A Simulation-Based Algorithm for Ergodic Control of Markov Chains Conditioned on Rare Events

S. Bhatnagar [*†], V. S. Borkar [‡] and A. Madhukar [§]

February 2006

Abstract

We study the problem of long-run average cost control of Markov chains conditioned on a rare event. In a related recent work, a simulation based algorithm for estimating performance measures associated with a Markov chain conditioned on a rare event has been developed. We extend ideas from this work and develop an adaptive algorithm for obtaining, online, optimal control policies conditioned on a rare event. Our algorithm uses three timescales or step-size schedules. On the slowest timescale, a gradient search algorithm for policy updates that is based on one-simulation simultaneous perturbation stochastic approximation (SPSA) type estimates is used. Deterministic perturbation sequences obtained from appropriate normalized Hadamard matrices are used here. The fast timescale recursions compute the conditional transition probabilities of an associated chain by obtaining solutions to the multiplicative Poisson equation (for a given policy estimate). Further, the risk parameter associated with the value function for a given policy estimate is updated on a timescale that lies in between the two scales above. We briefly sketch the convergence analysis of our algorithm and present a numerical application in the setting of routing multiple flows in communication networks.

Key Words: Markov decision processes, optimal control conditioned on a rare event, simulation based algorithms, SPSA with deterministic perturbations, reinforcement learning.

1 Introduction

Markov decision processes (MDPs) [5], [35] form a general framework for studying problems of control of stochastic dynamic systems (SDS). Many times, one encounters situations involving control of SDS conditioned on a rare event of asymptotically zero probability. This could be, e.g., a problem of damage control when faced with a catastrophic event. For instance, in the setting of a large communication network such as the internet, one may be interested in obtaining optimal flow

[*]Corresponding author

[†]Department of Computer Science and Automation, Indian Institute of Science, Bangalore 560 012, India. E-Mail: shalabh@csa.iisc.ernet.in

[‡]School of Technology and Computer Science, Tata Institute of Fundamental Research, Homi Bhabha Road, Mumbai 400 005, India. E-Mail: borkar@tifr.res.in

[§]Department of Electrical Engineering, Indian Institute of Science, Bangalore 560 012, India. E-Mail: madhukar@ee.iisc.ernet.in

and congestion control or routing strategies in a subnetwork given that an extremal event such as a link failure has occurred in another remote subnetwork. Our objective in this paper is to consider a problem of this nature wherein a rare event is specifically defined to be the time average of a function of the MDP and its associated control-valued process exceeding a threshold that is larger than its mean. We consider the infinite horizon long-run average cost criterion for our problem and devise an algorithm based on policy iteration for the same.

Research on developing simulation based methods for control of SDS has gathered momentum in recent times. These largely go under the names of neuro-dynamic programming (NDP) [7] or reinforcement learning (RL) [39] and are applicable in the case of systems for which model information is not known or computationally forbiddingly expensive, but output data obtained either through a real system or a simulated one is available. Our problem does not share this last feature, but we do borrow certain algorithmic paradigms from this literature. Before we proceed further, we first review some representative recent work along these lines. In [3], an algorithm for long-run average cost MDPs is presented. The average cost gradient is approximated using that associated with a corresponding infinite horizon discounted cost MDP problem. The variance of the estimates however increases rapidly as the discount factor is brought closer to one. In [4], certain variants based on the algorithm in [3] are presented and applications on some experimental settings shown. In [25], a perturbation analysis (PA) type approach is used to obtain the performance gradient based on sample path analysis. In [24], a PA-based method is proposed for solving long-run average cost MDPs. This requires keeping track of the regeneration epochs of the underlying process for any policy and aggregating data over these. The above epochs can however be very infrequent in most real life systems. In [32], the average cost gradient is computed by assuming that sample path gradients of performance and transition probabilities are known in functional form. Amongst other RL-based approaches, the temporal difference (TD) [39] and Q-learning [42] have been popular in recent times. These are based on value function approximations. A parallel development is that of actor-critic algorithms based on the classical policy iteration algorithm in dynamic programming. Note that the classical policy iteration algorithm proceeds via two nested loops – an outer loop in which the policy improvement step is performed and an inner loop in which the policy evaluation step for the policy prescribed by the outer loop is conducted. The respective operations in the two loops are performed one-after-the-other in a cyclic manner. The inner loop can in principle take a long time to converge, making the overall procedure slow in practice. In [29], certain simulation-based algorithms that use multi-timescale stochastic approximation are proposed. The idea is to use coupled stochastic recursions driven by different step-size schedules or timescales. The recursion corresponding to policy evaluation is run on the faster timescale while

that corresponding to policy improvement is run on the slower one. Thus while both recursions proceed simultaneously, the algorithm converges to the optimal policy. The algorithms of [29] (as with those described in the previous paragraph) are for finite state and finite action MDPs, under both the discounted and long-run average cost criteria. A variant of the above algorithms for the case of finite state but compact (non-discrete) action sets, in the setting of infinite horizon discounted cost MDPs is presented in [13], and performs gradient search in the space of stationary deterministic policies using a simultaneous perturbation stochastic approximation (SPSA) gradient estimate.

Standard SPSA [37] uses two simulations for estimating the performance/cost gradient regardless of the dimension N of the parameter vector, unlike Kiefer-Wolfowitz (K-W) based estimates that require $(N+1)$ simulations for the same. This it does by randomly perturbing all parameter components at each update epoch. The original SPSA algorithm [37] is, however, a one-timescale Robbins-Monro variant for parameter optimization and is not directly applicable when the cost to be optimized is for instance the long-run average of a running cost function, viz., the objective function for a given parameter value is derived only after viewing the entire sample path / trajectory of the system for that parameter value. Perturbation analysis (PA) schemes [26], [28] that were proposed for problems such as these use largely one simulation, however, they require certain constraining regularity conditions on the system dynamics and cost functions in order to allow for an interchange between the 'gradient' and 'expectation' operators. Moreover, many of these schemes update parameters only at certain regeneration epochs of the underlying process, making them slow in practice. In [8] and [9], certain two-timescale stochastic approximation algorithms were introduced as alternatives to PA type schemes. These do not require constraining regularity conditions like PA, while they also update parameters at certain deterministic epochs. The key in the algorithms of [8] and [9] is the use of two-timescale stochastic approximation, whereby on the faster timescale, data corresponding to a given parameter update is aggregated and on the slower timescale, the parameter is updated. These algorithms, however, use K-W estimates. In [11], variants that use SPSA estimates were proposed and were found to show significantly better performance. In [38], a one-simulation (one-timescale) variant of the original SPSA algorithm was proposed, which however does not show good performance because of the presence of an 'additional' bias term in its gradient estimate whose contribution to overall bias tends to be high. In [12], it was observed in a similar setting as [8], [9] and [11] that the use of deterministic perturbation sequences (instead of randomized) derived using normalized Hadamard matrices significantly alleviates this problem in the case of one-simulation SPSA with the latter subsequently showing good performance. It was shown that perturbation sequences derived using normalized Hadamard

matrices satisfy the desired properties on such sequences that result in all bias terms getting cancelled at regular intervals. Further, the space of perturbations derived as above has a cardinality of $2^{\log_2(N+1)}$ as against 2^N when randomized perturbations are used (the perturbation vectors in both spaces being $\{\pm 1\}^N$-valued). To sum up, the use of normalized Hadamard matrix based perturbations in the setting as described above has the inherent advantage that one may use a fast one-simulation SPSA based algorithm that updates all parameter components at each update epoch (the epochs themselves being deterministically spaced. In particular, the algorithms in [12] update the parameter once every L epochs for a given, arbitrarily chosen integer L) while working with a more general class of systems than what the PA based methods allow.

The works cited above represent some recent developments in the general area of simulation based optimization and control of SDS. We now review some of the work that is more directly related to the problem we study in this paper. In [21], a simulation-based algorithm for estimating performance measures of a Markov chain conditioned on a rare event of zero probability has been developed. This is based on the result that the transition probabilities of the Markov chain conditioned on a rare event as above are the same as those of another irreducible chain on the same state space whose transition probabilities are absolutely continuous w.r.t. those of the former chain. The calculation of these calls for the solution of an associated *multiplicative Poisson equation*, an object familiar from risk-sensitive control and large deviations theory [33], [2]. The simulation based algorithm in [21] recursively obtains the solution to this multiplicative Poisson equation and uses the same to learn, online, the new transition probabilities. In [1], a reinforcement learning based importance sampling scheme for estimating expectations associated with rare events has also been proposed.

A related work is [36], in which a simulation based technique for optimizing certain performance measures in discrete event systems conditioned on rare events is presented. The problem there is formulated as a constrained optimization problem with an importance sampling estimate in the objective function that is obtained by assuming the underlying processes to be regenerative. The constraint there corresponds to the occurrence of the given rare event. The above problem is then solved as a two-stage stochastic programming problem. Our work is fundamentally different from [36] in many ways. First, we consider the problem of obtaining an optimal control policy conditioned on a rare event and not just one of optimizing certain performance metrics within a parameterized class as with [36]. Next, even though we assume that our underlying process for any given stationary policy is ergodic Markov and hence regenerative, we do not use the regenerative structure per se in obtaining estimates of performance as [36] does. For the latter, one needs in particular to keep track of regeneration epochs of the underlying process that can be very infrequent

in the case of most systems. Finally, we use a stochastic approximation based recursive procedure that incorporates reinforcement learning type estimates, unlike (as already mentioned) [36] that formulates the problem as a stochastic program.

Our work can be viewed as an extension of [21] that addresses the important problem of optimal control of a Markov chain conditioned on a rare event. In our framework, the results of [21] correspond to policy evaluation for a *fixed* stationary deterministic policy. We develop and use a simulation-based algorithm to find the *optimal* randomized policy 'on top of' the algorithm of [21]. Our algorithm uses three timescales or step-size schedules and iterates in the space of stationary randomized policies. The policy itself, however, is updated on the slowest timescale. The value function updates for finding the solution to the multiplicative Poisson equation for a given policy, based on which the transition probabilities of an associated chain are obtained, are performed on the fastest timescale. The risk parameter associated with the multiplicative Poisson equation is updated on a timescale that is faster than the one on which policy is updated, but slower than that on which value function is updated. Finally, there is another recursion that is used for averaging the cost function with the latter average used in the policy update step. This proceeds on the fastest scale as well (same as the one on which the value function is updated). We show in the analysis that the difference in timescales of the various recursions results in the desired algorithmic behavior. For policy updates, we use a one-simulation SPSA based recursion with normalized Hadamard matrices [12]. Finally, we present numerical experiments using our algorithm in the setting of routing multiple flows in communication networks conditioned on a rare event. We observe that our algorithm exhibits good performance in this setting. It must be noted here that adaptive importance sampling (IS) schemes require storage of transition probabilities and our algorithm is no different in this regard. Thus it may not be applicable (as is also the case with other IS methods) in scenarios that involve very large state spaces for which storage of such information is not possible. Nevertheless, feature based methods as in RL may still be applied for ease of computation in the case of problems with state and action spaces that are moderately large but for which storage of vectors of the size of state space is not a major concern. Further, in many cases such as queueing networks, the transition probabilities are easy to compute and transitions easy to simulate using simple local dynamic laws. In such scenarios, storage of transition probability matrices may also not be a major concern as these are known to be highly sparse.

The rest of the paper is organized as follows: Section 2 presents the problem formulation and gives the basic results. Section 3 presents the simulation-based algorithm. Its convergence analysis is also briefly sketched here. The numerical results are presented in Section 4. Finally, Section 5 presents the concluding remarks.

2 Problem Formulation and Basic Results

Consider a Markov decision process (MDP) $\{X_n,\ n \geq 0\}$ on a finite state space $S = \{1, 2, \ldots, s\}$. For $X_n = i$, $i \in S$, let $A(i)$ be the set of feasible controls or actions. We assume $A(i)$ has the form $A(i) = \{a_i^1, a_i^2, \ldots, a_i^{N_i}\}$. Let $A = \cup_{i \in S} A(i)$ denote the action space (which is also finite). Let $\{Z_n,\ n \geq 0\}$ denote the associated control-valued sequence such that $Z_n \in A(X_n)\ \forall n$. Suppose $p(i, j, a)$ denotes the transition probability from state i to state j under action $a \in A(i)$. Then the evolution of $\{X_n\}$ is governed by

$$Pr(X_{n+1} = j \mid X_n = i, Z_n = a, X_{n-1} = i_{n-1}, Z_{n-1} = a_{n-1}, \ldots, X_0 = i_0, Z_0 = a_0) = p(i, j, a),$$

for any $i_0, \ldots, i_{n-1}$, i, j, $a_0, \ldots, a_{n-1}$, a, in appropriate sets.

A sequence of functions $\pi = \{\mu_1,\ \mu_2, \ldots\}$ with each $\mu_n : S \rightarrow A$, $n \geq 1$, is said to be an admissible policy if $\mu_n(i) \in A(i)$, $\forall i \in S$. This corresponds to the control choice $Z_n = \mu_n(X_n)\ \forall n$. An admissible policy $\pi = \{\mu_1,\ \mu_2, \ldots\}$ with each $\mu_n = \mu$, $n \geq 1$, is said to be a stationary deterministic policy (SDP). By a common abuse of notation, we simply refer to μ itself as the SDP. By a randomized policy (RP) ψ, we mean a sequence $\psi = \{\phi_1,\ \phi_2, \ldots\}$ with each $\phi_n : S \rightarrow \mathcal{P}(A)$, $n \geq 1$. Here $\mathcal{P}(A)$ is the set of all probability vectors on A such that for each $i \in S$, $n \geq 1$, $\phi_n(i) \in \mathcal{P}(A(i))$, with $\mathcal{P}(A(i))$ being the set of all probability vectors on $A(i)$. A stationary randomized policy (SRP) is an RP ψ for which $\phi_n(i) = \phi\ \forall n \geq 1$. By an abuse of notation, we refer to ϕ itself as the SRP. The $a-$th component of $\phi(i)$, $\phi(i)(a)$ is the probability of choosing action a when in state i. Thus this corresponds to picking Z_n with probability distribution $\phi(X_n)$ at time n, independent of all other random variables realized till n. We have

Assumption (A) Under any SDP μ, the process $\{X_n\}$ forms an irreducible Markov chain.

Let $E_\mu[\cdot]$ denote the expectation w.r.t. the stationary distribution of $\{X_n\}$ under SDP μ. Let $g : S \times A \rightarrow \mathcal{R}$ be a given function such that $E_\mu[g(X_n, \mu(X_n))] < \alpha < \infty$ for a given constant α, for every SDP μ. The rare event that we consider corresponds to

$$\lim_{n \rightarrow \infty} \frac{1}{n} \sum_{m=0}^{n-1} g(X_m, \mu(X_m)) \geq \alpha.$$

The choice of the function $g(\cdot, \cdot)$ and α will be, in practice, dictated by the application. For example, in reliability, one may want to look at the stationary probability of crossing a very large threshold, say, N. Then $g(X_m, \mu(X_m))$ can be chosen to be $I\{X_m \geq N\}$, where $I\{\cdot\}$ is the indicator function and α could be a convenient upper bound on the stationary expectation.

Let $h : S \times A \times S \rightarrow \mathcal{R}$ denote the cost function that we assume is bounded. For any SDP μ,

let for any (initial state) $X_0 \in S$,

$$J(\mu) = \lim_{n\to\infty} \frac{1}{n} \sum_{m=0}^{n-1} h(X_m, \mu(X_m), X_{m+1})$$

be the long-run average cost. Let D be the set of all possible stationary deterministic policies. The aim is to find

$$\mu^* = \arg\min_{\mu\in D} J(\mu),$$

conditioned on the rare event $\lim_{n\to\infty} \frac{1}{n} \sum_{m=0}^{n-1} g(X_m, \mu(X_m)) \geq \alpha, \forall \mu \in D$. Let $p^{\mu,*}(i,j) = \lim_{n\to\infty} P(X_1 = j \mid X_0 = i, Z_0 = \mu(i), \frac{1}{n} \sum_{m=0}^{n-1} g(X_m, \mu(X_m)) \geq \alpha)$ denote the transition probabilities under SDP μ conditioned on a rare event (as defined above). We now present the basic results for a given SDP μ. These have been directly adapted from [21] for a fixed SDP and are stated here for the sake of completeness. Some of these results are also available in the context of risk sensitive control of Markov chains, see for instance, [2], [27], [33]. We briefly explain the risk sensitive control problem in order to put things in perspective. Suppose (that instead of the original) the aim is simply to find an SDP μ that minimizes $J_\zeta(\mu)$ defined by

$$J_\zeta(\mu) = \lim_{n\to\infty} \frac{1}{n} \ln \left(E\left[\exp(\sum_{m=0}^{n-1} \zeta g(X_m, \mu(X_m))) \right] \right),$$

where ζ denotes the risk parameter. The cases $\zeta > 0$ and $\zeta < 0$ correspond to the risk-averse and risk-preferring cases, respectively. For a given μ, $J_\zeta(\mu)$ is obtained [2], [27] as the solution to the multiplicative Poisson equation: For $i \in S$,

$$V_\zeta^\mu(i) = \frac{\exp(\zeta g(i, \mu(i)))}{\rho_\zeta^\mu} \sum_j p(i,j,\mu(i)) V_\zeta^\mu(j), \;\; i \in S, \tag{1}$$

where $V_\zeta^\mu(\cdot)$ is a bounded function (that is unique up to a multiplicative constant) and ρ_ζ^μ corresponds to $\exp(J_\zeta(\mu))$ or that $J_\zeta(\mu) = \ln \rho_\zeta^\mu$. Note that solution of this equation is an eigenvalue problem for the positive matrix $[[\exp(\zeta g(i,\mu(i)))p(i,j,\mu(i))]]_{i,j\in S}$, and V_ζ^μ, resp. ρ_ζ^μ, are its Perron-Frobenius eigenvector and eigenvalue.

For the problem considered in this paper, as shown in [21], the multiplicative Poisson equation also arises via the conditional transition probabilities $p^{\mu,*}(i,j)$ (for given SDP μ), see (2) below. In fact, for any given $i \in S$, upon summing over all $j \in S$ on both sides of (2), one obtains the multiplicative Poisson equation (1). For any SDP μ and risk parameter ζ, $J_\zeta(\mu) = \ln \rho_\zeta^\mu$ corresponds to the infinite horizon risk-sensitive cost. As in [21], we fix the choice of $V_\zeta^\mu(\cdot)$ by setting $V_\zeta^\mu(i_0) = \rho_\zeta^\mu$ for a given $i_0 \in S$ in order to obtain unique $V_\zeta^\mu(i)$ $\forall i \in S$.

Theorem 1 [21]

(a) The map $\zeta \rightarrow \rho_\zeta^\mu$ is convex for each SDP μ and there exists a unique $\zeta_*^\mu \stackrel{\triangle}{=} \arg\max_{\zeta \geq 0}(\zeta\alpha - \ln(\rho_\zeta^\mu))$ for any μ.

(b) $p^{\mu,*}(i,j)$, $i,j \in S$ is given by

$$p^{\mu,*}(i,j) = \frac{\exp(\zeta_*^\mu g(i,\mu(i)))p(i,j,\mu(i))V_*^\mu(j)}{\rho_*^\mu V_*^\mu(i)} \tag{2}$$

(c) The regular conditional law of the MDP $\{X_m,\ m \geq 0\}$ under SDP μ, conditioned on the event $\{X_0 = x,\ \frac{1}{n}\sum_{k=0}^{n-1} g(X_k,\mu(X_k)) \geq \alpha\}$ converges to the law of a Markov chain starting at x with transition probabilities $p^{\mu,*}(\cdot,\cdot)$.

In the above, $\rho_*^\mu \stackrel{\triangle}{=} \rho_{\zeta_*^\mu}^\mu$ and $V_*^\mu \stackrel{\triangle}{=} V_{\zeta_*^\mu}^\mu$, respectively. It can be shown (cf. Lemma 2 of [21]) using a generalization of Theorem 6.3 of [33] that as $n \rightarrow \infty$,

$$P_x(\frac{1}{n}\sum_{m=0}^{n-1} g(X_m,\mu(X_m)) \geq \alpha_n) \sim \frac{V_*^\mu(x)\exp(-n(\zeta_*^\mu\alpha - \ln(\rho_*^\mu)))\exp(k\zeta_*^\mu)}{\zeta_*^\mu\sqrt{2\pi n\lambda_*^\mu}}$$

where $\alpha_n = \alpha - \frac{k}{n}$ and $\lambda_*^\mu = \sqrt{\frac{\partial^2 \ln \rho_\zeta^\mu}{\partial\zeta^2}\mid_{\zeta=\zeta^*}}$. The result in Theorem 1(b) follows in a straightforward manner from the above. Thus the transition probabilities $p^{\mu,*}(\cdot,\cdot)$ depend on the risk parameter ζ_*^μ given in Theorem 1(a).

For a given $\zeta > 0$ and SDP μ, let $\{X_n^{\zeta,\mu},\ n \geq 0\}$ represent a Markov chain on S with (suitably normalized) transition probabilities

$$p^{\mu,\zeta}(i,j) \stackrel{\triangle}{=} \frac{\exp(\zeta g(i,\mu(i)))p(i,j,\mu(i))V_\zeta^\mu(j)}{\rho_\zeta^\mu V_\zeta^\mu(i)},\ \ i,j \in S.$$

In particular, we consider here the corresponding risk-averse case ($\zeta > 0$). The risk-preferring case ($\zeta < 0$) is easier to handle and is not considered in this paper. In view of Assumption (A), $\{X_n^{\zeta,\mu}\}$ is irreducible. Let $\eta_\zeta^\mu(\cdot)$ denote its unique stationary distribution. We now have the following lemma whose proof follows as in Proposition 4.9 of [33].

Lemma 1 $\frac{\partial \ln(\rho_\zeta^\mu)}{\partial\zeta} = \sum_{i\in S}\eta_\zeta^\mu(i)g(i,\mu(i))$.

In classical Markov decision theory, one is minimizing expectation and not conditional expectation of the ergodic cost and one can prove that it suffices to consider only SDPs. Such a result is not proved here, so it is our *choice* to restrict to these. Finally, in principle, the requirement that the rare event condition hold for all SDPs μ (see the problem definition above) is not strictly needed in order for the theory to go through. However, one expects this to be true in typical applications. In the next section, we present an adaptive algorithm for finding optimal μ and ζ by building on the basic results of Theorem 1 – Lemma 1.

3 The Adaptive Algorithm

Given an SRP $\phi : S \to \mathcal{P}(A)$, one can identify $\phi(i)$ with a parameter vector $\theta_i = (\theta_i^1, \ldots, \theta_i^{N_i-1})^T$, where $\theta_i^j \geq 0$ are the probabilities of picking actions a_i^j, $j = 1, \ldots, N_i - 1$. Thus $\sum_{j=1}^{N_i-1} \theta_i^j \leq 1$. Further, $\theta_i^{N_i}$ (the probability of selecting action $a_i^{N_i}$) is directly obtained from the above representation of $\phi(i)$ as $\theta_i^{N_i} = 1 - \sum_{j=1}^{N_i-1} \theta_i^j$. Let $\theta = (\theta_1 \ldots, \theta_s)^T = (\theta_1^1, \ldots, \theta_1^{N_1-1}, \theta_2^1, \ldots, \theta_2^{N_2-1}, \ldots, \theta_s^1, \ldots, \theta_s^{N_s-1})^T$. Let $p^{\theta_i}(i,j)$, $i, j \in S$, be defined by $p^{\theta_i}(i,j) = \theta_i^1 p(i,j,a_i^1) + \ldots + \theta_i^{N_i} p(i,j,a_i^{N_i})$. Thus $p^{\theta_i}(i,j)$ correspond to the transition probabilities of the resulting Markov chain under SRP ϕ. Suppose $g^{\theta_i}(i) = \theta_i^1 g(i,a_i^1) + \ldots + \theta_i^{N_i} g(i,a_i^{N_i})$ and $h^{\theta_i}(i,j) = \theta_i^1 h(i,a_i^1,j) + \ldots + \theta_i^{N_i} h(i,a_i^{N_i},j)$, respectively, denote the expected values of the function $g(\cdot,\cdot)$ and the single-stage cost $h(\cdot,\cdot,\cdot)$ under SRP ϕ. Define three step-size sequences $\{a(n)\}$, $\{b(n)\}$ and $\{c(n)\}$ satisfying

Assumption (B)

$$\sum_n a(n) = \sum_n b(n) = \sum_n c(n) = \infty, \ \sum_n (a(n)^2 + b(n)^2 + c(n)^2) < \infty, \tag{3}$$

$$c(n) = o(b(n)), \ b(n) = o(a(n)). \tag{4}$$

Examples of $\{a(n)\}$, $\{b(n)\}$ and $\{c(n)\}$ that satisfy (3)-(4) are $a(n) = \frac{1}{n^{3/5}}$, $b(n) = \frac{1}{n^{4/5}}$, $c(n) = \frac{1}{n}$, and $a(n) = \frac{\log n}{n}$, $b(n) = \frac{1}{n}$, $c(n) = \frac{1}{n \log n}$, respectively. Let

$$T_i = \{x_i \stackrel{\triangle}{=} (x_i^1, \ldots, x_i^{N_i-1})^T \mid x_i^j \geq 0, \ j = 1, \ldots, N_i - 1, \text{ and } \sum_{j=1}^{N_i-1} x_i^j \leq 1\}$$

denote the policy simplex in state i onto which, after each policy update recursion, the vector of probabilities corresponding to the first $N_i - 1$ actions is projected. The probability $x_i^{N_i}$ of selecting the N_i−th action in state i is then set according to $x_i^{N_i} = 1 - \sum_{j=1}^{N_i-1} x_i^j$.

For any $i \in S$, let $\triangle_i^j(n)$, $j = 1, \ldots, N_i - 1$, $n \geq 0$, be ± 1-valued variables. These shall constitute the perturbations in SPSA type gradient estimates. Exact values of these for any given n are obtained using a normalized Hadamard matrix based construction as in [12] (see below). Let $\triangle_i(n) = (\triangle_i^1(n), \ldots, \triangle_i^{N_i-1}(n))^T$ denote the vector of perturbations at the nth epoch. In general, an $m \times m$ ($m \geq 2$) matrix H is said to be a Hadamard matrix of order m if its entries belong to $\{1, -1\}$ and $H^T H = mI_m$, where I_m is the $m \times m$ identity matrix. A Hadamard matrix is said to be normalized if all the elements in its first column are 1. The construction used in [12] that we also use here is the following:

- For $k = 1$, let

$$H_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

- For general $k > 1$,

$$H_{2^k} = \begin{bmatrix} H_{2^{k-1}} & H_{2^{k-1}} \\ H_{2^{k-1}} & -H_{2^{k-1}} \end{bmatrix}.$$

For an $(N_i - 1)$-dimensional parameter vector as above, the order of the Hadamard matrix used is $M_i = 2^{\lceil log_2(N_i) \rceil}$. It is easy to see that $N_i - 1 < M_i$. Next form a matrix $\hat{H}_i$ in the following manner: Remove the first column from the normalized Hadamard matrix constructed above. Next pick any $(N_i - 1)$ of the remaining $(M_i - 1)$ columns and all M_i rows to form the new matrix. If only $(N_i - 1)$ columns remain after deleting the first column above, then pick all the remaining columns. Thus $\hat{H}_i$ is an $M_i \times (N_i - 1)$ matrix. Let the M_i rows of this matrix be represented by $\hat{H}_i(1)$, ..., $\hat{H}_i(M_i)$, respectively. Finally, the perturbation sequence $\triangle_i(n)$ is cyclically moved through the sequence $\{\hat{H}_i(1), \ldots, \hat{H}_i(M_i)\}$ of vectors by setting $\triangle_i(0) = \hat{H}_i(n \bmod M_i + 1)$. In what follows, we present an adaptive single simulation stochastic approximation based algorithm that performs asynchronous updates. Suppose $\nu_i(n)$ denotes the number of times that state i is visited by the MDP $\{X_m\}$ in n epochs. Then, one can write, $\nu_i(n) = \sum_{m=1}^{n} I\{X_m = i\}$. We generate new $\triangle_i(n)$ only for those instants n for which state i is visited by the chain i.e., $X_n = i$. For all other instants, $\theta_i(n)$ and $\triangle_i(n)$ are held fixed. Let $\triangle_i(n)^{-1}$ denote the vector $\triangle_i(n)^{-1} = (\frac{1}{\triangle_i^1(n)}, \ldots, \frac{1}{\triangle_i^{N_i-1}(n)})^T$. We now present our algorithm.

3.1 The Algorithm

Suppose $\delta > 0$ is a given constant and $\Gamma_i : \mathcal{R}^{N_i-1} \rightarrow \mathcal{R}^{N_i-1}$ be the projection from $\mathcal{R}^{N_i-1}$ to the simplex T_i. Let $\theta_i(n)$, $n \geq 0$ denote the nth update of θ_i. Let $\bar{\theta}_i(n) = \Gamma_i(\theta_i(n) + \delta \triangle_i(n))$, where $\triangle_i(n)$, $n \geq 0$ are obtained using normalized Hadamard matrices as explained earlier. We analogously denote $\bar{\theta}_i(n)$ as the vector $\bar{\theta}_i(n) = (\bar{\theta}_i^1(n), \ldots, \bar{\theta}_i^{N_i-1}(n))^T$ and let $\bar{\theta}_i^{N_i}(n) = 1 - \sum_{j=1}^{N_i-1} \bar{\theta}_i^j(n)$. The simulated MDP $\{X_n\}$ is governed by the perturbed randomized policy in the following manner: If $X_n = i$, then an action from the set $A(i)$ is selected according to the randomized policy $\bar{\theta}_i(n)$. Let $Y_i(n)$, $n \geq 0$ be quantities defined via the recursions below that are used for averaging the cost function. Let $V_n(i)$, $i \in S$ denote the nth update of value function and ζ_n the nth update of the risk parameter, respectively. We also let $\theta_i^j(0) = \frac{1}{N_i}$, $\forall j = 1, \ldots, N_i$, $i \in S$, implying that the simulation is started with a policy that assigns equal weightage to every feasible action in each

state. Other initial values for the same could be selected as well. The algorithm is described as follows:

The Algorithm

- *Step 0 (Initialize): Fix* $\theta_i(0) \stackrel{\triangle}{=} (\theta_i^1(0), \ldots \theta_i^{N_i-1}(0))^T$, $i \in S$, *as the vectors of initial probabilities for selecting actions in states* i *with* $\theta_i^{N_i}(0) = 1 - \sum_{j=1}^{N_i-1} \theta_i^j$. *Fix integers* L *and (large)* P *arbitrarily. Fix a (small) constant* $\delta > 0$. *Set* $n := 0$ *and* $m := 0$. *Generate* $M_i \times M_i$, *normalized Hadamard matrices* (H_i) *where* $M_i = 2^{\lceil \log_2(N_i) \rceil}$, $i \in S$. *Let* $\hat{H}_i$, $i \in S$, *be* $M_i \times N_i$ *matrices formed from* H_i *by choosing any* N_i *of its columns other than the first and let* $\hat{H}_i(p)$, $p = 1, ..., M_i$ *denote the* M_i *rows of* $\hat{H}_i$. *Now set* $\Delta_i(0) := \hat{H}_i(1)$, $\forall i \in S$. *Set* $\bar{\theta}_i(0) = \Gamma_i(\theta_i(0) + \delta\Delta_i(0))$, $i \in S$ *as the initial value of the perturbed randomized policy. Alternatively, denote* $\bar{\theta}_i(0) = (\bar{\theta}_i^1(0), \ldots, \bar{\theta}_i^{N_i-1}(0))$ *and let* $\bar{\theta}_i^{N_i}(0) = 1 - \sum_{j=1}^{N_i-1} \theta_i^j(0)$. *Obtain initial transition probabilities* $p^{\bar{\theta}_i(0)}(i,j)$, $i,j \in S$ *by setting* $p^{\bar{\theta}_i(0)}(i,j) = \bar{\theta}_i^1(0)p(i,j,a_i^1) + \ldots + \bar{\theta}_i^{N_i}(0)p(i,j,a_i^{N_i})$. *Set* $p_0^{\bar{\theta}_i(0)}(i,j) \stackrel{\triangle}{=} p^{\bar{\theta}_i(0)}(i,j)$ *as the transition probabilities of the new Markov chain. Set* $g^{\bar{\theta}_i(0)}(i) = \bar{\theta}_i^1(0)g(i,a_i^1) + \ldots + \bar{\theta}_i^{N_i}(0)g(i,a_i^{N_i})$ *and* $h^{\bar{\theta}_i(0)}(i,j) = \bar{\theta}_i^1(0)h(i,a_i^1,j) + \ldots + \bar{\theta}_i^{N_i}(0)h(i,a_i^{N_i},j)$, *respectively. Set* $V_0(i)$, $\forall i \in S$ *as the initial estimates of the cost-to-go function. Also, set* $\zeta_0 = 0$. *Fix a state* $i_0 \in S$ *to be a given reference state and set* $Y_i(0) = 0, \forall i \in S$.

- *Step 1: For all states* $X_{nL+m} = i \in S$, *simulate the corresponding next states* X_{nL+m+1} *according to transition probabilities* $p_n^{\bar{\theta}_i(n)}(i,\cdot)$. *For all* $i \in S$, *perform the following updates:*

$$V_{nL+m+1}(i) = V_{nL+m}(i) + a(\nu_i(n))I\{X_{nL+m} = i\}(\frac{\exp(\zeta_{nL+m}g^{\bar{\theta}_i(n)}(i))}{V_{nL+m}(i_0)}V_{nL+m}(X_{nL+m+1})\times$$

$$(\frac{p^{\bar{\theta}_i(n)}(i, X_{nL+m+1})}{p_n^{\bar{\theta}_i(n)}(i, X_{nL+m+1})}) - V_{nL+m}(i)) \tag{5}$$

$$\zeta_{nL+m+1} = \zeta_{nL+m} + b(n)\left(\alpha - g^{\bar{\theta}_{X_{nL+m+1}}(n)}(X_{nL+m+1})\right) \tag{6}$$

$$Y_i(nL+m+1) = Y_i(nL+m) + a(\nu_i(n))I\{X_{nL+m} = i\}$$
$$\left(h^{\bar{\theta}_i(n)}(i, X_{nL+m+1})\left(\frac{p^{\bar{\theta}_i(n)}(i, X_{nL+m+1})}{p_n^{\bar{\theta}_i(n)}(i, X_{nL+m+1})}\right) - Y_i(nL+m)\right) \tag{7}$$

If $m = L-1$, *set* $nL := (n+1)L$, $m := 0$ *and go to Step 2;*

else, set $m := m+1$ *and repeat Step 1.*

- *Step 2: For all $i \in S$,*

$$\theta_i(n+1) = \Gamma_i\left(\theta_i(n) - c(\nu_i(n))I\{X_{nL} = i\}\frac{Y_i(nL)\triangle_i(\nu_i(n))^{-1}}{\delta}\right). \tag{8}$$

Set $n := n+1$. If $n = P$, go to Step 3;

else, for all $i \in S$, set $\Delta_i(n) := \hat{H}(n \bmod M_i + 1)$ as the new Hadamard matrix generated perturbation. Set $\bar{\theta}_i(n) = (\Gamma_i(\theta_i(n) + \delta\Delta_i(n))$, $i \in S$ as the new perturbed randomized policy. For all $i, j \in S$, set $p^{\bar{\theta}_i(n)}(i,j)$, $= \bar{\theta}_i^1(n)p(i,j,a_i^1) + \ldots + \bar{\theta}_i^{N_i}(n)p(i,j,a_i^{N_i})$. Set $g^{\bar{\theta}_i(n)}(i) = \bar{\theta}_i^1(n)g(i,a_i^1) + \ldots + \bar{\theta}_i^{N_i}(n)g(i,a_i^{N_i})$ and $h^{\bar{\theta}_i(n)}(i,j) = \bar{\theta}_i^1(n)h(i,a_i^1,j) + \ldots + \bar{\theta}_i^{N_i}(n)h(i,a_i^{N_i},j)$, respectively. Finally, for all $i, j \in S$, update estimates $p_n^{\bar{\theta}_i(n)}(i,j)$ of the transition probabilities for the new chain according to

$$p_n^{\bar{\theta}_i(n)}(i,j) = \frac{\exp(\zeta_{nL}g(i,\bar{\theta}_i(n)))}{V_{nL}(i)V_{nL}(i_0)}p^{\bar{\theta}_i(n)}(i,j)V_{nL}(j).$$

Normalize $p_n^{\bar{\theta}_i(n)}(i,j)$ such that $p_n^{\bar{\theta}_i(n)}(i,j) \geq 0, \quad \forall i,j$ and $\sum_{j\in S} p_n^{\bar{\theta}_i(n)}(i,j) = 1, \forall i$.

Go to Step 1.

- *Step 3 (termination): Terminate algorithm and output $\bar{\theta}_i(P)$, $i \in S$ as the final randomized policy.*

Remark 1: As described in the algorithm, it is observed that updating the slowest timescale recursion (8) every (given) $L \geq 1$ visits to state i, $i \in S$, and keeping the randomized policy fixed in between, enhances performance. This, in effect, amounts to an additional averaging over and above that resulting from the use of different step-size schedules, see also, [11], [12] for certain simulation based parametric optimization algorithms that use a similar 'additional' averaging. As observed in [38], [12], the one-simulation SPSA algorithms that use randomized perturbation sequences do not show good performance because of the presence of extra bias terms in the gradient estimates of these. As described in Section 1 (see also the discussion after Eq.(16) below), the use of normalized Hadamard matrices significantly improves performance since all bias terms get cancelled after regular deterministic intervals that are, in general, also significantly shorter in duration as compared to the case when randomized perturbations are used. Finally, even though we present our algorithm for the case when the number of iterations P is fixed apriori, it can be easily modified to allow for stopping criteria based on desired accuracy levels, a scenario that we consider in our numerical experiments in Section 4. The convergence analysis that follows carries through for this case with minor modifications.

3.2 Sketch of Convergence Analysis

The convergence analysis uses the following basic principle of two timescale, or more generally multiple timescale, stochastic approximation [15]: Each iteration in such a scheme can be analyzed separately by treating other iteration(s) on slower timescale(s) as quasi-static, i.e., freezing the parameter(s) updated by the latter; while treating other iteration(s) on faster timescale(s) as quasi-equilibrated, i.e., averaging the parameter(s) updated by the latter w.r.t. their equilibrium behavior, arrived at similarly by treating all slower components as constants and all faster components as equilibrated. For simplicity of presentation, we show here the analysis for the case corresponding to $L = 1$. The extension to the general case is straightforward [11], [12]. Let us first consider the synchronous version of the algorithm. Recursions (5)-(8) can be written as follows: For all $i \in S$,

$$V_{n+1}(i) = V_n(i) + a(n) \left(\frac{\exp(\zeta_n g^{\bar{\theta}_i(n)}(i))}{V_n(i_0)} V_n(X_{n+1}) \left(\frac{p^{\bar{\theta}_i(n)}(i, X_{n+1})}{p_n^{\bar{\theta}_i(n)}(i, X_{n+1})} \right) - V_n(i) \right), \tag{9}$$

$$\zeta_{n+1} = \zeta_n + b(n) \left(\alpha - g^{\bar{\theta}_{X_{n+1}(n)}}(X_{n+1}) \right), \tag{10}$$

$$Y_i(n+1) = Y_i(n) + a(n) \left(h^{\bar{\theta}_i(n)}(i, X_{n+1}) \left(\frac{p^{\bar{\theta}_i(n)}(i, X_{n+1})}{p_n^{\bar{\theta}_i(n)}(i, X_{n+1})} \right) - Y_i(n) \right), \tag{11}$$

$$\theta_i(n+1) = \Gamma_i \left(\theta_i(n) - c(n) \frac{Y_i(n) \triangle_i(n)^{-1}}{\delta} \right). \tag{12}$$

Iteration (9):

It can be shown that iteration (9) for fixed ζ_n and $\bar{\theta}_i(n)$ viz., $\zeta_n \equiv \zeta$ and $\bar{\theta}_i(n) \equiv \bar{\theta}_i$, respectively, asymptotically tracks the trajectories of the ordinary differential equation (ODE): For $i \in S$,

$$\dot{x}_t(i) = \frac{\exp(\zeta g^{\bar{\theta}_i}(i))}{x_t(i_0)} \sum_{j \in S} p^{\bar{\theta}_i}(i, j) x_t(j) - x_t(i). \tag{13}$$

The ODE (13) has a unique asymptotically stable fixed point in the positive quadrant (which is invariant under the ODE) which corresponds to the solution to the multiplicative Poisson equation. To see how this comes by, we use the fact that

$$E \left[\frac{\exp(\zeta g^{\bar{\theta}_i}(i))}{V_n(i_0)} V_n(X_{n+1}) \left(\frac{p^{\bar{\theta}_i}(i, X_{n+1})}{p_n^{\bar{\theta}_i}(i, X_{n+1})} \right) \mid X_n = i \right] = \frac{\exp(\zeta g^{\bar{\theta}_i}(i))}{V_n(i_0)} \sum_{j \in S} p^{\bar{\theta}_i}(i, j) V_n(j).$$

Thus (9) can be rewritten as

$$V_{n+1}(i) = V_n(i)$$

$$+ \quad a(n)\left(\frac{\exp(\zeta g^{\bar{\theta}_i}(i))}{V_n(i_0)}\sum_{j\in S}p^{\bar{\theta}_i}(i,j)V_n(j)) - V_n(i)\right)$$

$$+ \quad a(n)\left(\frac{\exp(\zeta_n g^{\bar{\theta}_i(n)}(i))}{V_n(i_0)}V_n(X_{n+1})\left(\frac{p^{\bar{\theta}_i(n)}(i,X_{n+1})}{p_n^{\bar{\theta}_i(n)}(i,X_{n+1})}\right) - \frac{\exp(\zeta g^{\bar{\theta}_i}(i))}{V_n(i_0)}\sum_{j\in S}p^{\bar{\theta}_i}(i,j)V_n(j))\right).$$

This is seen as a noisy discretization of the ODE (13) with decreasing stepsize $a(n)$ and a 'martingale difference' or 'noise' error term. The contribution to the net error due to the former vanishes asymptotically because $a(n) \to 0$ and so doe the contribution of the latter 'almost surely' following a standard martingale argument. This is a commonly used technique in reinforcement learning based algorithms [29], [13] with the idea being to replace conditional averages by evaluation at actual or simulated transitions and, then exploit the incremental nature of stochastic approximation scheme to do the averaging for you.

Iteration (10):

The iteration (10) is a stochastic gradient scheme that, for fixed $\bar{\theta}_i(n) \equiv \bar{\theta}_i$, can be seen, from the first part of Theorem 1 and Lemma 1, to asymptotically track the point $\zeta_*^{\bar{\theta}}$ corresponding to the given policy above (using again martingale type arguments and the latter part of (3) on $\{b(n)\}$ now).

Note from (4) that $c(n) = o(b(n))$ and $c(n) = o(a(n))$, respectively. This implies that recursions (9) and (10), respectively, proceed on faster timescales as compared to (12). Moreover, since $b(n) = o(a(n))$ as well, (9) proceeds on a faster scale than (10). Using standard analysis of multi-timescale stochastic approximations [15], one can show that the iterations (10) and (12) appear to be quasi-static when viewed from the timescale on which (9) is updated. Moreover, when viewed from either of the timescales on which (10) or (12) are updated, the recursion (9) appears to be essentially equilibrated. Similarly, when viewed from the timescale on which (10) is performed, the recursion (9) appears to be equilibrated while, as already stated, (12) appears to be quasi-static. The above justifies selecting time-invariant quantities $\zeta_n \equiv \zeta$ and $\bar{\theta}_i(n) \equiv \bar{\theta}_i$ (resp. $\bar{\theta}_i(n) \equiv \bar{\theta}_i$) in the convergence analysis of recursion (9) (resp. (10)).

Iteration (11):

The iteration (11) proceeds on the fastest timescale $\{a(n)\}$ as well and is merely used to perform averaging of the cost function. The updates from this recursion are then used in the gradient estimate for average cost in the slow timescale recursion (12).

Iteration (12):

Iteration (12) does policy update. Note that here one is interested in finding the minimizing policy parameters (i.e., the probabilities) for the long-run average cost albeit conditioned on the rare event. Thus one is interested in finding the gradient of the average cost. This is achieved by our slow timescale iteration as explained below.

For a bounded, continuous $v_i(\cdot) : \mathcal{R}^{N_i-1} \rightarrow \mathcal{R}^{N_i-1}$, define

$$\bar{\Gamma}_i(v_i(y)) = \lim_{\eta \downarrow 0} \left(\frac{\Gamma_i(y + \eta v_i(y)) - \Gamma_i(y)}{\eta} \right).$$

Suppose $\theta = (\theta_1^1, \ldots, \theta_1^{N_1-1}, \ldots, \theta_s^1, \ldots, \theta_s^{N_s-1})^T$ be a given SRP. Let $\hat{J}(\theta)$ denote the long-run average cost under SRP θ. Let $\nabla_i^j \hat{J}(\theta)$ denote the derivative of $\hat{J}(\theta)$ w.r.t. θ_i^j, $j = 1, \ldots, N_i - 1$, and let $\nabla_i \hat{J}(\theta)$ correspond to $\nabla_i \hat{J}(\theta) = (\nabla_i^1 \hat{J}(\theta), \ldots, \nabla_i^{N_i-1} \hat{J}(\theta))^T$. The policy update can be shown to track (in the limits as $P \rightarrow \infty$ and $\delta \rightarrow 0$) the trajectories of the ODE: For $i \in S$,

$$\dot{\theta}_i(t) = \bar{\Gamma}_i(-\nabla_i \hat{J}(\theta)). \tag{14}$$

The proof broadly proceeds as follows. A standard analysis of (11) [9], [11] using the fact that the chain under each stationary policy is irreducible (and hence positive recurrent) shows that

$$\parallel Y_i(n) - \hat{J}(\bar{\theta}(n)) \parallel \rightarrow 0 \text{ as } n \rightarrow \infty. \tag{15}$$

Here $\bar{\theta}(n) = (\bar{\theta}_1(n), \ldots, \bar{\theta}_s(n))^T$. Suppose for all $i \in S$, $\theta_i(n) \in T_i^0$, where T_i^0 corresponds to the interior of the simplex T_i. Then for δ sufficiently small, $\theta_i(n) + \delta \triangle_i(n) \in T_i^0$ as well. Hence $\bar{\theta}_i(n)$ $= \Gamma_i(\theta_i(n) + \delta \triangle_i(n)) = \theta_i(n) + \delta \triangle_i(n)$. Moreover, since $c(n) \rightarrow 0$ as $n \rightarrow \infty$, $\parallel \hat{J} \parallel < \infty$ and $\delta > 0$, one can ensure by choosing n large enough that

$$\Gamma_i \left(\theta_i(n) - c(n) \frac{\hat{J}(\bar{\theta}(n)) \triangle_i(n)^{-1}}{\delta} \right) = \theta_i(n) - c(n) \frac{\hat{J}(\theta(n) + \delta \triangle(n)) \triangle_i(n)^{-1}}{\delta}.$$

Using a Taylor series expansion of $\hat{J}(\theta(n) + \delta \triangle(n))$ around $\theta(n)$, one obtains

$$\hat{J}(\theta(n) + \delta \triangle(n)) = \hat{J}(\theta(n)) + \delta \sum_{l=1}^{s} \sum_{j=1}^{N_l - 1} \triangle_l^j(n) \nabla_l^j \hat{J}(\theta(n)) + O(\delta^2).$$

For a given $k \in \{1, \ldots, N_i - 1\}$,

$$\frac{\hat{J}(\theta(n) + \delta \triangle(n))}{\delta \triangle_i^k(n)} = \frac{\hat{J}(\theta(n))}{\delta \triangle_i^k(n)} + \nabla_i^k \hat{J}(\theta(n)) + \sum_{j=1, j \neq k}^{N_i - 1} \frac{\triangle_i^j(n) \nabla_i^j \hat{J}(\theta(n))}{\triangle_i^k(n)}$$

$$+ \sum_{l=1, l \neq i}^{s} \sum_{j=1}^{N_l - 1} \frac{\triangle_l^j(n) \nabla_l^j \hat{J}(\theta(n))}{\triangle_i^k(n)} + O(\delta). \tag{16}$$

The first term in the RHS above corresponds to the 'additional' bias term, described earlier, whose overall contribution to bias depends on the magnitude of δ and the frequency with which $\triangle_i^k(n)$ change sign as a function of n, for all k and i. It can be shown (cf. Theorem 2.5 of [12]) that for any $n \geq 0$, $\sum_{m=n}^{n+M_i} \frac{1}{\triangle_i^k(m)} = 0$, $\forall k = 1, \ldots, N_i$, and $\sum_{m=n}^{n+M_i} \frac{\triangle_i^j(m)}{\triangle_i^k(m)} = 0$, $\forall j \neq k$, $j, k \in \{1, \ldots, N_i\}$, respectively. Note that because of the use of Hadamard matrices, M_i is typically small, as a result of which the bias contributed by the above terms is not significant in general.

One can also show in a similar manner as Corollary 2.6 of [12] that

$$\| \sum_{m=n}^{n+\bar{M}} \sum_{l=1, l\neq i}^{s} \sum_{j=1}^{N_l-1} \frac{c(m)}{c(n)} \frac{\triangle_l^j(m) \nabla_l^j \hat{J}(\theta(m))}{\triangle_i^k(m)} \| \rightarrow 0 \text{ as } n \rightarrow \infty,$$

where $\bar{M} = \max(M_1, \ldots, M_s)$. (Recall that M_i is the number of rows in the $\hat{H}_i$, $i = 1, \ldots, s$, matrix defined earlier.) Thus (12) can be seen to be analogous to the recursion

$$\theta_i(n+1) = \Gamma_i(\theta_i(n) - c(n)(\nabla_i \hat{J}(\theta(n)) + \xi_1(n) + O(\delta))), \tag{17}$$

where $\xi_1(n) = o(n)$. In general, one can write $\Gamma_i(\theta_i(n) + \delta\triangle_i(n)) = \theta_i(n) + \delta\triangle_i(n) + \delta r_i(n)$ where $r_i(n)$ correspond to error terms because of the projection operator, such that $\| r_i(n) \| \leq \| \triangle_i(n) \|$ with equality only when $r_i(n) = -\triangle_i(n)$. In the latter case,

$$\| \sum_{m=n}^{n+M_i} \frac{c(m)}{c(n)} \frac{\hat{J}(\theta(m))}{\delta \triangle_i^k(m)} \| \rightarrow 0 \text{ as } n \rightarrow \infty, \ \forall \delta > 0. \tag{18}$$

Finally, we consider the case of any other $\theta_i(n)$ lying on the boundary of T_i. Suppose the correction term $r_i(n) \stackrel{\triangle}{=} (r_i^1(n), \ldots, r_i^{N_i-1}(n))^T$, $i \in S$. Now $\exists\ j \in \{1, \ldots, N_i - 1\}$ for which if sign of $\triangle_i^j(n)$ is such that the vector $\theta_i(n) + \delta\triangle_i(n)$ points outwards from the boundary, then $r_i^j(n) = -\triangle_i^j(n)$. For simplicity, suppose all other $\triangle_i^l(n)$ are such that components $\theta_i^l(n) + \delta\triangle_i^l(n)$ lie inside their respective regions. Then again one can see that (17) is valid. Also, for $k = j$, (18) continues to hold. Now the function $\hat{J}(\cdot)$ itself serves as a Liapunov function for the ODE (14) which has $K \stackrel{\triangle}{=} \{\theta \in T_1 \times T_2 \times \cdots \times T_s \mid \bar{\Gamma}_i(\nabla_i \hat{J}(\theta)) = 0 \ \forall i \in S\}$ as its asymptotically stable fixed points. A standard argument now shows that the iterations (12) converge to K almost surely in the limits as $P \rightarrow \infty$ and $\delta \rightarrow 0$. The equilibria for the projected gradient scheme here correspond to Kuhn-Tucker points with the stable ones being local minima. By 'avoidance of traps' results [19], [22], the scheme converges to one of these with probability one. (Strictly speaking, this requires some additional conditions on the noise component of the iterations that can be ensured by adding independent noise if necessary. Most often, as here, it is empirically observed that the existing noise suffices.)

For the asynchronous case that we actually work with, the step-size sequences are $\{a(\nu_i(n))\}$, $\{b(\nu_i(n))\}$ and $\{c(\nu_i(n))\}$, respectively, and the parameters corresponding to state i are updated only at instants when the MDP $\{X_n\}$ under the running policy visits state i. It can be shown (cf. [16], [17], [18], [20]) that the iterate (5) for fixed ζ and $\bar{\theta}$ as before, asymptotically tracks trajectories of the (combined) ODE

$$\dot{x}_t = \Pi(t) \begin{pmatrix} \frac{\exp(\zeta g^{\bar{\theta}_1}(1))}{x_t(i_0)} \sum_{j \in S} p^{\bar{\theta}_1}(1,j) x_t(j) - x_t(1) \\ . \\ . \\ . \\ \frac{\exp(\zeta g^{\bar{\theta}_s}(s))}{x_t(i_0)} \sum_{j \in S} p^{\bar{\theta}_s}(s,j) x_t(j) - x_t(s) \end{pmatrix}.$$

Here $\Pi(t)$ is an $s \times s$ scaling matrix which is a positive scalar in $[0,1]$ times the identity matrix under some additional technical conditions on the stepsize sequence (see $(i) - (iv)$, p. 842, [16]). Hence this ODE is a time-scaled version of the synchronous ODE. One thus obtains the same result here as before with the only difference being that the convergence to the desired limit points can now become slower as compared to the synchronous case. We now present our numerical results.

4 Numerical Results

The problem of routing multiple flows in communication networks has been well studied during the last few decades [6] with several approaches having been proposed for static and dynamic optimization of routing. In [40], [6], gradient based projection algorithms for optimal routing have been studied. More recently, [31], [34], [41], reinforcement learning techniques have also been applied to the problem of routing. We consider here an application of our algorithm to finding optimal routes for flows in communication networks, conditioned on a rare event. The basic setting is shown in Fig. 1. Nodes A and B are connected via two links. We assume that the system is slotted with time slots of equal length. Customers/flows arrive at the beginning of time slots at A, and have to be sent to B. There are two routes R_1 and R_2 from A to B. An arrival occurs with probability p in a given time slot independent of others. At the beginning of a time slot, decision on whether to route these arrivals onto R_1 or R_2 is made by a controller (at Node A). Thus, all new arrivals at the beginning of a time slot are routed either to R_1 or R_2. However, we also assume that both R_1 and R_2 can accommodate at most M customers (or flows) at any given instant. All flows that cannot be accommodated in a given slot immediately leave the system. Suppose each flow at any given instant (or a slot boundary) finishes service w.p. q_1 on R_1 and w.p. q_2 on R_2, respectively, independent of other flows. Further, if a flow does not finish service in a time slot, its service extends to the next slot independently of the number of flows in either route and the

number of slots the given flow has been in service for. The above process is repeated again in subsequent slots. Thus the number of slots that a customer is in service at node j, $j = 1, 2$ equals i with probability $(1 - q_j)^{i-1} q_j$, for $i \geq 1$. Let $X_n^{(1)}$ (resp. $X_n^{(2)}$) denote the number of flows on R_1 (resp. R_2) in time slot n. Let $\{A(n)\}$ with $A(n) \in \{a_1, a_2\}$ $\forall n \geq 1$, denote the associated action-valued process, where a_i corresponds to the action of routing new flows in a time slot on the route R_i, $i = 1, 2$. Then under a given SDP, $\{X_n\}$, where $X_n = (X_n^{(1)}, X_n^{(2)})$, $n \geq 0$, forms a discrete time Markov chain with state transition equation given by

$$\begin{pmatrix} X_{n+1}^{(1)} \\ X_{n+1}^{(2)} \end{pmatrix} = \begin{pmatrix} \min[X_n^{(1)} - Q_1(n) + I\{A(n) = a_1\}B(n), M] \\ \min[X_n^{(2)} - Q_2(n) + I\{A(n) = a_2\}B(n), M] \end{pmatrix},$$

where the departures from routes R_1 and R_2 during time slot n are denoted as $Q_1(n)$ and $Q_2(n)$, respectively, and satisfy $0 \leq Q_j(n) \leq N_j(n)$, $j = 1, 2$. Also, $B(n)$ denotes the number of new arrivals at Node A, at the beginning of time slot $(n+1)$. Note that since there are only two actions associated with each state here, the parameter vector $\theta_i(n)$ of the randomized policy is simply $\theta_i(n) = \theta_i^1(n)$. The simplex T_i associated with each state here corresponds to the interval $[0, 1]$ $\forall i$. The projection map Γ_i is thus defined by $\Gamma_i(x) = \max(0, \min(x, 1))$ $\forall i$. Also, $\bar{\theta}_i(n) = \Gamma_i(\theta_i^1(n) + \delta \triangle_i^1(n))$. The sequences $\{\triangle_i^1(n),\ n \geq 0\}$, $i \in S$ are generated using normalized Hadamard matrices. These turn out to be simply $\triangle_i^1(n) = (-1)^n$. The step-sizes are chosen as $a(n) = b(n) = c(n) = 1$, $n = 0, 1$, and for $n \geq 2$,

$$a(n) = \frac{\log(n)}{n}, b(n) = \frac{1}{n}, c(n) = \frac{1}{n \log(n)}.$$

The single-stage cost in state i under policy $\bar{\theta}_i(n)$ is given by $h^{\bar{\theta}_i(n)}(i, X_{n+1}) = |X_{n+1}^{(1)} - N_1| + |X_{n+1}^{(2)} - N_2|$, where N_1 and N_2 are given thresholds and (as before) $X_{n+1} = (X_{n+1}^{(1)},\ X_{n+1}^{(2)})$ corresponds to the state at the next instant. The cost function thus aims to keep the number of flows along R_1 to be near threshold N_1 and those along R_2 to be near N_2 for some $0 \leq N_1$, $N_2 \leq M$. Here the parameters N_1 and N_2 may be set arbitrarily. Note that since all new arrivals in a time slot are routed to either R_1 or R_2, N_1 and N_2 should be judiciously chosen. A value of N_1 or N_2 close to zero would lead to under-utilization while a value close to M would result in leaving less room for accommodating future flows on the corresponding route. The last is required, for instance, in cases where there are different categories of traffic flows in the network each having a possibly different pay off (a scenario not considered in this paper). Any other choice for the cost function may be used as well.

The function $g^{\cdot}(\cdot)$ used for defining the rare event is given as $g^{\bar{\theta}_{X_n}}(X_n) = I\{X_n^{(2)} > N\}$, where N is another (given integer) threshold. Thus $g^{\cdot}(\cdot)$ equals one if $X_n^{(2)} \in \{N + 1, \ldots, M\}$ and is zero otherwise. The long-run average $\lim_{n \to \infty} \frac{1}{n} \sum_{m=0}^{n-1} g^{\bar{\theta}_{X_m}}(X_m)$ in this case corresponds to the stationary

probability of the number of flows at the second node exceeding N. For any given SDP, the latter quantity would depend on the resulting transition probability matrix for the process $\{X_n\}$ under that SDP. We consider two different settings for our experiments that we refer to as settings (a) and (b), respectively. The input parameters for the two settings are given in Table 1 below.

Note that in the algorithm in Section 3.1, the number of iterations P is fixed apriori. However, for obtaining more accurate estimates, we use a different stopping criterion for the algorithm that is based on an accuracy parameter ϵ as explained below and not one based on a fixed value of P. For a given $\epsilon > 0$, let k_ϵ be the transition number of the Markov chain at which the estimate of $\rho_\zeta^{\mu^*} \equiv V_\zeta^{\mu^*}(i_0)$ converges to within ϵ of its previous value 100 times in succession. We let the value of ϵ to be 5×10^{-9} for setting (a) and 5×10^{-8} for setting (b), respectively. The above values of ϵ (for the two settings) will in fact be denoted as $\bar{\epsilon}$. More experiments using other values of ϵ are subsequently discussed.

In Figs. 2 and 4, we show the optimal policies $\theta^*(\cdot)$ for the two settings. The corresponding value functions are shown in Figs. 3 and 5. We observed from the optimal policies in both settings that for states (i_1, i_2), for given i_1, the value of $\theta^*(\cdot)$ i.e., the probability of selecting action a_1, on the whole seems to increase, starting from a low value, as i_2 is increased from 0 to M. Thus, in general, for low values of i_2, for given i_1, the preferred action is a_2 (i.e., to route customers on the second link) while for higher values of i_2, the preferred action becomes a_1. This is along expected lines given the form of the associated cost function. The value function $V^*(\cdot)$ (in both settings) takes low values for low values of (i_1, i_2) and gradually increases (overall) when either i_1 or i_2 is increased. What is more interesting, however, is that there is a step-increase in these values as soon as the set of rare event states is reached and it stays high over those states. This is not surprising since the conditional probabilities of the rare event states will be higher as we are conditioning on the rare event.

In Table 2, values of various performance metrics under the optimal policy are shown. Note that ζ^* corresponds to the converged value of the risk parameter obtained from the recursion (6). The quantities $E^{\theta_X^*}[X^{(1)}]$ and $E^{\theta_X^*}[X^{(2)}]$ denote the mean numbers of flows on the two routes. These, in general, depend on the parameters p, q_1, q_2, M and θ^*, and in the present case, can be seen to be less than the thresholds N_1 and N_2, in either setting. The mean cost $E^{\theta_X^*}[h^{\bar{\theta}_i}(i, X^{(1)}, X^{(2)})]$, is higher in Setting (b) as compared to Setting (a) since the values of thresholds N_1 and N_2 in the former setting are higher.

Next, we performed some additional experiments along similar lines as [21], [23], to estimate

Table 1: Input Parameters for the two settings

Input Parameter	Setting (a)	Setting (b)
Link Capacity, M	10	20
N_i	$N_1 = 3,\ N_2 = 5$	$N_1 = 6,\ N_2 = 12$
N	7	13
α	0.25	0.25
Arrival probability, p	0.65	0.85
Departure probability, q_i	$q_1 = 0.7,\ q_2 = 0.52$	$q_1 = 0.7,\ q_2 = 0.52$
δ	0.01	0.01
L	11	11
n (see Equation (19))	50	150
ζ_0	0	0
$V_0(i),\ \ \forall i \in S$	1	1
$Y_i(0),\ \ \forall i \in S$	0	0
Initial policy $\ \forall i \in S$	$\theta_i^1(0) = \theta_i^2(0) = 0.5$	$\theta_i^1(0) = \theta_i^2(0) = 0.5$
Reference state, i_0	$(2, 2)$	$(2, 2)$

the rare event probability $\hat{p}_n$ (see below) under both settings.

$$\hat{p}_n = P_x(\frac{1}{n}\sum_{m=0}^{n-1} g^{\theta^*_{X_m}}(X_m) \geq \alpha). \tag{19}$$

The values of n are described in Table 1 for the two settings. An importance sampling estimator for this probability is the average of the i.i.d. samples

$$I\{\frac{1}{n}\sum_{m=0}^{n-1} g^{\theta^*}(X_m) \geq \alpha\}\frac{p^{\theta^*_{X_0}}(X_0, X_1)p^{\theta^*_{X_1}}(X_1, X_2)\cdots p^{\theta^*_{X_{n-2}}}(X_{n-2}, X_{n-1})}{p_*^{\theta^*_{X_0}}(X_0, X_1)p_*^{\theta^*_{X_1}}(X_1, X_2)\cdots p_*^{\theta^*_{X_{n-2}}}(X_{n-2}, X_{n-1})}.$$

In practice, one is able to obtain the above estimate only upto a certain specified degree of accuracy as obtained from the quantity ϵ (see above). There is however a tradeoff involved in the choice of ϵ. The variance of the estimates tends to be high if ϵ is not chosen to be small enough, which may affect their accuracy. On the other hand, as the value of ϵ is decreased beyond a point, the amount of computational effort required increases rapidly.

We run the algorithm for different values of ϵ. For each value of ϵ, we obtain an estimate $p_*^\epsilon(\cdot,\cdot)$ of $p_*^{\theta^*}(\cdot,\cdot)$ that is then used to generate i.i.d. samples for the estimate of the rare event probability $\hat{p}_n$ (see above). The mean and variance of the rare event probability are then determined using the batch means method. The simulation is terminated when the 95% confidence interval (cf. [30]) of probability lies within 5% of its estimated mean value. Let T_ϵ denote the total computational effort

Table 2: Performance under optimal policy

Performance Metric	Setting (a)	Setting (b)
ζ^*	1.652923e+00	7.370684e-01
$\zeta^*\alpha - \ln(\rho_{\zeta^*})$	2.456064e-01	5.742653e-02
$E^{\theta^*_X}[X^{(1)}]$	1.092038e+00	2.836020e+00
$E^{\theta^*_X}[X^{(2)}]$	4.183547e+00	8.720516e+00
$E^{\theta^*_X}[h^{\theta_i}(i, X^{(1)}, X^{(2)})]$	5.488044e+00	1.096857e+01

Table 3: Rare Event Probability Experiments

Parameters/Performance Metrics	Setting (a)	Setting (b)
$\bar{\epsilon}$	5.000000e-09	5.000000e-08
$k_{\bar{\epsilon}}$	11287258742	1247427803
$p_{\bar{\epsilon}}$	5.785067e-07	1.704158e-05
ϵ^*	5.000000e-05	1.000000e-04
k_{ϵ^*}	9292162	1197983
T_{ϵ^*}	2760999897	92719997
$(k_{\epsilon^*} + T_{\epsilon^*})$	2770292059	93917980
p_{ϵ^*}	5.446732e-07	1.574290e-05

involved in terms of the number of simulated transitions of the MDP that are generated during this process. We show in Figs. 6 and 8, plots of k_ϵ, T_ϵ and $(k_\epsilon + T_\epsilon)$ as functions of ϵ for settings (a) and (b), respectively. The total computational effort (in terms of $(k_\epsilon + T_\epsilon)$) is found to be the least for $\epsilon \equiv \epsilon^* = 5 \times 10^{-5}$ in setting (a) and for $\epsilon \equiv \epsilon^* = 10^{-4}$ in setting (b), respectively. Also, Figs. 7 and 9 show the plots of the rare event probability $\hat{p}_n$ (described in the figures as p_ϵ) obtained for different accuracy levels ϵ. The values of ϵ in the above figures are shown on the log scale for convenience.

In Table 3, we describe the values of the various parameters and metrics obtained for the rare event probability experiments. The quantities k_{ϵ^*}, T_{ϵ^*} and $(k_{\epsilon^*} + T_{\epsilon^*})$, respectively, correspond to the case when $\epsilon = \epsilon^*$ is chosen for both settings. Also $\bar{\epsilon} = 5 \times 10^{-9}$ (resp. $\bar{\epsilon} = 5 \times 10^{-8}$) is the lowest value of ϵ for which the simulations were run for setting (a) (resp. setting (b)). This level of accuracy was obtained in about 1.18×10^{10} iterations in setting (a) and about 3.05×10^9 iterations in setting (b). As stated previously, the value of $\bar{\epsilon}$ is used as the accuracy parameter in the earlier experiments (cf. Figs. 2 to 5 and Table 2). In Table 3, p_{ϵ^*} (resp. $p_{\bar{\epsilon}}$) corresponds to the value of $\hat{p}_n$ obtained when $\epsilon = \epsilon^*$ (resp. $\epsilon = \bar{\epsilon}$). Note that these values are much lower for setting (a) than for setting (b) (see also Figs. 7 and 9). As a consequence of the above, the values of k_{ϵ^*} and T_{ϵ^*} are seen to be much less for setting (b) as compared to the corresponding values of these for setting (a).

5 Conclusions

We developed an adaptive simulation based stochastic approximation algorithm for ergodic control of Markov chains conditioned on a rare event of zero probability. Our algorithm uses coupled recursions that are driven by different timescales. We briefly sketched the convergence analysis of our algorithm and presented numerical experiments on a setting involving routing multiple flows in communication networks. The results obtained demonstrate the usefulness of the proposed algorithm in obtaining optimal policies conditioned on a rare event and in estimating the rare event probability. The numerical setting considered here was, however, a simple setting designed to demonstrate the usefulness of the proposed algorithm. More complex settings involving, say, networks with multiple nodes and more routes with large numbers of flows on each should be tried in order to study the scalability of the proposed algorithm. The SPSA technique, in general, is known to be highly scalable as has been demonstrated through several applications over the last decade. In the simulation based optimization framework, SPSA based multi-timescale algorithms have been found to perform well computationally in the case of high-dimensional parameter settings studied in [11] and [12] (by more than an order of magnitude over related K-W based algorithms). Implementations involving such high-dimensional settings (along the lines described above) need to be studied for the proposed algorithm in the setting of this paper. Recently, in [14], certain Newton-based multiscale SPSA algorithms that estimate both the gradient and Hessian of the average cost have been developed in the simulation optimization setting. Similar algorithms for the setting considered here may also be developed.

One may extend these ideas further by applying these for optimal control conditioned on multiple rare events. For problems with large action spaces, one may consider suitable parameterizations of the policy space. One may also use feature based methods for problems with moderately large state spaces. Our adaptive algorithm can be used to derive optimal parameterized policies using features in place of states. It must be noted here that adaptive importance sampling techniques require storage of transition probabilities and our algorithm is no different in this regard. Hence it cannot directly be applied in the case of problems with very large state spaces where storage of such information itself is computationally infeasible. However, in many cases such as queueing networks, the transition probabilities are easy to compute and transitions easy to simulate using simple local dynamic laws. Further, storage of transition probability matrices may not be a major concern in such scenarios since these are known to be highly sparse. Developing similar algorithms in general scenarios involving very large state spaces would be an interesting research direction to pursue.

Acknowledgements

The first author was supported by grant number SR/S3/EE/43/2002-SERC-Engg. from the Department of Science and Technology, Government of India. The second author was supported by grant number III.5(157)/99-ET from the Department of Science and Technology, Government of India.

References

[1] Ahamed, T.P.I., Borkar, V.S. and Juneja, S. (2006) "Adaptive importance sampling technique for Markov chains using stochastic approximation", *To appear in* Operations Research.

[2] Balaji, S. and Meyn, S.P. (2000) "Multiplicative ergodicity and large deviations for an irreducible Markov chain", *Stochastic Processes and their Appl.*, 90:123-144.

[3] Baxter, J. and Bartlett, P.L. (2001) "Infinite-horizon policy-gradient estimation", *Journal of Artificial Intelligence Research*, 15:319-350.

[4] Baxter, J., Bartlett, P.L. and Weaver, L. (2001) "Experiments with infinite-horizon, policy-gradient estimation", *Journal of Artificial Intelligence Research*, 15:351-381.

[5] Bertsekas, D.P. (2001) *Dynamic Programming and Optimal Control*, second edition, Athena Scientific, Belmont, MA.

[6] Bertsekas, D.P. and Gallager, R. (1991) *Data Networks*, Prentice Hall, New Jersey.

[7] Bertsekas, D.P. and Tsitsiklis J.N. (1996) *Neuro-Dynamic Programming*, Athena Scientific, Belmont, MA.

[8] Bhatnagar, S. and Borkar, V.S. (1997) "Multiscale stochastic approximation for parametric optimization of hidden Markov models", *Probability in the Engineering and Informational Sciences*, 11:509-522.

[9] Bhatnagar, S. and Borkar, V.S. (1998) "A two time scale stochastic approximation scheme for simulation based parametric optimization", *Probability in the Engineering and Informational Sciences*, 12:519-531.

[10] Bhatnagar, S. and Borkar, V.S. (2003) "Multiscale chaotic SPSA and smoothed functional algorithms for simulation optimization", *Simulation*, 79(10):568-580.

[11] Bhatnagar, S., Fu, M.C., Marcus, S.I. and Bhatnagar, S. (2001) "Two timescale algorithms for simulation optimization of hidden Markov models", *IIE Transactions*, 33(3):245-258.

[12] Bhatnagar, S., Fu, M.C., Marcus, S.I. and Wang, I-J. (2003) "Two-Timescale simultaneous perturbation stochastic approximation using deterministic perturbation sequences", *ACM Transactions on Modelling and Computer Simulation*, 13(2):180-209.

[13] Bhatnagar, S. and Kumar, S. (2004) "A simultaneous perturbation stochastic approximation based actor-critic algorithm for Markov decision processes", *IEEE Transactions on Automatic Control*, 49(4):592-598.

[14] Bhatnagar, S. (2005) "Adaptive multivariate three-timescale stochastic approximation algorithms for simulation based optimization", *ACM Transactions on Modeling and Computer Simulation*, 15(1):74-107.

[15] Borkar, V.S. (1997) "Stochastic approximation with two timescales", *System and Control Letters*, 29:291-294.

[16] Borkar, V.S. (1998) "Asynchronous stochastic approximations", *SIAM J. Control and Optimization*, 36:840-851. (Erratum in *ibid.* (2000), 38:662-663.)

[17] Borkar, V.S. (2001) "A sensitivity formula for risk-sensitive cost and the actor-critic algorithm", *Systems and Control Letters*, 44:339-346.

[18] Borkar, V.S. (2002) "Q-learning for risk-sensitive control", *Mathematics of Operations Research*, 27:294-311.

[19] Borkar, V.S. (2003) "Avoidance of traps in stochastic approximation", *Systems and Control Letters*, 50:1-9 and *ibid.*, 55(2):174-175, Feb. 2006.

[20] Borkar, V.S. and Meyn, S.P. (2002) "Risk-sensitive optimal control for Markov decision processes with monotone cost", *Mathematics of Operations Research*, 27:192-209.

[21] Borkar, V.S., Juneja, S. and Kherani, A.A. (2004) "Performance analysis conditioned on rare events: an adaptive simulation scheme", *Communications in Information and Systems*, 3:259-278.

[22] Brandiere, O. (1998) "Some pathological traps for stochastic approximation", *SIAM J. Contr. and Optim.*, 36:1293-1314.

[23] Bucklew, J. (1990) *Large Deviations Techniques in Decision, Simulation and Estimation*, John Wiley, New York.

[24] Cao, X.-R. (1998) "The relations among potentials, perturbation analysis, and Markov decision processes", *Discrete Event Dynamic Systems*, 8:71-87.

[25] Cao, X.-R. and Guo, X. (2004) "A unified approach to Markov decision problems and performance sensitivity analysis with discounted and average criteria: multichain cases", *Automatica*, 40:1749-1759.

[26] Chong, E.K.P. and Ramadge, P.J. (1994) Stochastic optimization of regenerative systems using infinitesimal perturbation analysis. *IEEE Trans. on Autom. Contr.*, 39(7):1400-1410.

[27] Hernández-Hernández, D. and Marcus, S.I. (1996) "Risk sensitive control of Markov processes in countable state space", *Systems and Control Letters*, 29:147-155 and *Corrigendum*, 34:105-106, 1998.

[28] Ho, Y.-C. and Cao, X.-R. (1991) *Perturbation Analysis of Discrete Event Dynamical Systems*, Kluwer, Boston.

[29] Konda, V.R. and Borkar, V.S. (1999) "Actor-critic like learning algorithms for Markov decision processes", *SIAM Journal on Control and Optimization*, 38(1):94-123.

[30] Law, A.M. and Kelton, W.D. (2000) *Simulation Modeling and Analysis*, 3rd edition, McGraw-Hill, New York.

[31] Marbach, P., Mihatsch, O. and Tsitsiklis, J.N. (2000) "Call admission control and routing in integrated services networks using neuro-dynamic programming", *IEEE J. Selected Areas in Communications*, 18(2):197-208.

[32] Marbach, P. and Tsitsiklis, J.N. (2001) "Simulation-based optimization of Markov reward processes" *IEEE Transactions on Automatic Control*, 46:191-209.

[33] Kontoyiannis, I. and Meyn, S.P. (2003) "Spectral theory and limit theorems for geometrically ergodic Markov processes", *Annals of Applied Probability*, 13:304-362.

[34] Nowe, A., Steenhaut, K., Fakir, M. and Veerbeck, K. (1998) "Q-learning for adaptive load based routing", *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, San Diego, California, USA.

[35] M.L.Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, John Wiley, New York, 1994.

[36] Rubinstein, R.Y. (1997) "Optimization of computer simulation models with rare events", *European Journal of Operations Research*, 19(1):89-112.

[37] Spall, J.C. (1992) "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation", *IEEE Trans. Autom. Contr.*, 37(3):332-341.

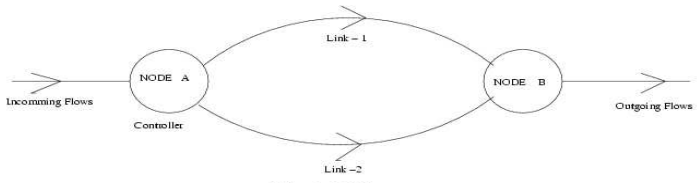
[38] Spall, J.C. (1997) "A one-measurement form of simultaneous perturbation stochastic approximation", *Automatica*, 33:109-112.

[39] Sutton, R. and Barto, A. (1998) *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA.

[40] Tsitsiklis, J.N. and Bertsekas, D.P. (1986) "Distributed asynchronous optimal routing in data networks", *IEEE Transactions on Automatic Control*, AC-31:325-332.

[41] Varadarajan, S., Ramakrishnan, N. and Thirunavukkarasu, M. (2003) "Reinforcing reachable routes", *Computer Networks*, 43(3):389-416.

[42] Watkins, C. and Dayan, P. (1992) "Q-learning", *Machine Learning*, 8:279-292.



Model Diagram

Figure 1: The Model

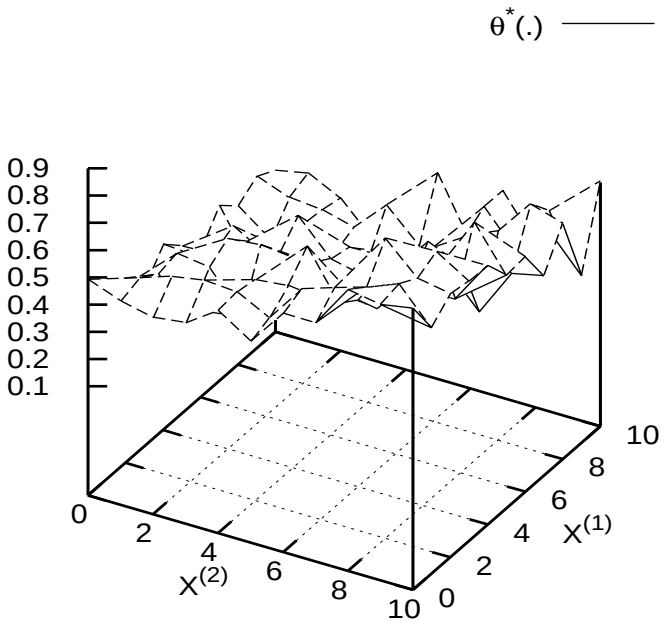


Figure 2: Setting (a): Optimal Policy $\theta^*(\cdot)$

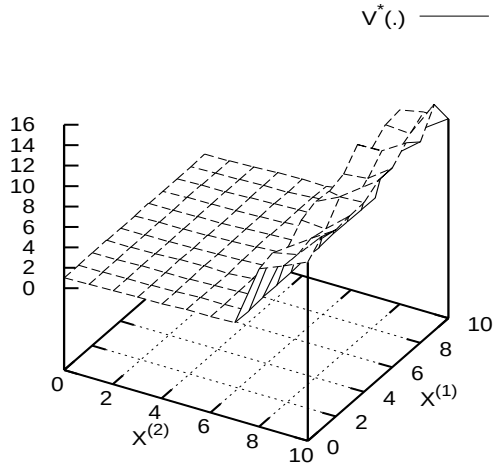


Figure 3: Setting (a): Value Function $V^*(\cdot)$

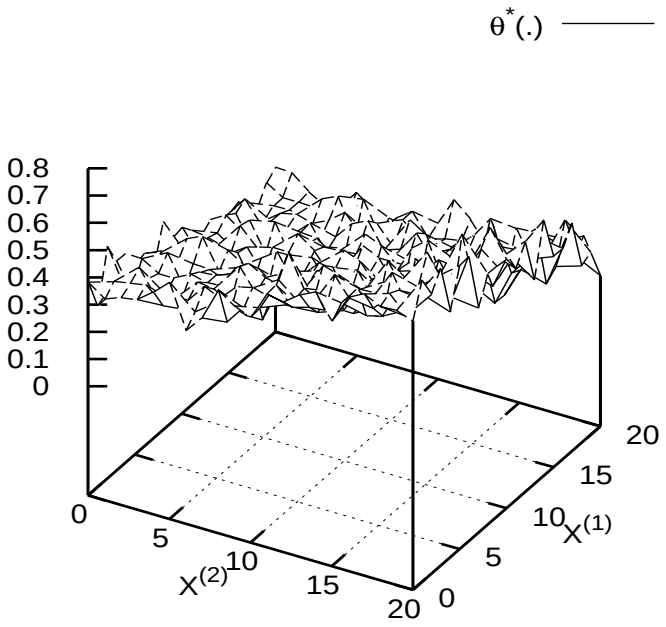


Figure 4: Setting (b): Optimal Policy $\theta^*(\cdot)$

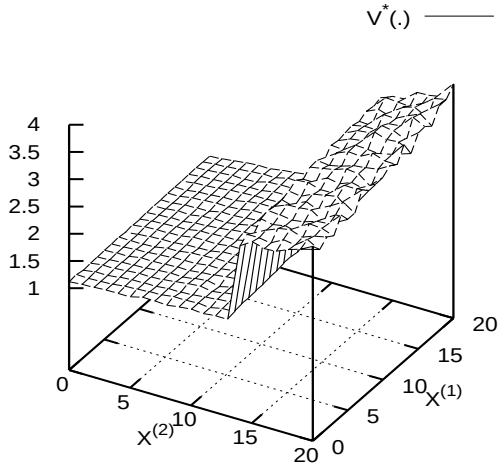


Figure 5: Setting (b): Value Function $V^*(\cdot)$

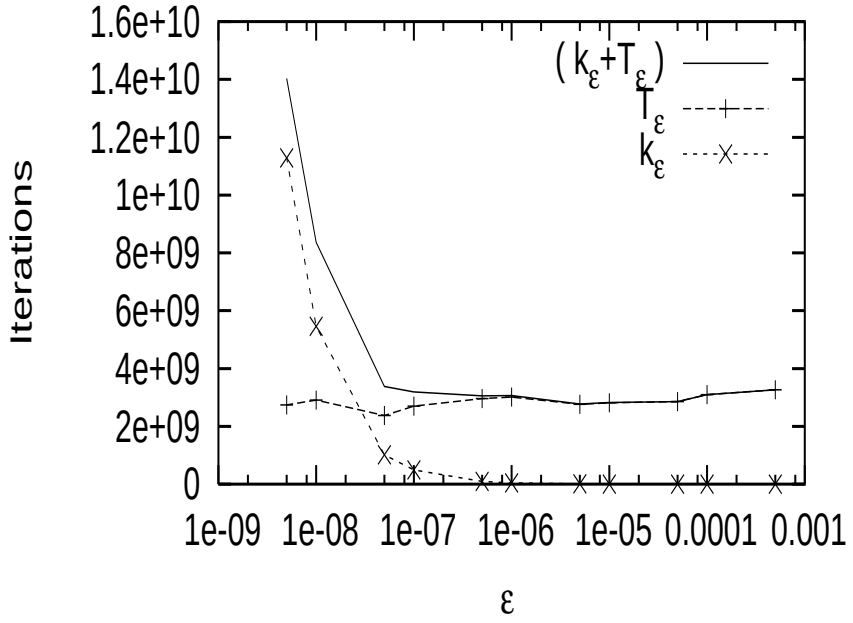


Figure 6: Setting (a): Plot of k_ϵ, T_ϵ and $(k_\epsilon + T_\epsilon)$ w.r.t. ϵ

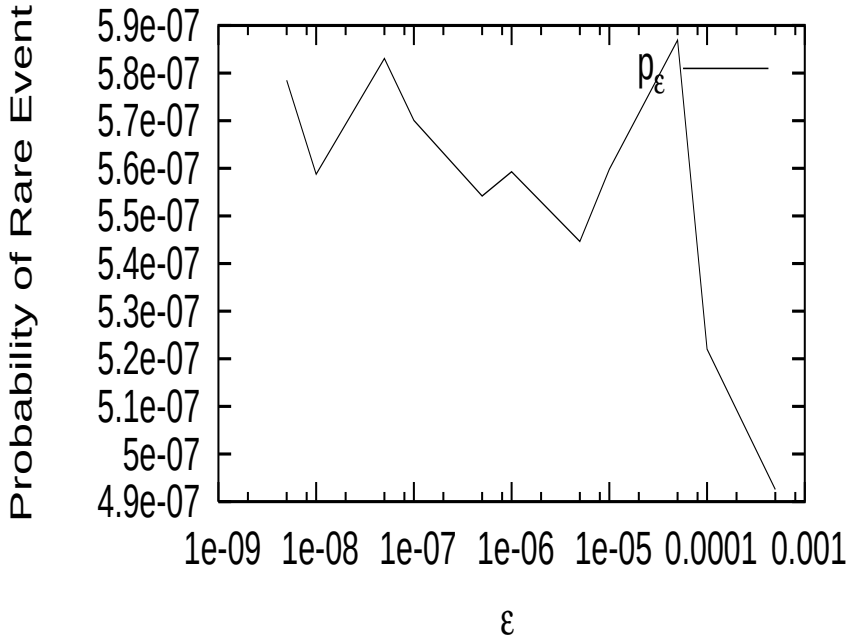


Figure 7: Setting (a): Variation of p_ϵ with ϵ

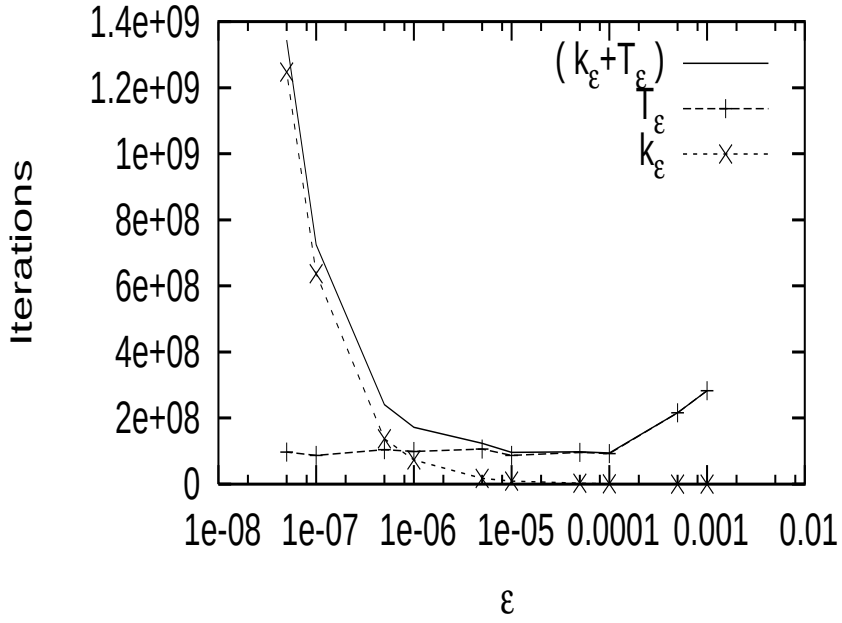


Figure 8: Setting (b): Plot of k_ϵ, T_ϵ and $(k_\epsilon + T_\epsilon)$ w.r.t. ϵ

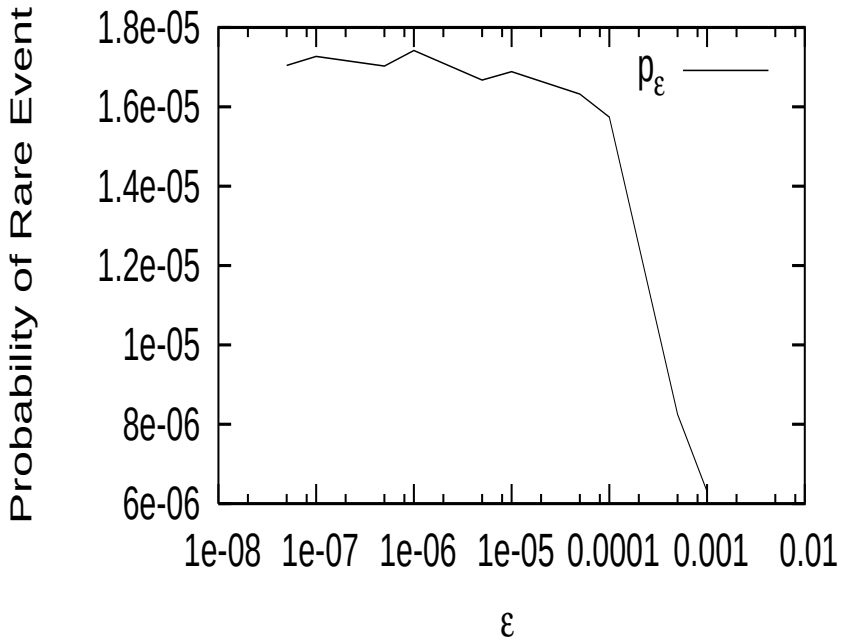


Figure 9: Setting (b): Variation of p_ϵ with ϵ