Available online at www.sciencedirect.com
SCIENCE DIRECT®
automatica
Automatica 42 (2006) 637–644
www.elsevier.com/locate/automatica

Brief paper

Actor-critic algorithms for hierarchical Markov decision processes[☆]

Shalabh Bhatnagar[a,*], J. Ranjan Panigrahi[b]

[a]*Department of Computer Science and Automation, Indian Institute of Science, Bangalore 560 012, India*
[b]*SoftJin Technologies Private Limited, Unit No.102, Mobius Tower, EPIP, White Field, Bangalore 560066, India*

Received 1 June 2004; received in revised form 4 May 2005; accepted 3 December 2005

Abstract

We consider the problem of control of hierarchical Markov decision processes and develop a simulation based two-timescale actor-critic algorithm in a general framework. We also develop certain approximation algorithms that require less computation and satisfy a performance bound. One of the approximation algorithms is a three-timescale actor-critic algorithm while the other is a two-timescale algorithm, however, which operates in two separate stages. All our algorithms recursively update randomized policies using the simultaneous perturbation stochastic approximation (SPSA) methodology. We briefly present the convergence analysis of our algorithms. We then present numerical experiments on a problem of production planning in semiconductor fabs on which we compare the performance of all algorithms together with policy iteration. Algorithms based on certain Hadamard matrix based deterministic perturbations are found to show the best results.
© 2006 Elsevier Ltd. All rights reserved.

Keywords: Hierarchical decision making; Learning algorithms; Markov decision processes; Stochastic approximation; Optimal control

1. Introduction

Markov decision processes (MDPs) (Bertsekas, 2001) form a general framework for the control of dynamic systems *under uncertainty*. However, traditional methodologies based on dynamic programming (DP) for solving MDPs suffer from the twin drawbacks of curse of dimensionality and lack of model information in most real world systems. Reinforcement learning (RL) (Bertsekas & Tsitsiklis, 1996) schemes have emerged in recent years as efficient alternatives to traditional solution methodologies for MDPs. These are applicable in scenarios where obtaining exact model information is hard; however, where transitions can be easily simulated. There are, however, situations where one is confronted with the problem of multiple decision makers with a clear hierarchy amongst them (Chang, Fard, Marcus, & Shayman, 2003; Forestier & Varaiya, 1978; Parr, 1998; Sutton, Precup, & Singh, 1999). The decisions made at the higher level (HL) have a greater consequence (in terms of the overall system performance) and are made much less often than those at the lower level (LL). In Parr (1998), MDPs with actions arranged in a hierarchy are considered. LL actions (LLA) in this take a unit time-step for execution; however, HL actions (HLA) are assumed to be ‘temporally abstract’ taking varying amounts of time. Similar ideas have also been used in Sutton et al. (1999) except for the difference that both HLA and LLA can be initiated at any time. In Forestier and Varaiya (1978), it is assumed that the HL controller issues a new stationary policy, each time the Markov chain visits a given subset of the state space, on the basis of which the LL controller chooses actions. In Chang et al. (2003), a framework for hierarchical control using a system of coupled MDPs that operate with different time schedules is provided. The transitions of the LL MDP occur every instant whereas those of the HL MDP occur once every T instants for some $T > 1$. Even though the transition dynamics of the LL MDP depends on the (current) state and action at both levels, that of the HL MDP is independent of those at the LL. However, the choice of HLA depends on the aggregate LL performance.

Reinforcement learning based actor-critic algorithms for MDPs have been studied in Bhatnagar and Kumar (2004), Konda and Borkar (1999), Konda and Tsitsiklis (2003). The

[☆] This paper was not presented at any IFAC meeting. This paper was recommended for publication in revised form by Associate Editor Xiren Cao under the direction of Editor Ian Petersen.

[*] Corresponding author. Fax: +91 80 236 02911.
E-mail addresses: shalabh@csa.iisc.ernet.in (S. Bhatnagar), jnana@softjin.com (J.R. Panigrahi).

0005-1098/$ - see front matter © 2006 Elsevier Ltd. All rights reserved.
doi:10.1016/j.automatica.2005.12.010

algorithm in Konda and Borkar (1999) performs policy iteration by using coupled stochastic recursions driven by different step-size schedules or timescales. The value function for a given policy update is iterated on the faster timescale while policy itself is updated on the slower one. In Bhatnagar and Kumar (2004), a two-timescale actor-critic algorithm for MDPs with finite state but compact (non-discrete) action set is proposed. This algorithm does gradient search using two-simulation, randomized difference simultaneous perturbation stochastic approximation (SPSA) (Spall, 1992) estimates on the slower timescale. The policy updates are performed in the space of stationary deterministic policies unlike (Konda & Borkar, 1999) that does so for stationary randomized policies. SPSA requires only two parallel simulations to estimate the performance gradient for any N-dimensional parameter vector. A one-simulation form of SPSA has also been proposed recently (Spall, 1997), however, which does not show good performance as regular SPSA. In Bhatnagar, Fu, Marcus, and Wang (2003), it is observed that the use of certain normalized Hadamard matrix based deterministic perturbations, in place of randomized, improves performance significantly in the case of one-simulation SPSA algorithms.

We develop in this paper certain actor-critic algorithms for hierarchical control. The basic framework we consider is similar to Chang et al. (2003). We assume both HL and LL MDPs to have finite state and action spaces. We first present a general two-timescale algorithm that requires an enumeration of all T-horizon LL policies for any given HL state (HLS) and action. For T large, we present two approximation (actor-critic) algorithms that update stationary randomized policies for both HL and LL MDPs. The first of these (AA1) is a three-timescale actor-critic algorithm. Here, on the fastest timescale, the value function for the LL MDP, and on the slowest one, the policies for both HL and LL MDPs are updated. Also, the HL value function is updated on the timescale that falls between the two timescales above. In the second approximation algorithm (AA2), the HL and LL recursions are run in separate stages. First, corresponding to each HLS and HLA pair, the optimal LL value function and policy are computed using an actor-critic algorithm for MDPs. Next, the optimal HL policy and value function are obtained using a similar algorithm, except that it takes as input the converged value estimates of the LL process. Since the algorithm is run in two different stages, the same set of step-size schedules can be used in both stages. All our algorithms use one-simulation and update in the space of stationary randomized policies. The slowest timescale recursions in our algorithms use one-simulation SPSA gradient estimates.

To the best of our knowledge, this is the first work that develops actor-critic algorithms for hierarchical MDPs. We give a sketch of convergence of our algorithms. We then present numerical experiments on a problem of production planning in semiconductor fabs using algorithms AA1 and AA2 under both randomized and Hadamard matrix based perturbations (for the SPSA estimates) with exact policy iteration. The use of Hadamard matrix based perturbations is seen to considerably improve performance in this setting. The rest of the paper is organized as follows: in Section 2, we briefly describe the framework for hierarchical MDPs. We then present our general adaptive algorithm and the approximation algorithms. This is followed by the convergence analysis. In Section 3, we present our numerical results.

2. Framework and the actor-critic algorithm

We consider a joint process $\{(Y_n^u, Y_n^l)\}$, $n \geqslant 0$, where $\{Y_n^u\}$ and $\{Y_n^l\}$ correspond to the HL and LL MDP, respectively, with state (action) spaces S^u and S^l (A^u and A^l), respectively. We assume that $S^u \cap S^l = \phi$. Further $A^u \cap A^l = \phi$. Let $A^u(x)$ ($A^l(y)$) denote the set of feasible actions in HLS x (LLS y). The LL MDP $\{Y_n^l\}$ is assumed to make transitions at instants $n = 0, 1, \ldots,$ while the HL MDP does so at instants $n = 0, T, 2T, \ldots,$ for some integer $T > 1$. Thus $Y_{mT}^u = Y_{mT+1}^u = \cdots = Y_{(m+1)T-1}^u$, $m \geqslant 0$. In general, T can be chosen to be a random variable, independent of the LLS and LLA. We assume that at instants mT, $m \geqslant 0$, the HL decisions are made prior to the LL ones by an infinitesimally small amount of time so that the LL decision maker has knowledge of HLS and HLA chosen at that instant. Let $\{Z_n^u\}$ ($\{Z_n^l\}$) denote the associated sequence of actions chosen at HL (LL). Also, let $p^u(y^u \mid x^u, z^u)$ ($p^l(y^l \mid x^l, z^l, x^u)$) denote the transition probabilities associated with the HL (LL) MDP. Here x^u and y^u (x^l and y^l) denote the HLS (LLS) at instants nT and $(n+1)T$ ($nT + j$ and $nT + j + 1$, $0 \leqslant j \leqslant n-1$), respectively.

Suppose $S \triangleq \{x = (x^u, x^l) \mid x^u \in S^u, x^l \in S^l$ s.t. x^l is a valid LLS corresponding to $x^u\}$. Then S is the set of all feasible HLS and LLS pairs. In the following, for any generic $x \in S$, x^u (x^l) shall correspond to its HL (LL) component. An admissible LL policy (LLP) $\gamma^l = \{\mu_0^l, \mu_1^l, \ldots\}$, is a sequence of functions $\mu_j^l : S \times A^u \to A^l$, $j \geqslant 0$, such that $\mu_j^l(x, a^u) \in A^l(x^l)$, $\forall x \in S$ and $a^u \in A^u(x^u)$. We define cyclic LLP γ^l as those for which $\mu_{nT+j}^l = \mu_{mT+j}^l$, $\forall n \neq m$, $j \in \{0, 1, \ldots, T-1\}$. By defining T-horizon policy segments (or simply T-segments) $d_n^l = \{\mu_{nT}^l, \mu_{nT+1}^l, \ldots, \mu_{(n+1)T-1}^l\}$, one can see that for a cyclic LLP, $d_n^l = d_m^l \equiv d^l$, $\forall n \neq m$. Thus a cyclic LLP may also be written as $\gamma^l = \{d^l, d^l, \ldots\}$, or one with time stationary T-segments. We say that $\gamma^u = \{\mu_0^u, \mu_1^u, \ldots\}$ is an admissible HL policy (HLP) if $\mu_j^u : S \to A^u$, $j \geqslant 0$, are such that $\mu_j^u(x) \in A^u(x^u)$, $\forall x \in S$. A stationary HLP is one with $\mu_j^u = \mu_k^u \equiv \mu^u$, $\forall j \neq k$. By a standard abuse of notation, we simply call μ^u (d^l) to be the stationary HLP (cyclic LLP). Let $R^l : S \times A^u \times A^l \to \Re$ and $K : S^u \times A^u \to \Re$ be the LL cost function and the immediate HL cost, respectively, that are both assumed to be nonnegative and bounded. Suppose $\alpha \in (0, 1]$. The expected single period HL cost is then given by

$$\begin{aligned} &R^u(x, a^u, d^l) \\ &\quad = E\left[\sum_{t=nT}^{(n+1)T-1} \alpha^{f(t)} R^l(x_t, a^u, \mu_t^l(x_t, a^u)) \,\middle|\, x_{nT} = x \right] \\ &\qquad + K(x^u, a^u), \end{aligned}$$

where for $r \in \{0, 1, \ldots, T-1\}$, $f(nT + r) = r$, $n \geqslant 0$. The expectation above is taken over all sample trajectories from $x \equiv (x^u, x^l)$ over T time steps when actions $\{a^u, d^l\}$ are used.

The aim is to find the optimal $\{\mu^u, d^l\}$ pair that attains $V^\star(x)$, $x \in S$, defined as

$$V^\star(x) = \min_{\mu^u} \min_{d^l} E\left[\sum_{n=0}^{\infty} \gamma^n R^u(x_{nT}, \mu^u(x_{nT}), d^l) \middle| x_0 = x \right],$$

with $0<\gamma<1$ as the discount factor. For given $x^u \in S^u$ and $a^u \in A^u(x^u)$, let $D^l(x^u, a^u)$ denote the set of all cyclic (T-horizon) policies in which d^l takes values. Suppose $p^T(\cdot \mid \cdot)$ represents the T-step transition probability of the LL process.

In Chang et al. (2003), the concept of an initialization function is used. The idea is that the state of the LL MDP is reinitialized using this function, once every T time epochs. This can be useful in cases where the 'next' HLS–LLS pair obtained after T epochs, does not lie in S. Suppose

$$\lambda_{d^l}(x, a^u; y^l) = \sum_{z \in S^l} p^{T-1}(z \mid x, a^u, d^l) \rho_{x^u, a^u}(y^l \mid z),$$

where $\rho_{x^u,a^u}(y^l \mid z)$ corresponds to the probability of choosing LLS y^l at some instant (say) nT ($n \geqslant 1$), given that the LLS at instant $nT-1$ is z, and HLS and HLA at instant $(n-1)T$ are x^u and a^u, respectively; see Chang et al. (2003) for a discussion on the possible forms of the initialization function. For our numerical experiments (Section 4), we simply choose the LLS at instants nT, $n \geqslant 0$, according to a uniform distribution on the set $\{\hat{y} \mid (y^u, \hat{y}) \in S\}$, where y^u is the 'next' HLS after x^u. One can write the Bellman equation in terms of λ_{d^l} as

$$V^\star(x) = \min_{a^u \in A^u(x^u)} \min_{d^l \in D^l(x^u, a^u)} \left[R^u(x, a^u, d^l) + \gamma \sum_{y^u \in S^u} \sum_{y^l \in S^l} \lambda_{d^l}(x, a^u; y^l) p^u(y^u \mid x^u, a^u) V^\star(y) \right]. \quad (1)$$

We now present an actor-critic algorithm for this problem. Suppose Φ^u and Φ^l are stationary randomized policies for the HL and LL MDPs. For notational simplicity, we assume in the following that the cardinalities of the sets $A^u(x^u)$ and $D^l(x^u, a^u)$ are constants $(N+1)$ and $(M+1)$, respectively. Let us enumerate the set of feasible actions $A^u(x^u)$ as $A^u(x^u) = \{a_x^0, a_x^1, \ldots, a_x^N\}$. Similarly, let $D^l(x^u, a^u) = \{d_{x,a}^0, d_{x,a}^1, \ldots, d_{x,a}^M\}$. Let T_P, $\hat{T}_P$, denote the simplexes

$$T_P = \left\{ (y^1, \ldots, y^N) \middle| y^j \geqslant 0, j = 1, 2, \ldots, N, \sum_{j=1}^N y^j \leqslant 1 \right\},$$

$$\hat{T}_P = \left\{ (y^1, \ldots, y^M) \middle| y^j \geqslant 0, j = 1, 2, \ldots, M, \sum_{j=1}^M y^j \leqslant 1 \right\},$$

respectively. Let $\Gamma : \Re^N \to T_P$ ($\hat{\Gamma} : \Re^M \to \hat{T}_P$) denote the projection from $\Re^N$ ($\Re^M$) to T_P ($\hat{T}_P$). The policy Φ^u (Φ^l) can be identified with vector of probabilities $\hat{\pi}^u = [[\pi^u(x; a^u)]]$, $x \in S$, $a^u \in A^u(x^u) \backslash \{a_x^0\}$ ($\hat{\pi}^l = [[\pi^l(x, a^u; d^l)]]$, $x \in S$, $a^u \in A^u(x^u)$, $d^l \in D^l(x^u, a^u) \backslash \{d_{x,a}^0\}$). The probability $\pi^u(x; a_x^0)$ ($\pi^l(x, a^u; d_{x,a}^0)$) gets automatically specified from the knowledge of Φ^u (Φ^l). Our algorithm updates stationary randomized policies with T_P, $\hat{T}_P$, as the relevant constraint sets. Let $\hat{\pi}_n^u$ and $\hat{\pi}_n^l$ denote the nth updates of $\hat{\pi}^u$ and $\hat{\pi}^l$. We consider variables $\Delta_n^u(x; a)$ and $\Delta_n^l(x, a^u; d)$, $n \geqslant 0$ that are used to perturb the probability vectors $\hat{\pi}_n^u$ and $\hat{\pi}_n^l$ so as to obtain optimal policies via a gradient search procedure. Suppose $\hat{\Delta}_n^u(x) = (\Delta_n^u(x; a_x^j),\ j = 1, \ldots, N)$, $x \in S$ and $\hat{\Delta}_n^l(x, a^u) = (\Delta_n^l(x, a^u;\ d_{x,a}^j),\ j = 1, \ldots, M)$, $x \in S$, $a^u \in A^u(x^u)$, respectively. We use one of the two constructions below for obtaining these variables.

Construction (A): randomized perturbations

$\Delta_n^u(x; a^u)$ and $\Delta_n^l(x, a^u; d^l)$, are independent, symmetric, mean-zero, ± 1-valued, Bernoulli distributed random variables. More general distributions can however be considered for these variables (see Spall, 1992).

Construction (B): deterministic perturbations

Let $\Delta_n^u(x; a^u)$ and $\Delta_n^l(x, a^u; d^l)$, be ± 1-valued variables. Exact values of these for any given n are obtained using an appropriate normalized Hadamard matrix based construction as in Bhatnagar et al. (2003). An $m \times m$ ($m \geqslant 2$) matrix H is said to be a normalized Hadamard matrix of order m if its entries belong to $\{1, -1\}$ and $H^T H = m I_m$, where I_m is the $m \times m$ identity matrix. Further, all the elements in the first column of this matrix are 1. For general $k > 1$, H_{2^k} can be obtained as $H_{2^k}(1,1) = H_{2^k}(1,2) = H_{2^k}(2,1) = H_{2^{k-1}}$ and $H_{2^k}(2,2) = -H_{2^{k-1}}$ as the four block elements each of dimension $2^{k-1} \times 2^{k-1}$. Also, the elements of H_2 are $H_2(1,1) = H_2(1,2) = H_2(2,1) = 1$ and $H_2(2,2) = -1$. For an N-dimensional vector, $\hat{\Delta}_n^u(x)$, the value of k used is $k = \lceil \log_2(N+1) \rceil$. Next remove the first column from the normalized Hadamard matrix constructed above and pick any N of the remaining $(P-1)$ columns and all P ($P = 2^k$) rows to form a new matrix, R, whose rows are denoted $R_0, \ldots, R_{P-1}$. Finally, the perturbations $\hat{\Delta}_n^u(x)$, $n \geqslant 0$, are obtained as $\hat{\Delta}_n^u(x) = R_{n \bmod P}$. The perturbations $\hat{\Delta}_n^l(x, a^u)$, $n \geqslant 0$, are obtained in a similar manner.

2.1. The general actor-critic algorithm

In what follows, for ease of notation, we denote $\hat{\Delta}_n^u \equiv \hat{\Delta}_n^u(x)$ and $\hat{\Delta}_n^l \equiv \hat{\Delta}_n^l(x, a^u)$. Let $\{\hat{\Delta}_n^u\}$ and $\{\hat{\Delta}_n^l\}$ be obtained using one of the constructions above (the same construction is however used for both sequences). Suppose $\delta > 0$ is a given small constant. Let $\bar{\pi}_n^u(x) = \Gamma(\hat{\pi}_n^u(x) - \delta \hat{\Delta}_n^u)$ and $\bar{\pi}_n^l(x, a^u) = \hat{\Gamma}(\hat{\pi}_n^l(x, a^u) - \delta \hat{\Delta}_n^l)$, respectively. In particular, we write $\bar{\pi}_n^u(x)$ (resp. $\bar{\pi}_n^l(x, a^u)$) as $\bar{\pi}_n^u(x) = (\bar{\pi}_n^u(x;\ a_x^j),\ j = 1, \ldots, N)^T$ (resp. $\bar{\pi}_n^l(x, a^u) = (\bar{\pi}_n^l(x, a^u;\ d_{x,a}^j),\ j = 1, \ldots, M)^T$). Let $\{\eta_n^u\}$ be a sequence of random variables with distribution $p^u(. \mid x^u, a^u)$. Let d^l denote the cyclic policy $\{\mu_0^l, \mu_1^l, \ldots, \mu_{T-1}^l\}$. For a given $j = 0, 1, \ldots, T-2$, let $\{\eta_{n,j}^l\}$ be a sequence of i.i.d. random variables with distribution $p^l(. \mid x, a^u, \mu_j^l(x, a^u))$. Also for

$j=T-1$, suppose $\{\eta^l_{n,T-1}\}$ be a sequence of i.i.d. random variables with distribution $\rho_{x^u,a^u}(. \mid x^l)$. Further we assume that all the above (T) sequences of random variables are mutually independent. Random variables η^u_n and $\eta^l_{n,j}$, $0 \leqslant j \leqslant T-1$, are used to simulate the next states of the corresponding HL and LL MDPs.

Suppose $L \geqslant 1$ is a given integer. Denote $\tilde{\pi}^u_n(x) = \hat{\pi}^u_{[n/L]}(x)$, $\tilde{\pi}^l_n(x,a^u) = \hat{\pi}^l_{[n/L]}(x,a^u)$, $\tilde{\Delta}^u_n = \hat{\Delta}^u_{[n/L]}$ and $\tilde{\Delta}^l_n = \hat{\Delta}^l_{[n/L]}$, respectively, $n \geqslant 0$. Let $\psi^u_n(\psi^l_n)$ be an $A^u(x^u)(D^l(x^u,a^u))$ valued random variable with conditional distribution given $\sigma(V_j(\cdot), \tilde{\pi}^u_j(\cdot), \tilde{\Delta}^u_j, \tilde{\pi}^l_j(\cdot,\cdot), \tilde{\Delta}^l_j, \eta^u_j,\ \eta^l_{j,r},\ j \leqslant n,\ 0 \leqslant r \leqslant T-1)$ being $\bar{\pi}^u_{[n/L]}(x)$ $(\bar{\pi}^l_{[n/L]}(x,a^u))$ on the set $A^u(x^u)\backslash\{a^0_x\}$ $(D^l(x^u,a^u)\backslash\{d^0_{x,a}\})$ with $\bar{\pi}^u_{[n/L]}(x;a^0_x)$ $(\bar{\pi}^l_{[n/L]}(x,a^u;d^0_{x,a}))$ automatically specified. Here $V_j(\cdot)$ is as in the algorithm below. Random variables ψ^u_n and ψ^l_n are used to simulate the actions chosen by the HL and LL MDPs according to the given randomized policies. Note that here too we suppress dependence of η^u_n, ψ^u_n etc. on the state/action variables to simplify notation. Define $(\hat{\Delta}^u_n)^{-1} = (1/\Delta^u_n(x;a^j_x),\ j=1,\ldots,N)$. Similarly let $(\hat{\Delta}^l_n)^{-1} = (1/\Delta^l_n(x,a^u;\ d^j_{x,a}),\ j=1,\ldots,M)$. Let $\{b(n)\}$ and $\{c(n)\}$ be two step-size sequences that satisfy

$$\sum_n b(n) = \sum_n c(n) = \infty, \quad \sum_n (b(n)^2 + c(n)^2) < \infty,$$
$$c(n) = o(b(n)). \tag{2}$$

Consider the hierarchical MDP (Y^u_n, Y^l_n), $n \geqslant 0$ that takes values in S and is governed according to the randomized policy pair $(\bar{\pi}^u_{[n/L]}, \bar{\pi}^l_{[n/L]})$. Let $V_0(x) = 0\ \forall x \in S$. Then,

$$\hat{\pi}^u_{n+1}(x) = \Gamma\left(\hat{\pi}^u_n(x) + c(n)\frac{V_{nL}(x)}{\delta}(\hat{\Delta}^u_n)^{-1}\right), \tag{3}$$

$$\hat{\pi}^l_{n+1}(x,a^u) = \hat{\Gamma}\left(\hat{\pi}^l_n(x,a^u) + c(n)\frac{V_{nL}(x)}{\delta}(\hat{\Delta}^l_n)^{-1}\right), \tag{4}$$

where for $m = 0,1,\ldots,L-1$,

$$V_{nL+m+1}(x) = (1-b(n))V_{nL+m}(x) + b(n)$$
$$(\tilde{R}^u(x,\psi^u_{nL+m},\psi^l_{nL+m}) + \gamma V_{nL+m}(\eta^u_{nL+m},\eta^l_{nL+m})). \tag{5}$$

In the above, η^l_n are independent random variables with distribution $\lambda_{d^l}(x,\ a^u;.)$. These are simulated using $\eta^l_{n,j}$, $j=0,1,\ldots,T-1$. Also, the estimate of R^u is

$$\tilde{R}^u(x,a^u,\psi^l_{nL+m}) = \sum_{j=0}^{T-1} \alpha^j R^l(x_j,a^u,\psi^l_{nL+m,j}) + K(x^u,a^u).$$

Here a^u is the HLA in state x chosen according to ψ^u_{nL+m} and is simulated only once. Also, $\psi^l_{nL+m,j}$ simulates the LLA $\mu^l_{(nL+m)T+j}(x_j,a^u)$ used at instant $(nL+m)T+j$. Further, x^l_j, $0 \leqslant j \leqslant T-1$, are defined recursively according to $x^l_0 = x^l$ and for $j \geqslant 1$, $x^l_j = \eta^l_{(nL+m)T,j}$ that in turn depends on x^l_{j-1}. The random variables $\eta^u_n, \eta^l_n, \psi^u_n, \psi^l_n$, etc. serve as estimators of conditional expectations, w.r.t. their respective distributions, of the corresponding quantities in whose arguments they appear. The averaging itself is, however, achieved using the appropriate stochastic approximation recursions.

2.2. Approximation algorithms

For ease of computation, we assume as in Chang et al. (2003) that the initialization function is independent of LLP. Thus at the LL, given $x_{nT} = x$ and $a^u_{nT} = a^u$, the goal is to find $R^\star(x,a^u) = \min_{d \in D^l(x^u,a^u)} R^u(x,a^u,d)$, which corresponds to solving a T-horizon problem for given x and a^u. Note that we do not consider the (original) value function $V^\star(\cdot)$ in the minimization above. Here, we consider a scenario where T is large and $\alpha < 1$. We propose to approximate in this case the T-horizon LL MDP with an infinite horizon MDP. The solution methodology would then give an optimal LLP that is stationary (instead of cyclic). The Bellman equation of optimality can be written as

$$V^\star(x) = \min_{a^u \in A^u(x^u)} \left[R^\star(x,a^u) + \gamma \sum_{y^u \in S^u} \sum_{y^l \in S^l} \hat{\lambda}'(x,a^u;y^l)p^u(y^u \mid x^u,a^u)V^\star(y) \right]. \tag{6}$$

Here $\hat{\lambda}'$ is the initialization function that is independent of LLP. The Bellman equation for the LL MDP is

$$V^{l,\star}(x,a^u) = \min_{a^l \in A^l(x^l)} [R^l(x,a^u,a^l) + \alpha \sum_{y^l \in S^l} p^l(y^l \mid x,a^u,a^l)V^{l,\star}(y,a^u)], \tag{7}$$

where $y = (y^u, y^l)$ with $y^u = x^u$. We have

$$R^\star(x,a^u) = V^{l,\star}(x,a^u) + K(x^u,a^u). \tag{8}$$

We now propose two actor-critic approximation algorithms for this setting. Both of these algorithms use the fact that solution of (6) requires (via (8)) the solution of (7). Our first approximation algorithm (AA1) uses three timescales since (cf. (6)–(8)) the upper-level value update requires the converged value of the LL value function. Hence, the latter is updated on a faster timescale than the former. Moreover, both HLP and LLP are updated on the slowest scale. The next algorithm (AA2) on the other hand, handles the HL and LL problems in two separate stages. In the first stage, for each upper-level state and action pair, a two-timescale actor-critic algorithm is used for computing the optimal LLP and value function. Next, another two-timescale actor-critic algorithm is used for the upper-level problem. For simplicity, let us assume that each set $A^l(x^l)$ has the form $A^l(x^l) = \{b^0_x,\ldots,b^R_x\}$ and has exactly $(R+1)$ elements. Let $\bar{T}_P$ denote the simplex

$$\bar{T}_P = \left\{ (y^1,\ldots,y^R) \mid y^j \geqslant 0,\ j=1,2,\ldots,R,\ \sum_{j=1}^R y^j \leqslant 1 \right\}.$$

Let $\bar{\Gamma} : \Re^R \to \bar{T}_P$ denote the projection map from $\Re^R$ to $\bar{T}_P$. We simply denote the LLP (in this case) by $\hat{\pi}^l = [[\pi^l(x, a^u; b^l)]]$, $x \in S$, $a^u \in A^u(x^u)$, $b^l \in A^l(x^l) \backslash \{b_x^0\}$. Here $\pi^l(x, a^u; b^l)$ is the probability of choosing LLA b^l in state $x \in S$ when the HLA is a^u. Also, $\pi^l(x, a^u; b_x^0)$ gets automatically specified. Let $\hat{\pi}_n^l$ denote the nth update of $\hat{\pi}^l$. Consider now perturbation variables (by abuse of notation) $\Delta_n^l(x, a^u; b)$, $x \in S$, $a^u \in A^u(x^u), b \in A^l(x^l), n \geqslant 0$, obtained using one of the two constructions above. Let $\hat{\Delta}_n^l = (\Delta_n^l(x, a^u; b_x^j), 1 \leqslant j \leqslant R)$ and $\bar{\pi}_n^l(x, a^u) = \bar{\Gamma}(\hat{\pi}_n^l(x, a^u) - \delta\hat{\Delta}_n^l)$, respectively. Denote $\bar{\pi}_n^l(x, a^u) \equiv (\bar{\pi}_n^l(x, a^u; b_x^j), 1 \leqslant j \leqslant R)$. Let $\{\xi_n^l\}$, be a sequence of independent random variables with distribution $p^l(\cdot \mid x, a^u, \mu^l(x, a^u))$. Here μ^l corresponds to the stationary LLP. Suppose for $n \geqslant 0$, $\tilde{\pi}_n^l(x, a^u) = \hat{\pi}_{[n/L]}^l(x, a^u)$ and $\tilde{\Delta}_n^l = \hat{\Delta}_{[n/L]}^l$, respectively. Let $\{\hat{\eta}_n^l\}$ be a sequence of independent random variables with distribution $\hat{\lambda}'(x, a^u; \cdot)$ over S^l. Let χ_n^l be an $A^l(x^l)$-valued random variable with conditional distribution given $\sigma(\hat{V}_j^l(\cdot, \cdot), \hat{V}_j^u(\cdot), \tilde{\pi}_j^u(\cdot), \tilde{\Delta}_j^u, \tilde{\pi}_j^l(\cdot, \cdot), \tilde{\Delta}_j^l, \eta_j^u, \xi_j^l, \hat{\eta}_j^l, j \leqslant n)$ being $\bar{\pi}_{[n/L]}^l(x, a^u)$ on the set $A^l(x^l) \backslash \{b_x^0\}$. Here $\tilde{\pi}_j^u(\cdot)$, $\tilde{\Delta}_j^u$, η_j^u and ψ_j^u are defined as before.

2.2.1. Approximation Algorithm 1 (AA1)

Let

$$(\hat{\Delta}_n^l)^{-1} = \left(\frac{1}{\Delta_n^l(x, a^u; b_x^1)}, \ldots, \frac{1}{\Delta_n^l(x, a^u; b_x^R)}\right)^{\mathrm{T}}.$$

Consider positive real numbers $\{a(n)\}$ satisfying

$$\sum_n a(n) = \infty, \quad \sum_n a(n)^2 < \infty, \quad b(n) = o(a(n)).$$

Let $\hat{V}_0^u(x) = \hat{V}_0^l(x, a^u) = 0 \ \forall x \in S, \ a^u \in A^u(x^u)$. Then,

$$\hat{\pi}_{n+1}^l(x, a^u) = \bar{\Gamma}\left(\hat{\pi}_n^l(x, a^u) + c(n)\frac{\hat{V}_{nL}^l(x, a^u)}{\delta}(\hat{\Delta}_n^l)^{-1}\right), \tag{9}$$

$$\hat{\pi}_{n+1}^u(x) = \Gamma\left(\hat{\pi}_n^u(x) + c(n)\frac{\hat{V}_{nL}^u(x)}{\delta}(\hat{\Delta}_n^u)^{-1}\right), \tag{10}$$

where for $m = 0, 1, \ldots, L-1$,

$$\begin{aligned}\hat{V}_{nL+m+1}^l(x, a^u) = \hat{V}_{nL+m}^l(x, a^u) + a(n)(R^l(x, a^u, \chi_{nL+m}^l) \\ + \alpha\hat{V}_{nL+m}^l((x^u, \xi_{nL+m}^l), a^u) \\ - \hat{V}_{nL+m}^l(x, a^u)),\end{aligned} \tag{11}$$

$$\begin{aligned}\hat{V}_{nL+m+1}^u(x) = (1 - b(n))\hat{V}_{nL+m}^u(x) \\ + b(n)(\hat{V}_{nL+m}^l(x, \psi_{nL+m}^u) \\ + K(x^u, \psi_{nL+m}^u) \\ + \gamma\hat{V}_{nL+m}^u(\eta_{nL+m}^u, \hat{\eta}_{nL+m}^l)).\end{aligned} \tag{12}$$

With AA1, one no longer needs to update policies in the set $D^l(x^u, a^u)$ as only stationary policies are considered and for which the set $A^l(x^l)$ suffices. Note that the cardinality of $D^l(x^u, a^u)$ increases exponentially with T. The infinite horizon approximation in (8) along with the use of different timescales allows one to use $\hat{V}_{nL+m}^l(x, a^u) + K(x^u, a^u)$ as an estimate for $R^\star(x, a^u)$ in (12). Suppose $V_T^{l,\star}(\cdot, \cdot)$ corresponds to the (actual) value function for the T-horizon LL MDP. Then it is easy to see that

$$\|V^{l,\star}(x, a^u) - V_T^{l,\star}(x, a^u)\| \leqslant \frac{\alpha^T G}{(1-\alpha)},$$

where

$$\|R^l(\cdot, \cdot, \cdot)\| \leqslant G < \infty.$$

Since,

$$V^\star(x) = \min E\left[\sum_{n=0}^{\infty} \gamma^n R^\star(Y_n, \mu^u(Y_n)\middle| Y_0 = x\right],$$

by (8), it is easy to see that $\|V_T^\star(x) - V^\star(x)\| \leqslant \alpha^T G/((1-\alpha)(1-\gamma))$, where $V_T^\star(x)$ is the optimal value function for the hierarchical MDP when the LL MDP problem has a time horizon of T. Thus for large T and $\alpha < 1$, it is reasonable to approximate the LL MDP with an infinite horizon, discounted cost problem.

2.2.2. Approximation Algorithm 2 (AA2)

We have similar recursions (here) as (9)–(12), except that these are run in two different stages. In the first stage, corresponding to each HLS and HLA pair, the optimal LL value function $V^{l,\star}(\cdot, \cdot)$ is computed using recursions (9) and (11), respectively. Next, recursions (10) and (12) are run, however, with $V^{l,\star}(\cdot, \cdot)$ in place of $\hat{V}_{nL+m}^l(\cdot, \cdot)$ in (12). One may choose $a(n) = b(n)$ here.

2.3. Convergence analysis

We sketch the analysis for the general algorithm. We first assume Construction (A). Let $G_n = \sigma(V_j(x), \tilde{\pi}_j^u(x), \tilde{\Delta}_j^u, \tilde{\pi}_j^l(x, a^u), \tilde{\Delta}_j^l, j \leqslant n; \eta_j^u, \eta_{j,k}^l, \eta_j^l, \psi_j^u, \psi_{j,k}^l, \psi_j^l, j < n, k = 0, 1, \ldots, T-1), n \geqslant 0$, denote a sequence of σ-fields. Note that

$$\begin{aligned}E[R^l(x_{kT}, \psi_k^u, \psi_{k,0}^l) \mid G_k, x_{kT} = x] \\ = \sum_{a^u \in A^u(x^u)} \sum_{d^l \in D^l(x^u, a^u)} \bar{\pi}_k^u(x; a^u)\bar{\pi}_k^l(x, a^u; d^l) \\ \times R^l(x, a^u, \mu_0^l(x, a^u)),\end{aligned}$$

$$\begin{aligned}E[R^l(x_{kT+1}, \psi_k^u, \psi_{k,1}^l) \mid G_k, x_{kT} = x] \\ = \sum_{a^u, d^l} \bar{\pi}_k^u(x; a^u)\bar{\pi}_k^l(x, a^u; d^l) \sum_{y \in S^l} p^l(y \mid x, a^u, \mu_0^l) \\ \times R^l((x^u, y), a^u, \mu_1^l(x, a^u)).\end{aligned}$$

Proceeding in this manner, one can see that

$$E[(\tilde{R}^u(x,\psi^u_k,\psi^l_k)+\gamma V_k(\eta^u_k,\eta^l_k)) \mid G_k] = \sum_{a^u,d^l}\bar{\pi}^u_k(x;a^u)\bar{\pi}^l_k(x,a^u;d^l)(R^u(x,a^u,d^l) + \gamma \sum_{r\in S^u,s\in S^l} p^u(r\mid x^u,a^u)\lambda_{d^l}(x,a^u;s) \times V_j((r,s))). \tag{13}$$

For $n \geqslant 0$, let $\tilde{b}(n)=b([n/L])$. Then $\sum_n \tilde{b}(n)=\infty$, $\sum_n \tilde{b}(n)^2 < \infty$ and $c(n)=o(\tilde{b}(n))$. The recursions (3)–(5) in the algorithm can now be rewritten using step-sizes $c(n)$ and $\tilde{b}(n)$. One can show as in Theorem 2.1 of Tsitsiklis (1994) that $\sup_k \|V_k(x)\| < \infty, \forall x \in S$. As a consequence of (13), one can form a suitable martingale sequence as the sum of terms involving the product of $\tilde{b}(n)$ with the difference of the term multiplying $b(n)$ on the RHS of (5) and the LHS of (13). The resulting martingale sequence can be seen to be convergent. Hence, the above terms asymptotically diminish. Note that (3)–(4) can be rewritten as

$$\hat{\pi}^u_{n+1}(x)=\Gamma(\hat{\pi}^u_n(x)+\tilde{b}(n)\xi^1(n)),$$

$$\hat{\pi}^l_{n+1}(x,a^u)=\hat{\Gamma}(\hat{\pi}^l_n(x,a^u)+\tilde{b}(n)\xi^2(n)),$$

where $\xi^1(n)$ and $\xi^2(n)$ are $o(1)$ since $c(n)=o(\tilde{b}(n))$. Now for given $\bar{\pi}^u$, $\bar{\pi}^l$, consider the following ODE: for $x \in S$,

$$\dot{V}_t(x)=\sum_{a^u,d^l}\bar{\pi}^u(x;a^u)\bar{\pi}^l(x,a^u;d^l)(R^u(x,a^u,d^l) + \gamma\sum_{r\in S^u,\ s\in S^l} p^u(r\mid x^u,a^u)\lambda_{d^l}(x,a^u;s) \times V_t(r,s)) - V_t(x). \tag{14}$$

It is easy to see that (14) is an asymptotically stable linear system with a unique fixed point corresponding to the solution of the Poisson equation: for $x \in S$,

$$\hat{V}(x)=\sum_{a^u,d^l}\bar{\pi}^u(x;a^u)\bar{\pi}^l(x,a^u;d^l)(R^u(x,a^u,d^l) + \sum_{r\in S^u,\ s\in S^l} p^u(r\mid x^u,a^u)\lambda_{d^l}(x,a^u;s) \times \hat{V}((r,s))). \tag{15}$$

Thus the faster timescale recursion (5), for given HLP and LLP updates, asymptotically tracks the solution of (15). The remainder of the analysis works towards showing that the slower timescale recursions (3)–(4) converge to corresponding optimal policies for the hierarchical MDP. For bounded, continuous $v(.)(w(.)) : \Re^N(\Re^M) \to \Re^N(\Re^M)$, define $\Gamma'(v(y)) = \lim_{\eta\downarrow 0}(\Gamma(y+\eta v(y))-\Gamma(y))/\eta(\bar{\Gamma}'(w(y)) = \lim_{\eta\downarrow 0}(\hat{\Gamma}(y+\eta w(y))-\hat{\Gamma}(y))/\eta)$. Consider the ODEs:

$$\dot{\hat{\pi}}^u_t(x)=\Gamma'(-\nabla_{\hat{\pi}^u_t(x)}V_{\hat{\pi}^u_t,\hat{\pi}^l_t}(x)), \tag{16}$$

$$\dot{\hat{\pi}}^l_t(x,a^u)=\bar{\Gamma}'(-\nabla_{\hat{\pi}^l_t(x,a^u)}V_{\hat{\pi}^u_t,\hat{\pi}^l_t}(x)). \tag{17}$$

Suppose $M=\{(\hat{\pi}^u,\hat{\pi}^l) \mid \Gamma'(\nabla_{\hat{\pi}^u(x)}V_{\hat{\pi}^u,\hat{\pi}^l}(x))=0,\ \forall x\in S,\ \bar{\Gamma}'(\nabla_{\hat{\pi}^l(x,a^u)}V_{\hat{\pi}^u,\hat{\pi}^l}(x))=0,\ \forall x\in S,\ a^u\in A^u(x^u)\}$. Suppose for $\varepsilon>0$, $M^\varepsilon=\{(\pi^u,\pi^l) \mid \exists(\hat{\pi}^u_0,\hat{\pi}^l_0)\in M$ with $\|(\pi^u,\pi^l)-(\hat{\pi}^u_0,\hat{\pi}^l_0)\|<\varepsilon\}$. Consider now σ-fields $\hat{G}_n=\sigma(V_{mL}(x), \hat{\pi}^u_m(x), \hat{\pi}^l_m(x,a^u),\ x\in S,\ a^u\in A^u(x^u),\ m\leqslant n;\ \hat{\Delta}^u_m,\ \hat{\Delta}^l_m,\ x\in S,\ a^u\in A^u(x^u),\ m<n)$, $n\geqslant 1$. We finally have

Theorem 1. *Given* $\varepsilon>0$, $\exists\delta_0>0$ *such that for all* $\delta\in(0,\delta_0]$, *the algorithm* (3)– (5) *converges to* M^ε *with probability one.*

Proof. Note that recursion (3) of the algorithm is asymptotically analogous to the recursion

$$\hat{\pi}^u_{n+1}(x)=\Gamma\left(\hat{\pi}^u_n(x)+c(n)E\left[\frac{V_{\bar{\pi}^u_n,\bar{\pi}^l_n}(x)}{\delta}(\hat{\Delta}^u_n)^{-1}\mid \hat{G}_n\right]\right).$$

Assume now that $\hat{\pi}^u_n(x)$ lies in the interior of the simplex T_P such that for δ small, $\Gamma(\hat{\pi}^u_n(x)-\delta\hat{\Delta}^u_n)=\hat{\pi}^u_n(x)-\delta\hat{\Delta}^u_n$. By a Taylor series expansion, we have

$$V_{\bar{\pi}^u_n,\bar{\pi}^l_n}(x)=V_{\hat{\pi}^u_n,\hat{\pi}^l_n}(x)-\delta\sum_{j=1}^{N}\Delta^u_n(x;a^j_x)\nabla_{\pi^u_n(x;a^j_x)}V_{\hat{\pi}^u_n,\hat{\pi}^l_n}(x) - \delta\sum_{l=1}^{M}\Delta^l_n(x,a^u;d^l_{x,a})\nabla_{\pi^l_n(x,a^u;d^l_{x,a})} \times V_{\hat{\pi}^u_n,\hat{\pi}^l_n}(x)+O(\delta^2).$$

It is now easy to verify from Construction (A) that

$$E\left[\left.\frac{V_{\bar{\pi}^u_n,\bar{\pi}^l_n}(x)}{\delta\Delta^u_n(x;a^u)}\right|\hat{G}_n\right] = -\nabla_{\pi^u_n(x;a^u)}V_{\hat{\pi}^u_n,\hat{\pi}^l_n}(x)+O(\delta),$$

for $a^u\in A^u(x^u)\backslash\{a^0_x\}$. A similar analysis can be seen to hold for recursion (4) of the algorithm as well. Finally, using standard arguments, it can be seen that $\sum_{x\in S}V_{\hat{\pi}^u_t,\hat{\pi}^l_t}(x)$ is a strict Liapunov function for (16)–(17). The claim follows by letting $\delta\to 0$. An argument similar to Bhatnagar and Kumar (2004) gives the claim for $\hat{\pi}^u_n$ on the boundary of simplex T_P. □

Consider now the case when Construction (B) is used for generating perturbations. The only changes needed are in the proof of Theorem 1 as the rest of the analysis proceeds along exactly the same lines. It can be shown as in Bhatnagar et al. (2003) that

$$\left\|\sum_{m=n}^{(n+P)}\frac{c(m)}{c(n)}\frac{V_{\hat{\pi}^u_n,\hat{\pi}^l_n}(x)}{\delta\Delta^u_n(x;a^k_x)}\right\|,$$

$$\left\|\sum_{m=n}^{(n+P)}\sum_{j=1,j\neq k}^{N}\frac{c(m)}{c(n)}\frac{\Delta^u_n(x;a^j_x)}{\Delta^u_n(x;a^k_x)}\nabla_{\pi^u_n(x;a^j_x)}V_{\hat{\pi}^u_n,\hat{\pi}^l_n}(x)\right\| \to 0 \quad \text{as } n\to\infty.$$

Note that perturbations $\Delta^u_n(x;a^k_x)$ are generated using a normalized Hadamard matrix with P rows while perturbations

$\Delta^l_n(x, a^l; d)$ are generated using a similar matrix with $\bar{P} = 2^{\lceil \log_2(M+1)\rceil}$ rows. However,

$$\frac{\Delta^l_n(x; a^u, d)}{\Delta^u_n(x; a^k_x)} = +1 \text{ or } -1$$

exactly half the number of times in a cycle with $P = \bar{P}$ if $\lceil\log_2(N+1)\rceil = \lceil\log_2(M+1)\rceil$. Otherwise, P is greater (lower) than $\bar{P}$ by at least a factor of 2 if $\lceil\log_2(N+1)\rceil > \lceil\log_2(M+1)\rceil$ ($\lceil\log_2(N+1)\rceil < \lceil\log_2(M+1)\rceil$). Thus one also obtains as in Bhatnagar et al. (2003) that

$$\left\| \sum_{m=n}^{(n+P)} \sum_{j=1}^{M} \frac{c(m)}{c(n)} \frac{\Delta^l_n(x, a^u; d^j_{x,a})}{\Delta^u_n(x; a^k_x)} \nabla_{\pi^l_n(x,a^u;d^j_{x,a})} V_{\hat{\pi}^u_n, \hat{\pi}^l_n}(x) \right\| \to 0 \quad \text{as } n \to \infty.$$

Thus, as $\delta \to 0$, (3) is analogous to

$$\hat{\pi}^u_{n+1}(x) = \Gamma(\hat{\pi}^u_n(x) - c(n)\nabla_{\hat{\pi}^u_n(x)} V_{\hat{\pi}^u_n, \hat{\pi}^l_n}(x)). \tag{18}$$

The rest follows as in Theorem 1. An analogous argument settles the claim for iteration (4). Finally, the analysis for the approximation algorithms can be shown in a similar manner with appropriate changes.

3. Numerical experiments

We consider the problem of planning and scheduling in a semiconductor fab. We consider a model similar to Bhatnagar, Fernandez-Gaucherand, Fu, Marcus, and He (1999), Panigrahi and Bhatnagar (2004). The HL decisions correspond to buying new and/or discarding old machines, while those at the LL correspond to capacity switch over from one type of production and/or operation to another. We assume that decisions on capacity switchover are made at each instant while those on buying/discarding are made once every T instants. In the numerical results in Bhatnagar et al. (1999), HL decisions are not considered for computational simplicity. In Panigrahi and Bhatnagar (2004), a similar framework as here is considered and TD(0) and Q-learning algorithms are applied for finding the optimal policy and value function pair. We refer the reader to Panigrahi and Bhatnagar (2004) for a detailed model description.

We consider a semiconductor fab manufacturing products of types A and B with each product requiring one 'litho' and one 'etch' operation that can be performed in any order. We assume that all machines are either litho or etch and can perform the corresponding operation on both products. Both types of machines require one (two) unit(s) of time on A (B). The HLS has the form (T_l, T_e). We assume that $T_l, T_e \in \{2, 3\}$. Thus the total number of HL states is 4. We also assume that at most one unit of litho and/or etch capacity can be bought or discarded at any instant. The HLA has the form (y_l, y_e) where $y_i = +1(-1)$ if type i capacity is bought (discarded). Further, $y_i = 0$ implies that the corresponding capacity is neither bought nor discarded. We also assume for simplicity that we do not buy and discard the same type of machine in a given period. The possible HLA are $a_1 = (1, 1)$, $a_2 = (1, 0)$, $a_3 = (1, -1)$, $a_4 = (0, 0)$, $a_5 = (0, 1)$, $a_6 = (0, -1)$, $a_7 = (-1, 0)$, $a_8 = (-1, 1)$ and $a_9 = (-1, -1)$, respectively. The LLS has four components—litho capacity for A ($X_{A,l}$), etch capacity for A ($X_{A,e}$), inventory of A (I_A) and inventory of B (I_B), respectively. The litho and etch capacities for B get automatically set. We assume that the inventories of A and B are constrained according to $I_A \in \{-1, 0, +1\}$ and $I_B \in \{-0.5, 0, +0.5\}$, respectively. We thus have a total of 81, 108, 108 and 144 LLS corresponding to the HLS (2, 2), (2, 3), (3, 2) and (3, 3), respectively. The optimal policy and value function is thus obtained for a total of 441 possible HLS and LLS combinations. We assume here that the demands D_A and D_B for A and B are independent random variables that are both distributed according to the Bernoulli distribution (though with different parameters).

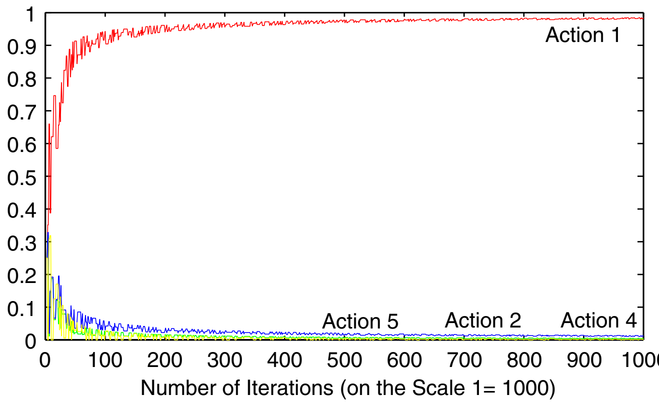

Fig. 1. Convergence of action selection probabilities in state (0,0) using AA2 with Hadamard matrix based perturbations.

For our experiments, we select $T = 100$. The cost for buying (discarding) unit capacity equals 100 (40). Further, $Pr(D_A = 1) = 0.4 = 1 - Pr(D_A = 2)$. Also, $Pr(D_B = 0.5) = 0.7 = 1 - Pr(D_B = 1)$. The unit inventory cost for A (B) equals 2 (1). The unit backlog cost for A (B) equals 10 (5). The unit operating cost on litho (etch) machines for both A and B is 0.2 (0.1). Also, the unit switch over cost on both litho and etch machines is 3. We use here a linear approximation architecture (Bertsekas & Tsitsiklis, 1996) for the value function for our algorithms AA1 and AA2, along with a feature based representation for states. We select a total of thirteen features for each HLS and LLS pair. The first feature chosen is 1 while the rest are the components of state values and their squares. The weight components are then updated instead of the value function estimates using the recursions in AA1 and AA2. We start both algorithms assuming equal probabilities for all actions in each state. We set $L = 10$, $c(n) = a/(n + b)$ and $b(n) = (a/(n + b))^{3/4}$, respectively, with $a, b > 0$. For algorithm AA1, we also set $a(n) = (a/(n + b))^{2/3}$. The values of δ, α and γ are chosen to be 0.1, 0.98 and 0.9, respectively. In Fig. 1, we show the convergence plot for the probabilities of choosing the various actions for a representative state when AA2 with Hadamard matrix based perturbations is used. Fig. 2 shows the optimal value function and policy obtained using the above algorithm while Fig. 3 shows the same for exact policy iteration. From Fig. 1, one can see that the probability of selecting the optimal action increases while

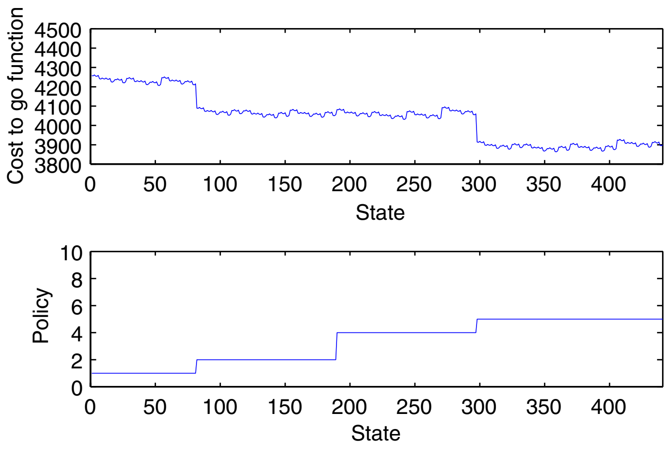

Fig. 2. Optimal value function and optimal policy using AA2 with Hadamard matrix based perturbations.

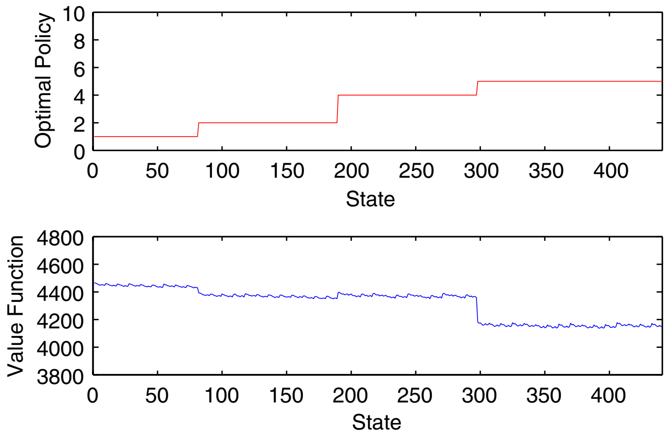

Fig. 3. Optimal value function and optimal policy with exact policy iteration.

the corresponding probabilities for all other feasible actions decrease with the number of iterations. Similar behaviour is also seen in the case of other algorithms as well. However, algorithms with randomized perturbations exhibit large oscillations unlike those with Hadamard matrix based perturbations. The latter show graceful and fast convergence. We do not show the other convergence plots for lack of space. As argued in Bhatnagar et al. (2003), algorithms based on Hadamard matrix based perturbations track the steepest descent direction more closely than those that use randomized perturbations. The optimal policy obtained using all our algorithms is the same as that using policy iteration. The value function plots also look similar except for some differences in the converged values obtained in these. The amount of computational (CPU) times required by these algorithms on an IBM workstation with Linux operating system are approximately 9 and 28 min (7 and 30 min) for the Hadamard matrix based and randomized perturbations, respectively, using AA2 (AA1). Algorithm AA1 is slightly better than AA2 when Hadamard matrix based perturbations are used while AA2 is better in the case of randomized perturbations. The policy iteration algorithm takes about 83 min here. Finally, the TD(0) and Q-learning algorithms in Panigrahi and Bhatnagar (2004) took about 15 and 30 min, respectively.

Acknowledgements

Work is supported in part by Grant number SR/S3/EE/43/2002-SERC-Engg. from Department of Science and Technology, Government of India.

References

Bertsekas, D. P. (2001). *Dynamic programming and optimal control*. (2nd ed.), Belmont, MA: Athena Scientific.

Bertsekas, D. P., & Tsitsiklis, J. N. (1996). *Neuro-dynamic programming*. Belmont, MA: Athena Scientific.

Bhatnagar, S., Fernandez-Gaucherand, E., Fu, M. C., Marcus, S. I., & He, Y. (1999). A Markov decision process model for capacity expansion and allocation. In *Proceedings of the 38th IEEE conference on decision and control* (pp. 1156–1161). Arizona: Phoenix.

Bhatnagar, S., Fu, M. C., Marcus, S. I., & 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.

Bhatnagar, S., & 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.

Chang, H. S., Fard, P., Marcus, S. I., & Shayman, M. (2003). Multi-time scale Markov decision processes. *IEEE Transactions on Automatic Control*, *48*(6), 976–987.

Forestier, J., & Varaiya, P. (1978). Multilayer control of large Markov chains. *IEEE Transactions on Automatic Control*, *AC-23*(2), 298–304.

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

Konda, V. R., & Tsitsiklis, J. N. (2003). Actor-critic algorithms. *SIAM Journal on Control and Optimization*, *42*(4), 1143–1166.

Panigrahi, J. R., & Bhatnagar, S. (2004). Hierarchical decision making in semiconductor fabs using multi-timescale Markov decision processes. In *Proceedings of IEEE conference on decision and control*. Paradise Island, Nassau, Bahamas.

Parr, R. E. (1998). *Hierarchical control and learning for Markov decision processes*. Ph.D. thesis, Department of Computer Science, University of California at Berkeley, USA.

Spall, J. C. (1992). Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, *37*(3), 332–341.

Spall, J. C. (1997). A one-measurement form of simultaneous perturbation stochastic approximation. *Automatica*, *33*, 109–112.

Sutton, R., Precup, D., & Singh, S. (1999). Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, *112*, 181–211.

Tsitsiklis, J. N. (1994). Asynchronous stochastic approximation and Q-learning. *Machine Learning*, *16*, 185–202.

Shalabh Bhatnagar received his Ph.D. in Electrical Engineering from the Indian Institute of Science, Bangalore, in 1998. He is currently an Assistant Professor in the Department of Computer Science and Automation at the Indian Institute of Science, Bangalore. His research interests are in stochastic approximation algorithms, Markov decision processes and simulation optimization.

Jnana Ranjan Panigrahi is a Senior Software Engineer at SoftJin Technologies Private Ltd., Bangalore. He received his Masters in System Science and Automation from the Indian Institute of Science, Bangalore, in 2003.