A Simultaneous Perturbation Stochastic Approximation-Based Actor–Critic Algorithm for Markov Decision Processes

Shalabh Bhatnagar and Shishir Kumar

***Abstract*—A two-timescale simulation-based actor-critic algorithm for solution of infinite horizon Markov decision processes with finite state and compact action spaces under the discounted cost criterion is proposed. The algorithm does gradient search on the slower timescale in the space of deterministic policies and uses simultaneous perturbation stochastic approximation-based estimates. On the faster scale, the value function corresponding to a given stationary policy is updated and averaged over a fixed number of epochs (for enhanced performance). The proof of convergence to a locally optimal policy is presented. Finally, numerical experiments using the proposed algorithm on flow control in a bottleneck link using a continuous time queueing model are shown.**

***Index Terms*—Actor-critic algorithms, Markov decision processes, simultaneous perturbation stochastic approximation (SPSA), two timescale stochastic approximation.**

I. Introduction

Dynamic programming (DP) is a general methodology for solving Markov decision process (MDP) models. However, in order to apply DP, one requires complete knowledge of transition probabilities of

Manuscript received December 18, 2002; revised November 9, 2003. Recommended by Associate Editor D. Li. The work of S. Bhatnagar was supported by a Faculty Startup Grant from the Indian Institute of Science.

The authors are with the Department of Computer Science and Automation, Indian Institute of Science, Bangalore 560 012, India (e-mail: shalabh@csa.iisc.ernet.in; shishir.kumar@alumnus.csa.iisc.ernet.in).

Digital Object Identifier 10.1109/TAC.2004.825622

the system. These are not easily obtainable in most cases of interest. Moreover, for solving the Bellman equation, the computational requirements become prohibitive in the presence of a large state space. Motivated by these considerations, in recent times, research on simulation based schemes for solving MDPs has gathered momentum. These schemes are particularly useful in the case of systems for which obtaining the transition probabilities is hard; however, where transitions can be easily simulated. For tackling the curse of dimensionality, schemes involving certain parametric representations of the value function and/or policy have also been proposed. These schemes largely go under the names of neuro-dynamic programming [2], or reinforcement learning [14]. Among these, methods such as temporal difference (TD) [14] and Q-learning [15], [16] involve parameterizations of the value function (also called critic). A scheme based solely on policy (also called actor) parameterization has been dealt with in [10]. Those involving parameterizations of both actor and critic [1], [8] have also been proposed.

In [7], a two timescale simulation based approach for policy iteration has been developed for MDPs with finite state and action spaces. The idea there is that one can use a coupled stochastic approximation algorithm that is driven by two different step-size schedules or timescales where both recursions proceed simultaneously, so as to get the same effect as that of the two nested loops in policy iteration.

In [10], a Markov reward process setting is primarily considered with a parameterized class of policies. Here, the parameterization and tuning of the value function is not considered explicitly; however, for finding the average cost gradient, knowledge of sample path gradients of performance and transition probabilities are assumed to be known in functional form. With the exception of [8], most of the gradient optimization based schemes studied so far have assumed knowledge of explicit functional form of the transition probabilities. In [8], however, the randomization probabilities are the parameters with regard to which optimization is performed. This is possible only for finite action spaces. Moreover, most of these schemes deal with the long run average cost setting and not discounted cost.

In this note, we consider infinite horizon MDPs with finite state and compact action sets. We present a two timescale simulation based, policy iteration type algorithm that does gradient search in the space of policies. The faster timescale update in our algorithm is similar to that in [7], except that we perform an additional averaging over a fixed number of epochs for enhanced performance. On the slower scale, gradient search using simultaneous perturbation stochastic approximation (SPSA) [13] type estimates is used for obtaining an optimal policy. Here, we are able to directly update the action taken in each set and thus can work with deterministic policies themselves instead of randomized. Note that most schemes (in the literature) mentioned above have been studied in the setting of finite state and finite action sets. However, there are several applications, for instance, flow control in communication networks, for which the setting of finite state and compact (nondiscrete) action sets is more appropriate. Our main motivation in this note therefore, is to develop an easily implementable algorithm for such a setting. We also show numerical experiments on an application of the above type. A key advantage in using an SPSA based gradient search in the policy space is that the algorithm is computationally efficient even when the state space is moderately large in size. In [4], an SPSA-based algorithm is used for solving long-run-average-reward MDP. For purposes of averaging, the estimates in [4] are taken over regenerative cycles making the overall scheme less effective. The speed of our algorithm, on the other hand, is significantly enhanced by using updates at deterministic epochs with two timescale averaging. Moreover, our algorithm is geared toward finding the solution to the Bellman equation (that too in the discounted cost setting) which is not the case with [4]. Finally, even though we do not consider scenarios where the size of the state space is enormous as is the case with schemes in [2], the required modifications there can quite easily be added on top of our proposed algorithm.

The rest of the note is organized as follows. The framework and algorithm are presented in Section II. The convergence analysis is presented in Section III. Numerical experiments are presented in Section IV. Finally, Section V provides the concluding remarks.

II. Framework and Algorithm

Consider a discrete-time dynamic system with s states that are denoted by $1, 2, \ldots, s$. Let $S = \{1, \ldots, s\}$ denote the state–space. At each state i, we are given a set of actions or controls $A(i)$. If the state is i and an action $a \in A(i)$ is chosen, the system moves to some state j with probability $p(i,j,a)$, irrespective of the previous values of states and actions taken. Also, the cost incurred during this transition is $K(i,a,j)$. We assume $K(i,a,j)$ is nonnegative for all i, a, j. Let $\{X_n, n \geq 1\}$ be a state-valued process (also called an MDP) that describes the evolution of this system. This process in turn depends on a control-valued sequence $\{Z_n\}$ with each $Z_n \in A(X_n)$, $n \geq 1$, and such that for any (states) $i_0, \ldots, i_{n-1}, i_n$, and corresponding actions $a_0, \ldots, a_{n-1}, a_n$

$$Pr(X_{n+1} = j | X_n = i_n, Z_n = a_n, \ldots, X_0 = i_0, Z_0 = a_0) = p(i_n, j, a_n) \quad (1)$$

$\forall\, j \in S$. We assume that all sets $A(i)$ are compact subsets of $\mathcal{R}^N$ and have the form $\prod_{j=1}^N \left[a^i_{j,\min}, a^i_{j,\max}\right]$, $i \in S$. Let $A = \cup_{i=1}^s A(i)$ represent the action or control space. We make the following assumption on the cost function and transition probabilities.

Assumption (A): $\forall\, i, j \in S, a \in A(i)$, both $K(i,a,j)$ and $p(i,j,a)$ are continuously differentiable w.r.t. a.

By an admissible policy π, we mean a sequence of functions $\pi = \{\mu_0, \mu_1, \mu_2, \ldots\}$ with each $\mu_k \colon S \to A$, such that $\mu_k(i) \in A(i)$, $i \in S$, $k \geq 0$. Let Π be the set of all admissible policies. If $\mu_k = \mu$, $\forall\, k \geq 1$, then we call $\pi = \{\mu, \mu, \mu, \ldots\}$ (or the function μ itself) a stationary policy. Let $\alpha \in (0,1)$ be a constant. The aim is to minimize over all admissible policies $\pi = \{\mu_0, \mu_1, \mu_2, \ldots\}$, the infinite horizon α-discounted cost

$$V_\pi(x_0) = E\left[\sum_{k=0}^{\infty} \alpha^k K\left(x_k, \mu_k(x_k), x_{k+1}\right)\right] \quad (2)$$

starting from a given initial state $x_0 \in S$ with evolution of x_k's governed according to (1). Let

$$V^*(i) = \min_{\pi \in \Pi} V_\pi(i), \qquad i \in S \quad (3)$$

denote the optimal cost. For a given stationary policy π, the function $V_\pi(\cdot)$ is called the value function corresponding to policy π that takes value $V_\pi(i)$ when the initial state is i. Under (A), one can show that an optimal stationary policy exists for this problem and the optimal cost V^* satisfies the Bellman equation

$$V^*(i) = \min_{a \in A(i)} \left(\sum_{j \in S} p(i,j,a)\left(K(i,a,j) + \alpha V^*(j)\right)\right). \quad (4)$$

Since any action $a_i \triangleq (a_i^1, \ldots, a_i^N)^T \in A(i) \subset \mathcal{R}^N$, $i \in S$, we can identify any stationary policy π directly with the vector $(a_1^1, \ldots, a_1^N, a_2^1, \ldots, a_2^N, \ldots, a_s^1, \ldots, a_s^N)^T$ or simply with the block vector $(a_1, \ldots, a_s)^T$ of actions ordered lexicographically according to states, i.e., the jth component $(j = 1, \ldots, s)$ of this vector corresponds to the action taken in state j. Thus, we simply write $\pi = (a_1, \ldots, a_s)^T$. Let $\|\cdot\|$ represent the sup norm. We call a policy $\pi_0 = (a_1^0, \ldots, a_s^0)^T$ to be locally optimal if there exists an $\epsilon > 0$ such that $V_{\pi_0}(i) \leq V_{\bar{\pi}}(i)$, $\forall\, i \in S$, $\forall\, \bar{\pi}$ with $\|\pi_0 - \bar{\pi}\| < \epsilon$. Here, $\|\pi_0 - \bar{\pi}\| = \sup_{i \in S, j = 1, \ldots, N} |a_i^{0,j} - \bar{a}_i^j|$. Let $V_\pi(i)$ be the stationary value or cost-to-go function corresponding to policy π starting from

$i \in S$. It is easy to see the following using (4) and an application of Cramer's rule.

Lemma 1: Under (A), $V_\pi(i), \forall\, i \in S$ are continuously differentiable functions of π.

For $i \in S$, let $a_i(n) \triangleq \left(a_i^1(n), \ldots, a_i^N(n)\right)^T$ denote the nth update of a_i (used in the algorithm). Then, $\pi(n) = (a_1(n), \ldots, a_s(n))^T$ corresponds to the nth update of policy π. Let for $n \geq 0$, $\triangle(n) \in \mathcal{R}^{Ns}$ be a vector of mutually independent and mean zero random variables $\triangle_i^j(n)$, $j = 1, \ldots, N$, $i \in S$ (viz., $\triangle(n) = (\triangle_1^1(n), \ldots, \triangle_1^N(n), \triangle_2^1(n), \ldots, \triangle_2^N(n), \ldots, \triangle_s^1(n), \ldots, \triangle_s^N(n))^T$) taking values in a compact set $E \subset \mathcal{R}^{Ns}$ and having a common distribution. One can alternatively define $\triangle(n)$ as $\triangle(n) = (\triangle_1(n), \ldots, \triangle_s(n))^T$, for $n \geq 0$, with suitably defined $\triangle_i(n)$, $i \in S$. We make the following standard assumption (see [13] for a similar condition) on the random variables $\triangle_i^j(n)$, $i \in S$, $j = 1, \ldots, N$, $n \geq 0$.

Assumption (B): $\exists \bar{K} < \infty$ such that for any $n \geq 0$, $i \in S$ and $j \in \{1, \ldots, N\}$, $E[\left(\triangle_i^j(n)\right)^{-2}] \leq \bar{K}$.

Let $\delta > 0$ be a given fixed constant. Let $\Gamma_i^j(x)$ for $x \in \mathcal{R}$ denote the projection $\Gamma_i^j(x) = \min\left(a^i_{j,\max}, \max(a^i_{j,\min}, x)\right)$. Let for $y = (y_1, \ldots, y_N)^T \in \mathcal{R}^N$, $\Gamma_i(y) = \left(\Gamma_i^1(y_1), \ldots, \Gamma_i^N(y_N)\right)^T$. Then, $\Gamma_i(y)$ is a projection of $y \in \mathcal{R}^N$ on the set $A(i)$. Further, let $\Gamma(z)$ for some $z \triangleq \left(z_1^1, \ldots, z_1^N, \ldots, z_s^1, \ldots, z_s^N\right)^T \in \mathcal{R}^{Ns}$ denote $\Gamma(z) = (\Gamma_1(z_1), \ldots, \Gamma_s(z_s))^T$, with each $z_i = \left(z_i^1, \ldots, z_i^N\right)^T$. Consider now two independent parallel simulations $\{X^1(n)\}$ and $\{X^2(n)\}$, respectively. Suppose the policy used at the nth update in the first simulation is $\pi^1(n) = (\Gamma_1(a_1(n) - \delta\triangle_1(n)), \ldots, (\Gamma_s(a_s(n) - \delta\triangle_s(n)))^T$, while that in the second simulation is $\pi^2(n) = (\Gamma_1(a_1(n) + \delta\triangle_1(n)), \ldots, \Gamma_s(a_s(n) + \delta\triangle_s(n)))^T$, respectively. We shall also represent policies $\pi^r(n)$, $r = 1, 2$, $n \geq 0$, simply as $\pi^r(n) = (\pi_1^r(n), \ldots, \pi_s^r(n))^T$, with $\pi_i^r(n)$, $r = 1, 2$, $i \in S$, suitably defined. Let $\{b(n)\}$ and $\{c(n)\}$ be two step-size sequences that satisfy

$$\sum_n b(n) = \sum_n c(n) = \infty, \sum_n b(n)^2, \sum_n c(n)^2 < \infty \text{ and } \\ c(n) = o(b(n)). \quad (5)$$

Suppose now that for any $i \in S$, $a \in A(i)$, $\{\eta_n^1(i,a)\}$ and $\{\eta_n^2(i,a)\}$ are independent families of independent and identically distributed (i.i.d.) random variables each with distribution $p(i, \cdot, a)$. Let $L \geq 1$ be a given fixed integer. Consider quantities $V_{nL+m}^r(i)$, $r = 1, 2$, $n \geq 0$, $m = 0, 1, \ldots, L-1$, $i \in S$, that are defined via recursions (7) below. These are used for estimating the corresponding stationary value functions for given policy updates in the two simulations. For all $i \in S$, $r = 1, 2$, we initialize $V_0^r(i) = V_1^r(i) = \cdots = V_{L-1}^r(i) = 0$. Then, $\forall\, i \in S$, $j = 1, \ldots, N$, we have

$$a_i^j(n+1) = \Gamma_i^j\left(a_i^j(n) + c(n)\left(\frac{V_{nL}^1(i) - V_{nL}^2(i)}{2\delta\triangle_i^j(n)}\right)\right) \quad (6)$$

where, for $m = 0, 1, \ldots, L-1$, $r = 1, 2$

$$\begin{aligned} V_{nL+m+1}^r(i) = V_{nL+m}^r(i) &+ b(n) \\ &\times \left(K\left(i, \pi_i^r(n), \eta_{nL+m}^r(i, \pi_i^r(n))\right)\right. \\ &+ \alpha V_{nL+m}^r\left(\eta_{nL+m}^r(i, \pi_i^r(n))\right) \\ &\left.- V_{nL+m}^r(i)\right). \end{aligned} \quad (7)$$

The quantities $V_{nL+m}^r(i), r = 1, 2, i \in S, n \geq 0, m \in \{0, 1, \ldots, L-1\}$, are the estimates of stationary value functions corresponding to policies $\pi^r(n)$. We perform here an additional averaging (on top of the two timescale averaging) over L epochs for better algorithmic behavior. The value of L is chosen arbitrarily. We let $L = 100$ in our numerical experiments in Section V.

Remark: Note that $\{\eta_n^r(i,a)\}$, $r = 1, 2$, can be simulated without an explicit knowledge of $p(i, \cdot, a)$. For instance, in the example considered in Section V, given the distributions of the arrival and service time processes, $\eta_n^r(i,a)$ simply corresponds to the simulated state in the (independently generated) nth sample, after T time units, when state at current instant is i and arrival rate is a. Thus, even though $p(i, \cdot, a)$ may not be known (or may be very hard to compute) in closed form, $\eta_n^r(i,a)$ can still be simulated.

III. Convergence Analysis

Let $\mathcal{F}_l = \sigma(\tilde{a}_i(p), \tilde{\triangle}_i(p), V_p^1(i), V_p^2(i), p \leq l, i \in S, \eta_p^1(i, \tilde{\pi}_i^1(p)), \eta_p^2(i, \tilde{\pi}_i^2(p)), p < l, i \in S)$, $l \geq 1$. Here, $\tilde{a}_i(p) = a_i(n)$ and $\tilde{\triangle}_i(p) = \triangle_i(n)$ for $nL \leq p \leq (n+1)L - 1$. Also, $\tilde{\pi}_i^1(p), \tilde{\pi}_i^2(p)$ are defined by $\tilde{\pi}_i^1(p) = \Gamma_i\left(\tilde{a}_i(p) - \delta\tilde{\triangle}_i(p)\right)$ and $\tilde{\pi}_i^2(p) = \Gamma_i\left(\tilde{a}_i(p) + \delta\tilde{\triangle}_i(p)\right)$, respectively. Thus, $\tilde{\pi}_i^r(p) = \pi_i^r(n)$ for $nL \leq p \leq (n+1)L - 1$, $r = 1, 2$. Note that iterates (7) can be written as follows: For $i = 1, \ldots, s$, $r = 1, 2$, $k \geq 0$

$$\begin{aligned} V_{k+1}^r(i) = V_k^r(i) &+ b(n) \\ &\times \left(\sum_{j \in S} p\left(i, j, \tilde{\pi}_i^r(k)\right)\right. \\ &\qquad \left.\times \left(K\left(i, \tilde{\pi}_i^r(k), j\right) + \alpha V_k^r(j)\right) - V_k^r(i)\right) \\ &+ b(n)\left(K\left(i, \tilde{\pi}_i^r(k), \eta_k^r\left(i, \tilde{\pi}_i^r(k)\right)\right)\right. \\ &\qquad + \alpha V_k^r\left(\eta_k^r(i, \tilde{\pi}_i^r(k))\right) \\ &\qquad - \sum_{j \in S} p(i, j, \tilde{\pi}_i^r(k)) \\ &\qquad \left.\times \left(K\left(i, \tilde{\pi}_i^r(k), j\right) + \alpha V_k^r(j)\right)\right). \end{aligned} \quad (8)$$

We now state the following result whose proof follows from [15, Th. 2.1], in a similar manner as [7, Cor. 5.2].

Lemma 2: The iterates $V_k^r(i)$, $r = 1, 2$, satisfy $\sup_k \|V_k^r(i)\| < \infty\ \forall\, i \in S$.

Consider now sequences $\{M_i^r(l), l \geq 1\}$, $r = 1, 2$, $i \in S$, defined by $M_i^r(l) \triangleq \sum_{k=0}^{l-1} b(k)\ [(K(i, \tilde{\pi}_i^r(k), \eta_k^r(i, \tilde{\pi}_i^r(k))) + \alpha V_k^r(\eta_k^r(i, \tilde{\pi}_i^r(k)))) - \sum_{j \in S} p(i, j, \tilde{\pi}_i^r(k))\ (K(i, \tilde{\pi}_i^r(k), j) + \alpha V_k^r(j))]$, $i \in S$, $r = 1, 2$, $l \geq 1$. We have the following.

Lemma 3: The sequences $\{M_i^r(l), l \geq 1\}$, $r = 1, 2$, $i \in S$, converge a.s.

Proof: It is easy to see that $\{M_i^r(l), \mathcal{F}_l\}$, $r = 1, 2$, $i \in S$, form martingale sequences. Moreover, it is easy to verify that their corresponding quadratic variation processes converge a.s. The claim now follows by [11, Prop. VII.2.3(c)]. ■

Define $\{s(n), n \geq 0\}$ as follows: $s(0) = 0$, $s(n) = \sum_{i=0}^{n-1} c(i)$, $n \geq 1$. For $j = 1, \ldots, N$, $i \in S$, let $\triangle_i^j(t) = \triangle_i^j(n)$ for $t \in [s(n), s(n+1)]$, $n \geq 0$. Further, let $\triangle_i(t) = \left(\triangle_i^1(t), \ldots, \triangle_i^N(t)\right)^T$, $i \in S$, and $\triangle(t) = (\triangle_1(t), \ldots, \triangle_s(t))^T$. Suppose for any bounded, continuous, real valued function $v(\cdot)$, $\hat{\Gamma}_i^j(v(y)) = \lim_{0 < \eta \to 0}\left(\Gamma_i^j(y + \eta v(y)) - y/\eta\right)$, $j = 1, \ldots, N$, $i \in S$. For any $x = (x_1, \ldots, x_N)^T \in \mathcal{R}^N$, let $\hat{\Gamma}_i(x) = \left(\hat{\Gamma}_i^1(x_1), \ldots, \hat{\Gamma}_i^N(x_N)\right)^T$. Also, for any $z = \left(z_1^1, \ldots, z_1^N, \ldots, z_s^1, \ldots, z_s^N\right)^T \in \mathcal{R}^{Ns}$, let $\hat{\Gamma}(z) = \left(\hat{\Gamma}_1^1(z_1^1), \ldots, \hat{\Gamma}_1^N(z_1^N), \ldots, \hat{\Gamma}_s^1(z_s^1), \ldots\ \hat{\Gamma}_s^N(z_s^N)\right)^T$.

For policy $\pi = (a_1, \ldots, a_s)^T$ and given $V = (V_1, \ldots, V_s)^T$, let $F_\pi(i, a_i, V) \triangleq \sum_{j \in S} p(i, j, a_i)\,(K(i, a_i, j) + \alpha V_j)$. Thus, given π, it follows from (4) that $V_\pi = (V_\pi(1), \ldots, V_\pi(s))^T$ is the solution of the linear system $V_\pi(i) = F_\pi(i, a_i, V_\pi)$, $\forall\, i \in S$. Let for i, $k \in S$, $j = 1, \ldots, N$, $\nabla_j^i V_\pi(k)$ denote the gradient of $V_\pi(k)$ w.r.t. a_i^j. For simplicity, let $\nabla^i V_\pi(k) = \left(\nabla_1^i V_\pi(k), \ldots, \nabla_N^i V_\pi(k)\right)^T$. Consider the systems of ODEs shown in (9) and (10) at the bottom of the page, respectively. In (9), $E[\cdot]$ denotes expectation with regard to the common c.d.f. of $\triangle_i^j(t)$, $j = 1, \ldots, N$, $i \in S$, $t \geq 0$. Also, $\pi_i^1(t) = \Gamma_i(a_i(t) - \delta\triangle_i(t))$ and $\pi_i^2(t) = \Gamma_i(a_i(t) + \delta\triangle_i(t)), i \in S$. Define $\{\tilde{b}(n)\}$ as follows: For $n \geq 0$, $\tilde{b}(n) = b([n/L])$, where $[n/L]$ denotes the integer part of n/L. It is easy to see that $\sum_n \tilde{b}(n) = \infty$, $\sum_n \tilde{b}(n)^2 < \infty$ and $c(n) = o(\tilde{b}(n))$. Note also that $\tilde{b}(n)$ is a faster step-size parameter than $b(n)$. In the following, we shall work with $\{\tilde{b}(n)\}$ as the natural step-size sequence in place of $\{b(n)\}$. Now, define $\{t(n)\}$ as follows: $t(0) = 0, t(n) = \sum_{i=0}^{n-1} \tilde{b}(i), n \geq 1$. Consider the following system of ODEs: For $i \in S$, $r = 1, 2$

$$\begin{aligned}\dot{\sigma}_i(t) &= 0\\ \dot{V}_i^r(t) &= F_{\tilde{\pi}^r(t)}(i, \tilde{\pi}_i^r(t), V^r) - V_i^r(t).\end{aligned} \tag{11}$$

Note that here we are solving s coupled minimization problems corresponding to (4). In the system of ODEs (11), the first recursion corresponds to a stationary policy π (that is independent of t). Now, consider ODEs

$$\dot{V}_i^r(t) = F_{\tilde{\pi}^r}(i, \tilde{\pi}_i^r, V^r) - V_i^r(t), \qquad r = 1, 2. \tag{12}$$

One can show as in [7, Lemma 5.3] that $V_{\tilde{\pi}^r}(i)$, $r = 1, 2$, are the unique asymptotically stable equilibrium points for (12). Suppose $M = \{\pi \mid \hat{\Gamma}_i\left(\nabla^i V_\pi(i)\right) = 0\ \forall\, i \in S\}$. Also, for given $\epsilon > 0$, $M^\epsilon = \{\pi \mid \exists \pi_0 \in M \text{ s.t. } \|\pi - \pi_0\| < \epsilon\}$. In order to prove our main result, Theorem 1, we proceed through a series of approximation steps that follow. Let us denote by $F\left(i, \pi_i^r(n), V_{nL+m}^r(\eta_{nL+m}^r(i, \pi_i^r(n)))\right)$ the quantity $K\left(i, \pi_i^r(n), \eta_{nL+m}^r(i, \pi_i^r(n))\right) + \alpha V_{nL+m}^r\ (\eta_{nL+m}^r\,(i, \pi_i^r(n)))$, $r = 1, 2$. Consider functions $\bar{x}^r(t) = (\bar{x}_1^r(t), \ldots, \bar{x}_s^r(t))^T$, $r = 1, 2$, defined by $\bar{x}_i^r(t(n)) = V_{nL}^r(i)$, with the maps $t \rightarrow \bar{x}^r(t)$ being continuous linear interpolations on $[t(n), t(n+1)]$, $n \geq 0$. Given $T > 0$, define $\{T_n\}$ as follows: $T_0 = 0$ and for $n > 1$, $T_n = \min\{t(m) \mid t(m) \geq T_{n-1} + T\}$. Let $I_n = [T_n, T_{n+1}]$. Define also functions $x^{r,n}(t) = (x_1^{r,n}(t), \ldots, x_s^{r,n}(t))^T$, $r = 1, 2$, $t \in I_n$, $n \geq 0$, according to $\dot{x}_i^{r,n}(t) = F_{\tilde{\pi}^r(t)}\,(i, \tilde{\pi}_i^r(t), x^{r,n}(t)) - x_i^{r,n}(t)$, with $x_i^{r,n}(T_n) = \bar{x}^r(t(m_n)) = V_{m_n}^r(i)$ for some m_n. We now have the following.

Lemma 4: $\lim_{n\rightarrow\infty} \sup_{t \in I_n} \|x^{r,n}(t) - \bar{x}^r(t)\| = 0, r = 1, 2$, w.p. 1.

Proof: It is easy to see that for $r = 1, 2$

$$\begin{aligned}\bar{x}_i^r(t(n+1)) = \bar{x}_i^r(t(n)) &+ \int_{t(n)}^{t(n+1)} F_{\tilde{\pi}^r(t)}\,(i, \tilde{\pi}_i^r(t), \bar{x}^r(t))\,dt\\ &+ \int_{t(n)}^{t(n+1)} \Big(F_{\tilde{\pi}^r(t(n))}\,(i, \tilde{\pi}_i^r(t(n)), \bar{x}^r(t(n)))\\ &\qquad - F_{\tilde{\pi}^r(t)}(i, \tilde{\pi}_i^r(t), \bar{x}^r(t))\Big)\,dt\\ &+ (M_i^r(n+1) - M_i^r(n)).\end{aligned}$$

Now, since $\bar{x}_i^r(\cdot), i \in S$, are bounded (by Lemma 2) and continuous, one can easily check that $\int_{t(n)}^{t(n+1)} (F_{\tilde{\pi}^r(t(n))}\,(i, \tilde{\pi}_i^r(t(n)),\bar{x}^r(t(n))) - F_{\tilde{\pi}^r(t)}\,(i, \tilde{\pi}_i^r(t), \bar{x}^r(t)))dt \sim O(\tilde{b}(n)^2)$. Also

$$x_i^{r,n}(t) = x_i^{r,n}(T_n) + \int_{T_n}^{t} F_{\tilde{\pi}^r(s)}(i, \tilde{\pi}_i^r(s), x^{r,n}(s))ds.$$

The claim now follows from (5), Lemma 3 and Gronwall's inequality. ■

Now, observe that the first iteration (6) of the algorithm can be written as follows:

$$a_i^j(n+1) = \Gamma_i^j\left(a_i^j(n) + \tilde{b}(n)\xi(n)\right)$$

where $\xi(n) = o(1)$ since $c(n) = o(\tilde{b}(n))$. From Lemma 4, note that the algorithm (6) and (7) asymptotically tracks trajectories of the ODE (11). Now, by [6, Th. 1], we have the following.

Lemma 5: For all $i \in S$, $r = 1, 2$, $\|V_n^r(i) - F_{\tilde{\pi}^r(n)}(i, \tilde{\pi}_i^r(n), V_{\tilde{\pi}^r(n)})\| \rightarrow 0$ a.s. as $n \rightarrow \infty$.

Consider now σ-fields $\mathcal{G}_l$, $l \geq 1$, defined by $\mathcal{G}_l = \sigma(a_i(p), V_{pL}^1(i), V_{pL}^2(i), p \leq l, i \in S, \triangle_i(p), p < l, i \in S)$. Defining appropriate martingale sequences (as before) w.r.t. $\mathcal{G}_l$, for the slower recursion using second term on the right-hand side of (6), one can again argue that these are a.s. convergent. Now, define $\tilde{x}^r(t) = (\tilde{x}_1^r(t), \ldots, \tilde{x}_s^r(t))^T$, $r = 1, 2$, and $a(t) = (a_1(t), \ldots, a_s(t))^T$ as follows: $\tilde{x}_i^r(s(n)) = V_{nL}^r(i)$, $r = 1, 2$ and $a_i(s(n)) = a_i(n)$, $n \geq 0$, with linear interpolation on intervals $[s(n), s(n+1)]$, $n \geq 0$. One can rewrite (6) as shown in the equation at the bottom of the page, where $\zeta(n)$ is $o(1)$ by the above and Lemma 5. Now, using a standard argument as in [9, pp. 191–194], it can be shown that (6) asymptotically tracks the trajectories of (9) on the slower scale. Let $A(i)^o$, $i \in S$, represent the interior of the action set $A(i)$. Also, let $\prod_{i=1}^s A(i)$ represent the cartesian product of all action sets. We now have the following.

Theorem 1: Given $\epsilon > 0, \exists \delta_0 > 0$ such that $\forall\ \delta \in (0, \delta_0]$, the algorithm (6) and (7) converges to M^ϵ with probability one.

Proof: Suppose $\pi(n) \in \prod_{i=1}^s A(i)^o$. Then, choose $\delta > 0$ small such that $\pi_i^1(n) = a_i(n) - \delta\triangle_i(n)$ and $\pi_i^2(n) = a_i(n) + \delta\triangle_i(n)$, respectively, for all $i \in S$. Now, using appropriate Taylor series expansions of terms in the numerator of $\left(F_{\pi^1(n)}(i, \pi_i^1(n), V_{\pi^1(n)}) - F_{\pi^2(n)}(i, \pi_i^2(n), V_{\pi^2(n)})\right)/2\delta\triangle_i^{k_0}(n)$ around the point $a_i(n)$ and from (B), one can derive that a.s.; see (13) at the bottom of the next page. For $\pi(n)$ on the boundary of $\prod_{i=1}^s A(i)$, a similar claim as above can be obtained except with a suitable scalar multiplying the second term on the left-hand side of (13). Now, $V_\pi(i) \geq 0, \forall\, i \in S$, since $K(i, a, j) \geq 0\ \forall\, i, j \in S, a \in A(i)$. For

$$\dot{\sigma}_i^j(t) = \hat{\Gamma}_i^j\left(E\left[\frac{F_{\pi^1(t)}(i, \pi_i^1(t), V_{\pi^1(t)}) - F_{\pi^2(t)}(i, \pi_i^2(t), V_{\pi^2(t)})}{2\delta\triangle_i^j(t)}\right]\right), \qquad j = 1, \ldots, N, i \in S \tag{9}$$

$$\dot{\sigma}_i(t) = \hat{\Gamma}_i\left(-\nabla^i V_{\pi(t)}(i)\right), \qquad i \in S. \tag{10}$$

$$a_i^j(n+1) = \Gamma_i^j\left(a_i^j(n) + c(n)\left(E\left[\frac{F_{\pi^1(n)}\left(i, \pi_i^1(n), V_{\pi^1(n)}\right) - F_{\pi^2(n)}\left(i, \pi_i^2(n), V_{\pi^2(n)}\right)}{2\delta\triangle_i^j(n)}\,\middle|\,\mathcal{G}_n\right]\right) + \zeta(n)\right)$$

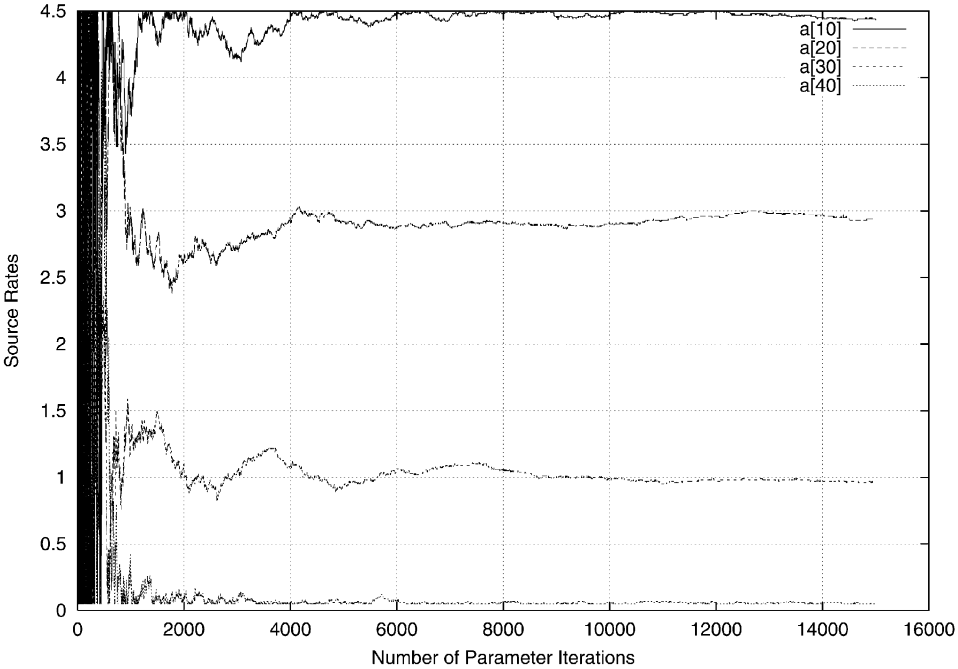

Fig. 1. Convergence behavior for $T = 5$ with $\lambda_u = 0.2$.

the ODE (10), observe that $\nabla^i V_\pi(i) \cdot \hat{\Gamma}_i\left(-\nabla^i V_\pi(i)\right) < 0$ outside the set M. It is easy to see that M serves as an asymptotically stable attractor set for (10) with $\sum_{i \in S} V_\pi(i)$, serving as the associated strict Lyapunov function for (10). The claim follows. ■

Remark: Note that M contains all Kuhn–Tucker points of (10) that include both stable and unstable equilibria. In principle, the stochastic approximation scheme may get trapped in an unstable equilibrium. In [12], with noise assumed to be sufficiently "omnidirectional" in addition, it is shown that convergence to unstable fixed points is not possible; see also [3] for conditions on avoidance of unstable equilibria that lie in certain *compact connected chain recurrent sets*. However, in most cases (even without extra noise conditions) due to the inherent randomness, stochastic approximation algorithms converge to stable equilibria. By continuity of $V_\pi(i), i \in S$, one then obtains an "ϵ-locally optimum" π. Next, note that Theorem 1 merely gives the existence of a $\delta_0 > 0$ for given $\epsilon > 0$ such that $\forall\ \delta \in (0, \delta_0]$, convergence to an ϵ-locally optimal policy is assured. We found from numerical experiments that a small enough δ chosen arbitrarily works well in most cases. A small δ has the same effect as that of a larger step-size and which results in a large variance during initial runs. On the other hand, however, it also helps to speed up convergence. Finally, for obtaining a globally optimal policy, one can use in addition, a "slowly decreasing Gaussian noise" as in simulated annealing algorithms [5].

IV. Numerical Experiments

We consider a continuous time queueing model of flow control in communication networks. A single bottleneck node is fed with two arrival streams, one an uncontrolled Poisson stream and the other a controlled Poisson process. Service times are assumed to be i.i.d., with exponential distribution. We assume that the node has a finite buffer of size B. Suppose $T > 0$ is a given constant. Let q_n denote the queue length observed at time nT, $n \geq 0$. The controlled source thus sends packets according to a Poisson process with rate $\lambda_c(n)$ during the time interval $[nT, (n+1)T)$ and at instant $(n+1)T$, upon observation of q_{n+1}, the source changes its rate to some $\lambda_c(n+1)$. The aim is to find a stationary optimal policy that assigns rates to individual states. We use our simulation based algorithm for this purpose.

In the experiments, we considered two cases for buffer size values, $B = 50$ and $B = 1000$, respectively. The constraint interval for the rate values was chosen to be $[0.05, 4.5]$ (same over all states). We chose two values for the uncontrolled traffic rate: $\lambda_u = 0.2$ and $\lambda_u = 0.8$, respectively. We show here results corresponding to $B = 50$ and $\lambda_u = 0.2$ as similar results were obtained for the other combinations of these. The service rate was chosen to be $\mu = 2.0$ in all cases. We considered the cost function to be $K(i, a_i, j) = |j - B/2|$. We observed similar behavior using two other cost functions $K(i, a_i, j) = r|i - B/2| + |j - B/2|$, $r \in (0, 1)$ and $K(i, a_i, j) = (|i - B/2| + |j - B/2|)/2$, respectively. Cost functions of the aforementioned type are useful in cases where the goal is to maximize throughput and minimize delays simultaneously as in multimedia applications involving integrated service networks. We denote the near-optimal rate as computed by our algorithm in state i by a_i^*, $i \in S$.

We ran all simulations for 15 000 iterations of the rate vector with initial rate values for all states in all cases set at 0.5. Convergence had been achieved in all cases we considered during this period. On a Pentium 4 personal computer, it took about 1.5 min for 15 000 iterations, for $B = 50$ and about 30 min for the same number of iterations, for $B = 1000$. Thus the amount of computation using this approach is seen to grow only linearly with the number of states. The same was also

$$\lim_{\delta \downarrow 0} \left| E\left[\frac{F_{\pi^1(n)}(i, \pi_i^1(n), V_{\pi^1(n)}) - F_{\pi^2(n)}(i, \pi_i^2(n), V_{\pi^2(n)})}{2\delta \triangle_i^{k_0}(n)} \,\middle|\, \mathcal{G}_n \right] + \nabla_{k_0}^i V_{\pi(n)}(i) \right| = 0. \qquad (13)$$

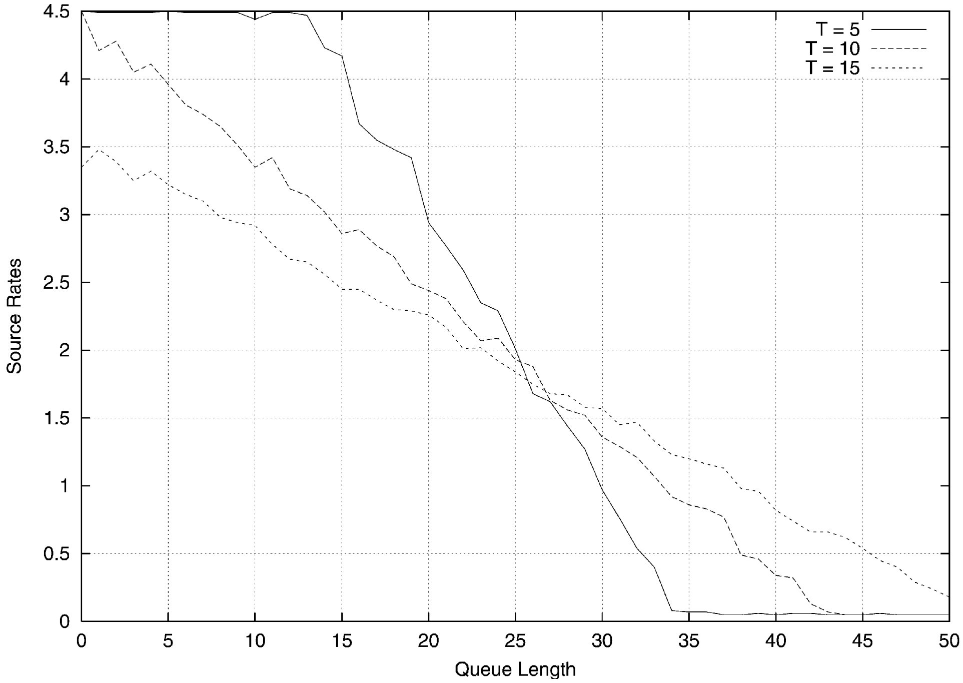

Fig. 2. Feedback policies after convergence for $\lambda_u = 0.2$.

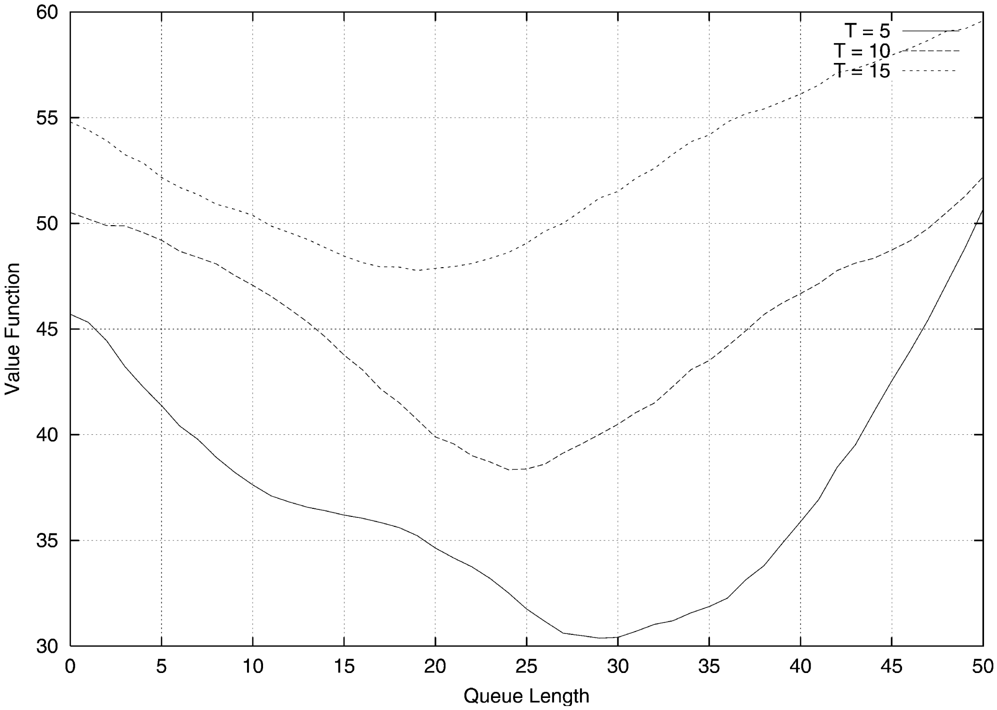

Fig. 3. Value functions for $\lambda_u = 0.2$.

verified using some other state–space cardinalities as well. We chose $L = 100$ in all cases. Also, the step-size sequences were chosen as $c(n) = 1/n$, $b(n) = 1/n^{2/3}$, $\forall\, n \geq 1$, with $c(0) = b(0) = 1$. The values of δ and α were set at 0.1 and 0.9, respectively. The perturbation sequences were chosen as i.i.d., Bernoulli distributed random variables $\triangle_i(n) = \pm 1$ w.p. 1/2, $\forall\, n \geq 0$. In Fig. 1, we plot the sample convergence behavior for rates corresponding to states 10, 20, 30 and 40, for $T = 5$, with $\lambda_u = 0.2$ and $B = 50$. We use the symbols $a[10]$, $a[20]$ etc., to denote these rates in the figure. In Figs. 2 and 3, we present plots of feedback policies and value functions (after convergence of algorithm), respectively, for $T = 5$, 10 and 15, with $\lambda_u = 0.2$. The value functions $V^*(i)$, $i = 0, \ldots, B$, were computed by averaging over 1000 independent sample paths of the Markov chain under the stationary policy obtained after convergence of algorithm for the various cases, with each path terminated after 30 000 "T-instants."

From Fig. 2, observe that for given T, the rates are high for low queue length values and become low when queue length values are high. This is obvious because of the form of the cost function. Also, the rate curves become flatter as T is increased. Note that $T = \infty$ corresponds to the open loop or state invariant policy case. Thus, as T is increased, the effect of the controller tends to die down. From Fig. 3, note that the values obtained are lowest for $T = 5$. Also, in most cases, the value function dips somewhere near $B/2$ and rises on either side. This is expected because of the form of the cost function and the discounted

TABLE I
STEADY STATE PERFORMANCE METRICS FOR $\lambda_u = 0.2$

T	$E[q]$	$\bar{\lambda}_c^*$	$Var(q)$	$J(\pi^*)$	P_{band}
2	26.24	1.80	9.01	2.58	0.555
3	26.34	1.79	12.65	3.01	0.487
4	26.06	1.80	16.48	3.32	0.452
5	26.07	1.79	20.24	3.65	0.418
6	25.85	1.79	24.25	3.94	0.389
10	26.15	1.80	40.39	5.12	0.306
15	25.95	1.80	59.09	6.18	0.253
20	25.98	1.80	78.65	7.13	0.220

cost criterion, because of which a significant contribution to the value function comes from the initial stages as costs accrued from later stages get discounted heavily.

Next, in Table I, we use the rates obtained after convergence of our algorithm for various values of T, for computing certain steady state performance measures of interest. We use these rates to compute the steady-state mean queue length $E[q]$, average rate of the controlled source $\bar{\lambda}_c^*$, long run average cost J^*, variance of queue length $Var(q)$ and the stationary probability (P^*) that the queue length is in the region $= \{(B/2) - 2, \ldots, (B/2) + 2\}$. The previous metrics are obtained using another simulation that runs for 30 000 "T-instants" using standard Monte-Carlo estimates. We computed these metrics in order to get an idea of the steady-state system performance (in addition) using our algorithm. From Table I, it is clear that the steady state performance degrades as T increases in both cases. For instance, $Var(q)$ and J^* increase while P^* decreases. Note that the average rate for all cases when $\lambda_u = 0.2$ is close to 1.8 which ensures almost complete bandwidth utilization in all cases.

V. CONCLUSION

In this note, we developed a two timescale gradient search based actor–critic algorithm for solving infinite horizon MDPs with finite-state and compact action spaces under the discounted cost criterion. The algorithm was theoretically shown to converge to a locally optimal policy. We showed numerical experiments using a continuous time queueing model for rate-based flow control. The algorithm is found to be computationally efficient with the computational effort growing only linearly with the number of states.

ACKNOWLEDGMENT

The first author would like to thank Prof. V. S. Borkar for many helpful comments on an earlier version of this manuscript which led, in particular, to the relaxation of a technical assumption.

REFERENCES

[1] A. Barto, R. Sutton, and C. Anderson, "Neuron-like elements that can solve difficult learning control problems," *IEEE Trans. Syst., Man, Cybern.*, vol. 13, pp. 835–846, Dec. 1983.

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

[3] O. Brandiere, "Some pathological traps for stochastic approximation," *SIAM J. Control Optim.*, vol. 36, pp. 1293–1314, 1998.

[4] H.-T Fang, H.-F Chen, and X.-R. Cao, "Recursive approaches for single sample path based Markov reward processes," *Asian J. Control*, vol. 3, no. 1, pp. 21–26, 2001.

[5] S. B. Gelfand and S. K. Mitter, "Recursive stochastic algorithms for global optimization in $\mathcal{R}^{d*}$," *SIAM J. Control Optim.*, vol. 29, no. 5, pp. 999–1018, 1991.

[6] M. W. Hirsch, "Convergent activation dynamics in continuous time networks," *Neural Networks*, vol. 2, pp. 331–349, 1989.

[7] V. R. Konda and V. S. Borkar, "Actor-critic like learning algorithms for Markov decision processes," *SIAM J. Control Optim.*, vol. 38, no. 1, pp. 94–123, 1999.

[8] V. R. Konda and J. N. Tsitsiklis, "Actor-critic algorithms," *SIAM J. Control Optim.*, vol. 42, no. 4, pp. 1143–1166, 2003.

[9] H. J. Kushner and D. S. Clark, *Stochastic Approximation Methods for Constrained and Unconstrained Systems*. New York: Springer-Verlag, 1978.

[10] P. Marbach and J. N. Tsitsiklis, "Simulation-based optimization of Markov reward processes," *IEEE Trans. Automat. Contr.*, vol. 46, pp. 191–209, Feb. 2001.

[11] J. Neveu, *Discrete Parameter Martingales*. Amsterdam, The Netherlands: North Holland, 1975.

[12] R. Permantle, "Nonconvergence to unstable points in urn models and stochastic approximations," *Ann. Probab.*, vol. 18, pp. 698–712, 1990.

[13] J. C. Spall, "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation," *IEEE Trans. Automat. Contr.*, vol. 37, pp. 332–341, Mar. 1992.

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

[15] J. N. Tsitsiklis, "Asynchronous stochastic approximation and Q-learning," *Mach. Learn.*, vol. 16, pp. 185–202, 1994.

[16] C. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, pp. 279–292, 1992.

Digital Object Identifier 10.1109/TAC.2004.825628