A constrained optimization perspective on actor critic algorithms and application to network routing

Prashanth L.A.*[1], H. L. Prasad†[2], Shalabh Bhatnagar‡[3] and Prakash Chandra§[4]

[1]Institute for Systems Research, University of Maryland
[2]Astrome Technologies Pvt Ltd, Bangalore, India
[3]Department of Computer Science and Automation, Indian Institute of Science, Bangalore, India
[4]System Sciences and Automation, Indian Institute of Science, Bangalore, India

Abstract

We propose a novel actor-critic algorithm with guaranteed convergence to an optimal policy for a discounted reward Markov decision process. The actor incorporates a descent direction that is motivated by the solution of a certain non-linear optimization problem. We also discuss an extension to incorporate function approximation and demonstrate the practicality of our algorithms on a network routing application.

1 Introduction

We consider a discounted MDP with state space $\mathcal{S}$, action space $\mathcal{A}$, both assumed to be finite. A randomized policy π specifies how actions are chosen, i.e., $\pi(s)$, for any $s \in \mathcal{S}$ is a distribution over the actions $\mathcal{A}$. The objective is to find the optimal policy π^* that is defined as follows:

$$\pi^*(s) = \underset{\pi \in \Pi}{\operatorname{argmax}} \left\{ v^\pi(s) := E\left[\sum_n \beta^n \sum_{a \in \mathcal{A}(s_n)} r(s_n, a)\pi(s_n, a) | s_0 = s \right] \right\}, \tag{1}$$

where $r(s,a)$ is the instantaneous reward obtained in state s upon choosing action a, $\beta \in (0,1)$ is the discount factor and Π is the set of all admissible policies. We shall use $v^*(= v^{\pi^*})$ to denote the optimal value function.

Actor-critic algorithms (cf. [8], [4] and [9]) are popular stochastic approximation variants of the well-known policy iteration procedure for solving (1). The *critic* recursion provides estimates of the value function using the well-known temporal-difference (TD) algorithm, while the *actor* recursion performs a gradient search over the policy space. We propose an actor-critic algorithm with a novel descent direction for the actor recursion. The novelty of our approach is that we can motivate the actor-recursion in the following manner: the descent direction for the actor update is such that it (globally) minimizes the objective of a non-linear optimization problem, whose minima coincide with the optimal policy π^*. This descent direction

*prashla@isr.umd.edu
†prasad@astrome.co
‡shalabh@csa.iisc.ernet.in
§pchandra@ee.iisc.ernet.in

is similar to that used in Algorithm 2 in [8], except that we use a different exponent for the policy and a similar interpretation can be used to explain Algorithm 2 (and also 5) of [8]. Using multi-timescale stochastic approximation, we provide global convergence guarantees for our algorithm.

While the proposed algorithm is for the case of full state representations, we also briefly discuss a function approximation variant of the same. Further, we conduct numerical experiments on a shortest-path network problem. From the results, we observe that our actor-critic algorithm performs on par with the well-known Q-learning algorithm on a smaller-sized network, while on a larger-sized network, the function approximation variant of our algorithm does better than the algorithm in [1].

2 The Non-Linear Optimization Problem

With an objective of finding the optimal value and policy tuple, we formulate the following problem:

$$\left.\begin{array}{l} \min\limits_{v\in \mathrm{R}^{|\mathcal{S}|}} \min\limits_{\pi\in\Pi} \left(J(v,\pi) := \sum\limits_{s\in\mathcal{S}} \left[v(s) - \sum\limits_{a\in\mathcal{A}} \pi(s,a) Q(s,a) \right] \right) \\ \text{s.t. } \forall s\in\mathcal{S}, a\in\mathcal{A} \\ (a)\, \pi(s,a) \geq 0, \quad (b) \sum\limits_{a\in\mathcal{A}} \pi(s,a) = 1, \quad \text{and} \quad (c)\, g(s,a) \leq 0. \end{array}\right\} \tag{2}$$

In the above, $g(s,a) := Q(s,a) - v(s)$, with $Q(s,a) := r(s,a) + \beta \sum\limits_{s'} p(s'|s,a)v(s')$. Here $p(s'|s,a)$ denotes the probability of a transition from state s to s' upon choosing action a.

The objective in (2) is to ensure that there is no Bellman error, i.e., the value estimates v are correct for the policy π. The constraints (2(a))–(2(b)) ensure that π is a distribution, while the constraint (2(c)) is a proxy for the max in (1). Notice that the non-linear problem (2) has a quadratic objective and linear constraints.

From the definition of π^*, it is easy to infer the following claim:

Theorem 1. *Let $g^*(s,a) := Q^*(s,a) - v^*(s)$, with $Q^*(s,a) := r(s,a) + \beta \sum\limits_{s'} p(s'|s,a)v^*(s')$, $\forall s \in \mathcal{S}, a \in \mathcal{A}$. Then,*
(i) *Any feasible (v^*, π^*) is optimal in the sense of (1) if and only if $J(v^*, \pi^*) = 0$.*
(ii) *π^* is an optimal policy if and only if $\pi^*(s,a)g^*(s,a) = 0$, $\forall a \in \mathcal{A}, s \in \mathcal{S}$.*

3 Descent direction.

Proposition 1. *For the objective in (2), the direction $\sqrt{\pi(s,a)}g(s,a)$ is a non-ascent and in particular, a descent direction along $\pi(s,a)$ if $\sqrt{\pi(s,a)}g(s,a) \neq 0$, for all $s \in \mathcal{S}, a \in \mathcal{A}$.*

Proof. Consider any action $a \in \mathcal{A}$ for some $s \in \mathcal{S}$. We show that $\sqrt{\pi(s,a)}g(s,a)$ is a descent direction by the following Taylor series argument. Let

$$\hat{\pi}(s,a) = \pi(s,a) + \delta\sqrt{\pi(s,a)}g(s,a),$$

for a small $\delta > 0$. We define $\hat{\pi}$ to be the same as π except with the probability of picking action a in state $s \in \mathcal{S}$ being changed to $\hat{\pi}(s,a)$ (and the rest staying the same). Then by Taylor's expansion of $J(\pi)$ upto the first order term, we have that

$$J(v,\hat{\pi}) = J(v,\pi) + \delta\sqrt{\pi(s,a)}g(s,a)\frac{\partial J(v,\pi)}{\partial \pi(s,a)}.$$

Note that higher order terms are all zero since $J(v, \pi)$ is linear in π. It should be easy to see from definition of the objective that $\frac{\partial J(v,\pi)}{\partial \pi(s,a)} = -g(s,a)$. So,

$$J(v, \hat{\pi}) = J(v, \pi) - \delta\sqrt{\pi(s,a)}(g(s,a))^2.$$

Thus, for $a \in \mathcal{A}$ and $s \in \mathcal{S}$ where $\pi(s,a) > 0$ and $g(s,a) \neq 0$, $J(v,\hat{\pi}) < J(v,\pi)$, while when $\sqrt{\pi(s,a)}g(s,a) = 0$, $J(v,\hat{\pi}) = J(v,\pi)$. □

The next section utilizes the descent direction to derive an actor-critic algorithm.

4 The Actor-Critic Algorithm

Combining the descent procedure in π from the previous section, with a $TD(0)$ [11] type update for the value function v on a faster time-scale, we have the following update scheme:

Q-Value: $Q_n(s,a) = r(s,a) + \beta v_n(s')$, **TD Error:** $g_n(s,a) = Q_n(s,a) - v_n(s)$,

Critic: $v_{n+1}(s) = v_n(s) + c(n)g_n(s,a)$, **Actor:** $\pi_{n+1}(s,a) = \Gamma\left(\pi_n(s,a) + b(n)\sqrt{\pi_n(s,a)}g_n(s,a)\right)$. (3)

In the above, Γ is a projection operator that ensures that the updates to π stay within the simplex $\mathcal{D} = \{(x_1,\ldots,x_q) \mid x_i \geq 0, \forall i = 1,\ldots,q, \sum_{j=1}^{q} x_j \leq 1\}$, where $q = |\mathcal{A}|$. Further, the step-sizes $b(n)$ and $c(n)$ satisfy

$$\sum_{n=1}^{\infty} c(n) = \sum_{n=1}^{\infty} b(n) = \infty, \sum_{n=1}^{\infty}\left(c^2(n) + b^2(n)\right) < \infty \text{ and } b(n) = o(c(n)).$$

Remark 1. ***(Connection to Algorithm 2 of [8])*** *From Proposition 1, we have that $\sqrt{\pi(s,a)}g(s,a)$ is a descent direction for $\pi(s,a)$. This implies $\pi(s,a)^\alpha \times \sqrt{\pi(s,a)}g(s,a)$ for any $\alpha \geq 0$, is also a descent direction. Hence,*

a generic update rule for π is: $$\pi_{n+1}(s,a) = \Gamma\left(\pi_n(s,a) + b(n)(\pi_n(s,a))^{\alpha'}g_n(s,a)\right), \textit{ for any } \alpha' \geq \frac{1}{2}.$$

The special case of $\alpha' = 1$ coincides with the π-recursion in Algorithm 2 of [8].

5 Convergence Analysis

For the purpose of analysis, we assume that the underlying Markov chain for any policy $\pi \in \Pi$ is irreducible.

Main result Let $v^\pi = [I - \beta P_\pi]^{-1} R_\pi$, where $R_\pi = \langle r(s,\pi), s \in \mathcal{S}\rangle^T$ is the column vector of rewards and $P_\pi = [p(y|s,\pi), s \in \mathcal{S}, y \in \mathcal{S}]$ is the transition probability matrix, both for a given π. Consider the ODE:

$$\frac{d\pi(s,a)}{dt} = \bar{\Gamma}\left(\sqrt{\pi(s,a)}g^\pi(s,a)\right), \forall a \in \mathcal{A}, s \in \mathcal{S}, \text{ where} \tag{4}$$

$$g^\pi(s,a) := r(s,a) + \beta \sum_{y \in U(s)} p(y|s,a)v^\pi(y) - v^\pi(s). \tag{5}$$

In the above, $\bar{\Gamma}$ is a projection operator defined by $\bar{\Gamma}(\epsilon(\pi)) := \lim_{\alpha \downarrow 0} \frac{\Gamma(\pi + \alpha\epsilon(\pi)) - \pi}{\alpha}$, for any continuous $\epsilon(\cdot)$.

Theorem 2. *Let K denote the set of all equilibria of the ODE (4), G the set of all feasible points of the problem* (2) *and $\hat{K} := K \cap G$. Then, the iterates $(v_n, \pi_n), n \geq 0$ governed by* (3) *satisfy*

$$(v_n, \pi_n) \to K^* \text{ a.s. as } n \to \infty, \text{ where } K^* = \{(v^*, \pi^*) \mid \pi^* \in \hat{K}\}.$$

The algorithm (3) comprises of updates to v on the faster time-scale and to π on the slower time-scale. Using the theory of two time-scale stochastic approximation [5, Chapter 6], we sketch the convergence of these recursions as well as prove global optimality in the following steps (the reader is referred to the appendix for proof details):

Step 1: Critic Convergence We assume π to be time-invariant owing to time-scale separation. Consider the ODE:

$$\frac{dv(s)}{dt} = r(s, \pi) + \beta \sum_{s' \in \mathcal{S}} p(s'|s, \pi)v(y) - v(s), \forall s \in \mathcal{S}, \tag{6}$$

where $r(s, \pi) = \sum_{a \in \mathcal{A}} \pi(s, a)r(s, a)$ and $p(s'|s, \pi) = \sum_{a \in \mathcal{A}} \pi(s, a)p(s'|s, a)$. It is well-known (cf. [2]) that the above ODE has a unique globally asymptotically stable equilibrium v^π. We now have the main result regarding the convergence of v_n on the faster time-scale.

Theorem 3. *For a given π, the critic recursion in* (3) *satisfies $v_n \to v^\pi$ a.s. as $n \to \infty$.*

Step 2: Actor Convergence Due to timescale separation, we can assume that the critic has converged in the analysis of the actor recursion. We first provide a useful characterization for the set K of equilibria of the ODE (4).

Lemma 4. *Let $L = \{\pi | \pi(s) \text{ is a probability vector over } \mathcal{A}, \forall s \in \mathcal{S}\}$ denote the set of policies that are distributions over the actions for each state. Then,*

$$\pi \in K \text{ if and only if } \pi \in L \text{ and } \sqrt{\pi(s, a)}g^\pi(s, a) = 0, \forall a \in \mathcal{A}, s \in \mathcal{S}.$$

From Lemma 4, the set K can be redefined as follows: $K = \left\{ \pi \in L \,\middle|\, \sqrt{\pi(s, a)}g(s, a) = 0, \forall a \in \mathcal{A}, s \in \mathcal{S} \right\}$. The set K can be partitioned using the feasible set G of (2) as $K = \hat{K} \cup \hat{K}^\mathsf{c}$, where $\hat{K} = K \cap G$.

Lemma 5. *All $\pi^* \in \hat{K}^\mathsf{c}$ are unstable equilibrium points of the system of ODEs (4).*

Proof. For any $\pi^* \in K^\mathsf{c}$, there exists some $a \in \mathcal{A}(s), s \in \mathcal{S}$, such that $g^\pi(s, a) > 0$ and $\pi(s, a) = 0$ because K^c is not in the feasible set G. Let $B_\delta(\pi^*) = \{\pi \in L | \, \|\pi - \pi^*\| < \delta\}$. Choose $\delta > 0$ such that $g^\pi(s, a) > 0$ for all $\pi \in B_\delta(\pi^*) \setminus K$. So, $\bar{\Gamma}(\sqrt{\pi(s, a)}g^\pi(s, a)) > 0$ for any $\pi \in B_\delta(\pi^*) \setminus K$ which suggests that $\pi(s, a)$ will be increasingly moving away from π^*. Thus, π^* is an unstable equilibrium point for the system of ODEs (4). □

Remark 2. ***($G = \hat{K}$)*** *We already have that $\hat{K} \subseteq G$. So, it is sufficient to show that $G \subseteq \hat{K}$. A policy π belongs to G if $g^\pi(s, a) \leq 0$ for all $a \in \mathcal{A}(s)$ and $s \in \mathcal{S}$. By definition, v^π is obtained from $\sum_{a \in \mathcal{A}(s)} \pi(s, a)g^\pi(s, a) = 0, \forall s \in \mathcal{S}$. Since each term in the summation is negative, we have that*

$$\pi(s, a)g^\pi(s, a) = 0 = \sqrt{\pi(s, a)}g^\pi(s, a), \forall a \in \mathcal{A}(s), s \in \mathcal{S} \text{ and hence } G = \hat{K}.$$

Proof of Theorem 2

Proof. The update of π on the slower time-scale can be re-written as

$$\pi_{n+1}(s,a) = \Gamma\left(\pi_n(s,a) + b(n)(H(\pi_n) + \eta_n)\right), \text{ where} \tag{7}$$

$H(\pi_n) = \sqrt{\pi_n(s,a)}g^{\pi}(s,a)$ and $\eta_n = \sqrt{\pi_n(s,a)}g_n(s,a) - H(\pi_n)$. We can infer the claim regarding convergence of π_n governed by (7) using Kushner-Clark lemma (Theorem 2.3.1 in [10]), if we verify the following:

(i) H is a continuous function. **(ii)** The sequence $\eta_n, n \geq 0$ is a bounded random sequence with $\eta_n \to 0$ almost surely as $n \to \infty$. **(iii)** The step-sizes $b(n), n \geq 0$ satisfy $b(n) \to 0$ as $n \to \infty$ and $\sum_n b(n) = \infty$.

Now, (i) follows by definition of H and (iii) by assumption on step-sizes. Consider (ii): η_n is bounded since we consider a finite state-action space setting ($\Rightarrow g(s,a)$ is bounded) and π is trivially upper-bounded. From Theorem 3, $v_n \to v^{\pi}$ a.s. as $n \to \infty$ and hence, $\eta_n \to 0$ a.s. The claim follows. □

Remark 3. *(**Avoidance of traps**) Note that from the foregoing, the set K comprises of both stable and unstable attractors and in principle from Lemma 5, the iterates π_n governed by (4) can converge to an unstable equilibrium. A standard trick to avoid such traps, as discussed in Chapter 4 of [5], is to introduce additional noise in the iterates. For this purpose, we perturb the policy every $\tau > 0$ iterations to obtain a new policy $\hat{\pi}$ as follows:*

$$\hat{\pi}(s,a) = \frac{\pi(s,a) + \eta}{\sum\limits_{a \in \mathcal{A}} (\pi(s,a) + \eta)}, a \in \mathcal{A}. \tag{8}$$

The above scheme ensures that the convergence of the policy sequence π_n governed by (3) *is to the stable set $\hat{K}$.*

Step 3: Global Optimality Here we establish that our algorithm converges to a globally optimal policy.

Lemma 6. *If $\pi \in \hat{K}$, then π is globally optimal and the corresponding value function v^{π} is the same as the optimal value v^*.*

Proof.

$$\text{If } \pi(s,a) > 0, \text{ then } g(s,a) = 0 \Rightarrow v^{\pi}(s) = r(s,a) + \beta \sum_{y \in U(s)} p(y|s,a) v^{\pi}(y).$$

$$\text{If } \pi(s,a) = 0, \text{ then } g(s,a) \leq 0 \Rightarrow v^{\pi}(s) \geq r(s,a) + \beta \sum_{y \in U(s)} p(y|s,a) v^{\pi}(y).$$

$$\text{Thus, it follows that } \forall s \in \mathcal{S}, \quad v^{\pi}(s) = \max_{a \in \mathcal{A}(s)} \left[r(s,a) + \beta \sum_{y \in U(s)} p(y|s,a) v^{\pi}(y) \right].$$

□

6 Extension to incorporate function approximation

The actor-critic algorithm described in Section 4 is infeasible for implementation in high-dimensional settings where the state and action spaces are large. A standard approach to alleviate this problem is to employ function approximation techniques and parameterize the value function and policies as follows:

Value function Using a linear architecture, the value function is approximated as $v^\pi(s) \approx f(s)^\mathsf{T} w$, for any given policy π. Here $f(s)$ is the *state feature vector* and w is the *value function parameter*, both in some low-dimensional subspace R^{d_1}, with $d_1 << |\mathcal{S}|$.

Policies We consider a parameterized class of policies such that each policy is continuously differentiable in its parameter. A common approach is to employ the Boltzmann distribution to obtain the following form for policies: $\pi^\theta(s,a) \approx \dfrac{e^{\theta^T \phi(s,a)}}{\sum\limits_{b \in \mathcal{A}} e^{\theta^T \phi(s,b)}}$. Here $\phi(s,a)$ is a *state-action feature vector* and θ is the *policy parameter vector*, both assumed to be in a compact subset $\mathcal{C} \in \mathrm{R}^{d_2}$.

Update rule Choose $a_n \sim \pi^{\theta_n}(\cdot, s_m)$ and observe the reward $r(s_n, a_n)$. Then, update the critic parameter w_n and policy parameter θ_n as follows:

$$\textbf{TD Error:} \quad g_n(s_n,a_n) := r(s_n,a_n) + \beta f(s_{n+1})^\mathsf{T} w_n - f(s_n)^\mathsf{T} w_n, \tag{9}$$

$$\textbf{Critic:} \quad w_{n+1} = w_n + c(n) g_n(s_n,a_n) f(s_n), \tag{10}$$

$$\textbf{Actor:} \quad \theta_{n+1} = \hat{\Gamma}\big(\theta_n + b(n)\pi_n(s_n,a_n)^{3/2}\psi_n(s_n,a_n) g_n(s_n,a_n)\big). \tag{11}$$

In the above, $\hat{\Gamma}$ projects any θ onto a compact set $\mathcal{C} \subset \mathrm{R}^{d_2}$ and $\psi_n(s_n,a_n) = \dfrac{\partial \log \pi_n(s_n,a_n)}{\partial \theta_n}$ are the *compatible features*. For Boltzmann policies, $\psi_n(s_n,a_n) = \phi_n(s_n,a_n) - \sum\limits_{b \in \mathcal{A}} \pi_n(s_n,b)\phi_n(s_n,b)$.

The critic recursion above follows from the standard TD(0) with function approximation update. The idea is to have the increment $\Delta w_n \propto \left[v_t(s_n) - f(s_n)^T w_n\right]^2$, where $v_t(s_n) = r(s_n,a_n) + \beta f(s_{n+1})^\mathsf{T} w_n$ is the current estimate of the return. A natural update increment for the actor recursion is to have
$\Delta\theta_n \propto -\dfrac{\partial J}{\partial \theta_n} = -\dfrac{\partial J}{\partial \pi_n} \cdot \dfrac{\partial \pi_n}{\partial \theta_n} = \sqrt{\pi_n(s_n,a_n)}\, g_n(s_n,a_n)\pi_n(s_n,a_n)\psi_n(s_n,a_n)$.

Preliminary result:

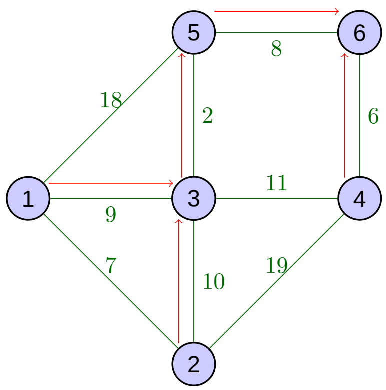
In addition to irreducibility of the underlying Markov chain for any policy and differentiability of the policy, we assume that the feature matrix Φ with rows $f(s)^\mathsf{T}, \forall s \in \mathcal{S}$ is full rank. These assumptions are standard in the analysis of actor-critic algorithms (cf. [4]). Let $d^{\pi^\theta}(s) = (1-\beta)\sum_{n=0}^\infty \beta^n \Pr(s_n = s | s_0; \pi^\theta)$ for any policy $\theta \subset \mathcal{C}$. Let $\bar{K}$ denote the set of all equilibria of the ODE:

$$\dot{\theta}(t) = \check{\Gamma}\left(\sum_{s \in \mathcal{S}} d^{\pi^{\theta(t)}}(s) \sum_{a \in \mathcal{A}} \pi^{\theta(t)}(s,a) \nabla \pi^{\theta(t)} \big(r(s,a) + \beta \sum_{s' \in \mathcal{S}} p(s' \mid s,a) {w^{\theta(t)}}^\mathsf{T} f(s') - {w^{\theta(t)}}^\mathsf{T} f(s)\big)\right). \tag{12}$$

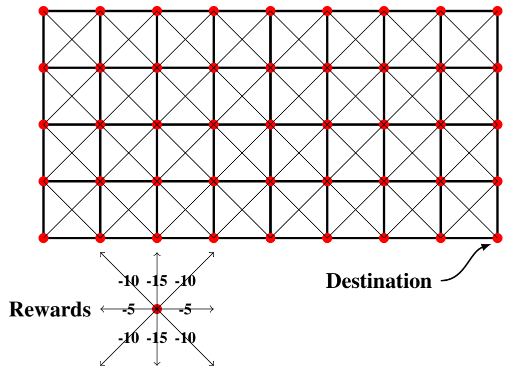
Theorem 7. *The iterates* $(w_n, \theta_n), n \geq 0$ *governed by* (11) *satisfy*

$$(w_n,\theta_n) \to \tilde{K} \text{ a.s. as } n \to \infty, \text{ where } \tilde{K} = \{(w^\theta, \theta) \mid \theta \in \bar{K}\}.$$

In the above, w^θ *is the solution to* $Aw^\theta = b$, *where* $A = \Phi^\mathsf{T}\Psi_\theta(I - \beta P)\Phi$ *and* $b = \Phi^\mathsf{T}\Psi_\theta r$ *with* Ψ_θ *is a diagonal matrix with the stationary distribution of the Markov chain underlying policy with parameter* θ *as the diagonal entries and* r *is a column vector with entries* $\sum_a \pi^\theta(s,a) r(s,a)$, *for each* $s \in \mathcal{S}$.



(a) Six node graph



(b) 44 node graph

Figure 1: Network graphs with associated rewards

Node	Value function	MPA[1]	Probability
1	-17.83	2	0.87
2	-19.64	2	0.96
3	-9.24	1	0.95
4	-6.00	1	0.96
5	-8.22	1	0.92

(a) AC-OPT algorithm

Node	Q(s,1)	Q(s,2)	Q(s,3)	Q(s,4)
1	-24.4	**-15.72**	-20.376	N.A
2	-25.72	**-16.72**	-19.576	N.A
3	**-8.4**	-15.8	-23.376	-21.576
4	**-6**	-17.72	-32.376	N.A
5	**-8**	-8.72	-30.576	N.A

(b) Q-learning algorithm

Figure 2: Performance of Q-learning and actor-critic algorithms on six node network graph

7 Simulation Experiments

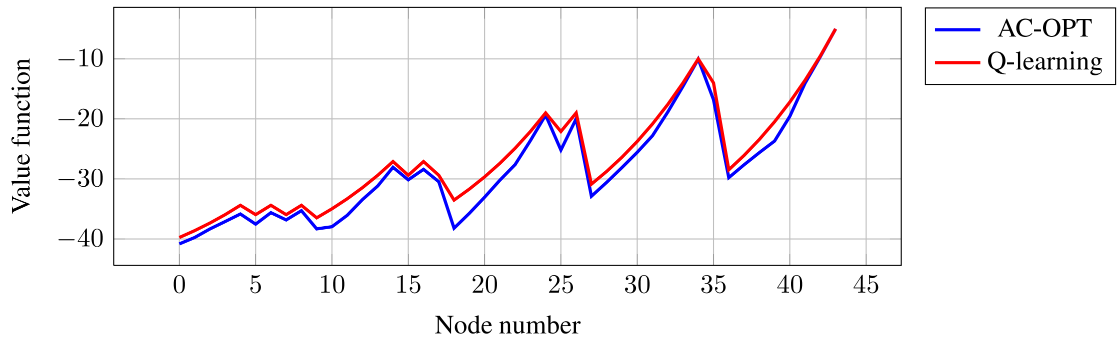
Setup Routing packets through a communication network is a natural application for reinforcement learning algorithms. Q-routing, that is, using Q-learning for routing packets in dynamically changing networks has been investigated among others by [6] and [3]. We have considered a highly simplified version of the problem over two network graph settings:

Six node graph As shown in Fig. 1a, the state space here consists of the nodes themselves, that is $\mathcal{S} = \{1, 2, 3, 4, 5, 6\}$, and the number of actions in a state corresponds to the number of neighbouring nodes to which a packet can be routed from the given node. The next state is chosen randomly and node 6 is the *absorbing* destination node. Further, each run started from state 1 and the initial estimate of the Q-value was 0 for all states. Rewards in each transition are negative of the edge weight (as depicted in Fig. 1a).

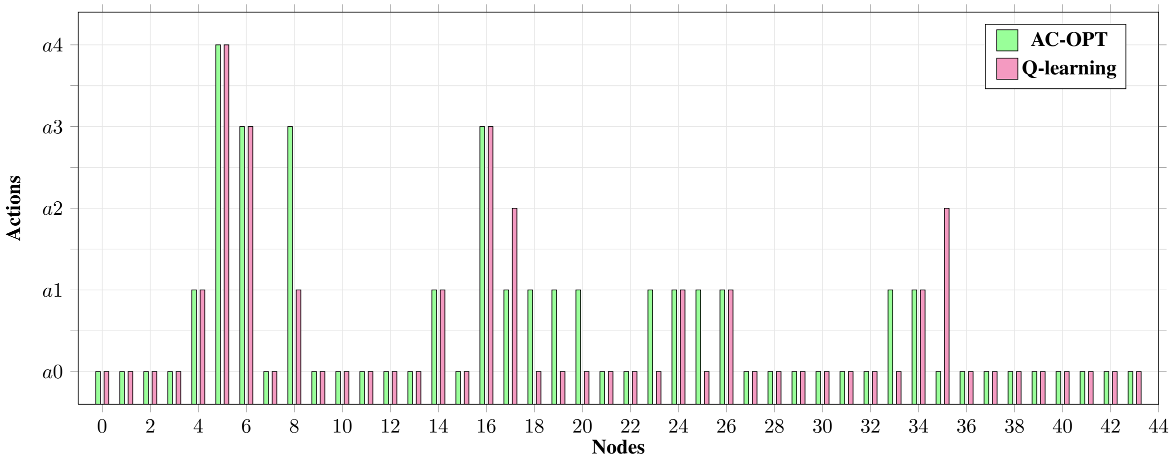
44 **node graph** As shown in Fig. 1b, the state space here is $\mathcal{S} = \{0, 1, 2, ..., 43, 44\}$, with 44 being the destination node. The actions are as follows: at any node start from direction east and move in clockwise direction. 1^{st} action is $a0$, second action is $a1$ and so on. For all actions, rewards are shown in Fig. 1b.

On these two settings, we implemented both the Q-learning and our actor-critic algorithm (henceforth, referred to as AC-OPT). For both algorithms, we set the discount factor $\beta = 0.8$. The initial randomized

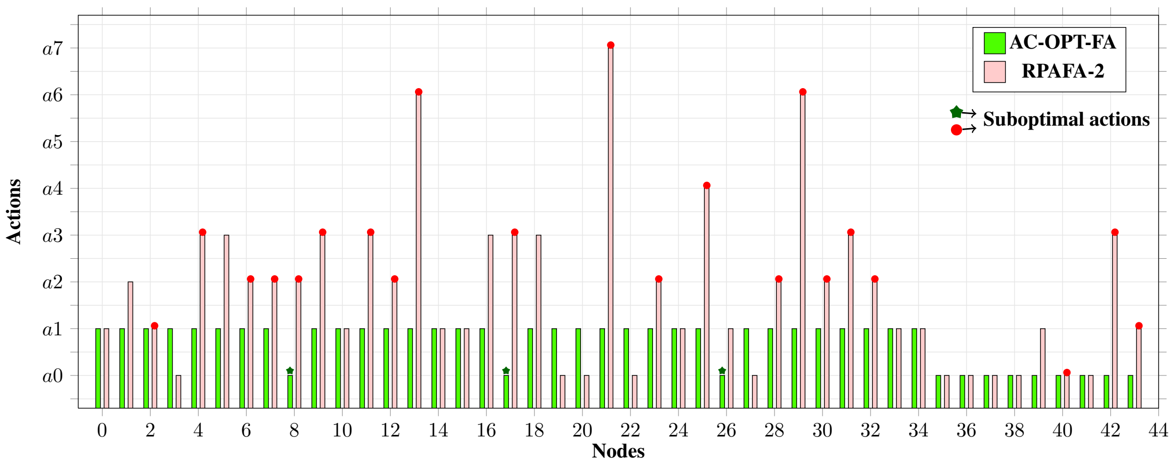
[1] MPA stands for "Most probable action".



(a) Value function



(b) Recommended actions, state-wise,for full-state algorithms: Q-learning and AC-OPT.



(c) Recommended actions, state-wise, for function approximation algorithms: AC-OPT-FA and RPAFA-2

Figure 3: Performance comparison on a 44-node network graph

policy was set to the uniform distribution. For AC-OPT, the policy was perturbed every $\tau = 10$ iterations (see Remark 3). All the results presented are averaged over 50 independent runs of the respective algorithm.

Results The tales in Figs. 2a–2b present the results obtained upon convergence of the AC-OPT and Q-learning algorithms for the six node network graph setting, respectively. It is evident that both algorithms converge to the optimal policy. While Q-learning recommends the best action using Q-values, AC-OPT, being randomized, suggests the optimal action with high probability.

Fig. 3a presents the value function estimates obtained from both algorithms on the 44 node network graph, while Fig. 3b compares the actions suggested by both algorithms upon convergence, for each state(=node) in the network graph. It is evident that AC-OPT recommends the same (as well as optimal) actions as Q-learning on almost all the states. Even though there is change in the recommended actions on a small number of states, the difference in value estimates here is negligible.

Function approximation We show here the results the function approximation variant of our actor-critic algorithm (henceforth referred to as AC-OPT-FA) and the RPAFA-2 algorithm from [1]. For any state s, let $a \equiv \lfloor \frac{s}{9} \rfloor$ and $b \equiv s \mod 9$. Then, the state features are chosen as: $f(s) = (4 - a, 8 - b, 4 + a - b, 1)^{\mathsf{T}}$. Along similar lines, the state-action feature $\phi(s, a) = (4 - a, 8 - b, 4 + a - b, r(x, y), 1)^{\mathsf{T}}$.

Fig. 3c compares the actions recommended by AC-OPT-FA and RPAFA-2 algorithms, while also highlighting the sub-optimal actions. It is evident that AC-OPT-FA recommends with high probability (≈ 0.9 on the average) the best action with a 93% accuracy. On the other hand. RPAFA-2 achieved only a 50% accuracy, i.e., sub-optimal actions suggested over half of the state space.

8 Conclusions

In this paper, we proposed a new actor-critic algorithm with guaranteed convergence to the optimal policy in a discounted MDP. The proposed algorithm was validated through simulations on a simple shortest path problem in networks. A topic of future study is to strengthen the convergence result of the function approximation variant of our actor-critic algorithm.

Appendix

A Proofs for the actor-critic algorithm

Lemma 8. *Let $R_\pi = \langle r(s, \pi), s \in \mathcal{S} \rangle^T$ be a column vector of rewards and $P_\pi = [p(y|s, \pi), s \in \mathcal{S}, y \in \mathcal{S}]$ be the transition probability matrix, both for a given π. Then, the system of ODEs (6) has a unique globally asymptotically stable equilibrium given by*

$$\mathbf{v}_\pi = [I - \beta P_\pi]^{-1} R_\pi. \tag{13}$$

Proof. The system of ODEs (6) can be re-written in vector form as given below.

$$\frac{dv}{dt} = R_\pi + \beta P_\pi v - v. \tag{14}$$

Rearranging terms, we get

$$\frac{dv}{dt} = R_\pi + (\beta P_\pi - I)v,$$

where I is the identity matrix of suitable dimension. Note that for a fixed π, this ODE is linear in v and moreover, all the eigenvalues of $(\beta P_\pi - I)$ have negative real parts. Thus by standard linear systems theory, the above ODE has a unique globally asymptotically stable equilibrium which can be computed by setting $\frac{dv}{dt} = 0$, that is, $R_\pi + (\beta P_\pi - I)v = 0$. The trajectories of the ODE (14) converge to the above equilibrium starting from any initial condition in lieu of the above. □

Proof of Theorem 3

For establishing the proof, we require the notion of (T, δ)-perturbation of an ODE, defined as follows:

Definition 1. *Consider the ODE*

$$\dot{x}(t) = f(x(t)). \tag{15}$$

Given $T, \delta > 0$, *we say that* $\bar{x}(\cdot)$ *is a* (T, δ)*-perturbation of* (15)*, if there exist* $0 = T_0 < T_1 < T_2 < \cdots < T_n \uparrow \infty$ *such that* $T_{n+1} - T_n \geq T$*, for all* $n \geq 0$ *and* $\sup_{t \in [T_n, T_{n+1}]} \| \bar{x}(t) - x(t) \| < \delta$*, for all* $n \geq 0$.

Let $\mathbb{Z}$ be the globally asymptotically stable attractor set for (15) and $\mathbb{Z}^\epsilon$ be the ϵ-neighborhood of $\mathbb{Z}$. Then, the following lemma by Hirsch (see Theorem 1 on pp. 339 of [7]) is useful in establishing the convergence of a (T, δ)-perturbation to the limit set Z^ϵ.

Lemma 9 (Hirsch Lemma)**.** *Given* ϵ, $T > 0$, $\exists \bar{\delta} > 0$ *such that for all* $\delta \in (0, \bar{\delta})$*, every* (T, δ)*-perturbation of (15) converges to* $\mathbb{Z}^\epsilon$.

Proof. ***(Theorem 3)*** Fix a state $s \in \mathcal{S}$. Let $\{\bar{n}\}$ represent a sub-sequence of iterations in algorithm (3) when the state is $s \in \mathcal{S}$. Also, let $Q_n = \{\bar{n} : \bar{n} < n\}$. For a given π, the updates of v on the slower time-scale $\{c(n)\}$ given in algorithm (3) can be re-written as

$$v_{\bar{n}+1}(s) = v_{\bar{n}}(s) + c(n) \left[\sum_{a \in \mathcal{A}(s)} \pi_{\bar{n}}(s,a) g_{\pi_{\bar{n}}}(s,a) + \tilde{\chi}_{\bar{n}} \right], \tag{16}$$

where $\tilde{\chi}_{\bar{n}} = r(s,a) + \beta v_{\bar{n}}(s') - \sum\limits_{a \in \mathcal{A}(s)} \pi_{\bar{n}}(s,a) g_{\pi_{\bar{n}}}(s,a)$, is the noise term. Let $\tilde{M}_n = \sum\limits_{m \in Q_n} c(m) \tilde{\chi}_m$. Then, $\tilde{M}_n, n \geq 0$, is a convergent martingale sequence by the martingale convergence theorem (since $\sum\limits_{\bar{n}} c^2(\bar{n}) < \infty$ and $\|g\| \stackrel{\triangle}{=} |g_{(\cdot)}(s,a)| < \infty$). The equation (16) can now be seen to be a (T, δ)-perturbation of the system of ODEs (6). Thus, by Lemma 9, it can be seen that v_n converges to the globally asymptotically stable equilibrium v_π (see equation (13)) of the system of ODEs (6). □

Proof of Lemma 4

Proof.

If part: If $\pi \in L$ and $\sqrt{\pi(s,a)} g^\pi(s,a) = 0, \forall a \in \mathcal{A}, s \in \mathcal{S}$ holds, then by definition of operators Γ and $\bar{\Gamma}$, the result follows.

Only if part: The operator $\bar{\Gamma}$, by definition, ensures that $\pi \in L$. Suppose for some $a \in \mathcal{A}(s), s \in \mathcal{S}$, we have $\bar{\Gamma}(\sqrt{\pi(s,a)} g_\pi(s,a)) = 0$ but $\sqrt{\pi(s,a)} g_\pi(s,a) \neq 0$. Then, $g_\pi(s,a) \neq 0$ and since $\pi \in L$, $1 \geq \pi(s,a) > 0$. We analyze this by considering the following two cases:
(i) $1 > \pi(s,a) > 0$ and $g_\pi(s,a) \neq 0$: In this case, it is possible to find a $\Delta > 0$ such that for all $\delta \leq \Delta$,

$$1 > \pi(s,a) + \delta \sqrt{\pi(s,a)} g_\pi(s,a) > 0.$$

This implies that

$$\bar{\Gamma}\left(\sqrt{\pi(s,a)}g_\pi(s,a)\right) = \sqrt{\pi(s,a)}g_\pi(s,a) \neq 0,$$

which contradicts the initial supposition.

(ii) $\pi(s,a) = 1$ and $g_\pi(s,a) \neq 0$: Since v_π is solution to the system of ODEs (6), the following should hold:

$$\sum_{\hat{a}\in\mathcal{A}(s)} \pi(s,\hat{a})g_\pi(s,\hat{a}) = \pi(s,a)g_\pi(s,a) = 0.$$

This again leads to a contradiction.

The result follows. □

B Proofs for the function approximation variant

Proof of Theorem 7

Proof. Due to timescale separation, we can assume that the policy parameter θ is constant for the sake of analysis of the critic recursion in (11). For any fixed policy given as parameter θ, the critic recursion in (11) converges to w^θ, which is the TD fixed point (see Theorem 7 statement for the explicit form of w^θ). This is a standard claim for TD(0) with function approximation - see [12] for a detailed proof.

Let $\mathcal{F}_n = \sigma(\theta_m, m \leq n)$. The actor recursion (17) in the main paper can be re-written as

$$\begin{aligned}\theta_{n+1} =& \hat{\Gamma}\Bigg(\theta_n + b(n)\mathbb{E}[\pi_n(s_n,a_n)^{3/2}\psi_n(s_n,a_n)\bar{g}(s_n,a_n) \mid \mathcal{F}_n] \\ &+ b(n)\left(\pi_n(s_n,a_n)^{3/2}\psi_n(s_n,a_n)g_n(s_n,a_n) - \mathbb{E}[\pi_n(s_n,a_n)^{3/2}\psi_n(s_n,a_n)g_n(s_n,a_n) \mid \mathcal{F}_n]\right) \\ &+ b(n)\mathbb{E}\left[\pi_n(s_n,a_n)^{3/2}\psi_n(s_n,a_n)\big(g_n(s_n,a_n) - \bar{g}(s_n,a_n)\big) \mid \mathcal{F}_n\right]\Bigg),\end{aligned} \tag{17}$$

where $\bar{g}(s,a) := r(s,a) + \beta\sum_{s'\in\mathcal{S}} p(s' \mid s,a)w^{\theta(t)^\intercal}f(s') - w^{\theta(t)^\intercal}f(s)$.

Since the critic converges, i.e., $w_n \to w^\theta$ a.s. as $n \to \infty$, the last term in (17) vanishes asymptotically. Let $M_n = \sum_{m=0}^{n-1} \pi_m(s_m,a_m)^{3/2}\psi_m(s_m,a_m)g_m(s_m,a_m) - \mathbb{E}[\pi_m(s_m,a_m)^{3/2}\psi_m(s_m,a_m)g_m(s_m,a_m) \mid \mathcal{F}_n]$. Using arguments similar to the proof of Theorem 2 in [4], it can be seen that M_n is a convergent martingale sequence that converges to zero. So, that leaves out the first term multiplying $b(n)$ in (17). A simple calculation shows that

$$\begin{aligned}&\mathbb{E}[\pi_n(s_n,a_n)^{3/2}\psi_n(s_n,a_n)\bar{g}(s_n,a_n) \mid \mathcal{F}_n] \\ =&\sum_{s\in\mathcal{S}} d^{\pi^{\theta(t)}}(s)\sum_{a\in\mathcal{A}}\pi^{\theta(t)}(s,a)\nabla\pi^{\theta(t)}\big(r(s,a) + \beta\sum_{s'\in\mathcal{S}} p(s' \mid s,a)w^{\theta(t)^\intercal}f(s') - w^{\theta(t)^\intercal}f(s)\big).\end{aligned}$$

The rest of the proof amounts to showing that the RHS above is Lipschitz continuous and that the recursion (17) is a (T,δ) perturbation of the ODE (12) in the main paper. These facts can be verified in a similar manner as in the proof of Theorem 2 in [4] and the final claim follows from Hirsch lemma (see Lemma 9 above). □

C Simulation Experiments

Results for full state representation based algorithms on 44 node graph

Tables. 1–2 present detailed results for our AC-OPT algorithm and Q-learning, respectively on the 44-node network graph setting. For Q-learning results in Table 2, the action achieving the maximum in $\max_a Q(s,a)$) is boldened. It is evident that AC-OPT suggests the same (as well as optimal) actions as that of Q-learning, on almost all the states.

Node no.	Value function	MPA: Probability	Node no.	Value function	MPA: Probability
0	-40.824	$0 : 0.974759$	22	-27.6105	$0 : 0.952729$
1	-39.7619	$0 : 0.940369$	23	-23.6213	$1 : 0.965307$
2	-38.3387	$0 : 0.954584$	24	-19.3607	$1 : 0.956485$
3	-37.1019	$0 : 0.934279$	25	-25.1828	$1 : 0.917481$
4	-35.8406	$1 : 0.977405$	26	-19.9879	$1 : 0.973978$
5	-37.5327	$4 : 0.775096$	27	-32.8828	$0 : 0.962421$
6	-35.618	$3 : 0.726475$	28	-30.5635	$0 : 0.963262$
7	-36.8312	$0 : 0.699411$	29	-28.1035	$0 : 0.935406$
8	-35.2874	$3 : 0.986148$	30	-25.5654	$0 : 0.951051$
9	-38.3211	$0 : 0.966336$	31	-22.8029	$0 : 0.965918$
10	-37.9592	$0 : 0.937302$	32	-18.8625	$0 : 0.955858$
11	-36.0614	$0 : 0.959576$	33	-14.5632	$1 : 0.929352$
12	-33.4332	$0 : 0.95668$	34	-10.0406	$1 : 0.9742$
13	-31.1697	$0 : 0.961255$	35	-16.8062	$0 : 0.928148$
14	-28.057	$1 : 0.95864$	36	-29.7862	$0 : 0.989813$
15	-30.1452	$0 : 0.951196$	37	-27.6444	$0 : 0.966042$
16	-28.4007	$3 : 0.940799$	38	-25.6189	$0 : 0.94836$
17	-30.4659	$1 : 0.863991$	39	-23.6847	$0 : 0.972548$
18	-38.2062	$1 : 0.937154$	40	-19.5683	$0 : 0.99494$
19	-35.7315	$1 : 0.94369$	41	-14.0438	$0 : 0.981092$
20	-33.0474	$1 : 0.930422$	42	-9.6131	$0 : 0.994136$
21	-30.2144	$0 : 0.941161$	43	-5.00005	$0 : 0.939764$

Table 1: Performance of the AC-OPT algorithm (MPA stands for "most probable action") on the 44-node network graph

Results for function approximation based algorithms

Tables. 3 – 4 present the detailed results for the function approximation based algorithms: RPAFA-2 from [1] and our AC-OPT-FA. States that are shown in bold in these tables correspond to those where the respective algorithm recommended a sub-optimal action. It is evident that AC-OPT-FA results in 93% accuracy, i.e., on 93% of the state space, AC-OPT-FA recommended the optimal action with high probability (around 0.9 in almost all states). On the other hand, RPAFA-2 achieved only 50% accuracy.

Node no.(s)	**Q(s, 0)**	**Q(s, 1)**	**Q(s, 2)**	**Q(s, 3)**	**Q(s, 4)**	**Q(s, 5)**	**Q(s, 6)**	Q(s, 7)
0	**−39.7583**	−41.4778	−47.83	N.A	N.A	N.A	N.A	N.A
1	**−38.6203**	−39.9753	−46.4778	−42.83	−40.7824	N.A	N.A	N.A
2	**−37.3559**	−38.3059	−44.9753	−41.4778	−39.7583	N.A	N.A	N.A
3	**−35.951**	−36.451	−43.3059	−39.9753	−38.6203	N.A	N.A	N.A
4	−37.3559	**−34.39**	−41.451	−38.3059	−37.3559	N.A	N.A	N.A
5	−35.951	−36.451	−39.39	−36.451	**−35.951**	N.A	N.A	N.A
6	−37.3559	−34.39	−41.451	**−34.39**	−37.3559	N.A	N.A	N.A
7	**−35.951**	−36.451	−39.39	−36.451	−35.951	N.A	N.A	N.A
8	−41.451	**−34.39**	−37.3559	N.A	N.A	N.A	N.A	N.A
9	**−36.4778**	−38.5253	−45.1728	−50.7824	−44.7583	N.A	N.A	N.A
10	**−34.9753**	−36.6948	−43.5253	−40.1728	−37.83	−45.7824	−49.7583	−43.6203
11	**−33.3059**	−34.6609	−41.6948	−38.5253	−36.4778	−44.7583	−48.6203	−42.3559
12	**−31.451**	−32.401	−39.6609	−36.6948	−34.9753	−43.6203	−47.3559	−40.951
13	**−29.39**	−29.89	−37.401	−34.6609	−33.3059	−42.3559	−45.951	−42.3559
14	−31.451	**−27.1**	−34.89	−32.401	−31.451	−40.951	−47.3559	−40.951
15	**−29.39**	−29.89	−32.1	−29.89	−29.39	−42.3559	−45.951	−42.3559
16	−31.451	−27.1	−34.89	**−27.1**	−31.451	−40.951	−47.3559	−40.951
17	−32.1	−29.89	**−29.39**	−42.3559	−45.951	N.A	N.A	N.A
18	**−33.5253**	−35.8681	−42.7813	−47.83	−41.4778	N.A	N.A	N.A
19	**−31.6948**	−33.7424	−40.8681	−37.7813	−35.1728	−42.83	−46.4778	−39.9753
20	**−29.6609**	−31.3804	−38.7424	−35.8681	−33.5253	−41.4778	−44.9753	−38.3059
21	**−27.401**	−28.756	−36.3804	−33.7424	−31.6948	−39.9753	−43.3059	−36.451
22	**−24.89**	−25.84	−33.756	−31.3804	−29.6609	−38.3059	−41.451	−34.39
23	**−22.1**	−22.6	−30.84	−28.756	−27.401	−36.451	−39.39	−36.451
24	−24.89	**-** 19	−27.6	−25.84	−24.89	−34.39	−41.451	−34.39
25	**−22.1**	−22.6	- 24	−22.6	−22.1	−36.451	−39.39	−36.451
26	−27.6	**-** 19	−24.89	−34.39	−41.451	N.A	N.A	N.A
27	**−30.8681**	−33.4766	−40.629	−45.1728	−38.5253	N.A	N.A	N.A
28	**−28.7424**	−31.0852	−38.4766	−35.629	−32.7813	−40.1728	−43.5253	−36.6948
29	**−26.3804**	−28.4279	−36.0852	−33.4766	−30.8681	−38.5253	−41.6948	−34.6609
30	**−23.756**	−25.4755	−33.4279	−31.0852	−28.7424	−36.6948	−39.6609	−32.401
31	**−20.84**	−22.195	−30.4755	−28.4279	−26.3804	−34.6609	−37.401	−29.89
32	**−17.6**	−18.55	−27.195	−25.4755	−23.756	−32.401	−34.89	−27.1
33	**-** 14	−14.5	−23.55	−22.195	−20.84	−29.89	−32.1	−29.89
34	−17.6	**-** 10	−19.5	−18.55	−17.6	−27.1	−34.89	−27.1
35	- 15	−14.5	**-** 14	−29.89	−32.1	N.A	N.A	N.A
36	**−28.4766**	−42.7813	−35.8681	N.A	N.A	N.A	N.A	N.A
37	**−26.0852**	−30.629	−37.7813	−40.8681	−33.7424	N.A	N.A	N.A
38	**−23.4279**	−28.4766	−35.8681	−38.7424	−31.3804	N.A	N.A	N.A
39	**−20.4755**	−26.0852	−33.7424	−36.3804	−28.756	N.A	N.A	N.A
40	**−17.195**	−23.4279	−31.3804	−33.756	−25.84	N.A	N.A	N.A
41	**−13.55**	−20.4755	−28.756	−30.84	−22.6	N.A	N.A	N.A
42	**−9.5**	−17.195	−25.84	−27.6	- 19	N.A	N.A	N.A
43	**-** 5	−13.55	−22.6	- 24	−22.6	N.A	N.A	N.A

Table 2: Performance of Q-learning algorithm on the 44-node network graph

Node	Value function	MPA: Probability	Node	Value function	MPA: Probability
0	-52.8351	$1 : 0.975949$	22	-28.674	$1 : 0.96989$
1	-50.4398	$1 : 0.969893$	23	-26.2787	$1 : 0.969891$
2	-48.0445	$1 : 0.969893$	24	-23.8834	$1 : 0.96989$
3	-45.6493	$1 : 0.969893$	25	-21.4882	$1 : 0.96989$
4	-43.254	$1 : 0.969893$	**26**	-19.0929	$0 : 0.513957$
5	-40.8587	$1 : 0.969893$	27	-30.965	$1 : 0.975946$
6	-38.4635	$1 : 0.969893$	28	-28.5698	$1 : 0.96989$
7	-36.0682	$1 : 0.969893$	29	-26.1745	$1 : 0.96989$
8	-33.6729	$0 : 0.513958$	30	-23.7792	$1 : 0.969891$
9	-45.545	$1 : 0.975946$	31	-21.384	$1 : 0.96989$
10	-43.1498	$1 : 0.96989$	32	-18.9887	$1 : 0.96989$
11	-40.7545	$1 : 0.96989$	33	-16.5934	$1 : 0.96989$
12	-38.3592	$1 : 0.96989$	34	-14.1982	$1 : 0.969891$
13	-35.964	$1 : 0.969891$	35	-11.8029	$0 : 0.513957$
14	-33.5687	$1 : 0.96989$	36	-23.675	$0 : 0.999869$
15	-31.1734	$1 : 0.96989$	37	-21.2797	$0 : 0.993623$
16	-28.7782	$1 : 0.96989$	38	-18.8845	$0 : 0.993624$
17	-26.3829	$0 : 0.513957$	39	-16.4892	$0 : 0.993624$
18	-38.255	$1 : 0.975946$	40	-14.0939	$0 : 0.993623$
19	-35.8598	$1 : 0.96989$	41	-11.6987	$0 : 0.993623$
20	-33.4645	$1 : 0.969891$	42	-9.30341	$0 : 0.993624$
21	-31.0692	$1 : 0.96989$	43	-6.90814	$0 : 0.993624$

Table 3: Performance of the function approximation variant AC-OPT-FA on the 44-node network graph

Node	MPA: Probability	Node	MPA: Probability
0	$1 : 0.504191$	22	$0 : 0.984263$
1	$2 : 0.330269$	**23**	$2 : 0.497062$
2	$1 : 0.496113$	24	$1 : 0.49855$
3	$0 : 0.330723$	**25**	$4 : 0.996063$
4	$3 : 0.331711$	26	$1 : 0.499916$
5	$3 : 0.50029$	27	$0 : 0.329259$
6	$2 : 0.332378$	**28**	$2 : 0.249082$
7	$2 : 0.498791$	**29**	$6 : 0.255686$
8	$2 : 0.499996$	**30**	$2 : 0.25075$
9	$3 : 0.330108$	**31**	$3 : 0.500413$
10	$1 : 0.201589$	**32**	$2 : 0.249539$
11	$3 : 0.491524$	33	$1 : 0.20215$
12	$2 : 0.249318$	34	$1 : 0.249613$
13	$6 : 0.253784$	35	$0 : 0.999038$
14	$1 : 0.249081$	36	$0 : 0.969508$
15	$1 : 0.249349$	37	$0 : 0.978052$
16	$3 : 0.249717$	38	$0 : 0.330178$
17	$3 : 0.33322$	**39**	$1 : 0.336035$
18	$3 : 0.330103$	40	$0 : 0.996688$
19	$0 : 0.20268$	41	$0 : 0.989921$
20	$0 : 0.202288$	**42**	$3 : 0.498579$
21	$7 : 0.33527$	**43**	$1 : 0.49913$

Table 4: Performance of RPAFA-2 algorithm from [1] on the 44-node network graph

References

[1] M.S. Abdulla and S. Bhatnagar. Reinforcement learning based algorithms for average cost Markov decision processes. *Discrete Event Dynamic Systems*, 17(1):23–52, 2007.

[2] D.P. Bertsekas and J.N. Tsitsiklis. *Neuro-Dynamic Programming (Optimization and Neural Computation Series, 3)*. Athena Scientific, May 1996. ISBN 1886529108.

[3] S. Bhatnagar and K.M. Babu. New algorithms of the Q-learning type. *Automatica*, 44(4):1111–1119, 2008.

[4] S. Bhatnagar, R.S. Sutton, M. Ghavamzadeh, and M. Lee. Natural actor-critic algorithms. *Automatica*, 45(11):2471–2482, 2009.

[5] V.S. Borkar. *Stochastic approximation: a dynamical systems viewpoint*. Cambridge Univ Pr, 2008.

[6] J.A. Boyan and M.L. Littman. Packet routing in dynamically changing networks: A reinforcement learning approach. *Advances in neural information processing systems*, pages 671–671, 1994.

[7] M. W. Hirsch. Convergent activation dynamics in continuous time networks. *Neural Networks*, 2: 331–349, 1989.

[8] V.R. Konda and V.S. Borkar. Actor-Critic–Type Learning Algorithms for Markov Decision Processes. *SIAM Journal on Control and Optimization*, 38:94, 1999.

[9] V.R. Konda and J.N. Tsitsiklis. On actor-critic algorithms. *SIAM Journal on Control and Optimization*, 42(4):1143–1166, 2004.

[10] H.J. Kushner and D.S. Clark. *Stochastic Approximation Methods for Constrained and Unconstrained Systems*, volume 6. Springer-Verlag New York, 1978.

[11] R.S. Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988.

[12] J. N. Tsitsiklis and B. Van Roy. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42(5):674–690, 1997.