Semi-Markov Adaptive Critic Heuristics with Application to Airline Revenue Management

Ketaki Kulkarni
Abhijit Gosavi [*]
Email: gosavia@mst.edu
Susan Murray
Katie Grantham
Department of Engineering Management and Systems Engineering
Missouri University of Science and Technology
Rolla, MO 65409

Abstract

The adaptive critic heuristic has been a popular algorithm in reinforcement learning (RL) and approximate dynamic programming (ADP) alike. It is one of the first RL and ADP algorithms. RL and ADP algorithms are particularly useful for solving Markov decision processes (MDPs) that suffer from the curses of dimensionality and modeling. Many real-world problems however tend to be semi-Markov decision processes (SMDPs) in which the time spent in each transition of the underlying Markov chains is itself a random variable. Unfortunately for the average reward case, unlike the discounted reward case, the MDP does not have an easy extension to the SMDP. Examples of SMDPs can be found in the area of supply chain management, maintenance management, and airline revenue management. In this paper, we propose an adaptive critic heuristic for the SMDP under the long-run average reward criterion. We present the convergence analysis of the algorithm which shows that under certain mild conditions, which can be ensured within a simulator, the algorithm converges to an optimal solution with probability 1. We test the algorithm extensively on a problem of airline revenue management in which the manager has to set prices for airline tickets over the booking horizon. The problem has a large scale, suffering from the curse of dimensionality, and hence it is difficult to solve it via classical methods of dynamic programming. Our numerical results are encouraging and show that the algorithm outperforms an existing heuristic used widely in the airline industry.

Keywords: adaptive critics, actor critics, Semi-Markov, approximate dynamic programming, reinforcement learning.

[*]Corresponding author

1 Introduction

Markov decision problems (MDPs) are problems of sequential decision making in which the system dynamics are governed by Markov chains, and in each state visited by the system, the controller has to select from two or more actions. The goal for the controller is to maximize some function of the rewards earned in each state visited by the system over a finite or infinite time horizon. The MDP was invented in the 1950s by Richard Bellman [1], who developed what is now called the Bellman optimality equation. Outside of operations management, where the MDP has been widely applied, more recently MDPs have found applications in other areas of engineering (autonomous helicopter control [2]) and artificial intelligence (playing computer games [3, 4]).

Classical methods to solve this problem include linear programming and dynamic programming (DP), e.g., value iteration [5] and policy iteration [6]. DP methods break down when the number of state-action pairs is large, e.g., more than a few thousand, which is referred to as the *curse of dimensionality*, and also in the absence of the transition probabilities of the underlying Markov chains, which is called the *curse of modeling*. Typically, on large-scale problems encountered in the real-world, the number of state-action pairs is too large (curse of dimensionality) and the transition probability model too complex (curse of modeling) for classical DP methods to work. This is essentially because it is difficult, if not impossible, to store or process all the elements of the transition probability matrices, which is required in DP. In particular, these matrices are an integral part of the so-called Bellman equation, the solution of which leads to an optimal solution.

It is on problems that suffer from these curses that reinforcement learning (RL) and adaptive/approximate dynamic programming (ADP) methods become useful. RL/ADP methods bypass the transition probability matrices and solve a variant of the underlying Bellman equation without generating the transition probability matrices. These methods usually rely on a simulator of the system which is often easier to generate than the transition probabilities. For textbook references to this topic, see e.g., [7, 8, 9].

In this paper, we present a new adaptive critic algorithm that is intended for use with Semi-MDPs (SMDPs) in which the time spent in each state is a random variable and the performance metric takes the time into account. For the so-called discounted reward case, in which the performance metric is the sum of the net present value of the rewards earned over an infinite time horizon, the adaptive critic algorithm for the MDP has a simple extension to the SMDP that has been studied in [10]; all that changes for the SMDP is the discount factor. However, for the average reward case, where one seeks to maximize the expected reward per unit time, the algorithm for the SMDP *cannot* be developed by a simple change in the MDP algorithm because the update contains the *optimal* value of the performance metric which is unknown at the start. To this end, we introduce an additional step in the algorithm that iteratively updates a scalar to the optimal value of the performance metric. We also establish the convergence of the algorithm to an optimal solution. As mentioned above, RL algorithms are useful when the problem suffers from the curses of dimensionality and modeling. Therefore we test the algorithm on a problem from airline revenue management in which the state-action space is huge and the transition probabilities are hard to obtain. Our algorithm shows encouraging performance on this problem, outperforming an industrial heuristic that is widely used by most airlines. To the best of our knowledge this is the first paper to present a convergent adaptive-critic algorithm for semi-Markov control under average reward.

The rest of this paper is organized as follows. Section 2 provides a background on SMDPs and RL. The new algorithm is described in Section 3. Convergence properties of the algorithm are studied in Section 4. The application of the algorithm to the airline revenue management problem along with numerical results are presented in Section 5, while conclusions drawn from this research are presented in Section 6.

2 SMDPs and RL

The SMDP is a problem of finding the optimal action in each state when the time taken in each state transition is a random variable and this random time is a part of the objective function, namely the long-run average reward in our case. We begin with a discussion of long-run average reward.

2.1 Long-run average reward

We first present some notation needed for our discussion. Let $\mathcal{S}$ denote the finite set of states in the SMDP, $\mathcal{A}(i)$ the finite set of actions allowed in state i, and $\mu(i)$ the action chosen in state i when policy μ is followed, where $\cup_{i\in\mathcal{S}}\mathcal{A}(i) = \mathcal{A}$. Further let $r(.,.,.) : \mathcal{S}\times\mathcal{A}\times\mathcal{S} \to \Re$ denote the one-step immediate reward, $t(.,.,.) : \mathcal{S}\times\mathcal{A}\times\mathcal{S} \to \Re$ denote the time spent in one transition, and $p(.,.,.) : \mathcal{S}\times\mathcal{A}\times\mathcal{S} \to [0,1]$ denote the associated transition probability. Then the *expected* immediate reward earned in state i when action a is chosen in it can be expressed as: $\bar{r}(i,a) = \sum_{j=1}^{|\mathcal{S}|} p(i,a,j)r(i,a,j)$ and the expected time of the associated transition: $\bar{t}(i,a) = \sum_{j=1}^{|S|} p(i,a,j)t(i,a,j)$.

We can now define long-run average reward, or simply average reward, as follows.

Definition 1 *Let*

$$R_\mu(i) \equiv \lim_{k\to\infty} \frac{E_{\hat{\mu}}\left[\sum_{s=1}^{k} \bar{r}(x_s,\mu(x_s))|x_1 = i\right]}{k},$$

and

$$T_\mu(i) \equiv \lim_{k\to\infty} \frac{E_{\hat{\mu}}\left[\sum_{s=1}^{k} \bar{t}(x_s,\mu(x_s))|x_1 = i\right]}{k}.$$

Then for regular Markov chains, from Theorem 7.5 in [11] (pg. 160), the long-run average reward of a policy μ in an SMDP starting at state i is

$$\rho_\mu(i) = \frac{R_\mu(i)}{T_\mu(i)}.$$

For regular Markov chains, $\rho_\mu(.)$ is independent of the starting state.

2.2 SMDP Bellman equation

The average reward SMDP is all about finding the policy μ that maximizes ρ_μ. The optimal average reward will be denoted in this paper by ρ^*. The following is a central result that shows the existence of an optimal solution to the SMDP. The result can be found in any standard text on DP, e.g., Bertsekas [12].

Theorem 1 *For an average-reward SMDP in which all Markov chains are regular, there exists a vector $V \equiv \{V(1), V(2), \ldots, V(|\mathcal{S}|)\}$ and a scalar ρ that solve the following system of equations:*

$$V(i) = \max_{a\in\mathcal{A}(i)} \left[\bar{r}(i,a) - \rho\bar{t}(i,a) + \sum_{j=1}^{|\mathcal{S}|} p(i,a,j)V(j)\right] \text{ for all } i \in \mathcal{S}. \quad (1)$$

Further ρ equals ρ^* the optimal average reward of the SMDP.

Equation (1) is often called the Bellman optimality equation for SMDPs. The above result paves a way for solving the average reward SMDP, since it implies that if one can find a solution to the vector V and the scalar ρ^*, then the following policy d is optimal, where

$$d(i) \in \arg\max_{a \in \mathcal{S}} \left[\bar{r}(i,a) - \rho^* \bar{t}(i,a) + \sum_{j=1}^{|\mathcal{S}|} p(i,a,j)V(j) \right] \text{ for all } i \in \mathcal{S}.$$

3 Adaptive Critics

Werbos [13] invented a framework that is now well-known as *H*euristic *D*ynamic *P*rogramming (HDP). An integral part of this framework is the adaptive critic algorithm which was based on policy iteration. HDP led to the development of numerous significant algorithms, e.g., dual heuristic programming and action-dependent HDP [14]. A parallel development in this area was the remarkable algorithm of Barto et al. [15], which used notions of reinforcement learning. It was shown later in [16] that the algorithm in [15] does converge with probability 1.

We note that the adaptive critic is just one of the many algorithms in ADP/RL. Other well-known algorithms include Q-Learning [17] and approximate policy iteration [18, 7]. Much of this literature has been surveyed in the literature in the textbooks named above. The interested reader is also referred to some recent survey papers: see [19, 20] for a control-theoretic viewpoint and see [21, 22] for an operations research and machine learning viewpoint. ADP/RL currently forms an active area of study within the continuous-time control community; see e.g., [23, 24].

The central idea in adaptive (actor) critics is that a policy is selected (read acting), then the policy is enacted, i.e., simulated (read critiquing the policy), and then it is improved in an adaptive manner. The power of adaptive critics lies in their ability to solve the MDP without having to generate the transition probabilities. This is how the adaptive critics break the curses of modeling and dimensionality. In the rest of this section, we discuss our new algorithm. In subsection 3.1, we present a derivation of the algorithm and the intuition underlying it, and in subsection 3.2, we present the step-by-step details of the algorithm.

3.1 Semi-Markov adaptive critics

The adaptive critic heuristic for the MDP has two main components: an "actor" that selects a policy and the "critic" that evaluates the value function associated with the policy. For the SMDP, we need to add a third step in which the critic in addition to estimating the value function also evaluates the average reward of the policy. This additional step is needed due to the element of average reward and transition times in the Bellman equation for SMDPs.

We will attempt to solve the following version of the Bellman equation:

$$V(i) = \max_{a \in \mathcal{A}(i)} \left[\bar{r}(i,a) - \rho \bar{t}(i,a) + \lambda \sum_{j \in \mathcal{S}} p(i,a,j) V(j) \right], \tag{2}$$

where ρ is a constant. That the above equation has a unique solution follows directly from the theory of discounted reward MDPs. If we replace ρ by ρ^*, as λ tends to 1, the above results in the Bellman equation for SMDPs, i.e., Equation (1). In practice, if the value of λ is set close to 1, Equation (2) behaves like the Bellman optimality equation for SMDPs. We formalize this statement using the following assumption:

Assumption A1: There exists a value for $\bar{\lambda}$ in the interval $(0,1)$ such that for all $\lambda \in (\bar{\lambda}, 1)$, the unique solution, J, of Equation (2) with $\rho \equiv \rho^*$ produces a policy d defined as follows

$$d(i) \in \arg\max_{a \in \mathcal{S}} \left[\bar{r}(i,a) - \rho^* \bar{t}(i,a) + \lambda \sum_{j} p(i,a,j) J(j) \right] \qquad \forall i$$

whose average reward equals ρ^*.

This assumption can be easily verified in practice for problems whose transition structure, i.e., collectively the transition probabilities, rewards, and times, is known. Then, the value of $\bar{\lambda}$ can be determined *empirically*. In section 5, we will verify this assumption on small problems whose transition structures are known. In this paper, we seek to use the above equation on problems whose transition structures are unknown, and hence we will work with a guestimate of the value of $\bar{\lambda}$. Guestimating the value of $\bar{\lambda}$ is a common practice in the literature on policy gradients [25, 26], where it has been shown that the actual value of $\bar{\lambda}$ may depend on the eigenvalues of the transition matrix, and that the value of $\bar{\lambda}$ does not have to be very close to 1. However, since the eigenvalues can be determined only when the structure is available, it is a standard practice in the literature on policy gradients to use a guestimate of $\bar{\lambda}$. In the same spirit, we will use a guestimate in our algorithm.

Discounted ρ-MDP: We will for the sake of analysis construct an auxiliary (or fictitious) discounted MDP for analyzing our SMDP. Note that the equation in (2) can be viewed as the Bellman equation for a discounted MDP in which the reward structure is altered; the immediate reward is a function of ρ, which is fixed and known, and the transition time:

$$r(i,a,j) - \rho t(i,a,j).$$

This auxiliary problem, which we call discounted ρ-MDP, will come in handy when showing the convergence of the algorithm.

The policy that is selected in every step in the algorithm is usually a stochastic policy whose action-selection probabilities are a function of the function P, where $P(i,a)$ is associated with state i and action a. In the algorithm, the value function, V, and the action-selection function, P, are updated via a feedback/reinforcement mechanism that works as follows:

$$X \leftarrow X + \mu[F], \tag{3}$$

where F denotes the feedback and μ is a step size, typically a small scalar less than 1 that is decayed to 0. To derive the reinforcement mechanism for the SMDP, we must exploit Equation (2). Using the step-size-based update shown above in (3), from Equation (2), one can derive the following update for V:

$$V(i) \leftarrow V(i) + \mu \sum_{j \in \mathcal{S}} p(i,a,j)\left[r(i,a,j) - \rho t(i,a,j) + \lambda V(j) - V(i)\right].$$

Using the above and (3), we define our feedback term as follows:

$$F = \left[r(i,a,j) - \rho t(i,a,j) + \lambda V(j) - V(i)\right]. \tag{4}$$

Note the feedback intentionally does not contain the transition probabilities, since we wish to avoid them. Since the feedback is repeatedly sampled in the simulator, the averaging effect of the sampling allows us to leave out the transition probabilities from the update. We can now re-write our update of V as:

$$V(i) \leftarrow V(i) + \mu[F] = V(i) + \mu\left[r(i,a,j) - \rho t(i,a,j) + \lambda V(j) - V(i)\right].$$

Also, P will be updated as follows:

$$P(i,a) \leftarrow P(i,a) + \mu[F] = P(i,a) + \mu\left[r(i,a,j) - \rho t(i,a,j) + \lambda V(j) - V(i)\right].$$

The updating of ρ is done via the update of R and T, where R is the total accumulated reward over the entire simulation and T the total accumulated time. Details of these updates are provided below in the algorithm description. Although we used the symbol μ for all step-sizes for the purpose of exposition of the main idea, we note that convergence criteria require that the step-sizes used for P, V, and ρ be not the same; details are in the description below.

3.2 Steps in algorithm

Step 1. For all l, where $l \in \mathcal{S}$, and $u \in \mathcal{A}(l)$, set $V(l) \leftarrow 0$ and $P(l,u) \leftarrow 0$. Set k, the number of state changes, to 0. Set R, T, and ρ to 0. Set λ to a value close to 1, e.g., 0.99. Set q_0 to a large, computer-permissible, positive value. Run the algorithm for $k_{\max}$ iterations, where $k_{\max}$ is chosen to be a sufficiently large number. Start system simulation at any arbitrary state.

Step 2. Let the current state be i. Select action a with a probability of
$\exp(P(i,a))/\sum_{b\in\mathcal{A}(i)}\exp(P(i,b))$.

Step 3. Simulate action a. Let the next state be j. Let $r(i,a,j)$ be the immediate reward earned in going to j from i under a and $t(i,a,j)$ the time in the same transition. Set $k \leftarrow k+1$ and update $P(i,a)$ using a step size, α:

$$P(i,a) \leftarrow P(i,a) + \alpha\left[r(i,a,j) - \rho t(i,a,j) + \lambda V(j) - V(i)\right]. \tag{5}$$

If $P(i,a) < -q_0$, set $P(i,a) = -q_0$. And if $P(i,a) > q_0$, set $P(i,a) = q_0$.

Step 4. Update V as follows using step size β:

$$V(i) \leftarrow (1-\beta)V(i) + \beta\left[r(i,a,j) - \rho t(i,a,j)) + \lambda V(j)\right]. \tag{6}$$

Step 5. Update ρ, R and T as follows:

$$R \leftarrow R + r(i,a,j);$$

$$T \leftarrow T + t(i,a,j).$$

$$\rho \leftarrow (1-\gamma)\rho + \gamma R/T; \tag{7}$$

Step 6. If $k < k_{max}$, set $i \leftarrow j$ and then go to Step 2. Otherwise, go to Step 7.

Step 7. For each $l \in \mathcal{S}$, select $d(l) \in \arg\max_{b \in \mathcal{A}(l)} P(l,b)$. The policy (solution) generated by the algorithm is $\hat{d}$. Stop.

The idea above is for the algorithm to use a suitable value of λ less than 1 in order to generate a solution to Equation (2) such that ρ approaches the average reward of the policy contained in the value function V. This under Assumption A1 ensures that we obtain the optimal solution.

In the above algorithm description, to increase clarity of presentation, we have suppressed the superscript, k, in V, ρ, P, and also in the step sizes α, β, and γ. However, we will use these superscripts when needed. The "actor" in the algorithm above which selects the policy is in Step 3, while the "critic" updates the value function in Step 4 and the average reward in Step 5. The step sizes α, β and γ must satisfy the following conditions.

Assumption A2:

$$\lim_{k\to\infty} \frac{\gamma^k}{\beta^k} = 0 \text{ and } \lim_{k\to\infty} \frac{\beta^k}{\alpha^k} = 0. \tag{8}$$

The above implies that:

$$\lim_{k\to\infty} \frac{\gamma^k}{\alpha^k} = 0.$$

These conditions are necessary to show convergence of the algorithm mathematically, and it is not difficult to find step-sizes that satisfy these conditions. Examples of step-sizes that obey these conditions will be provided in Section 5 (see Equation (14)). Finally, we note that the action-selection strategy in Step 2 is called Boltzmann strategy and is well-studied in the literature on RL [8].

4 Algorithm Convergence

We now study the convergence of this algorithm. Consider the time-interpolated continuous-time processes (see Kushner and Clark [27] for a classical description of this process) underlying each class of iterates. We will name the iterates x, y, and z, where x will correspond to P, y to V, and z to R and T. Let $x(t), y(t)$, and $z(t)$ denote these continuous-time processes. Consider a synchronous updating of the iterates on two time scales under the following scheme:

$$x^{k+1}(m) = x^k(m) + \alpha^k \left[f_m(x^k, y^k, z^k) + M_x^k(m) \right] \text{ for } m = 1, \ldots.., |\mathcal{S}| \times |\mathcal{A}|$$

$$y^{k+1}(m) = y^k(m) + \beta^k \left[g_m(x^k, y^k, z^k) + M_y^k(m) \right] \text{ for } m = 1, \ldots.., |\mathcal{S}|$$

$$\text{and } z^{k+1} = z^k + \gamma^k \left[h_m(x^k, y^k, z^k) \right].$$

In the above, f, g, and h are functions that will depend on the algorithm, while M^k is the martingale difference sequence in the kth iteration. Under asynchronous updating, the above schemes will be modified by an indicator function multiplied by the step-size; the indicator function will return a 1 if the mth component of the iterate is updated in the kth iteration of the simulator and 0 otherwise.

Note that in the context of our algorithm, for x, $m \equiv (i, a)$, for y, $m \equiv i$, and $z \equiv \rho$, where i denotes the state and a the action. We define the functions f, g, h and the martingale difference sequences as follows:

$$f_m(x^k, y^k, z^k) = f_{i,a}(P^k, V^k, \rho^k) = \sum_{j \in \mathcal{S}} p(i, a, j) \left[r(i, a, j) - \rho^k t(i, a, j) + \lambda V^k(j) - V^k(i) \right];$$

$$M_x^k(m) = M_x^k(i, a) = \left[r(i, a, j) - \rho^k t(i, a, j) + \lambda V^k(j) - V^k(i) \right] - f_{i,a}(P^k, V^k, \rho^k);$$

$$g_m(x^k, y^k, z^k) = g_i(P^k, V^k, \rho^k) = \sum_{j \in \mathcal{S}} p(i, a, j) \left[r(i, a, j) - \rho^k t(i, a, j) + \lambda V^k(j) - V^k(i) \right];$$

$$M_y^k(m) = M_y^k(i) = \left[r(i, a, j) - \rho^k t(i, a, j) + \lambda V^k(j) - V^k(i) \right] - f_{i,a}(P^k, V^k, \rho^k);$$

$$h(x^k, y^k, z^k) = h(P^k, V^k, \rho^k) = R/T - \rho^k.$$

Assumption B1: For fixed $z \in \Re$, the ODE $\frac{dx(t)}{dt} = f(x(t), y(t), z(t))$ and the ODE $\frac{dy(t)}{dt} = g(x(t), y(t), z(t))$ have globally asymptotically stable critical points, $\kappa(z)$ and $\psi(z)$, respectively, such that maps κ and ψ are Lipschitz continuous.

Assumption B2: The ODE $\frac{dz(t)}{dt} = h\left(\kappa(x(t)), \psi(y(t)), z(t)\right)$ has a globally asymptotically stable critical point.

Assumption B3: Iterates x, y, and z remain bounded.

We now present our main result.

Theorem 2 *The adaptive critic algorithm proposed above converges with probability 1 to the optimal policy of the SMDP under Assumption A1.*

Proof For a fixed value of ρ, we have from Theorem 5.13 in [16] that the algorithm converges to a solution that is optimal for the discounted ρ-MDP. If it can be proved that ρ converges to ρ^*, then if Assumptions A2, B1, B2, and B3 hold, the main result in [28] implies that the algorithm converges to the optimal solution of the discounted ρ^*-MDP. Now, if Assumption A1 holds, with a suitable choice of $\bar{\lambda}$, the algorithm will then converge to the optimal solution of the SMDP. Hence what remains to be shown is that the algorithm satisfies Assumptions A2, B1, B2, and B3, and that ρ converges to ρ^*.

Assumption A2 holds as long as step-sizes follow the rules defined in Equation (8). That Assumption B1 is true for *any* value of ρ follows from the result un [16]. Also, that the iterates P and V remain bounded for any value of ρ has been established in [16]. What remains regarding Assumption B3 is to show that ρ will remain bounded. Let

$$\frac{\max_{i,a,j} |r(i,a,j)|}{\min_{i,a,j} t(i,a,j)} = Q < \infty,$$

where we assume that $t(i,a,j) > 0$ always. We can show that $|\rho^k| \leq Q$ for all k. Since R and T are initialized to 0, $|R^k| < k \max_{i,a,j} |r(i,a,j)|$ and $|T^k| < k \min_{i,a,j} t(i,a,j)$. Since $\rho^1 = 0$, we have:

$$|\rho^2| \leq (1-\gamma^1)|\rho^1| + \gamma^1 \frac{\max_{i,a,j} |r(i,a,j)|}{\min_{i,a,j} t(i,a,j)} = \gamma^1 Q < Q;$$

$$\begin{aligned} |\rho^{k+1}| &\leq (1-\gamma^k)|\rho^k| + \gamma^k \left|\frac{R^k}{T^k}\right| \\ &\leq (1-\gamma^k)Q + \gamma^k \left(\frac{k \max_{i,a,j} |r(i,a,j)|}{k \min_{i,a,j} t(i,a,j)}\right) = Q. \end{aligned}$$

We now show that Assumption B2 is true and that ρ tends to ρ^*. For fixed values of ρ, vector P^k will converge to a fixed point that we will denote by $P(\rho)$ and vector V^k will converge to a fixed point to be denoted by $V(\rho)$. If we define

$$\delta^k = h(P^k, V^k, \rho^k) - h(P(\rho^k), V(\rho^k), \rho^k), \tag{9}$$

then the result in [28] implies that δ^k tends to 0. We can define the update of ρ in Step 5 of the algorithm as follows:

$$\rho^{k+1} \leftarrow \rho^k + \gamma^k h(P^k, V^k, \rho^k); \tag{10}$$

We define $\Delta^k = \rho^k - \rho^*$. Then, from the definition of δ^k in Equation (9) and Equation (10), we have that:

$$\Delta^{k+1} = \Delta^k + \gamma^k h(P(\rho^k), V(\rho^k), \rho^k) + \gamma^k \delta^k. \tag{11}$$

Now

$$\frac{\partial h(P, V, \rho)}{\partial \rho} = -1, \text{ since } R \text{ and } T \text{ are not explicitly functions of } \rho,$$

and hence we have upper and lower bounds on the above derivative. Therefore there exist $L_1, L_2 \in \Re$, where $0 < L_1 \leq L_2$ such that:

$$\begin{aligned} -L_2(\rho_1 - \rho_2) &\leq h_1(P(\rho_1), V(\rho_1), \rho_1) - h_1(P(\rho_2), V(\rho_2), \rho_2) \\ &\leq -L_1(\rho_1 - \rho_2) \end{aligned}$$

for any scalar values of ρ_1, and ρ_2. If the update in Steps 3 and 4 are employed with $\rho^k \equiv \rho^*$, then under Assumption A1, $V^k \to V^*$ and $P^k \to P^*$, which will ensure that algorithm will converge to the optimal policy. As a result, ρ will converge to ρ^*, since it measures the average reward. Thus from (10), $h(P(\rho^*), V(\rho^*), \rho^*) = 0$. So if, $\rho_2 = \rho^*$ and $\rho_1 = \rho^k$, the above will lead to:

$$-L_2 \Delta^k \leq h(P(\rho^k), V(\rho^k), \rho^k) \leq -L_1 \Delta^k.$$

Because $\gamma^k > 0$, the above leads to:

$$-L_2 \Delta^k \gamma^k \leq \gamma^k h(P(\rho^k), V(\rho^k), \rho^k) \leq -L_1 \Delta^k \gamma^k.$$

If we add $\Delta^k + \gamma^k\delta^k$ to all sides of the above, then the result combined with (11) leads to:

$$(1 - L_2\gamma^k)\Delta^k + \gamma^k\delta^k \leq \Delta^{k+1} \leq (1 - L_1\gamma^k)\Delta^k + \gamma^k\delta^k.$$

Then for any $\epsilon > 0$, we have that:

$$\begin{aligned} (1 - L_2\gamma^k)\Delta^k + \gamma^k\delta^k - \epsilon &\leq \Delta^{k+1} \\ &\leq (1 - L_1\gamma^k)\Delta^k + \gamma^k\delta^k + \epsilon. \end{aligned}$$

Then, under using Lemma 4.4 of [29], like in [29], we have that a.s., as $k \to \infty$, $\Delta^k \to 0$, i.e., $\rho^k \to \rho^*$; i.e., Assumption B2 is also satisfied. ∎

Note that whether Assumption A1 holds can be verified when the problem structure is known. In practice, on large-scale problems, this is ruled out, and we must assume that one can guestimate a value for $\bar{\lambda}$ such that Assumption A1 holds.

5 Airline Revenue Management

The problem of setting prices for airline tickets is one that has been studied since 1978 when the airline industry was deregulated in the US. However, it is only in recent times that due to progress in simulation and dynamic programming that has become possible to study this problem using optimal or near-optimal techniques.

The revenue management problem is a problem of sequential decision making in which the decision-maker has to decide on whether to accept or reject an incoming customer. The customer who arrives (virtually) on the web-site of the airline carrier or some other travel agency's website (e.g., *Expedia*) is offered several fares for a given origin-destination. Each airline that offers a seat typically updates its offering regularly, i.e., every fortnight, every week, or every night. The price of the seat offered changes with how much time is remaining for the flight to depart, the preferences of the customer, and a number of other factors. It turns out that unless prices are changed in an intelligent manner, airlines are not able to make the profits needed to survive in what is a business with fierce competition, expensive resources, and low profit-to-expenditure ratios. In what follows, we present an operational perspective of this problem, along with an account of the literature on this topic.

Essentially the problem of setting prices boils down to solving an allocation problem in which a number of seats is allocated to a given fare f_i or all fares greater than f_i. Passengers that pay the same fare are said to belong to the same fare class. When seats for a given fare are sold, the next higher fare class is opened up for the customer. In addition to this is the issue of cancellations and overbooking. Customers often cancel their reservations and pay a price depending on the nature of the ticket they have purchased. As a result, airlines overbook their planes, i.e., they sell more seats than are in the plane. Usually, because of the cancellations, people with reservations are not denied boarding requests (bumping). However, an excessive amount of overbooking can result in many passengers being bumped, which can be very expensive. If no overbooking is done, the plane usually flies with some empty seats, which hurts the carrier, especially on their high demand flights, since the competition is likely to use overbooking and fly with near capacity. Flying with empty seats especially on high-demand flights forces airlines to raise their fares, which in general reduces demand.

The idea of segmentation of the customer demand into different fare classes has roots in the fact that different passengers have different expectations. Some passengers are willing to buy more expensive seats with a lower penalty for cancellation. Of course, such passengers tend to cancel with a higher probability and tend to be business travelers. Leisure travelers plan their trips in advance and hence tend to appear in higher numbers earlier in the booking horizon.

The above described problem can be solved in a heuristic manner via what is popularly called the Expected Marginal Seat Revenue (EMSR) heuristic. It has roots in the Littlewood's equation [30]. The version popular in the industry goes by the name EMSR-b [31]. Excellent reviews of the literature on revenue management have appeared in [32, 33]; see also [34, 35] for textbook references to revenue management. Reinforcement learning has been used for solving the revenue management problem in [36, 37], but the algorithms used were not based on adaptive critics. The algorithm in [36] is a heuristic based on multi-step updates, while the approximate-policy iteration-based algorithm in [37] does have convergence guarantees, but it is both slow and susceptible to a phenomenon called chattering. To the best of our knowledge this research presents the first numerical tests of adaptive critics on a large-scale airline revenue management problem.

5.1 Notation

We now present some of the notation that we will need repeatedly in this section of the paper.

- s_i: the number of seats sold in fare class i
- n: the number of fare classes
- f_i: the fare of class i
- c: class of the current customer
- t: the time remaining for the departure of the plane
- H: the length of the booking horizon
- Y_i: the demand for the number of customers in class i
- C: the capacity of the plane
- Λ: Poisson rate of arrival of all customers

5.2 SMDP

We now present the SMDP model for solving the problem. We assume that the problem has infinite time horizon in which the booking horizon is rolled repeatedly. The goal is to maximize the average reward per unit time. The action space for this problem is composed of two actions: $(Accept, Deny)$. The state space of the problem can be defined as follows:

$$(c, s_1, s_2, \ldots, s_n, \omega_1, \omega_2, \ldots, \omega_n, t) \tag{12}$$

where ω_i is the a vector of size s_i containing the times of arrivals of all the customers with a ticket of class i. The cardinality of the finite component of the state space should not exceed nA^n where

A denotes the maximum number of customers in any given class. Even the finite component of the state space tends to be huge. For the sake of mapping the high-dimensional state space into a lower manageable dimension, we use the following function that was used in [37].

$$PI \equiv \frac{\sum_{i=1}^{n} f_i s_i}{D}, \tag{13}$$

where f_i denotes the fare of class i and D is a scalar whose value must be determined via trial and error. Our state space is now defined as (c, PI) instead of the definition in (12). The result of the above is that we have a state-space with a smaller dimension, which leads us to a manageable number and allows us to use a look-up table.

5.3 EMSR-b

We now explain the EMSR-b heuristic, which we have used as a benchmark for our adaptive critic algorithm. We note that in our notation: $f_1 < f_2 < \cdots < f_n$, where f_i denotes the fare of the ith class. Now

$$\bar{Y}_i = \sum_{j=i}^{n} Y_j$$

will denote the sum of the demands of all fare classes above i and including i. Now, the aggregate revenue for the ith class is defined to be the weighted mean revenue of all fare classes above i and including i as follows:

$$\bar{f}_i = \frac{\sum_{j=i}^{n} f_j \mathsf{E}[Y_j]}{\sum_{j=i}^{n} \mathsf{E}[Y_j]}.$$

The famous Littlewood's equation [30] in the context of the EMSR-b is:

$$\bar{f}_i = \bar{f}_{i+1} Pr\left(\bar{Y}_{i+1} > P_{i+1}\right)$$

for $i = 1, 2, \ldots, n-1$, where P_i, the protection level, denotes the number of seats to protect for fare classes $i, i+1, \ldots, n$. There is no protection level for the lowest class 1. The booking limit for the ith class is defined to be:

$$BL_i = \max\{C - P_{i+1}, 0\}$$

for $i = 1, 2, \ldots, n-1$; note that the booking limit for the highest class n should be the capacity of the plane. Cancellations can be accounted for in Littlewood's equation by replacing C in the above equation by $C/(1-q)$, where q is the mean cancellation probability over all the fare classes. One should note that one requires the distribution of each random variable $\bar{Y}_i$ to solve the above equation.

5.4 Empirical results

We first empirically verify Assumption A1 (subsubsection 5.4.1) and then present empirical results on the revenue management problem (subsubsection 5.4.2).

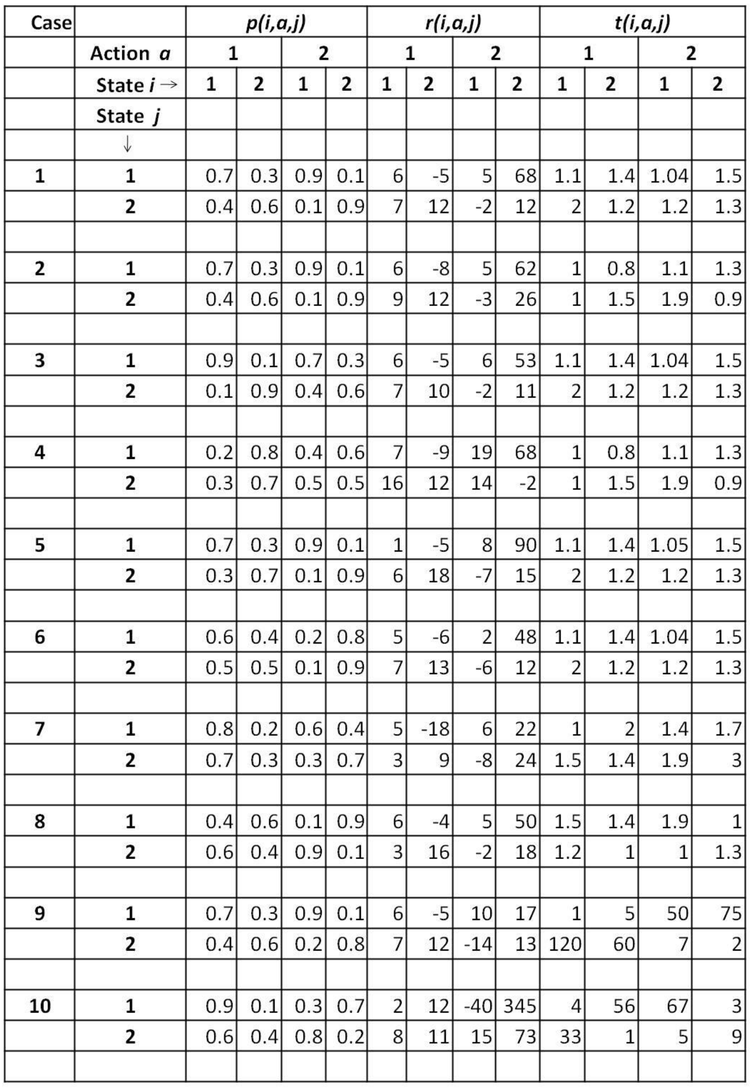
5.4.1 Empirical Verification of Assumption A1

We study 10 small SMDPs whose transition probabilities, rewards, and times are depicted in Figure 1. The value of ρ^* for each SMDP was determined via exhaustive evaluation and then classical discounted value iteration was performed with decreasing values of λ starting from 0.9. The transformation for the discounted value iteration was based on Equation (2) with $\rho = \rho^*$. The policy resulting from this value iteration was compared to the optimal policy obtained from exhaustive evaluation. The value of λ at which this policy differed from the optimal policy was thus determined empirically; this value of course was defined above to be $\bar{\lambda}$. Table 1 shows the value of $\bar{\lambda}$ for all the 10 SMDPs depicted in Figure 1. The interpretation of the results in Table 1 is that solving a λ-SMDP for any value of λ greater than $\bar{\lambda}$ but less than 1 is equivalent to solving the original average reward SMDP. For cases 9 and 10, we find that λ can assume any value between 0 and 1. The results show that in general λ must be chosen carefully, and that in practice a value close to 1 should be a safe choice.

5.4.2 Revenue Management Case Study

We conducted an extensive empirical analysis of our new algorithm on the revenue management problem described above. We conducted a series of experiments on problems with 3 fare classes and four fare classes. The fares for the different fare classes have been detailed in Table 2.

We assume a homogenous Poisson process with a rate $\Lambda = 1.4$ per day for a plane with a capacity of 100 seats. The booking horizon is 100 days long. The Poisson process for each fare



Case		*p(i,a,j)*				*r(i,a,j)*				*t(i,a,j)*			
	Action *a*	1		2		1		2		1		2	
	State *i* →	1	2	1	2	1	2	1	2	1	2	1	2
	State *j* ↓												
1	1	0.7	0.3	0.9	0.1	6	-5	5	68	1.1	1.4	1.04	1.5
	2	0.4	0.6	0.1	0.9	7	12	-2	12	2	1.2	1.2	1.3
2	1	0.7	0.3	0.9	0.1	6	-8	5	62	1	0.8	1.1	1.3
	2	0.4	0.6	0.1	0.9	9	12	-3	26	1	1.5	1.9	0.9
3	1	0.9	0.1	0.7	0.3	6	-5	6	53	1.1	1.4	1.04	1.5
	2	0.1	0.9	0.4	0.6	7	10	-2	11	2	1.2	1.2	1.3
4	1	0.2	0.8	0.4	0.6	7	-9	19	68	1	0.8	1.1	1.3
	2	0.3	0.7	0.5	0.5	16	12	14	-2	1	1.5	1.9	0.9
5	1	0.7	0.3	0.9	0.1	1	-5	8	90	1.1	1.4	1.05	1.5
	2	0.3	0.7	0.1	0.9	6	18	-7	15	2	1.2	1.2	1.3
6	1	0.6	0.4	0.2	0.8	5	-6	2	48	1.1	1.4	1.04	1.5
	2	0.5	0.5	0.1	0.9	7	13	-6	12	2	1.2	1.2	1.3
7	1	0.8	0.2	0.6	0.4	5	-18	6	22	1	2	1.4	1.7
	2	0.7	0.3	0.3	0.7	3	9	-8	24	1.5	1.4	1.9	3
8	1	0.4	0.6	0.1	0.9	6	-4	5	50	1.5	1.4	1.9	1
	2	0.6	0.4	0.9	0.1	3	16	-2	18	1.2	1	1	1.3
9	1	0.7	0.3	0.9	0.1	6	-5	10	17	1	5	50	75
	2	0.4	0.6	0.2	0.8	7	12	-14	13	120	60	7	2
10	1	0.9	0.1	0.3	0.7	2	12	-40	345	4	56	67	3
	2	0.6	0.4	0.8	0.2	8	11	15	73	33	1	5	9

Figure 1: Inputs for problems used for verifying Assumption A1.

Case	Optimal Policy	$\bar{\lambda}$
1	(2,1)	0.93
2	(1,2)	0.77
3	(2,2)	0.65
4	(2,2)	0.98
5	(2,1)	0.27
6	(2,1)	0.4
7	(2,1)	0.7
8	(2,2)	0.76
9	(1,2)	(0,1)
10	(2,2)	(0,1)

Table 1: Values of $\bar{\lambda}$ for 10 small SMDPs.

System	FARES (3 classes)	FARES (4 classes)
FS1a	100,175,250	75,200,400,550
FS1b	100,150,250	80,200,400,500
FS1c	100,175,240	75,150,300,550
FS1d	110,140,255	80,150,400,550
FS1e	120,160,245	70,150,350,550
FS2a	199,275,350	125,180,225,400
FS2b	200,250,335	100,175,250,400
FS2c	200,280,350	100,150,200,450
FS2d	199,260,340	119,139,239,430
FS2e	205,275,330	145,209,280,350

Table 2: Fares in dollars for 3-fare-class and 4-fare-class problems

	3-fare-class systems	4-fare-class systems
Arrival Probability ($Pr(i)$)	0.7,0.2,0.1	0.6,0.25,0.09,0.06
Cancellation Probability	0.1,0.2,0.3	0.1,0.2,0.2,0.4
Cancellation Penalty FS1	100,50,10	70,50,30,10
Cancellation Penalty FS2	120,70,30	100,90,60,40
Bumping Penalty FS1	300	400
Bumping Penalty FS2	400	400

Table 3: Input parameters for systems studied.

class is an independent Poisson process with a rate $\Lambda Pr(i)$ where $Pr(i)$ is the arrival probability of class i. The cancellation probability for a fare class is the probability with which a customer in a given fare class cancels tickets. The cancellation can occur any time with a uniform distribution from the time of booking until the plane departure. The details of the arrival probabilities, the cancellation probabilities and penalties, and the bumping penalties are provided in Table 3.

For the 3-fare-class systems, we used $Div = 500$, and for the 4-fare-class systems, we use $Div = 1000$; Div is an important scaling constant in Equation (13) for generating a lower dimensional state space for our problem. These values were arrived at after much experimentation. We used $\lambda = 0.9$ in our computations, although we find that values as low as 0.5 do not impact the policy obtained. The following step-sizes were used for the three main updates in the algorithm:

$$\alpha^k = 0.01\frac{\log k}{k}; \quad \beta^k = \frac{0.01}{k}; \quad \gamma^k = \frac{0.01}{k \log k}. \tag{14}$$

The algorithm was run in all the systems described above (10 for the 3 fare classes and 10 for the

System	ρ_{EMSR}	ρ_{AC}	IMP (%)
FS1a	116.17	130.72	12.53
FS1b	109.36	118.31	8.18
FS1c	115.18	121.73	5.68
FS1d	114.85	122.68	6.82
FS1e	126.45	135.89	7.47
FS2a	215.23	232.40	7.97
FS2b	208.07	219.82	5.65
FS2c	217.18	230.31	6.05
FS2d	210.89	225.16	6.77
FS2e	218.18	234.16	7.34

Table 4: Results for the 3-fare-class systems

4 fare classes), and the learning phase took at most 26 seconds on an Intel Pentium Processor with a speed of 2.66 GHz on a 64-bit operating system. Once the optimal policy was obtained, it was re-run with the policy frozen for 100 replications, and the values for average reward (ρ) obtained from each replication were averaged to compute the average reward of the policy generated by our algorithm, which we denote by ρ_{AC} for the adaptive-critic algorithm. The results presented in Tables 4 and 5 are from the frozen phase with 100 replications. The solution from EMSR-b, denoted by ρ_{EMSR}, was also estimated by using 100 replications. A t-test was a performed to determine if the results are statistically different, and the results were positive in every case studied at 95% confidence. The improvement in Tables 4 and 5 is defined in percent as:

$$IMP = \frac{\rho_{AC} - \rho_{EMSR}}{\rho_{EMSR}} \times 100.$$

As is clear from the Tables, the improvement in the average reward ranges from 1.35 % to 18.5 %. The average revenue per flight is equal to $\rho \cdot H$. As samples, we present some figures to demonstrate how the average reward improves with the iterations during the learning phase. Figures 2 and 3 show the learning curves for a 3-fare-class and a 4-fare-class system, respectively.

6 Conclusions

The adaptive critic has become an integral part of the literature on ADP/RL since the publication of the influential paper by Werbös [13], where policy iteration was proposed as an important

System	ρ_{EMSR}	ρ_{AC}	IMP (%)
FS1a	142.49	146.22	2.62
FS1b	140.93	146.65	4.05
FS1c	115.87	123.23	6.36
FS1d	129.48	134.66	3.99
FS1e	117.62	121.22	3.05
FS2a	140.64	160.94	14.43
FS2b	127.42	129.14	1.34
FS2c	116.27	132.71	14.13
FS2d	126.89	150.38	18.50
FS2e	164.59	184.89	12.34

Table 5: Results for the 4-fare-class systems

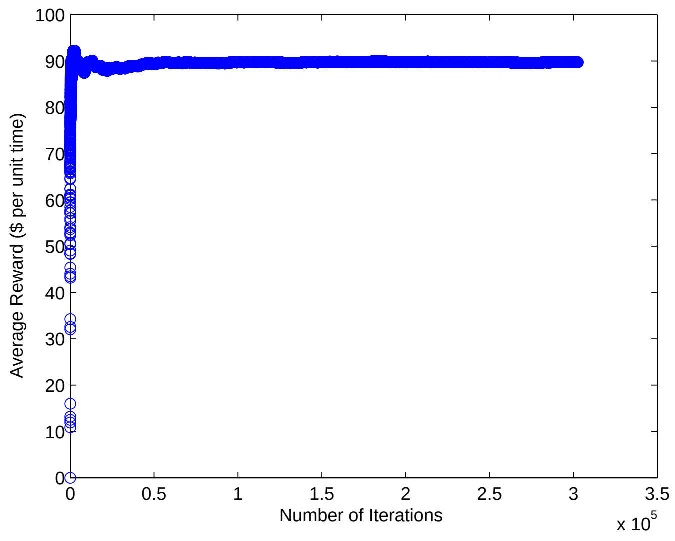


Figure 2: The graph shows how the average reward improves with the iterations during the learning phase for System FS1d for the 3-fare-class problems.

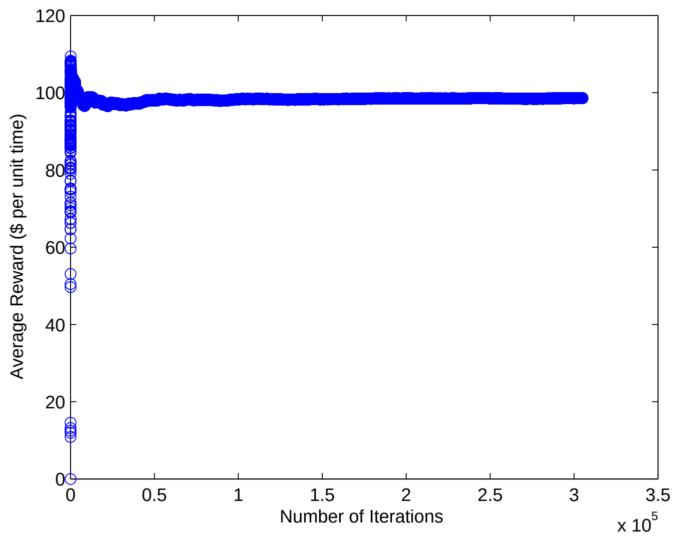


Figure 3: The graph shows how the average reward improves with the iterations during the learning phase for System FS2c for the 4-fare-class problems.

mechanism for ADP/RL. The paper of Barto et al. [15] was a remarkable development in this field. Their algorithm was designed for MDPs, and the convergence of their algorithm was established in [16]. Somewhat surprisingly, although algorithms of this nature preceded the Q-Learning algorithm [17] historically, they have received less attention in the literature. Also, one of the gaps in the literature is an adaptive critic algorithm for the SMDP under average reward. In this paper, we attempt to fill this gap by proposing an algorithm that estimates the optimal average reward simultaneously with the value function. We prove the convergence of our algorithm mathematically using an ordinary differential equations mechanism. We also test it empirically on a large-scale problem of airline revenue management, where the power of ADP/RL becomes more obvious. The empirical performance of the algorithm is encouraging; it outperforms a well-known industrial heuristic.

Acknowledgements: The authors are deeply grateful to all reviewers of this paper for their useful feedback. The second author would like to acknowledge support from the NSF grant ECS: 0841055, which partially funded this research.

References

[1] R. Bellman. The theory of dynamic programming. *Bull. Amer. Math. Soc*, 60:503–516, 1954.

[2] P. Abbeel, A. Coates, M. Quigley, and A. Y. Ng. An application of reinforcement learning to aerobatic helicopter flight. In *In Advances in Neural Information Processing Systems 19*, page 2007. MIT Press, 2007.

[3] G. Tesaru. Td-gammon, a self-teaching backgammon program, achieves master-level play. *Neural Computation*, 6(2):215–219, 1994.

[4] N. Jan van Eck and M. wan Wezel. Application of reinforcement learning to the game of othello. *Computers and Operations Research*, 35(6):1999–2017, 2008.

[5] R. E. Bellman. *Dynamic Programming.* Princeton University Press, Princeton, NJ, 1957.

[6] R. Howard. *Dynamic Programming and Markov Processes.* MIT Press, Cambridge, MA, 1960.

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

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

[9] A. Gosavi. *Simulation-based Optimization: Parametric Optimization Techniques and Reinforcement Learning.* Kluwer Academic, Boston, MA, 2003.

[10] A. Gosavi. Adaptive critics for airline revenue management. In *Conference Proceedings of the Production and Operations Management Society, Dallas*, 2007.

[11] S. Ross. *Applied Probability Models with Optimization Applications.* Dover, New York, 1992.

[12] D. P. Bertsekas. *Dynamic Programming and Optimal Control.* Athena, Belmont, 2nd edition, 2000.

[13] P. J. Werbös. Building and understanding adaptive systems: A statistical/numerical approach to factory automation and brain research. *IEEE Transactions on Systems, Man., and Cybernetics*, 17:7–20, 1987.

[14] G. Venayagamoorthy, R. Harley, and D. Wunsch. Comparison of heuristic dynamic programming and dual heuristic programming adaptive critics for neuro-control of a turbogenerator. *IEEE Transactions on Neural Networks*, 13 (3):764–773, 2002.

[15] A.G. Barto, R.S. Sutton, and C.W. Anderson. Neuronlike elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, 13:835–846, 1983.

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

[17] C.J. Watkins. *Learning from Delayed Rewards.* PhD thesis, Kings College, Cambridge, England, May 1989.

[18] G.A. Rummery and M. Niranjan. On-line Q-learning using connectionist systems. Technical Report CUED/F-INFENG/TR 166. Engineering Department, Cambridge University, 1994.

[19] F.L. Lewis and D. Vrabie. Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE Circuits and Systems Magazine*, 3rd Quarter:32–50, 2009.

[20] S. N. Balakrishnan, J. Ding, and F. L. Lewis. Issues on stability of adp feedback controllers for dynamical systems. *IEEE Transactions on Systems Man and Cybernetics (Special Issue on ADP/RL)*, 38(4):913–917, 2008.

[21] Csaba Szepesvari. *Algorithms for Reinforcement Learning (Synthesis Lectures on Artificial Intelligence and Machine Learning).* Morgan Claypool Publishers, 2010.

[22] A. Gosavi. Reinforcement learning: A tutorial survey and recent advances. *INFORMS Journal on Computing*, 21(2):178–192, 2009.

[23] T. Dierks and S. Jagannathan. Optimal control of affine nonlinear discrete-time systems. In *Proc. of the IEEE Mediterranean Conference on Control and Automation.* IEEE, 2009.

[24] D. Liu, X. Xiong, and Y. Zhang. Action-dependent adaptive critic designs. In *Proc. INNS-IEEE Int. Joint Conf. Neural Networks, Washington, DC*, pages 990–995. IEEE, 2001.

[25] J. Baxter and P. Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence*, 15:319–350, 2001.

[26] S. Singh, V. Tadic, and A. Doucet. A policy-gradient method for semi-Markov decision processes with application to call admission control. *European Journal of Operational Research*, 178(3):808–818, 2007.

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

[28] V. S. Borkar. Stochastic approximation with two-time scales. *Systems and Control Letters*, 29:291–294, 1997.

[29] J. Abounadi, D.P. Bertsekas, and V. Borkar. Learning algorithms for Markov decision processes with average cost. *SIAM Journal of Control and Optimization*, 40:681–698, 2001.

[30] K. Littlewood. Forecasting and control of passenger bookings. *In Proceedings of the 12th AGIFORS (Airline Group of the International Federation of Operational Research Societies Symposium)*, pages 95–117, 1972.

[31] P.P. Belobaba. Application of a probabilistic decision model to airline seat inventory control. *Operations Research*, 37:183–197, 1989.

[32] J.I. McGill and G. J. van Ryzin. Revenue management: Research overview and prospects. *Transportation Science*, 33(2):233–256, 1999.

[33] W-C Chiang, J.C.H. Chen, and X. Xu. An overview of research on revenue management: current issues and future research. *International Journal of Revenue Management*, 1(1):97 – 128, 2007.

[34] K. Talluri and G. van Ryzin. *The Theory and Practice of Revenue Management.* Kluwer Academic, Boston, MA, 2004.

[35] R. Phillips. *Pricing and Revenue Optimization.* Stanford Business Books, 2005.

[36] A. Gosavi, N. Bandla, and T. K. Das. A reinforcement learning approach to a single leg airline revenue management problem with multiple fare classes and overbooking. *IIE Transactions*, 34(9):729–742, 2002.

[37] A. Gosavi. A reinforcement learning algorithm based on policy iteration for average reward: Empirical results with yield management and convergence analysis. *Machine Learning*, 55(1):5–29, 2004.

About the Authors

Ketaki Kulkarni received her B.Tech in Instrumentation and Control from the College of Engineering, Pune, India in May 2007. After completing her undergraduate studies, she worked at Tata Consultancy Services Ltd., Mumbai, India as an Assistant Systems Engineer. She will receive her MS from the department of Engineering Management and Systems Engineering at the Missouri University of Science and Technology, Rolla in May 2011. Her research interests include reinforcement learning, simulation-based optimization, revenue management, and pricing techniques. She is a member of INFORMS and ASEM.

Abhijit Gosavi has a Ph.D. in Industrial Engineering from the University of South Florida, an M. Tech in Mechanical Engineering from IIT Madras, and a BE in Mechanical Engineering from Jadavpur University, Calcutta. He joined Missouri University of Science and Technology in 2008 as an assistant professor. His research interests include simulation-based optimization, revenue management, supply chain management, and production management. He is a member of ASEM, INFORMS, IIE, IEEE, ASEE, and POMS.

Susan Murray is an associate professor in the Engineering Management and Systems Engineering Department at the Missouri University of Science and Technology. Dr. Murray received her B.S. and Ph.D. in industrial engineering from Texas A & M University. Her MS is also in industrial engineering from the University of Texas-Arlington. Her research and teaching interests include productivity improvement, human performance, ergonomics, project management, and engineering education. Prior to her academic position, she spent seven years working in the industry.

Katie Grantham is currently an Assistant Professor in the Department of Engineering Management and Systems Engineering at the Missouri University of Science and Technology. She joined the Interdisciplinary Engineering Department of The University of Missouri-Rolla (now the Missouri University of Science and Technology) as an Assistant Professor in January of 2006. At Missouri S & T she develops and applies risk assessment methodologies to product design principles and leads efforts in the design of a variety of products from bio-medical devices, to the sustainable technologies of solar houses. Before joining the S & T faculty, Dr. Grantham served as a research scientist for 21st Century Systems where she has added risk assessment techniques to their existing defense software products. Also, she was involved with projects to identify both hardware and software failures in mechatronic systems. She received her Ph.D. in Mechanical Engineering at University of Missouri-Rolla in 2005.