2])$ is the probability of this event before the network empties, starting from state corresponding to y_1 customers in the first queue and y_2 customers in the second queue. This example was considered in De Boer (2001). As noted in Glasserman and Kou

(1995) and De Boer (2001), this is a difficult network to simulate under static importance sampling.

The simulation starts when an arrival to the network finds it empty, i.e., at state $[1,0]$. We initialize $J^{(0)}([y_1,y_2]) = 0.1$ for $0 < y_1 + y_2 < N$, $J^{(0)}([0,0]) = 0$ and $J^{(0)}([y_1,y_2]) = 0$ for $y_1 + y_2 = N$ (recall that $g([x_1,x_2],[y_1,y_2]) = 1$ if $x_1 + x_2 < N$ and $y_1 + y_2 = N$). All the transition probabilities are initially set equal to the original transition probabilities. Every time the total queue length hits N, the simulation is reset to the state $[1,0]$. A large number m of transitions are generated using the proposed algorithm. An estimate of $J^*([1,0])$ at the final transition, i.e., $J^{(m)}([1,0])$, is recorded. To develop confidence intervals of $J^*([1,0])$, we conduct many such independent simulation runs and use the i.i.d. samples of $J^{(m)}([1,0])$, as discussed in Section 2.3.

5.1.1 Varying step-sizes

We run the proposed algorithm with different values of a for $N = 25$. In each run 300,000 iterations are conducted (equivalently, 300,000 transitions are generated), and 200 such independent runs are used to construct confidence intervals. The amount 300,000 was chosen because visually it appears that the bias vanishes by 300,000 iterations for $a \geq 0.2$. (A more sophisticated criterion is to stop when a suitable average of the estimators stabilizes in the sense that it does not vary by more than $\epsilon = 1\%$ over a large number n of transitions. Empirically we observed that the stopping time so generated for $n = 1,000$ was smaller than 300,000 for $a \geq 0.2$.) We chose 200 independent runs so that the relative confidence width becomes stable. It was observed that the relative confidence interval width is extremely small but variable and hence more than 200 samples may be needed to estimate it accurately. Nonetheless its small value indicates that pretty much all samples are very close to the true value.

We also run the algorithm with decreasing step-sizes $(a_n : n \geq 0)$. The rationale for selecting step-sizes in this manner is well known: Initially step-sizes should be large so that the bias is removed quickly. Thereafter, step-sizes should be reduced so that the estimate becomes less variable.

The step-sizes were chosen as follows: $a_n = c$ for $n \leq 100,000$ and $a_n = c * (100,000/n)^{3/4}$ for $n > 100,000$ for $c = 0.9$ and 0.98. The rationale for selecting a_n proportional to $1/n^{3/4}$ rather than the customary rate $1/n$ in Monte-Carlo simulations is that in stochastic approximation settings (see, e.g., Singh and Dayan 1998, Randhawa and Juneja 2004), typically the step-sizes proportional to $1/n$ have no special advantage over other reasonable choices. Randhawa and Juneja (2004), found this to be true through extensive experimentation with step-sizes proportional to $1/n^\alpha$ for $\alpha \in (0.6, 1)$ in a similar problem setting. In our experiments step-sizes $c/n^{0.6}$, $c/n^{0.75}$ and c/n gave almost identical confidence intervals for the values of c considered.

The results are displayed in Table 1. As may be expected, for small $a = 0.1$, the bias has not vanished in 300,000 iterations. A large value of a (e.g., $a = 0.98$), on the other hand, may give a bad estimate due to large variance. The algorithm is remarkably stable for all intermediate values of a. The algorithm with decreasing step-sizes performs somewhat better than the corresponding algorithm with fixed step-sizes.

The plot of estimated $J^*([1,0])$ as a function of the number of transitions for one simulation run having 800,000 transitions is given in Fig. 1. From the figure, we see that except when

a is close to 1 or it is close to 0, (e.g., $a = 0.98$ and 0.1, respectively), the simulation output maybe expected to quickly converge to the correct value.

Note from Table 1 and Fig. 1 that for large values of a, i.e., $a = 0.98$ and $a =$ '0.98 and decreasing' for initial transitions, typically the iterates $J^{(k)}([1,0])$ underestimate $J^*([1,0])$ while occasionally a spike in the output leads to an overestimation. A heuristic reasoning for this is easily seen. For a close to 1, a very small weight of $(1-a) \approx 0$ is assigned to the previous iterate, hence the update step in (4) is governed primarily by the second term on the right hand side. This term may typically be small, as from a state x_n, a state x_{n+1} is selected frequently if $p^{(n)}_{x_n x_{n+1}}$ is large. In this case, the ratio $\frac{p_{x_n x_{n+1}}}{p^{(n)}_{x_n x_{n+1}}}$ assigns a low weight to the output and hence $J^{(n)}(x_n)$ will be underestimated. Rarely, a correction event for which $p^{(n)}_{x_n x_{n+1}}$ is small is observed leading to a spike in the value of the corresponding $J^{(n)}(x_n)$. In the case where step-sizes eventually start decreasing, the bias underestimation introduced in the constant step-size phase may take a long time to vanish. This explains the underestimation for '0.98 and decreasing' case in Table 1.

It is also interesting to note that for larger values of a, at least for the problem of estimating rare event probabilities, the error estimate $O(a)$ of Theorem 2 above is of little value. This is because the quantity to be estimated is typically of much smaller magnitude. This in fact works in our favor: The theoretical analysis of Theorem 2 is an application of general stochastic approximation theory where usually small a are warranted. Despite the fact that this error estimate does not mean much for larger a, our empirical evidence of excellent convergence properties of the algorithm for larger a only shows that the error estimate has been very conservative. Larger a implies faster convergence (lower bias) in view of the interpretation of a as a time step. Thus the empirical observation that this is permissible without compromising variance too much is an added advantage of our scheme.

a	Estimator ($\times 10^{25}$)	a	Estimator ($\times 10^{25}$)
0.1	4.15 $\pm$0.025	0.2	2.87 $\pm 1.72 \times 10^{-5}$
0.5	2.87 $\pm 1.72 \times 10^{-5}$	0.7	2.87 $\pm 5.74 \times 10^{-5}$
0.9	2.87 $\pm$0.017	0.98	2.69 $\pm$1.4
0.9 and decreasing	2.87 $\pm 5.17 \times 10^{-3}$	0.98 and decreasing	2.13 $\pm$0.11

Table 1: Estimators ($\times 10^{25}$) and 95% confidence intervals for different choices of step-sizes. The exact value of the probability, 2.87×10^{-25}, was determined using analytical methods in De Boer 2001.

5.1.2 Performance comparison between ASA, AMC and CE algorithms on a small network

De Boer et al. (2000) and De Boer (2001) propose various elegant modifications to their basic state dependent cross entropy algorithm that considerably improve the performance over the basic algorithm. These allow them to solve Jackson networks with over a million states. However, in this section we restrict comparison of ASA algorithm to the basic CE algorithm and the AMC algorithm for the following reasons:

1. The basic algorithms AMC, CE and ASA represent three substantially different approaches to adaptive simulation. Thus their comparison is insightful in identifying their relative advantages for various classes / sizes of problems.

2. The modifications suggested in the above references for the CE algorithm may, after suitable adjustments, be used in the ASA and the AMC algorithms as well: Performing a fair comparison that involves such modifications and further modifications such as function approximations (discussed in Section 6) is an enormous task that may be an object of future study.

3. The modifications proposed in the above references rely critically on the fact that queues in open Jackson networks behave like reflected random walks. Thus consider for example the set of states where a subset of queues is non-empty in a network. For all these states, the probabilistic jump structure is independent of the state. This allows for clever state aggregation techniques for updating the change of measure in an iteration of the CE method as proposed in the above references. However, for more general Markov chains, e.g., queueing networks where arrival and service rates are state dependent, such aggregation techniques may be less useful. The ASA and the AMC algorithms, however, do not appear to depend critically on such uniformity in the probabilistic structure of the Markov chain.

Similarly, Kollman et al. (1999) suggest some modifications to the basic AMC algorithm that include using 'path information'. This involves getting samples for cost till termination for all the states in I visited along a generated path. Note that these samples are no longer independent. Computationally this may be more efficient then generating independent paths from each state in I. They also suggest a modification whereby the estimate of the value function after iteration $n+1$ is taken to be the weighted average of the estimate after iteration n and the estimate generated at iteration $n+1$ (in the basic method, only the latter is considered). Along with these they also suggest other useful ideas. However, to maintain simplicity and focus, we implement only their basic algorithm.

To begin with, we assume that the initial distribution for all three algorithms is the original distribution of the network. Later, this is relaxed for the AMC and the CE algorithms. We compare the three algorithms for $N = 5$ and 12.

For implementing the AMC algorithm, we further need to decide the number of independent samples l to generate from each state at every iteration. We do this by trying values $l = 20, 50, 100, 200, \cdots$, till the best amongst these is identified. Unless reported otherwise, we use the best value. The minimum value of 20 is chosen so that reliable confidence intervals may be constructed. Every iteration starts by generating l independent samples from the state $[1, 0]$. We test to check if the generated 95% confidence interval length is within 0.1% of the estimated value, in which case the simulation terminates. The small value of 0.1% is chosen to avoid termination at incorrect estimated values. If this stopping criterion is not met, then we proceed to generate independent samples starting from each of the remaining interior states in order to complete the iteration.

For $N = 5$, we selected $l = 20$, and 8 iterations were conducted before the stopping condition was met. For $N = 12$, we note that for $l = 20, 100$ and $1,000$, the stopping criteria was not met even after two hours. The reason for this is that in the first iteration under the original measure, almost all samples generated from states in I hit state $[0, 0]$ before hitting

the prescribed threshold (recall that the probability of the latter equals 1.47×10^{-11}). After the first iteration, the probability of hitting the state $[0, 0]$ from any state in I becomes zero (since we update $J^{(n)}([0, 0])$ to zero for all $n \geq 1$, and this is used in updating the transition probabilities). Thus from the second iteration onwards, from any starting state a single replication will terminate only if the threshold is reached. In our simulation, we note that in the second iteration, even the first replication starting from state $[1, 0]$ did not hit the prescribed threshold even once in two hours of computation time (after 8.9×10^9 transitions).

We implement the state dependent version of the CE algorithm described in Section 2.2 (see De Boer 2001, Chapter 8. As noted in this reference, the state independent CE method fails miserably for the example considered. The reference also shows examples where the state independent CE method performs remarkably well.) Again, we stop when after an iteration the generated 95% confidence interval is within 0.1% of the actual value. For $N = 5$, we use 10^4 replications in every iteration of the CE algorithm. With 10^3 replications, the algorithm stopped at a wrong estimated value of the probability (its estimate was 1.7×10^{-4}, while the actual value is 2.17×10^{-4}), while for 10^5 and 10^6 replications, the time taken to meet the stopping criterion was larger. For $N = 12$, greater than 10^{11} replications are needed on an average to observe the desired rare event under the original probability measure, while it took two hours to generate about 3.85×10^8 replications and no such event was observed.

The ASA algorithm is implemented with $a = 0.5$. Twenty independent runs are conducted to construct confidence intervals. The number of transitions are selected so that the bias becomes negligible. Table 2 reports the results under the three approaches. Note that the ASA algorithm converges very quickly even for $N = 12$.

N		ASA Method	AMC Method	CE Method
5	Prob. est. $\times 10^4$	$2.17 \pm 1.09 \times 10^{-6}$	$2.17 \pm 4.34 \times 10^{-4}$	$2.17 \pm 9.8 \times 10^{-4}$
	Transitions	10^5	2.00×10^5	4.6×10^5
	CPU (in sec.)	0.10	0.15	0.61
12	Prob. est. $\times 10^{11}$	$1.47 \pm 2.79 \times 10^{-4}$	no convergence	no convergence
	Transitions	5×10^5	9.7×10^9	1.56×10^9
	CPU (in sec.)	0.47	7,200	7,200

Table 2: Comparison of three approaches when initial probability measure is the original one for all three methods. The range displayed along with the point estimator corresponds to 95% confidence interval. The exact value of the probability is numerically determined in De Boer (2001). For $N = 5$ and 12, the respective values are: 2.17×10^{-4} and 1.47×10^{-11}.

As mentioned in De Boer (2001), for successful implementation, the CE algorithm typically requires an initial change of measure where the network is unstable. The same may be expected to be true for the AMC algorithm. Hence we also perform comparisons wherein the CE algorithm and the AMC algorithm begin simulation with the change of measure proposed in De Boer (2001), under which the network is unstable. Under this measure, the arrival rate to the first queue equals 0.522 and the service rates to the two queues are respectively $(0.412, 0.066)$. As recommended in De Boer (2001), we use 10^4 replications in every iteration of their algorithm.

Table 3 shows the comparison between the three methods for $N = 12, 50, 100$ and 150. In the AMC algorithm, the number of samples generated per state per iteration and the

N		ASA Method	AMC Method	CE Method
12	Prob. est. $\times 10^{11}$	$1.47 \pm 2.79 \times 10^{-4}$	$1.47 \pm 2.06 \times 10^{-4}$	$1.47 \pm 2.5 \times 10^{-4}$
	Transitions	5×10^5	9.9×10^4	6.2×10^5
	CPU (in sec.)	0.47	0.08	1.06
50	Prob. est. $\times 10^{52}$	$6.03 \pm 4.94 \times 10^{-8}$	$6.03 \pm 2.29 \times 10^{-3}$	$3.11 \pm 8.7 \times 10^{-4}$
	Transitions	4×10^7	1.35×10^7	7×10^8
	CPU (in sec.)	35.9	9.47	$3,600$
100	Prob. est. $\times 10^{105}$	$1.33 \pm 6.5 \times 10^{-9}$	$1.33 \pm 5.59 \times 10^{-4}$	no convergence
	Transitions	2×10^8	3.8×10^8	4.6×10^8
	CPU (in sec.)	193	306	$7,200$
150	Prob. est. $\times 10^{159}$	$2.19 \pm 5.9 \times 10^{-4}$	$2.19 \pm 1.75 \times 10^{-3}$	no convergence
	Transitions	6×10^8	3.1×10^9	3.4×10^8
	CPU (in sec.)	612	$2,825$	$7,200$

Table 3: Comparison of three approaches when initial probability measure is the original one for the ASA algorithm while it is the unstable one for the AMC and the CE algorithm. The range displayed along with the point estimator corresponds to 95% confidence interval. The exact value of the probability is numerically determined in De Boer (2001). For $N = 12, 50, 100$ and 150, the respective values are: 1.47×10^{-11}, 6.03×10^{-52}, 1.33×10^{-105} and 2.19×10^{-159}.

number of iterations till the convergence criterion is met equal (20,7),(20, 14) (100, 9) and (200,12), for $N = 12, 50, 100$ and 150, respectively. The CE method required 3 iterations for buffer size 12. For $N = 50$, about 800 iterations were conducted, although the result did not converge to the correct value.

It can be seen that while the AMC algorithm performs better for smaller N, the ASA algorithm remains the most viable method for large N. Note that the ASA algorithm may also be further improved by selecting initial value functions carefully (e.g., by conducting an iteration of the AMC algorithm under an unstable measure); however, in such attempts the simplicity of implementation of the proposed method is lost. Also note that the average computer time required per transition is the highest for the CE method. This is due to the large computational effort needed in the CE method to update the transition probabilities after every iteration.

5.1.3 Performance comparison between ASA, AMC and CE algorithms on a larger network

We now consider the probability of the total queue length reaching a threshold N in a busy cycle of a larger 5 queue open Jackson network with single servers at each queue. To aid in validating the results, we consider the network considered in Section 8.3.5 of De Boer (2001). This network has an external arrival rate of 3 to queue 1. Other queues do not have external arrivals. The service rates $(\mu_1, \ldots, \mu_5)$ equal $(40, 20, 50, 50, 60)$. From the first queue with probability 1/2 a customer departs to queue 2 and with probability 1/2 it departs to queue 3. From queue 2 with probability 1/2 a customer departs to queue 5 and with probability

Buffer Size		ASA Method	AMC Method	CE Method
5	Prob. est. $\times 10^3$	$5.67 \pm 1.42 \times 10^{-4}$	$5.67 \pm 1.71 \times 10^{-3}$	$5.67 \pm 2.27 \times 10^{-3}$
	Transitions	4×10^6	1.7×10^6	5.9×10^6
	CPU (in sec.)	10.5	3.6	$1,106$
10	Prob. est. $\times 10^7$	$6.14 \pm 6.14 \times 10^{-5}$	no convergence	no convergence
	Transitions	2×10^8	1.8×10^{10}	1.9×10^7
	CPU (in sec.)	552	$36,000$	$36,000$
20	Prob. est. $\times 10^{16}$	$7.79 \pm 1.17 \times 10^{-3}$	no convergence	no convergence
	Transitions	2×10^9	1.4×10^{10}	8.6×10^6
	CPU (in sec.)	$6,593$	$36,000$	$36,000$

Table 4: Comparison of three approaches for a large network when initial probability measure is the original one for all three methods. The range displayed along with the point estimator corresponds to 95% confidence interval.

1/2 it returns to queue 1. From queue 3 with probability 1 the customer goes to queue 4 and from queue 4 with probability 1 it goes to queue 5. From queue 5 the customer leaves the system with probability 1/2 and returns to queue 3 with probability 1/2. Under these parameters each queue has a traffic intensity equal to 0.1. Figure 2 graphically illustrates this network. As noted by De Boer (2001), state-independent methods do not work well with this problem.

Again we compare the three algorithms for this example and we set the initial distribution to the original distribution in the three cases. We compare the algorithms for the total queue length threshold $N = 5, 10$ and 20. It is easy to see that the number of states so that the total network population equals j in a network with q queues is given by $\frac{(j+q-1)!}{j!(q-1)!}$. Using this, we compute that the Markov chain has $127,2003$ and $42,505$ states, respectively for $N = 5, 10$ and 20 (considering the set of overflow states as a single state). Table 4 reports the results under the three approaches.

The methodology for the three approaches is similar to that described in Section 5.1.2. Under the AMC approach, for $N = 5$, we selected $l = 50$, and 11 iterations were conducted before the stopping condition is met (again these were the best values compared to those obtained with $l = 20$ and 100). For $N = 10$ and 20 for $l = 20, 100$ and $1,000$, the stopping criteria was not met even after 10 hours. The reasons were similar to those given in Section 5.1.2 for a smaller example.

As in Section 5.1.2, we implement the state dependent version of the CE algorithm for this example as well. For $N = 5$, we use 10^5 replications per iteration. Again, this performed better than the cases where 10^4 and 10^6 replications were used. Again for $N = 10$ and 20 no convergence was seen in 10 hours for reasons similar to those given in Section 5.1.2.

For ASA method we generate 20 independent runs. The step a is set to 0.5. Note that for $N = 20$ the resulting estimate of the probability and the 95% confidence interval equalled $7.79 \times 10^{-16} \pm 1.17 \times 10^{-3}$%. This is within the confidence interval reported by De Boer (2001), where the estimate was less precise because the confidence width was wider.

De Boer (2001) also considers N equal to 50 and 100 and uses various sophisticated

techniques to handle that situation. However, for $N = 50$ the $|I|$ is close to 3.2 million; hence the storage requirements make it difficult to implement the proposed methodology in its present form (see the discussion in Section 6).

5.1.4 Performance comparison between ASA algorithm and deterministic numerical methods

A well known 'rule of thumb' is that the Monte-Carlo based methods provide quicker ball park estimates compared to the deterministic methods when the state space is large, and therefore in such settings, are a viable alternative. In Ahamed, Borkar and Juneja (2005), an online companion to this paper, we illustrate the validity of this rule for ASA algorithm vis-a-vis the value iteration method (a deterministic iterative procedure to solve the 'fixed point' linear equations satisfied by the value function J^*, see Bertsekas and Tsitsitklis 1996) through an example. The results there suggest that the value iteration method performs better if greater accuracy is required, while the ASA algorithm gives quicker ball-park estimates. Also, as the state-space increases, the ASA algorithm improves in comparison to the value iteration method.

Note that the deterministic methods such as value iteration, converge to the correct value at an exponential rate in terms of iterates required (see Golub and Van Loan 1996 for more advanced deterministic methods such as the Successive Over-Relaxation method). Nevertheless, the convergence can be slow in practice since each iteration requires $O(d^2)$ computations (recall that $d = |S|$, also, without loss of generality we may take $|T| = 1$ so that $|I| = d - 1$). This $O(d^2)$ computation corresponds to the number of positive entries in the transition matrix and may be reduced when the transition matrix of the Markov chain is sparse (for example, in Jackson networks the transition matrix of the associated Markov chain has $O(dq^2)$ positive entries where q denotes the number of queues in the network, and thus computations per iteration are $O(dq^2)$). On the other hand, the ASA algorithm requires $O(d)$ operations per transition (this reduces to $O(q^2)$ for Jackson networks). However, note that under value iteration the value function corresponding to each state is updated while under the ASA algorithm the value function corresponding to a single state (from which the transition is made) is updated. Thus, the number of transitions required under the ASA algorithm may be large; however, their dependence upon d cannot be captured in any transparent way. Note that both the AMC algorithm and the state-dependent CE algorithm also require $O(d)$ operations per transition. In these cases as well, the dependence of number of transitions needed on d is not straightforward.

Also note that for general Markov chains, $O(d^2)$ memory is needed to store the original transition matrix. For specialized Markov chains such as those associated with Jackson networks, the original probability transition matrix need not be stored. In deterministic numerical methods such as value iteration, and in the ASA and the AMC method, $O(d)$ memory is required to store the estimate of value function. In the ASA and the AMC method, the importance sampling transition probabilities can then be generated using this estimate. However, in these methods, this may lead to some degradation in performance vis-a-vis storing importance sampling probabilities (that takes $O(d^2)$ storage), as in the former case, $O(d)$ computational effort is needed to generate the importance sampling transition probabilities at every transition, while in the latter case, these may be generated in $O(1)$ computations (as discussed for the ASA method in Section 2.3; for the AMC method these

probabilities are computed only once in each iteration at its beginning). In the state dependent CE method, $O(d^2)$ memory is required to store the importance sampling transition probabilities.

Also note that Kuruganti and Strickland (1996) suggest running a few iterations of a deterministic numerical method such as value iteration to estimate the value function and then using the associated estimate of the zero variance measure as the static importance sampling distribution to further refine estimates of value function via simulation.

5.2 Mean Time to Hit a Rare Set

Buffer Size		ASA Method	AMC Method
3	mttf	$4.72 \pm 1.8 \times 10^{-3}$	$4.72 \pm 2.12 \times 10^{-3}$
	Transitions	1×10^8	2.2×10^8
	CPU (in sec.)	275	309
5	mttf	$99.05 \pm 1.48 \times 10^{-2}$	no convergence
	Transitions	2×10^8	1.1×10^{10}
	CPU (in sec.)	548	$36,000$
6	mttf	$540.01 \pm 5.4 \times 10^{-2}$	no convergence
	Transitions	2×10^9	1.8×10^{10}
	CPU (in sec.)	$5,880$	$36,000$
7	mttf	$3199.4 \pm 3.2 \times 10^{-1}$	no convergence
	Transitions	2×10^{10}	4.4×10^{10}
	CPU (in sec.)	$54,660$	$72,000$

Table 5: Comparison of ASA and AMC methods in estimating the mean time to hit a rare set for the 5-queue open Jackson network example. The range displayed along with the point estimator corresponds to 95% confidence interval.

In the previous section we used the three methods to estimate certain probabilities of rare events. We now apply the ASA and the AMC method to estimate a more general performance measure.

Again consider the five queue network example described in Section 5.1.3. For this example, we now consider the problem of determining the mean time to hit the set of states corresponding to the total queue length at the five queues equal to N for $N = 3, 5, 6$ and 7 (call these set of states R_N; let A denote the set containing a single 'all queues empty' state). We restrict ourselves to small values of N as the computational time requirement grows very fast with N (we give a heuristic explanation for this later). In this setting, $g(x, y)$ denotes the mean time spent in state x before jumping to state y. This is well known to be independent of y and is given by:

$$g(x, y) = \frac{1}{\sum_{i=1}^{5}(\lambda_i + \mu_i I(x_i > 0))},$$

where $x = (x_i : i \leq 5)$.

In the AMC method for $N = 3$ the best performance was obtained for $l = 50$. Here 9 iterations were conducted before the stopping condition (same as in the previous examples) was met. For $N = 5, 6$ and 7, we tried $l = 50, 100$ and 200. The estimator did not meet the stopping criteria even in ten hours for $N = 5, 6$ and for twenty hours for $N = 7$.

For the ASA method, we conduct 20 independent runs for each value of N. The step-size a is again set to 0.5. The initial value $J_0(x)$ for all x not in R_N is kept at 100 (we also tried initial values set to $1, 1000$ and $10,000$ for each x not in R_N. The results were quite insensitive to these values). $J_0(x)$ is fixed at zero for states $x \in R_N$. We start our algorithm with all queues empty. Every time the total queue length equals N, the simulation is reset to A. The results are shown in Table 5. While the ASA method performs better than the AMC method, especially for $N \geq 5$, its performance appears to deteriorate quickly with increasing N. This differs from its behavior in estimating the probability of hitting R_N before hitting A, shown in Table 4, where the corresponding probability for $N = 20$ was estimated in a reasonable time.

Note that an alternate method to estimate mean time to hit R_N exists that involves efficient estimation of the probability of hitting R_N before hitting A. To see this, let T_N denote the time to hit the set R_N starting from an empty network. Let T_0 denote the length of the busy cycle of the queuing network, i.e., starting from an empty network, time for it to re-empty after an arrival takes place. It is well known that (see, e.g., Heidelberger 1995)

$$ET_N = \frac{E\min(T_0, T_N)}{P(T_N < T_0)}. \tag{15}$$

Using this ratio representation we validate our estimate for $N = 5$ in Table 5. Note that $E\min(T_0, T_N)$ may be easily estimated via naive simulation, while the ASA method may be used to estimate $P(T_N < T_0)$. As reported in Table 5, the direct application of the ASA method to estimate ET_5 provides an approximate 95% confidence interval $99.05 \pm 1.48 \times 10^{-2}$ in 548 seconds. We estimate $E\min(T_0, T_5)$ using naive simulation by generating 10^6 independent samples of $\min(T_0, T_5)$. This took 21 seconds. The approximate 97.5% confidence interval of $E\min(T_0, T_5)$ equals $0.5617 \pm 4.11 \times 10^{-4}$. From Table 4 it is easy to infer that that the approximate 97.5% confidence interval for $P(T_5 < T_0)$ equals $5.67 \times 10^{-3} \pm 1.62 \times 10^{-7}$, and this is estimated in 10.5 seconds. Thus, the point estimator of ET_5 from the ratio representation equals 99.065 and is within acceptable error limit of the estimator from the direct application of the ASA method. To see that the estimator from the ratio representation does not have very high variability, note that a conservative confidence interval for it may be easily developed using the following observation: Consider independent random variables X and Y and suppose that there exist positive constants $\alpha, a, \epsilon, b, \delta$ such that $a > \epsilon$, $b > \delta$, $P(X \in (a \pm \epsilon)) = \alpha$ and $P(Y \in (b \pm \delta)) = \alpha$. Then

$$P(\frac{X}{Y} \in (\frac{a-\epsilon}{b+\delta}, \frac{a+\epsilon}{b-\delta})) \geq \alpha^2.$$

In our settings, by selecting $\alpha = 0.975$, and noting that $\alpha^2 \approx 0.95$, an approximate and conservative confidence interval for ET_5 using the ratio representation equals $(98.99, 99.14)$. Asymptotically more accurate confidence interval may be developed using the techniques analogous to those in the regenerative settings, where such ratio representations arise (see, e.g., Glynn and Iglehart 1993). However, we avoid this as some adjustments to the standard analysis may be needed in our settings (to account for small value of the denominator and

the different number of samples used in estimating the numerator and the denominator). These experiments suggest that for larger values of N, e.g., $N = 20$, under the ASA method, the ratio representation is better suited for efficient estimation of ET_N vis-a-vis its direct estimation.

A heuristic justification for the better performance of the ASA method in learning $P(T_N < T_0)$ vis-a-vis its directly learning ET_N is as follows: It is reasonable to suppose that visits to R_N provide useful information for updating J, the estimates of value functions, towards their true values, J^*. Recall that in estimating $P(T_N < T_0)$, $g(x, y) = 0$ except when $y \in R_N$, in which case $g(x, y) = 1$. Also note that for $x, y \in I$, if $y \geq x$ componentwise, then $J^*(y) \geq J^*(x)$. In view of this and the representation of the transition probabilities associated with the zero variance estimator in (3), it is reasonable to expect that as the estimates J improve, the probability of hitting the rare set in a busy cycle may increase, leading to faster learning of useful information. Now consider the case of direct estimation of ET_N. Here, $g(x, y)$ is independent of y, and for $x, y \in I$, if $y \geq x$ componentwise, then $J^*(y) \leq J^*(x)$. Thus, it is reasonable to expect that as the estimates J improve, the probability of hitting the rare set in a busy cycle may reduce, leading to slower learning of useful information.

Empirically, we illustrate this via the following experiment. We run the ASA method for the two separate cases, one corresponding to learning $P(T_N < T_0)$ (as in Section 5.1.3) and the other to directly learning ET_N (as in this section) for a sufficiently long time so that the estimated J in the two cases are close to the corresponding J^* (10^6 transitions were generated in the first case and 10^8 transitions in the second). Then, in both the cases we count the number of transitions to reach R_N starting from A. We consider $N = 5$. In the case of learning $P(T_N < T_0)$, the approximate 95% confidence interval for the expected number of transitions to hit R_N starting from A, from $1,000$ observations, equalled 28.38 ± 1.06 (this confidence interval is valid under the assumption that the observations are i.i.d. This would be true if the zero variance transition probabilities were used to generate samples; in our case this is approximately correct. However, this issue is minor relative to the point considered). In the case of directly learning ET_N, the approximate 95% confidence interval, from $1,000$ observations, equalled $1.36 \times 10^7 \pm 8.43 \times 10^5$ (enormously larger than in the former case). Interestingly, under the naive probability measure, the approximate 95% confidence interval, from $1,000$ observations, equalled $2,384 \pm 146$.

As the results of the AMC algorithm illustrate, this drawback may be valid for all schemes that attempt to learn and use the zero-variance transition probabilities in simulation. One approach to ameliorate this drawback may be to use the ratio representation based on regenerative theory, as in (15), to develop an estimate. Alternatively, one may consider using a change of measure that emphasizes the paths to the rare event and keep this measure fixed as the simulation continues to learn the value function at every transition (as in Randhawa and Juneja 2004). However, further research is needed to analyze this drawback carefully and come up with a good solution.

6 Conclusion

In this paper, we presented an easily implementable and efficient adaptive importance sampling technique for estimating common performance measures associated with finite state

Markov chains. A potential drawback of our approach is that it requires computation and storage of the estimated value function of each state generated via simulation. While this is an inevitable feature of any adaptive scheme as pointed out earlier, it could require enormous memory and computational effort when the state space is large with hundreds of thousands of states. One way to deal with this problem may be to use function approximation techniques to approximate the value function (see, e.g., Bertsekas and Tsitsiklis 1996). This issue will be further explored by the authors in a separate paper.

Acknowledgements: *The authors would like to thank the area editor, the associate editor and the referees for their inputs that led to considerable improvements in the paper.*

REFERENCES

Ahamed, T. P. I., Borkar, V. S., Juneja, S. 2005. Adaptive importance sampling technique for Markov chains using stochastic approximation. Online Companion *Operations Research* http://or.pubs.informs.org/Pages/collect.html.

Andradottir, S., D. P. Heyman and T. Ott. 1995. On the choice of alternative measures in importance sampling with Markov chains. *Operations Research* **43**, 3, 509-519.

Bertsekas, D. P. and J. N. Tsitsiklis. 1989. *Parallel and Distributed Computation*, Prentice Hall, Englewood Cliffs, New Jersey.

Bertsekas, D.P. and J.N. Tsitsiklis. 1996. *Neuro-Dynamic Programming*, Athena Scientific, Belmont, Massachusetts.

Booth, T. E. 1985. Exponential convergence for Monte Carlo particle transport. *Transactions of the American Nuclear Society* **50** 267-268.

Borkar, V. S. 1998. Asynchronous stochastic approximation. *SIAM Journal of Control and Optimization* **36**, 840-851.

Borkar, V. S. 2002. On the lock-in probability of stochastic approximation. *Combinatorics, Probability and Computing* **11**, 11-20.

Borkar, V. S. and S. P. Meyn. 2000. The O.D.E. method for convergence of stochastic approximation and reinforcement learning. *SIAM Journal of Control and Optimization* **38**, 447-469.

Borkar, V. S. and K. Soumyanath. 1997. An analog scheme for fixed point computation, Part 1: Theory. *IEEE Trans. on Circuits and Systems 1: Fundamental Theory and Appl.* **44**, 351-355.

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

Cao, X. R. and Chen, H. F. 1997. Perturbation realization, potentials, and sensitivity analysis of Markov processes. *IEEE Trans. on Automatic Control* **42**, 1382-1393.

De Boer, P.T. 2001. *Analysis and Efficient Simulation of Queueing Models of Telecommunication Systems.* Ph. D. Thesis, University of Twente.

De Boer, P.T., V. F. Nicola and R. Y. Rubinstein. 2000. Adaptive importance sampling simulation of queueing networks. *Proceedings of 2000 Winter Simulation Conference*, 646-655.

Desai, P. Y. and P. W. Glynn. 2001. A Markov chain perspective on adaptive Monte Carlo algorithms. *Proceedings of 2000 Winter Simulation Conference*, 379-384.

Derevitskii, D. P., and Fradkov, A. L. 1974. Two models for analyzing the dynamics of adaptation algorithms. *Automation and Remote Control* **35**, 59-67.

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

Glynn, P.W. and D.L. Iglehart. 1989. Conditions for the applicability of the regenerative method. *Management Science* **39**, 1108-1111.

Golub, G. H., and C. F. Loan, Matrix Computations, 3rd. edition, Johns Hopkins, 1996.

Fishman, G. S. 1996. *Monte Carlo: Concepts, Algorithms, and Applications*, Springer, New York.

Heidelberger, P. 1995. Fast simulation of rare events in queuing and reliability models. *ACM Transactions on Modeling and Computer Simulation* **5**, 1, 43-85.

Hsieh, M. and P. W. Glynn. 2002. Confidence regions for stochastic approximation algorithms. *Proc. 2002 Winter Simulation Conference*, 370-376.

Juneja, S. 2003. Efficient rare event simulation using importance sampling: an introduction. *Computational Mathematics, Modelling and Algorithms* (J. C. Misra, ed.), Narosa Publishing House, New Delhi. 357-396.

Juneja, S., and V. Nicola. 2003. Efficient simulation of buffer overflow probabilities in Jackson networks with feedback. Submitted for publication.

Kollman, C., K. Baggerly, D. Cox and R. Picard. 1999. Adaptive importance sampling on discrete Markov chains. *Annals of Applied Probability* **9**, 391-412.

Konda, V. R. and Borkar, V. S. 1999. Actor-critic-type learning algorithms for Markov decision processes. *SIAM Journal of Control and Optimization* **38**, 94-123.

Kuruganti, I. and S. Strickland. 1996. Importance sampling for Markov chains: computing variance and determining optimal measures. *Proc. 1996 Winter Simulation Conference*, 273-280.

Kuruganti, I. and S. Strickland. 1995. Optimal importance sampling for Markovian systems. *Proc. of the 1995 IEEE Conf.on Systems, Man and Cybernetics.* 195-200.

Kushner, H. J. and G. Yin. 1997. *Stochastic Approximation Algorithms and Applications.* Springer Verlag, New York, 1997.

Precup, D., Sutton, R. S. and S. P. Singh. 2000. Eligibility traces for off-policy policy evaluation. *Proc. of the 17th Intl. Conf. on Machine Learning (ICML'00)*, Morgan Kaufman, 759-766.

Randhawa, R. S. and S. Juneja. 2004. Simulating rare events by combining temporal difference methods and importance sampling. *ACM TOMACS*, **14**, 1-30.

Singh, S. and P. Dayan. 1998. Analytical Mean Squared Error Curves for Temporal Difference Learning. *Machine Learning* **32**, 5-40.

Su Y. and Fu M. C. 2000. Importance sampling in derivatives security pricing. *Proc. of the 2000 Winter Simulation Conference*, J. A. Joines, R. R. Barton, K. Jang, P. A. Fishwick (eds.)

Su Y. and Fu. M. C. 2002. Optimal importance sampling in securities pricing. *Journal of Computational Finance* **5**, 27-50.

Tsitsiklis, J. T. 1994. Asynchronous stochastic approximation and Q-learning. *Machine Learning* **16**, 185-202.

Vázquez-Abad F. J. and Dufresne D. 1998. Accelerated simulation for pricing Asian options. *Proc. of the 1998 Winter Simulation Conference*, D. J. Medeiros, E. F. Watson, J. S. Carson and M. S. Manivannan (eds.)

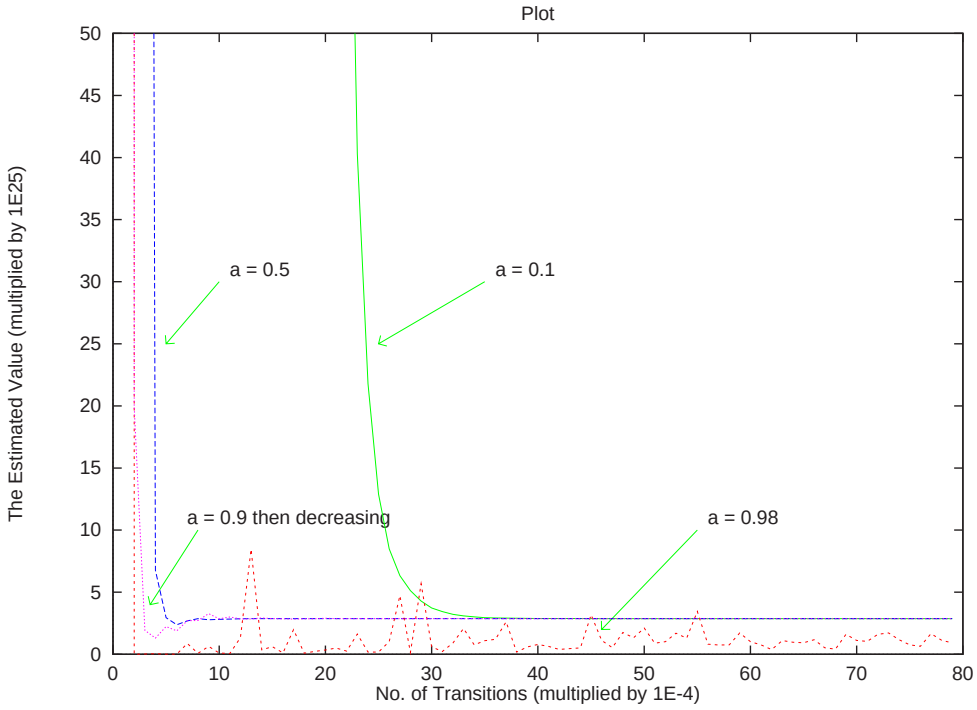

Figure 1: Plot of estimates of probability that the total queue length reaches $N = 25$ in a busy cycle for various step-sizes

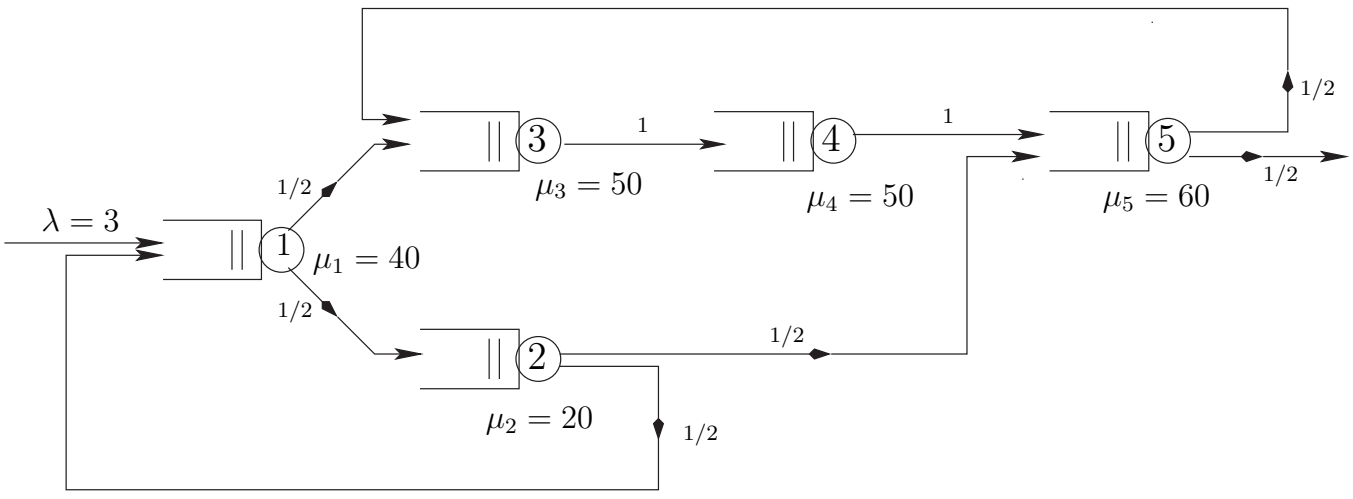

Figure 2: A five-queue open Jackson network