The Linear Programming Approach to Approximate Dynamic Programming

D.P. de Farias and B. Van Roy
Department of Management Science and Engineering
Stanford University, Stanford, CA 94305-4023
{pucci,bvr}@stanford.edu

Abstract

The curse of dimensionality gives rise to prohibitive computational requirements that render infeasible the exact solution of large-scale stochastic control problems. We study an efficient method based on linear programming for approximating solutions to such problems. The approach "fits" a linear combination of pre-selected basis functions to the dynamic programming cost-to-go function. We develop error bounds that offer performance guarantees and also guide the selection of both basis functions and "state-relevance weights" that influence quality of the approximation. Experimental results in the domain of queueing network control provide empirical support for the methodology.

1 Introduction

Dynamic programming offers a unified approach to solving problems of stochastic control. Central to the methodology is the cost-to-go function, which is obtained via solving Bellman's equation. The domain of the cost-to-go function is the state space of the system to be controlled, and dynamic programming algorithms compute and store a table consisting of one cost-to-go value per state. Unfortunately, the size of a state space typically grows exponentially in the number of state variables. Known as the *curse of dimensionality*, this phenomenon renders dynamic programming intractable in the face of problems of practical scale.

One approach to dealing with this difficulty is to generate an approximation within a parameterized class of functions, in a spirit similar to that of statistical regression. In particular, to approximate a cost-to-go function J^* mapping a state space $\mathcal{S}$ to reals, one would design a parameterized class of functions $\tilde{J} : \mathcal{S} \times \Re^K \mapsto \Re$, and then compute a parameter vector $r \in \Re^K$ to "fit" the cost-to-go function; i.e., so that

$$\tilde{J}(\cdot, r) \approx J^*.$$

Note that there are two important preconditions to the development of an effective approximation. First, we need to choose a parameterization $\tilde{J}$ that can closely approximate the desired cost-to-go function. In this respect, a suitable choice requires some practical experience or theoretical analysis that provides rough information on the shape of the function to be approximated. "Regularities" associated with the function, for example, can guide the choice of representation. Designing an approximation architecture is a problem-specific task and it is not the main focus of this paper; however, we provide some general guidelines and illustration via case studies involving queueing problems.

Given a parameterization for the cost-to-go function approximation, we need an efficient algorithm that computes appropriate parameter values. The focus of this paper is on an algorithm for computing parameters for linearly parameterized function classes. Such a class can be represented by

$$\tilde{J}(\cdot, r) = \sum_{k=1}^{K} r_k \phi_k,$$

where each ϕ_k is a "basis function" mapping $\mathcal{S}$ to $\Re$ and the parameters $r_1, \ldots, r_K$ represent basis function weights. The algorithm we study is based on a linear programming formulation, originally proposed by Schweitzer and Seidman [27], that generalizes the linear programming approach to exact dynamic programming [5, 12, 13, 14, 19, 22].

Over the years, interest in approximate dynamic programming has been fueled to a large extent by stories of empirical success in applications such as backgammon [30], job shop scheduling [37], elevator scheduling [8] and pricing of American options [21, 33]. These case studies point towards approximate dynamic programming as a potentially powerful tool for large-scale stochastic control. However, significant trial and error is involved in most of the success stories found in the literature, and duplication of the same success in other applications has proven difficult. Factors leading to such difficulties include poor understanding of how and why approximate dynamic programming algorithms work and a lack of streamlined guidelines for implementation. These deficiencies pose a barrier to the use of approximate dynamic programming in industry. Limited understanding also affects the linear programming approach; in particular, though the algorithm was introduced by Schweitzer and Seidmann more than fifteen years ago, there has been virtually no theory explaining its behavior.

We develop a variant of approximate linear programming which represents a significant improvement over the original formulation. While the original algorithm may exhibit poor scaling properties, our version enjoys strong theoretical guarantees and is provably well-behaved for a fairly general class of problems involving queueing networks — and we expect the same to be true for other classes of problems. Specifically, our contributions can be summarized as follows:

- We develop an error bound that characterizes the quality of approximations produced by approximate linear programming. The error is characterized in relative terms, compared against the "best possible" approximation of the optimal cost-to-go function given the selection of basis functions — "best possible" is taken under quotations because it involves choice of a metric by which to compare different approximations. In addition to providing performance guarantees, the error bounds and associated analysis offer new interpretations and insights pertaining to approximate linear programming. Furthermore, insights from the analysis offer guidance in the selection of basis functions, motivating our variant of the algorithm.

  Our error bound is the first to link quality of the approximate cost-to-go function to quality of the " best" approximate cost-to-go function within the approximation architecture not only for the linear programming approach but also for any algorithm that approximates cost-to-go functions of general stochastic control problems by computing weights for arbitrary collections of basis functions.

- We provide analysis, theoretical results and numerical examples that explain the impact of state-relevance weights on the performance of approximate linear programming and offer guidance on how to choose them for practical problems. In particular, appropriate choice of state-relevance weights is shown to be of fundamental importance for the scalability of the algorithm.

- We develop a bound on the cost increase due to using policies generated by the approximation of the cost-to-go function instead of the optimal policy. The bound suggests a natural metric by which to compare different approximations to the cost-to-go function and provides further guidance on the choice of state-relevance weights.

The linear programming approach has been studied in the literature, but almost always with a focus different from ours. Much of the effort has been directed toward efficient implementation of the algorithm. Trick and Zin [31, 32] developed heuristics for combining the linear programming approach with successive state aggregation/grid refinement in two-dimensional problems. Some of their grid generation techniques are based on stationary state distributions, which also

appear in our analysis of state-relevance weights. Paschalidis and Tsitsiklis [24] also apply the algorithm to two-dimensional problems. An important feature of the linear programming approach is that it generates *lower bounds* as approximations to the cost-to-go function; Gordon [15] discusses problems that may arise from that and suggests constraint relaxation heuristics. One of these problems is that the linear program used in the approximate linear programming algorithm may be overly constrained, which may lead to poor approximations or even infeasibility. The approach taken in our work prevents this — part of the different between our variant of approximate linear programming and the original one proposed by Schweitzer and Seidmann is that we include certain basis functions that guarantee feasibility and also lead to improved bounds on the approximation error. Morrison and Kumar [23] develop efficient implementations in the context of queueing network control. Guestrin et al. [17] and Schuurmans and Patrascu [26] develop efficient implementations of the algorithm to factored MDP's. The linear programming approach involves linear programs with a prohibitive number of constraints, and the emphasis of the previous three articles is on exploiting problem-specific structure that allows for the constraints in the linear program to be represented compactly. Alternatively, de Farias and Van Roy [11] suggest an efficient constraint sampling algorithm.

This paper is organized as follows. We first formulate in Section 2 the stochastic control problem under consideration and discuss linear programming approaches to exact and approximate dynamic programming. In Section 3, we discuss the significance of "state-relevance weights," and establish a bound on the performance of policies generated by approximate linear programming. Section 4 contains the main results of the paper, which offer error bounds for the algorithm, as well as associated analyses. The error bounds involve problem-dependent terms, and in Section 5, we study characteristics of these terms in examples involving queueing networks. Presented in Section 6 are experimental results involving problems of queueing network control. A final section offers closing remarks, including a discussion of merits of the linear programming approach relative to other methods for approximate dynamic programming.

2 Stochastic Control and Linear Programming

We consider discrete-time stochastic control problems involving a finite state space $\mathcal{S}$ of cardinality $|\mathcal{S}| = N$. For each state $x \in \mathcal{S}$, there is a finite set of available actions $\mathcal{A}_x$. Taking action $a \in \mathcal{A}_x$ when the current state is x incurs cost $g_a(x)$. State transition probabilities $p_a(x, y)$ represent, for each pair (x, y) of states and each action $a \in \mathcal{A}_x$, the probability that the next state will be y given that the current state is x and the current action is $a \in \mathcal{A}_x$.

A *policy* u is a mapping from states to actions. Given a policy u, the dynamics of the system follow a Markov chain with transition probabilities $p_{u(x)}(x, y)$. For each policy u, we define a transition matrix P_u whose (x, y)th entry is $p_{u(x)}(x, y)$.

The problem of stochastic control amounts to selection of a policy that optimizes a given criterion. In this paper, we will employ as an optimality criterion infinite-horizon discounted cost of the form

$$J_u(x) = \mathrm{E}\left[\sum_{t=0}^{\infty} \alpha^t g_u(x_t) \Big| x_0 = x\right],$$

where $g_u(x)$ is used as shorthand for $g_{u(x)}(x)$ and the discount factor $\alpha \in (0, 1)$ reflects intertemporal preferences. It is well known that there exists a single policy u that minimizes $J_u(x)$ simultaneously for all x, and the goal is to identify that policy.

Let us define operators T_u and T by

$$T_u J = g_u + \alpha P_u J \quad \text{and} \quad TJ = \min_u \left(g_u + \alpha P_u J\right),$$

where the minimization is carried out component-wise. Dynamic programming involves solution

of Bellman's equation

$$J = TJ.$$

The unique solution J^* of this equation is the optimal cost-to-go function

$$J^* = \min_u J_u,$$

and optimal control actions can be generated based on this function, according to

$$u(x) = \operatorname*{argmin}_{a \in \mathcal{A}_x} \left(g_a(x) + \alpha \sum_{y \in \mathcal{S}} p_a(x,y) J^*(y) \right).$$

Dynamic programming offers a number of approaches to solving Bellman's equation. One of particular relevance to our paper makes use of linear programming, as we will now discuss. Consider the problem

$$\begin{aligned} \max \quad & c'J \\ \text{s.t.} \quad & TJ \geq J, \end{aligned} \tag{1}$$

where c is a vector with positive components, which we will refer to as *state-relevance weights*. It can be shown that any feasible J satisfies $J \leq J^*$. It follows that, for any set of positive weights c, J^* is the unique solution to (1).

Note that T is a nonlinear operator, and therefore the constrained optimization problem written above is not a linear program. However, it is easy to reformulate the constraints to transform the problem into a linear program. In particular, noting that each constraint

$$(TJ)(x) \geq J(x)$$

is equivalent to a set of constraints

$$g_a(x) + \alpha \sum_{y \in \mathcal{S}} p_a(x,y) J(y) \geq J(x) \qquad \forall a \in \mathcal{A}_x,$$

we can rewrite the problem as

$$\begin{aligned} \max \quad & c'J \\ \text{s.t.} \quad & g_a(x) + \alpha \sum_{y \in \mathcal{S}} p_a(x,y) J(y) \geq J(x), \quad \forall x \in S, a \in \mathcal{A}_x. \end{aligned}$$

We will refer to this problem as the *exact LP*.

As mentioned in the introduction, state spaces for practical problems are enormous due to the curse of dimensionality. Consequently, the linear program of interest involves prohibitively large numbers of variables and constraints. The approximation algorithm we study reduces dramatically the number of variables.

Let us now introduce the linear programming approach to approximate dynamic programming. Given pre-selected basis functions $\phi_1, \ldots, \phi_K$, define a matrix

$$\Phi = \begin{bmatrix} | & & | \\ \phi_1 & \vdots & \phi_K \\ | & & | \end{bmatrix}.$$

With an aim of computing a weight vector $\tilde{r} \in \Re^K$ such that $\Phi\tilde{r}$ is a close approximation to J^*, one might pose the following optimization problem

$$\begin{aligned} \max \quad & c'\Phi r \\ \text{s.t.} \quad & T\Phi r \geq \Phi r. \end{aligned} \tag{2}$$

Given a solution $\tilde{r}$, one might then hope to generate near-optimal decisions according to

$$u(x) = \operatorname*{argmin}_{a \in \mathcal{A}_x} \left(g_a(x) + \alpha \sum_{y \in \mathcal{S}} p_a(x,y)(\Phi\tilde{r})(y) \right).$$

We will call such a policy a *greedy* policy with respect to $\Phi\tilde{r}$. More generally, a greedy policy u with respect to a function J is one that satisfies

$$u(x) = \operatorname*{argmin}_{a \in \mathcal{A}_x} \left(g_a(x) + \alpha \sum_{y \in \mathcal{S}} p_a(x,y) J(y) \right).$$

As with the case of exact dynamic programming, the optimization problem (2) can be recast as a linear program

$$\begin{array}{ll} \max & c'\Phi r \\ \text{s.t.} & g_a(x) + \alpha \displaystyle\sum_{y \in \mathcal{S}} p_a(x,y)(\Phi r)(y) \geq (\Phi r)(x), \quad \forall x \in S, a \in \mathcal{A}_x. \end{array}$$

We will refer to this problem as the *approximate LP*. Note that, though the number of variables is reduced to K, the number of constraints remains as large as in the exact LP. Fortunately, most of the constraints become inactive, and solutions to the linear program can be approximated efficiently. In numerical studies presented in Section 6, for example, we sample and use only a relatively small subset of the constraints. We expect that subsampling in this way suffices for most practical problems, and have developed sample-complexity bounds that qualify this expectation [11]. There are also alternative approaches studied in the literature for alleviating the need to consider all constraints. Examples include heuristics presented in [31] and problem-specific approaches making use of constraint generation methods (e.g., [16, 26]) or structure allowing constraints to be represented compactly (e.g., [23, 17]).

In the next four sections, we assume that the approximate LP can be solved, and we study the quality of the solution as an approximation to the cost-to-go function.

3 The Importance of State-Relevance Weights

In the exact LP, for any vector c with positive components, maximizing $c'J$ yields J^*. In other words, the choice of state-relevance weights does not influence the solution. The same statement does not hold for the approximate LP. In fact, the choice of state-relevance weights may bear a significant impact on the quality of the resulting approximation, as suggested by theoretical results in this section and demonstrated by numerical examples later in the paper.

To motivate the role of state-relevance weights, let us start with a lemma that offers an interpretation of their function in the approximate LP. The proof can be found in the appendix.

Lemma 3.1. *A vector $\tilde{r}$ solves*

$$\begin{array}{ll} \max & c'\Phi r \\ \text{s.t.} & T\Phi r \geq \Phi r, \end{array}$$

if and only if it solves

$$\begin{array}{ll} \min & \|J^* - \Phi r\|_{1,c} \\ \text{s.t.} & T\Phi r \geq \Phi r. \end{array}$$

The preceding lemma points to an interpretation of the approximate LP as the minimization of a certain weighted norm, with weights equal to the state-relevance weights. This suggests that c imposes a tradeoff in the quality of the approximation across different states, and we can lead the algorithm to generate better approximations in a region of the state space by assigning relatively larger weight to that region.

Underlying the choice of state-relevance weights is the question of how to compare different approximations to the cost-to-go function. A possible measure of quality is the distance to the optimal cost-to-go function; intuitively, we expect that the better the approximate cost-to-go function captures the real long-run advantage of being in a given state, the better the policy it generates. A more direct measure is a comparison between the actual costs incurred by using the greedy policy associated with the approximate cost-to-go function and those incurred by an optimal policy. We now provide a bound on the cost increase incurred by using approximate cost-to-go functions generated by approximated linear programming.

We consider as a measure of the quality of policy u the expected increase in the infinite-horizon discounted cost, conditioned on the initial state of the system being distributed according to a probability distribution ν; i.e.,

$$\mathrm{E}_{X\sim\nu}\left[J_u(X) - J^*(X)\right] = \|J_u - J^*\|_{1,\nu}.$$

It will be useful to define a measure $\mu_{u,\nu}$ over the state space associated with each policy u and probability distribution ν, given by

$$\mu_{u,\nu}^T = (1-\alpha)\nu^T \sum_{t=0}^{\infty} \alpha^t P_u^t. \tag{3}$$

Note that, since $\sum_{t=0}^{\infty} \alpha^t P_u^t = (I - \alpha P_u)^{-1}$, we also have

$$\mu_{u,\nu}^T = (1-\alpha)\nu^T (I - \alpha P_u)^{-1}.$$

The measure $\mu_{u,\nu}$ captures the expected frequency of visits to each state when the system runs under policy u, conditioned on the initial state being distributed according to ν. Future visits are discounted according to the discount factor α.

Proofs for the following lemma and theorem can be found in the appendix.

Lemma 3.2. *$\mu_{u,\nu}$ is a probability distribution.*

We are now poised to prove the following bound on the expected cost increase associated with policies generated by approximate linear programming. Henceforth we will use the norm $\|\cdot\|_{1,\gamma}$, defined by

$$\|J\|_{1,\gamma} = \sum_{x\in\mathcal{S}} \gamma(x)|J(x)|.$$

Theorem 3.1. *Let $J : \mathcal{S} \mapsto \Re$ be such that $TJ \geq J$. Then*

$$\|J_{u_J} - J^*\|_{1,\nu} \leq \frac{1}{1-\alpha}\|J - J^*\|_{1,\mu_{u_J,\nu}}. \tag{4}$$

Theorem 3.1 offers some reassurance that, if the approximate cost-to-go function J is close to J^*, the performance of the policy generated by J should similarly be close to the performance of the optimal policy. Moreover, the bound (4) also establishes how approximation errors in different states in the system map to losses in performance, which is useful for comparing different approximations to the cost-to-go function.

Contrasting Lemma 3.1 with the bound on the increase in costs (4) given by Theorem 3.1, we may want to choose state-relevance weights c that capture the (discounted) frequency with which different states are expected to be visited. Note that the frequency with which different states

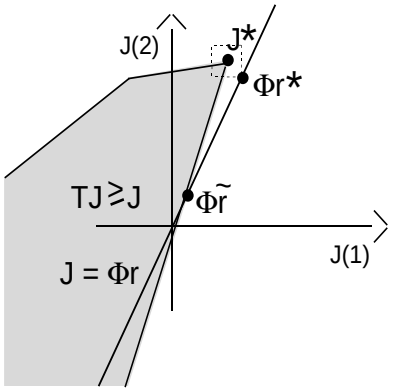

Figure 1: Graphical interpretation of approximate linear programming

are visited in general depends on the policy being used. One possibility is to have an iterative scheme, where the approximate LP is solved multiple times with state-relevance weights adjusted according to the intermediate policies being generated. Alternatively, a plausible conjecture is that some problems will exhibit structure making it relatively easy to make guesses about which states are desirable and therefore more likely to be visited often by reasonable policies, and which ones are typically avoided and rarely visited. We expect structures enabling this kind of procedure to be reasonably common in large-scale problems, in which desirable policies often exhibit some form of "stability," guiding the system to a limited region of the state space and allowing only infrequent excursions from this region. Selection of state-relevance weights in practical problems is illustrated in Sections 5 and 6.

4 Error Bounds for the Approximate LP

When the optimal cost-to-go function lies within the span of the basis functions, solution of the approximate LP yields the exact optimal cost-to-go function. Unfortunately, it is difficult in practice to select a set of basis functions that contains the optimal cost-to-go function within its span. Instead, basis functions must be based on heuristics and simplified analysis. One can only hope that the span comes close to the desired cost-to-go function.

For the approximate LP to be useful, it should deliver good approximations when the cost-to-go function is near the span of selected basis functions. Figure 1 illustrates the issue. Consider an MDP with states 1 and 2. The plane represented in the figure corresponds to the space of all functions over the state space. The shaded area is the feasible region of the exact LP, and J^* is the pointwise maximum over that region. In the approximate LP, we restrict attention to the subspace $J = \Phi r$.

In Figure 1, the span of the basis functions comes relatively close to the optimal cost-to-go function J^*; if we were able to perform, for instance, a maximum-norm projection of J^* onto the subspace $J = \Phi r$, we would obtain the reasonably good approximate cost-to-go function Φr^*. At the same time, the approximate LP yields the approximate cost-to-go function $\Phi\tilde{r}$. In this section, we develop bounds guaranteeing that $\Phi\tilde{r}$ is not too much farther from J^* than Φr^* is.

We begin in Section 4.1 with a simple bound capturing the fact that, if e is within the span of the basis functions, the error in the result of the approximate LP is proportional to the minimal error given the selected basis functions. Though this result is interesting in its own right, the bound is very loose — perhaps too much so to be useful in practical contexts. In Section 4.2,

however, we remedy this situation by providing a refined bound, which constitutes the main result of the paper. The bound motivates a modification to the original approximate linear programming formulation so that the basis functions span *Lyapunov functions*, defined later.

4.1 A Simple Bound

Let $\|\cdot\|_\infty$ denote the maximum norm, defined by $\|J\|_\infty = \max_{x\in\mathcal{S}} |J(x)|$, and e denote the vector with every component equal to 1. Our first bound is given by the following theorem.

Theorem 4.1. *Let e be in the span of the columns of Φ and c be a probability distribution. Then, if $\tilde{r}$ is an optimal solution to the approximate LP,*

$$\|J^* - \Phi\tilde{r}\|_{1,c} \leq \frac{2}{1-\alpha} \min_r \|J^* - \Phi r\|_\infty.$$

Proof: Let r^* be one of the vectors minimizing $\|J^* - \Phi r\|_\infty$ and define $\epsilon = \|J^* - \Phi r^*\|_\infty$. The first step is to find a feasible point $\bar{r}$ such that $\Phi\bar{r}$ is within distance $O(\epsilon)$ of J^*. Since

$$\|T\Phi r^* - J^*\|_\infty \leq \alpha\|\Phi r^* - J^*\|_\infty,$$

we have

$$T\Phi r^* \geq J^* - \alpha\epsilon e. \tag{5}$$

We also recall that for any vector J and any scalar k,

$$\begin{aligned} T(J-ke) &= \min_u \{g_u + \alpha P_u(J-ke)\} \\ &= \min_u \{g_u + \alpha P_u J - \alpha k e\} \\ &= \min_u \{g_u + \alpha P_u J\} - \alpha k e \\ &= TJ - \alpha k e. \end{aligned} \tag{6}$$

Combining (5) and (6), we have

$$\begin{aligned} T(\Phi r^* - ke) &= T\Phi r^* - \alpha k e \\ &\geq J^* - \alpha\epsilon e - \alpha k e \\ &\geq \Phi r^* - (1+\alpha)\epsilon e - \alpha k e \\ &= \Phi r^* - ke + [(1-\alpha)k - (1+\alpha)\epsilon]\, e. \end{aligned}$$

Since e is within the span of the columns of Φ, there exists a vector $\bar{r}$ such that

$$\Phi\bar{r} = \Phi r^* - \frac{(1+\alpha)\epsilon}{1-\alpha} e,$$

and $\bar{r}$ is a feasible solution to the approximate LP. By the triangular inequality,

$$\|\Phi\bar{r} - J^*\|_\infty \leq \|J^* - \Phi r^*\|_\infty + \|\Phi r^* - \Phi\bar{r}\|_\infty \leq \epsilon\left(1 + \frac{1+\alpha}{1-\alpha}\right) = \frac{2\epsilon}{1-\alpha}.$$

If $\tilde{r}$ is an optimal solution to the approximate LP, by Lemma 3.1, we have

$$\begin{aligned} \|J^* - \Phi\tilde{r}\|_{1,c} &\leq \|J^* - \Phi\bar{r}\|_{1,c} \\ &\leq \|J^* - \Phi\bar{r}\|_\infty \\ &\leq \frac{2\epsilon}{1-\alpha} \end{aligned}$$

where the second inequality holds because c is a probability distribution. The result follows. □

This bound establishes that when the optimal cost-to-go function lies close to the span of the basis functions, the approximate LP generates a good approximation. In particular, if the error $\min_r \|J^* - \Phi r\|_\infty$ goes to zero (e.g., as we make use of more and more basis functions) the error resulting from the approximate LP also goes to zero.

Though the above bound offers some support for the linear programming approach, there are some significant weaknesses:

1. The bound calls for an element of the span of the basis functions to exhibit uniformly low error over all states. In practice, however, $\min_r \|J^* - \Phi r\|_\infty$ is typically huge, especially for large-scale problems.
2. The bound does not take into account the choice of state-relevance weights. As demonstrated in the previous section, these weights can significantly impact the approximation error. A sharp bound should take them into account.

In Section 4.2, we will state and prove the main result of this paper, which provides an improved bound that aims to alleviate the shortcomings listed above.

4.2 An Improved Bound

To set the stage for development of an improved bound, let us establish some notation. First, we introduce a weighted maximum norm, defined by

$$\|J\|_{\infty,\gamma} = \max_{x \in \mathcal{S}} \gamma(x)|J(x)|, \tag{7}$$

for any $\gamma : \mathcal{S} \mapsto \Re^+$. As opposed to the maximum norm employed in Theorem 4.1, this norm allows for uneven weighting of errors across the state space.

We also introduce an operator H, defined by

$$(HV)(x) = \max_{a \in \mathcal{A}_x} \sum_y P_a(x,y)V(y),$$

for all $V : \mathcal{S} \mapsto \Re$. For any V, $(HV)(x)$ represents the maximum expected value of $V(y)$ if the current state is x and y is a random variable representing the next state. For each $V : \mathcal{S} \mapsto \Re$, we define a scalar β_V given by

$$\beta_V = \max_x \frac{\alpha(HV)(x)}{V(x)}. \tag{8}$$

We can now introduce the notion of a "Lyapunov function."

Definition 4.1 (Lyapunov function). *We call $V : \mathcal{S} \mapsto \Re^+$ a Lyapunov function if $\beta_V < 1$.*

Our definition of a Lyapunov function translates into the condition that there exist $V > 0$ and $\beta < 1$ such that

$$\alpha(HV)(x) \leq \beta V(x), \qquad \forall x \in \mathcal{S}. \tag{9}$$

If α were equal to 1, this would look like a Lyapunov stability condition: the maximum expected value $(HV)(x)$ at the next time step must be less than the current value $V(x)$. In general, α is less than 1, and this introduces some slack in the condition.

Our error bound for the approximate LP will grow proportionately with $1/(1-\beta_V)$, and we therefore want β_V to be small. Note that β_V becomes smaller as the $(HV)(x)$'s become small relative to the $V(x)$'s; β_V conveys a degree of "stability," with smaller values representing stronger stability. Therefore our bound suggests that, the more stable the system is, the easier it may be for the approximate LP to generate a good approximate cost-to-go function.

We now state our main result. For any given function V mapping $\mathcal{S}$ to positive reals, we use $1/V$ as shorthand for a function $x \mapsto 1/V(x)$.

Theorem 4.2. *Let $\tilde{r}$ be a solution of the approximate LP. Then, for any $v \in \Re^K$ such that $(\Phi v)(x) > 0$ for all $x \in \mathcal{S}$ and $\alpha H \Phi v < \Phi v$,*

$$\|J^* - \Phi\tilde{r}\|_{1,c} \leq \frac{2c'\Phi v}{1-\beta_{\Phi v}} \min_r \|J^* - \Phi r\|_{\infty,1/\Phi v}. \tag{10}$$

Proof: We will first present three preliminary lemmas leading to the main result. Omitted proofs can be found in the appendix.

The first lemma bounds the effects of applying T to two different vectors.

Lemma 4.1. *For any J and $\overline{J}$,*

$$|TJ - T\bar{J}| \leq \alpha \max_u P_u |J - \bar{J}|.$$

Based on the preceding lemma, we can place the following bound on constraint violations in the approximate LP.

Lemma 4.2. *For any vector V with positive components and any vector J,*

$$TJ \geq J - (\alpha HV + V)\,\|J^* - J\|_{\infty,1/V}. \tag{11}$$

The next lemma establishes that subtracting an appropriately scaled version of a Lyapunov function from any Φr leads us to the feasible region of the approximate LP.

Lemma 4.3. *Let v be a vector such that Φv is a Lyapunov function, r be an arbitrary vector, and*

$$\overline{r} = r - \|J^* - \Phi r\|_{\infty,1/\Phi v} \left(\frac{2}{1-\beta_{\Phi v}} - 1\right) v.$$

Then,

$$T\Phi\bar{r} \geq \Phi\bar{r}.$$

Given the preceding lemmas, we are poised to prove Theorem 4.2.

Proof of Theorem 4.2 From Lemma 4.3, we know that $\overline{r} = r^* - \|J^* - \Phi r^*\|_{\infty,1/\Phi v} \left(\frac{2}{1-\beta_{\Phi v}} - 1\right) v$ is a feasible solution for the approximate LP. From Lemma 3.1, we have

$$\begin{aligned}
\|J^* - \Phi\tilde{r}\|_{1,c} &\leq \|J^* - \Phi\bar{r}\|_{1,c} \\
&= \sum_x c(x)(\Phi v)(x) \frac{|J^*(x) - (\Phi\bar{r})(x)|}{(\Phi v)(x)} \\
&\leq \left(\sum_x c(x)(\Phi v)(x)\right) \max_x \frac{|J^*(x) - (\Phi\bar{r})(x)|}{(\Phi v)(x)} \\
&= c^T\Phi v \|J^* - \Phi\bar{r}\|_{\infty,1/\Phi v} \\
&\leq c^T\Phi v \left(\|J^* - \Phi r^*\|_{\infty,1/\Phi v} + \|\Phi\bar{r} - \Phi r^*\|_{\infty,1/\Phi v}\right) \\
&\leq c^T\Phi v \left(\|J^* - \Phi r^*\|_{\infty,1/\Phi v} + \|J^* - \Phi r^*\|_{\infty,1/\Phi v} \left(\frac{2}{1-\beta_{\Phi v}} - 1\right) \|\Phi v\|_{\infty,1/\Phi v}\right) \\
&\leq \frac{2}{1-\beta_{\Phi v}} c^T\Phi v \|J^* - \Phi r^*\|_{\infty,1/\Phi v},
\end{aligned}$$

and Theorem 4.2 follows. □

Let us now discuss how this new theorem addresses the shortcomings of Theorem 4.1 listed in the previous section. We treat in turn the two items from the aforementioned list.

1. The norm $\|\cdot\|_\infty$ appearing in Theorem 4.1 is undesirable largely because it does not scale well with problem size. In particular, for large problems, the cost-to-go function can take on huge values over some (possibly infrequently visited) regions of the state space, and so can approximation errors in such regions.

   Observe that the maximum norm of Theorem 4.1 has been replaced in Theorem 4.2 by $\|\cdot\|_{\infty,1/\Phi v}$. Hence, the error at each state is now weighted by the reciprocal of the Lyapunov function value. This should to some extent alleviate difficulties arising in large problems. In particular, the Lyapunov function should take on large values in undesirable regions of the state space - regions where J^* is large. Hence, division by the Lyapunov function acts as a normalizing procedure that scales down errors in such regions.

2. As opposed to the bound of Theorem 4.1, the state-relevance weights do appear in our new bound. In particular, there is a coefficient $c'\Phi v$ scaling the right-hand side. In general, if the state-relevance weights are chosen appropriately, we expect that this factor of $c'\Phi v$ will be reasonably small and independent of problem size. We defer to Section 5 further qualification of this statement and a discussion of approaches to choosing c in contexts posed by concrete examples.

5 On the Choice of Lyapunov Function

The Lyapunov function Φv plays a central role in the bound of Theorem 4.2. Its choice influences three terms on the right-hand side of the bound:

1. the error $\min_r \|J^* - \Phi r\|_{\infty,1/\Phi v}$;
2. the Lyapunov stability factor $k_{\Phi v}$;
3. the inner product $c'\Phi v$ with the state-relevance weights.

An appropriately chosen Lyapunov function should make all three of these terms relatively small. Furthermore, for the bound to be useful in practical contexts, these terms should not grow much with problem size.

In the following subsections, we present three examples involving choices of Lyapunov functions in queueing problems. The intention is to illustrate more concretely how Lyapunov functions might be chosen and that reasonable choices lead to practical error bounds that are independent of the number of states, as well as the number of state variables. The first example involves a single autonomous queue. A second generalizes this to a context with controls. A final example treats a network of queues. In each case, we study the three terms enumerated above and how they scale with the number of states and/or state variables.

5.1 An Autonomous Queue

Our first example involves a model of an autonomous (i.e., uncontrolled) queueing system. We consider a Markov process with states $0, 1, ..., N-1$, each representing a possible number of jobs in a queue. The system state x_t evolves according to

$$x_{t+1} = \begin{cases} \min(x_t + 1, N-1), & \text{with probability } p, \\ \max(x_t - 1, 0), & \text{otherwise,} \end{cases}$$

and it is easy to verify that the steady-state probabilities $\pi(0), \ldots, \pi(N-1)$ satisfy

$$\pi(x) = \pi(0) \left(\frac{p}{1-p}\right)^x.$$

If the state satisfies $0 < x < N-1$, a cost $g(x) = x^2$ is incurred. For the sake of simplicity, we assume that costs at the boundary states 0 and $N-1$ are chosen to ensure that the cost-to-go function takes the form

$$J^*(x) = \rho_2 x^2 + \rho_1 x + \rho_0,$$

for some scalars ρ_0, ρ_1, ρ_2 with $\rho_0 > 0$ and $\rho_2 > 0$[1]. We assume that $p < 1/2$ so that the system is "stable." Stability here is taken in a loose sense indicating that the steady-state probabilities are decreasing for all sufficiently large states.

Suppose that we wish to generate an approximation to the optimal cost-to-go function using the linear programming approach. Further suppose that we have chosen the state-relevance weights c to be the vector π of steady-state probabilities and the basis functions to be $\phi_1(x) = 1$ and $\phi_2(x) = x^2$.

How good can we expect the approximate cost-to-go function $\Phi\tilde{r}$ generated by approximate linear programming to be as we increase the number of states N? First note that

$$\begin{aligned}
\min_r \|J^* - \Phi r\|_{1,c} &\leq \|J^* - (\rho_0\phi_1 + \rho_2\phi_2)\|_{1,c} \\
&= \sum_{x=0}^{N-1} \pi(x)|\rho_1|x \\
&= |\rho_1| \sum_{x=0}^{N-1} \pi(0) \left(\frac{p}{1-p}\right)^x x \\
&\leq |\rho_1| \frac{\frac{p}{1-p}}{1 - \frac{p}{1-p}},
\end{aligned}$$

for all N. The last inequality follows from the fact that the summation in the third line corresponds to the expected value of a geometric random variable conditioned on its being less than N. Hence, $\min_r \|J^* - \Phi r\|_{1,c}$ is uniformly bounded over N. One would hope that $\|J^* - \Phi\tilde{r}\|_{1,c}$, with $\tilde{r}$ being an outcome of the approximate LP, would be similarly uniformly bounded over N. It is clear that Theorem 4.1 does not offer a uniform bound of this sort. In particular, the term $\min_r \|J^* - \Phi r\|_\infty$ on the right-hand-side grows proportionately with N and is unbounded as N increases. Fortunately, this situation is remedied by Theorem 4.2, which does provide a uniform bound. In particular, as we will show in the remainder of this section, for an appropriate Lyapunov function $V = \Phi v$, the values of $\min_r \|J^* - \Phi r\|_{\infty,1/V}$, $1/(1-\beta_V)$ and $c^T V$ are all uniformly bounded over N, and together these values offer a bound on $\|J^* - \Phi\tilde{r}\|_{1,c}$ that is uniform over N.

We will make use of a Lyapunov function

$$V(x) = x^2 + \frac{2}{1-\alpha},$$

which is clearly within the span of our basis functions ϕ_1 and ϕ_2. Given this choice, we have

$$\min_r \|J^* - \Phi r\|_{\infty,1/V} \leq \max_{x \geq 0} \frac{|\rho_2 x^2 + \rho_1 x + \rho_0 - \rho_2 x^2 - \rho_0|}{x^2 + 2/(1-\alpha)}$$

[1] It is easy to verify that such a choice of boundary conditions is possible. In particular, given the desired functional form for J^*, we can solve for ρ_0, ρ_1, and ρ_2, based on Bellman's equation for states $1, \ldots, N-2$:

$$J^*(x) = x^2 + \alpha(pJ^*(x+1) + (1-p)J^*(x-1)), \qquad \forall x = 1, \ldots, N-2.$$

Note that the solution is unique as long as $N > 5$. We can then set $g(0) \equiv J^*(0) - \alpha(pJ^*(1) + (1-p)J^*(0))$ and $g(N-1) \equiv J^*(N-1) - \alpha(pJ^*(N-1) + (1-p)J^*(N-2))$ so that Bellman's equation is also satisfied for states 0 and $N-1$.

$$
\begin{aligned}
&= \max_{x\geq 0} \frac{|\rho_1| x}{x^2 + 2/(1-\alpha)} \\
&\leq \frac{|\rho_1|}{2\sqrt{2/(1-\alpha)}}.
\end{aligned}
$$

Hence, $\min_r \|J^* - \Phi r\|_{\infty,1/V}$ is uniformly bounded over N.

We next show that $1/(1-\beta_V)$ is uniformly bounded over N. In order to do that, we find bounds on HV in terms of V. For $0 < x < N-1$, we have

$$
\begin{aligned}
\alpha(HV)(x) &= \alpha\left[p\left(x^2 + 2x + 1 + \frac{2}{1-\alpha}\right) + (1-p)\left(x^2 - 2x + 1 + \frac{2}{1-\alpha}\right)\right] \\
&= \alpha\left[x^2 + \frac{2}{1-\alpha} + 1 + 2x(2p-1)\right] \\
&\leq \alpha\left(x^2 + \frac{2}{1-\alpha} + 1\right) \\
&= V(x)\left(\alpha + \frac{\alpha}{V(x)}\right) \\
&\leq V(x)\left(\alpha + \frac{1}{V(0)}\right) \\
&= V(x)\frac{1+\alpha}{2}.
\end{aligned}
$$

For $x = 0$, we have

$$
\begin{aligned}
\alpha(HV)(0) &= \alpha\left[p\left(1 + \frac{2}{1-\alpha}\right) + (1-p)\frac{2}{1-\alpha}\right] \\
&= \alpha p + \alpha\frac{2}{1-\alpha} \\
&\leq V(0)\left(\alpha + \frac{1-\alpha}{2}\right) \\
&= V(0)\frac{1+\alpha}{2}.
\end{aligned}
$$

Finally, we clearly have

$$
\alpha(HV)(N-1) \leq \alpha V(N-1) \leq V(N-1)\frac{1+\alpha}{2},
$$

since the only possible transitions from state $N-1$ are to states $x \leq N-1$ and V is a nondecreasing function. Therefore, $\beta_V \leq (1+\alpha)/2$ and $1/(1-\beta_V)$ is uniformly bounded on N.

We now treat $c^T V$. Note that for $N \geq 1$,

$$
\begin{aligned}
c^T V &= \sum_{x=0}^{N-1} \pi(0)\left(\frac{p}{1-p}\right)^x \left(x^2 + \frac{2}{1-\alpha}\right) \\
&= \frac{1 - p/(1-p)}{1 - [p/(1-p)]^N} \sum_{x=0}^{N-1} \left(\frac{p}{1-p}\right)^x \left(x^2 + \frac{2}{1-\alpha}\right) \\
&\leq \frac{1 - p/(1-p)}{1 - p/(1-p)} \sum_{x=0}^{\infty} \left(\frac{p}{1-p}\right)^x \left(x^2 + \frac{2}{1-\alpha}\right) \\
&= \frac{1-p}{1-2p}\left(\frac{2}{1-\alpha} + 2\frac{p^2}{(1-2p)^2} + \frac{p}{1-2p}\right),
\end{aligned}
$$

so $c^T V$ is uniformly bounded for all N.

5.2 A Controlled Queue

In the previous example, we treated the case of an autonomous queue and showed how the terms involved in the error bound of Theorem 4.2 are uniformly bounded on the number of states N. We now address a more general case in which we can control the queue service rate. For any time t and state $0 < x_t < N-1$, the next state is given by

$$x_{t+1} = \begin{cases} x_t - 1, & \text{with probability } q(x_t), \\ x_t + 1, & \text{with probability } p, \\ x_t, & \text{otherwise.} \end{cases}$$

From state 0, a transition to state 1 or 0 occurs with probabilities p or $1-p$, respectively. From state $N-1$, a transition to state $N-2$ or $N-1$ occurs with probabilities $q(N-2)$ or $1-q(N-2)$, respectively. The arrival probability p is the same for all states and we assume that $p < 1/2$. The action to be chosen in each state x is the departure probability or service rate $q(x)$, which takes values in a finite set $\{q_i, i = 1, ..., A\}$. We assume that $q_A = 1 - p > p$, therefore the queue is "stabilizable". The cost incurred at state x if action q is taken is given by

$$g(x, q) = x^2 + m(q),$$

where m is a nonnegative and increasing function.

As discussed before, our objective is to show that the terms involved in the error bound of Theorem 4.2 are uniformly bounded over N. We start by finding a suitable Lyapunov function based on our knowledge of the problem structure. In the autonomous case, the choice of the Lyapunov function was motivated by the fact that the optimal cost-to-go function was a quadratic. We now proceed to show that in the controlled case, J^* can be bounded above by a quadratic

$$J^*(x) \leq \rho_2 x^2 + \rho_1 x + \rho_0$$

for some $\rho_0 > 0$, ρ_1 and $\rho_2 > 0$ that are constant independent of the queue buffer size $N-1$. Note that J^* is bounded above by the value of a policy $\bar{\mu}$ that takes action $q(x) = 1 - p$ for all x, hence it suffices to find a quadratic upper bound for the value of this policy. We will do so by making use of the fact that for any policy μ and any vector J, $T_\mu J \leq J$ implies $J \geq J_\mu$. Take

$$\begin{aligned} \rho_2 &= \frac{1}{1-\alpha}, \\ \rho_1 &= \frac{\alpha\left[2\rho_2(2p-1)\right]}{1-\alpha}, \\ \rho_0 &= \max\left(\frac{\alpha p(\rho_2+\rho_1)}{1-\alpha}, \frac{m(1-p)+\alpha\left[\rho_2+\rho_1(2p-1)\right]}{1-\alpha}\right). \end{aligned}$$

For any state x such that $0 < x < N-1$, we can verify that

$$\begin{aligned} J(x) - (T_{\bar{\mu}}J)(x) &= \rho_0(1-\alpha) - m(1-p) - \alpha\left[\rho_2 + \rho_1(2p-1)\right] \\ &\geq \frac{m(1-p)+\alpha\left[\rho_2+\rho_1(2p-1)\right]}{1-\alpha}(1-\alpha) - m(1-p) - \\ &\quad -\alpha\left[\rho_2+\rho_1(2p-1)\right] \\ &= 0. \end{aligned}$$

For state $x = N-1$, note that if $N > 1 - \rho_1/2\rho_2$ we have $J(N) > J(N-1)$ and

$$\begin{aligned} J(N-1) - (T_{\bar{\mu}}J)(N-1) &= J(N-1) - (N-1)^2 - m(1-p) - \\ &\quad -\alpha\left[(1-p)J(N-2) + pJ(N-1)\right] \end{aligned} \tag{12}$$

$$
\begin{aligned}
&\geq J(N-1) - (N-1)^2 - m(1-p) - \\
&\quad -\alpha\left[(1-p)J(N-2) + pJ(N)\right] \\
&= \rho_0(1-\alpha) - m(1-p) - \alpha\left[\rho_2 + \rho_1(2p-1)\right] \\
&\geq 0.
\end{aligned} \tag{13}
$$

Finally, for state $x = 0$ we have

$$
\begin{aligned}
J(0) - (T_{\bar{\mu}}J)(0) &= (1-\alpha)\rho_0 - \alpha p(\rho_2 + \rho_1) \\
&\geq (1-\alpha)\frac{\alpha p(\rho_2+\rho_1)}{1-\alpha} - \alpha p(\rho_2+\rho_1) \\
&= 0.
\end{aligned}
$$

It follows that $J \geq T_\mu J$, and for all $N > 1 - \rho_1/2\rho_2$,

$$0 \leq J^* \leq J_{\bar{\mu}} \leq J = \rho_2 x^2 + \rho_1 x + \rho_0.$$

A natural choice of Lyapunov function is, as in the previous example, $V(x) = x^2 + C$ for some $C > 0$. It follows that

$$
\begin{aligned}
\min_r \|J^* - \Phi r\|_{\infty,1/V} &\leq \|J^*\|_{\infty,1/V} \\
&\leq \max_{x\geq 0} \frac{\rho_2 x^2 + \rho_1 x + \rho_0}{x^2 + C} \\
&< \rho_2 + \frac{\rho_1}{2\sqrt{C}} + \frac{\rho_0}{C}.
\end{aligned}
$$

Now note that

$$
\begin{aligned}
\alpha(HV)(x) &\leq \alpha\left[p(x^2+2x+1+C) + (1-p)(x^2+C)\right] \\
&= V(x)\left(\alpha + \frac{\alpha p(2x+1)}{x^2+C}\right)
\end{aligned}
$$

and for C sufficiently large and independent of N, there is $\beta < 1$ also independent of N such that $\alpha HV \leq \beta V$ and $1/(1-\beta)$ is uniformly bounded on N.

It remains to be shown that $c^T V$ is uniformly bounded on N. For that, we need to specify the state-relevance vector c. As in the case of the autonomous queue, we might want it to be close to the steady-state distribution of the states under the optimal policy. Clearly, it is not easy to choose state-relevant weights in that way since we do not know the optimal policy. Alternatively, we will use the general shape of the steady-state distribution to generate sensible state-relevance weights.

Let us analyze the infinite buffer case and show that, under some stability assumptions, there should be a geometric upper bound for the tail of steady-state distribution; we expect that results for finite (large) buffers should be similar if the system is stable, since in this case most of the steady-state distribution will be concentrated on relatively small states. Let us assume that the system under the optimal policy is indeed stable – that should generally be the case if the discount factor is large. For a queue with infinite buffer the optimal service rate $q(x)$ is nondecreasing in x [1], and stability therefore implies that

$$q(x) \geq q(x_0) > p$$

for all $x \geq x_0$ and some sufficiently large x_0. It is easy then to verify that the tail of the steady-state distribution has an upper bound with geometric decay since it should satisfy

$$\pi(x)p = \pi(x+1)q(x+1),$$

and therefore

$$\frac{\pi(x+1)}{\pi(x)} \leq \frac{p}{q(x_0)} < 1,$$

for all $x \geq x_0$. Thus a reasonable choice of state-relevance weights is $c(x) = \pi(0)\xi^x$, where $\pi(0) = \frac{1-\xi}{1-\xi^N}$ is a normalizing constant making c a probability distribution. In this case,

$$\begin{aligned} c^T V &= \mathrm{E}\left[X^2 + C \mid X < N\right] \\ &\leq 2\frac{\xi^2}{(1-\xi)^2} + \frac{\xi}{1-\xi} + C, \end{aligned}$$

where X represents a geometric random variable with parameter $1-\xi$. We conclude that $c^T V$ is uniformly bounded on N.

5.3 A Queueing Network

Both previous examples involved one-dimensional state spaces and had terms of interest in the approximation error bound uniformly bounded over the number of states. We now consider a queueing network with d queues and finite buffers of size B to determine the impact of dimensionality on the terms involved in the error bound of Theorem 4.2.

We assume that the number of exogenous arrivals occuring in any time step has expected value less than or equal to Ad, for a finite A. The state $x \in \Re^d$ indicates the number of jobs in each queue. The cost per stage incurred at state x is given by

$$g(x) = \frac{|x|}{d} = \frac{1}{d}\sum_{i=1}^{d} x_i,$$

the average number of jobs per queue.

Let us first consider the optimal cost-to-go function J^* and its dependency on the number of state variables d. Our goal is to establish bounds on J^* that will offer some guidance on the choice of a Lyapunov function V that keeps the error $\min_r \|J^* - \Phi r\|_{\infty,1/V}$ small. Since $J^* \geq 0$, we will only derive upper bounds.

Instead of carrying the buffer size B throughout calculations, we will consider the infinite buffer case. The optimal cost-to-go function for the finite buffer case should be bounded above by that of the infinite buffer case, as having finite buffers corresponds to having jobs arriving at a full queue discarded at no additional cost.

We have

$$\mathrm{E}_x\left[|x_t|\right] \leq |x| + Adt,$$

since the expected total number of jobs at time t cannot exceed the total number of jobs at time 0 plus the expected number of arrivals between 0 and t, which is less than or equal to Adt. Therefore we have

$$\begin{aligned} \mathrm{E}_x\left[\sum_{t=0}^{\infty} \alpha^t |x_t|\right] &= \sum_{t=0}^{\infty} \alpha^t \mathrm{E}_x\left[|x_t|\right] \\ &\leq \sum_{t=0}^{\infty} \alpha^t (|x| + Adt) \\ &= \frac{|x|}{1-\alpha} + \frac{Ad}{(1-\alpha)^2}. \end{aligned} \tag{14}$$

The first equality holds because $|x_t| \geq 0$ for all t; by the monotone convergence theorem, we can interchange the expectation and the summation. We conclude from (14) that the optimal

cost-to-go function in the infinite buffer case should be bounded above by a linear function of the state; in particular,

$$0 \leq J^*(x) \leq \frac{\rho_1}{d}|x| + \rho_0,$$

for some positive scalars ρ_0 and ρ_1 independent of the number of queues d.

As discussed before, the optimal cost-to-go function in the infinite buffer case provides an upper bound for the optimal cost-to-go function in the case of finite buffers of size B. Therefore, the optimal cost-to-go function in the finite buffer case should be bounded above by the same linear function regardless of the buffer size B.

As in the previous examples, we will establish bounds on the terms involved in the error bound of Theorem 4.2. We consider a Lyapunov function $V(x) = \frac{1}{d}|x| + C$ for some constant $C > 0$, which implies

$$\begin{aligned}
\min_r \|J^* - \Phi r\|_{\infty,1/V} &\leq \|J^*\|_{\infty,1/V} \\
&\leq \max_{x \geq 0} \frac{\rho_1|x| + d\rho_0}{|x| + dC} \\
&\leq \rho_1 + \frac{\rho_0}{C},
\end{aligned}$$

and the bound above is independent of the number of queues in the system.

Now let us study β_V. We have

$$\begin{aligned}
\alpha(HV)(x) &\leq \alpha\left(\frac{|x| + Ad}{d} + C\right) \\
&\leq V(x)\left(\alpha + \frac{\alpha A}{\frac{|x|}{d} + C}\right) \\
&\leq V(x)\left(\alpha + \frac{\alpha A}{C}\right),
\end{aligned}$$

and it is clear that, for C sufficiently large and independent of d, there is a $\beta < 1$ independent of d such that $\alpha HV \leq \beta V$, and therefore $\frac{1}{1-\beta_V}$ is uniformly bounded on d.

Finally, let us consider $c^T V$. We expect that under some stability assumptions, the tail of the steady-state distribution will have an upper bound with geometric decay [3] and we take $c(x) = \left(\frac{1-\xi}{1-\xi^{B+1}}\right)^d \xi^{|x|}$. The state-relevance weights c are equivalent to the conditional joint distribution of d independent and identically distributed geometric random variables conditioned on the event that they are all less than $B+1$. Therefore,

$$\begin{aligned}
c^T V &= E\left[\frac{1}{d}\sum_{i=1}^{d} X_i + C \;\middle|\; X_i < B+1, i = 1, ..., d\right] \\
&< E[X_1] + C \\
&= \frac{\xi}{1-\xi} + C,
\end{aligned}$$

where $X_i, i = 1, ..., d$ are identically distributed geometric random variables with parameter $1 - \xi$. It follows that $c^T V$ is uniformly bounded over the number of queues.

6 Application to Controlled Queueing Networks

In this section, we discuss numerical experiments involving application of the linear programming approach to controlled queueing problems. Such problems are relevant to several industries

including manufacturing and telecommunications and the experimental results presented here suggest approximate linear programming as a promising approach to solving them.

In all examples, we assume that at most one event (arrival/departure) occurs at each time step. We also choose basis functions that are polynomial in the states. This is partly motivated by the analysis in the previous section and partly motivated by the fact that, with linear(quadratic) costs, our problems have cost-to-go functions that are asymptotically linear(quadratic) functions of the state. Hence our approach is to exploit the problem structure to select basis functions. It may not be straightforward to identify properties of the cost-to-go function in other applications; in Section 7, we briefly discuss an alternative approach.

The first example illustrates how state-relevance weights influence the solution of the approximate LP.

6.1 Single Queue with Controlled Service Rate

In Section 5.2, we studied a queue with a controlled service rate and determined that the bounds on the error of the approximate LP were uniform over the number of states. That example provided some guidance on the choice of basis functions; in particular, we now know that including a quadratic and a constant function guarantees that an appropriate Lyapunov function is in the span of the columns of Φ. Furthermore, our analysis of the (unknown) steady-state distribution revealed that state-relevance weights of the form $c(x) = (1-\xi)\xi^x$ are a sensible choice. However, how to choose an appropriate value of ξ was not discussed there. In this section, we present results of experiments with different values of ξ for a particular instance of the model described in Section 5.2. The values of ξ chosen for experimentation are motivated by ideas developed in Section 3.

We assume that jobs arrive at a queue with probability $p = 0.2$ in any unit of time. Service rates/probabilities $q(x)$ are chosen from the set $\{0.2, 0.4, 0.6, 0.8\}$. The cost incurred at any time for being in state x and taking action q is given by

$$g(x, q) = x + 60q^3.$$

We take the buffer size to be 49999 and the discount factor to be $\alpha = 0.98$. We select basis functions $\phi_1(x) = 1$, $\phi_2(x) = x$, $\phi_3(x) = x^2$, $\phi_4(x) = x^3$ and state-relevance weights $c(x) = (1-\xi)\xi^x$. The approximate LP is solved for $\xi = 0.9$ and $\xi = 0.999$ and we denote the solution of the approximate LP by r^ξ. The numerical results are presented in Figures 2, 3, 4 and 5.

Figure 2 shows the approximations Φr^ξ to the cost-to-go function generated by the approximate LP. Note that the results agree with the analysis developed in Section 3; small states are approximated better when $\xi = 0.9$ whereas large states are approximated almost exactly when $\xi = 0.999$.

In Figure 3 we see the greedy action with respect to Φr^ξ. We get the optimal action for almost all "small" states with $\xi = 0.9$. On the other hand, $\xi = 0.999$ yields optimal actions for all relatively large states in the relevant range.

The most important result is illustrated in Figure 4, which depicts the cost-to-go functions associated with the greedy policies. Note that despite taking suboptimal actions for all relatively large states, the policy induced by $\xi = 0.9$ performs better than that generated with $\xi = 0.999$ in the range of relevant states, and it is close in value to the optimal policy even in those states for which it does not take the optimal action. Indeed, the average cost incurred by the greedy policy with respect to $\xi = 0.9$ is 2.92, relatively close to the average cost incurred by the optimal (discounted cost) policy, which is 2.72. The average cost incurred when $\xi = 0.999$ is 4.82, which is significantly higher.

Steady-state probabilities for each of the different greedy policies, as well as the corresponding (rescaled) state-relevance weights are shown in Figure 5. Note that setting ξ to 0.9 captures the

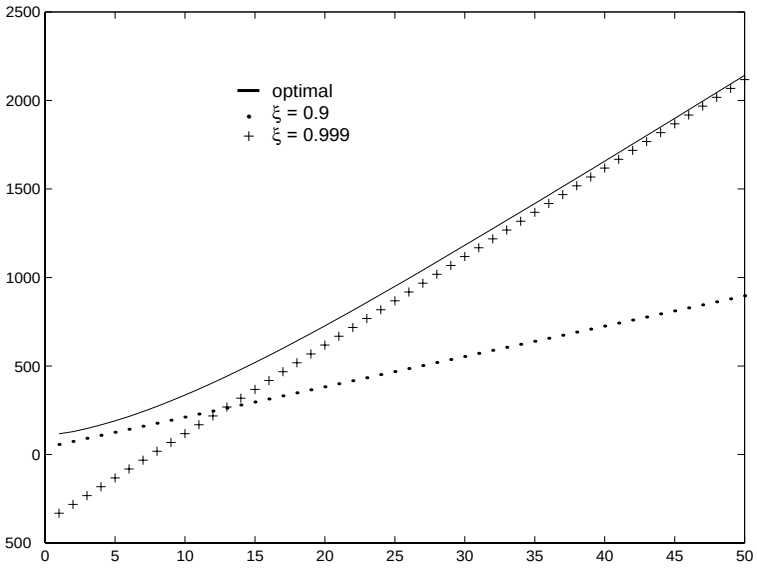

Figure 2: Approximate cost-to-go function for the example in Section 6.1.

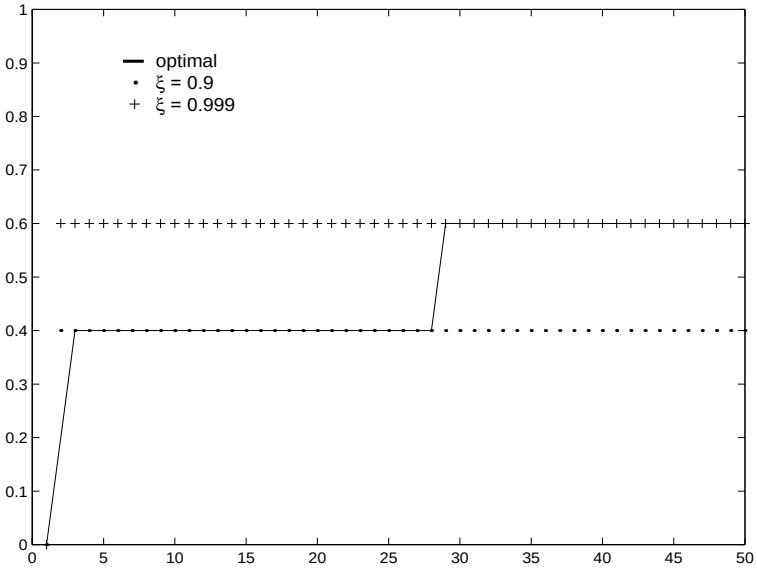

Figure 3: Greedy action for the example in Section 6.1.

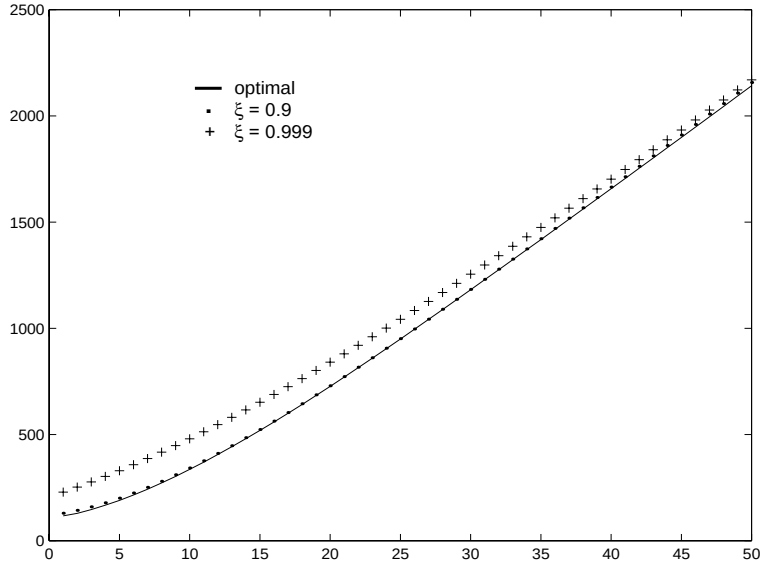

Figure 4: Cost-to-go function for the example in Section 6.1.

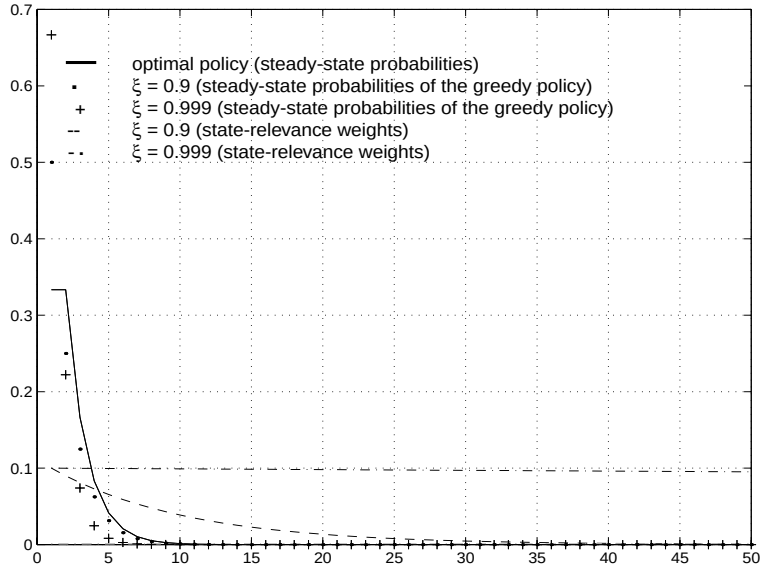

Figure 5: Steady-state probabilities for the example in Section 6.1.

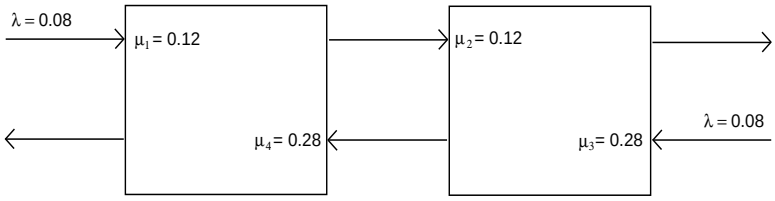

Figure 6: System for the example in Section 6.2.

Policy	Average cost
$\xi = 0.95$	33.37
LONGEST	45.04
FIFO	45.71
LBFS	144.1

Table 1: Performance of different policies for Example 6.2. Average cost estimated by simulation after 50000000 iterations, starting with empty system.

relative frequencies of states, whereas setting ξ to 0.999 weights all states in the relevant range almost equally.

6.2 A Four-Dimensional Queueing Network

In this section we study the performance of the approximate LP algorithm when applied to a queueing network with two servers and four queues. The system is depicted in Figure 6 and has been previously studied in [6, 20, 25]. Arrival (λ) and departure ($\mu_i, i = 1, ..., 4$) probabilities are indicated. We assume a discount factor $\alpha = 0.99$. The state $x \in \Re^4$ indicates the number of jobs in each queue and the cost incurred in any period is $g(x) = |x|$, the total number of jobs in the system. Actions $a \in \{0, 1\}^4$ satisfy $a_1 + a_4 \leq 1$, $a_2 + a_3 \leq 1$ and the non-iddling assumption, i.e., a server must be working if any of its queues is nonempty. We have $a_i = 1$ iff queue i is being served.

Constraints for the approximate LP are generated by sampling 40000 states according to the distribution given by the state-relevance weights c. We choose the basis functions to span all of the polynomials in x of degree 3; therefore, there are

$$\binom{4}{0} + \binom{4}{1} + \left[\binom{4}{1} + \binom{4}{2}\right] + \left[\binom{4}{1} + 2\binom{4}{2} + \binom{4}{3}\right] = 35$$

basis functions. The terms in the above expression denote the number of basis functions of degree 0, 1, 2, and 3, respectively.

We choose the state-relevance weights to be $c(x) = (1-\xi)^4 \xi^{|x|}$. Experiments were performed for a range of values of ξ. The best results were generated when $0.95 \leq \xi \leq 0.99$. The average cost was estimated by simulation with 50,000,000 iterations, starting with an empty system.

We compare the average cost obtained by the greedy policy with respect to the solution of the approximate LP with that of several different heuristics, namely, first-in-first-out (FIFO), last-buffer-first-served (LBFS), and a policy that always serves the longest queue (LONGEST). Results are summarized in Table 1 and we can see that the approximate LP yields significantly better performance than all of the other heuristics.

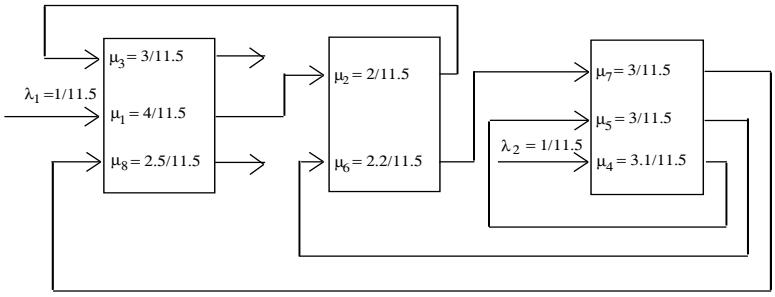

Figure 7: System for the example in Section 6.3.

6.3 An Eight-Dimensional Queueing Network

In our last example, we consider a queueing network with eight queues. The system is depicted in Figure 7, with arival $(\lambda_i, i = 1, 2)$ and departure $(\mu_i, i = 1, ..., 8)$ probabilities indicated.

The state $x \in \Re^8$ represents the number of jobs in each queue. The cost-per-state is $g(x) = |x|$, and the discount factor α is 0.995. Actions $a \in \{0, 1\}^8$ indicate which queues are being served; $a_i = 1$ iff a job from queue i is being processed. We consider only non-iddling policies and, at each time step, a server processes jobs from one of its queues exclusively.

We choose state-relevance weights of the form $c(x) = (1 - \xi)^8 \xi^{|x|}$. The basis functions are chosen to span all polynomials in x of degree at most 2; therefore, the approximate LP has 47 variables. Due to the relatively large number of actions per state (up to 18), we choose to sample a relatively small number of states. Note that we take a slightly different approach from that proposed in [11] and include constraints relative to all actions associated with each state in the system. Constraints for the approximate LP are generated by sampling 5000 states according to the distribution associated with the state-relevance weights c. Experiments were performed for $\xi = 0.85, 0.9$ and 0.95, and $\xi = 0.9$ yielded the policy with smallest average cost. We do not specify a maximum buffer size. The maximum number of jobs in the system for states sampled in the LP was 235, and the maximum single queue length, 93. During simulation of the policy obtained, the maximum number of jobs in the system was 649, and the maximum number of jobs in any single queue, 384.

To evaluate the performance of the policy generated by the approximate LP, we compare it with first-in-first-out (FIFO), last-buffer-first-serve (LBFS) and a policy that serves the longest queue in each server (LONGEST). LBFS serves the job that is closest to leaving the system; for example, if there are jobs in queue 2 and in queue 6, a job from queue 2 is processed since it will leave the system after going through only one more queue, whereas the job from queue 6 will still have to go through two more queues. We also choose to assign higher priority to queue 8 than to queue 3 since queue 8 has higher departure probability.

We estimated the average cost of each policy with 50,000,000 simulation steps, starting with an empty system. Results appear in Table 2. The policy generated by the approximate LP performs significantly better than each of the heuristics, yielding more than 10% improvement over LBFS, the second best policy. We expect that even better results could be obtained by refining the choice of basis functions and state-relevance weights.

The constraint generation step took 74.9 seconds and the resulting LP was solved in approximately 3.5 minutes of CPU time with CPLEX 7.0 running on a Sun Ultra Enterprise 5500 machine with Solaris 7 operating system and a 400 MHz processor.

Policy	Average cost
ALP	136.67
LBFS	153.82
FIFO	163.63
LONGEST	168.66

Table 2: Average number of jobs in the system for the example in Section 6.3, after 50,000,000 simulation steps.

7 Closing Remarks and Open Issues

In this paper we studied the linear programming approach to approximate dynamic programming for stochastic control problems as a means of alleviating the curse of dimensionality. We provided an error bound for a variant of approximate linear programming based on certain assumptions on the basis functions. The bounds were shown to be uniform in the number of states and state variables in certain queueing problems. Our analysis also led to some guidelines in the choice of the so-called "state-relevance weights" for the approximate LP.

An alternative to the approximate LP are temporal-difference learning (TD) methods [2, 9, 10, 28, 29, 34, 35, 36]. In such methods, one tries to find a fixed point for an "approximate dynamic programming operator" by simulating the system and learning from the observed costs and state transitions. Experimentation is necessary to determine when TD can offer better results than the approximate LP. However, it is worth mentioning that due to its complexity, much of TD's behavior is still to be understood; there are no convergence proofs or effective error bounds for general stochastic control problems. Such poor understanding leads to implementation difficulties; a fair amount of trial and error is necessary in order to get the method to perform well or even to converge. The approximate LP, on the other hand, benefits from the inherent simplicity of linear programming: its analysis is simpler, and error bounds such as those provided here provide guidelines on how to set the algorithm's parameters most efficiently. Packages for large-scale linear programming developed in the recent past also make the approximate LP relatively easy to implement.

A central question in approximate linear programming not addressed here is the choice of basis functions. In the applications to queueing networks, we have chosen basis functions polynomial in the states. This was largely motivated by the fact that, with linear/quadratic costs, it can be shown in these problems that the optimal cost-to-go function is asymptotically linear/quadratic. Reasonably accurate knowledge of structure the cost-to-go function may be difficult in other problems. An alternative approach is to extract a number of *features* of the states which are believed to be relevant to the decision being made. The hope is that the mapping from features to the cost-to-go function might be smooth, in which case certain sets of basis functions such as polynomials might lead to good approximations.

We have motivated many of the ideas and guidelines for choice of parameters through examples in queueing problems. In future work, we intend to explore how these ideas would be interpreted in other contexts, such as portfolio management and inventory control.

Several other questions remain open and are the object of future investigation: Can the state-relevance weights in the objective function be chosen in some adaptive way? Can we add robustness to the approximate LP algorithm to account for errors in the estimation of costs and transition probabilities, i.e., design an alternative LP with meaningful performance bounds when problem parameters are just known to be in a certain range? How do our results extend to the average cost case? How do our results extend to the infinite-state case?

Finally, in this paper we utilize linear architectures to represent approximate cost-to-go func-

tions. It may be interesting to explore algorithms using nonlinear representations. Alternative representations encountered in the literature include neural networks [4, 18] and splines [7, 32], among others.

Acknowledgements

The authors would like to thank John Tsitsiklis and Sean Meyn for valuable comments. This research was supported by NSF CAREER Grant ECS-9985229, by the ONR under Grant MURI N00014-00-1-0637, and by an IBM Research Fellowship.

References

[1] D. Bertsekas. *Dynamic Programming and Optimal Control.* Athena Scientific, 1995.

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

[3] D. Bertsimas, D. Gamarnik, and J.N. Tsitsiklis. Performance of multiclass Markovian queueing networks via piecewise linear Lyapunov functions. *Annals of Applied Probability*, 11(4):1384–1428, 2001.

[4] C.M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, 1995.

[5] V. Borkar. A convex analytic approach to Markov decision processes. *Probability Theory and Related Fields*, 78:583–602, 1988.

[6] R-R. Chen and S. Meyn. Value iteration and optimization of multiclass queueing networks. *Queueing Systems*, 32:65–97, 1999.

[7] V.C.P. Chen, D. Ruppert, and C.A. Shoemaker. Applying experimental design and regression splines to high-dimensional continuous-state stochast dynamic programming. *Operations Research*, 47(1):38–53, 1999.

[8] R.H. Crites and A.G. Barto. Improving elevator performance using reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 8, 1996.

[9] P. Dayan. The convergence of TD(λ) for general λ. *Machine Learning*, 8:341–362, 1992.

[10] D.P. de Farias and B. Van Roy. On the existence of fixed points for appproximate value iteration and temporal-difference learning. *Journal of Optimization Theory and Applications*, 105(3), 2000.

[11] D.P. de Farias and B. Van Roy. On constraint sampling in the linear programming approach to approximate dynamic programming. Conditionally accepted to *Mathematics of Operations Research*, 2001.

[12] G. de Ghellinck. Les problèmes de décisions séquentielles. *Cahiers du Centre d'Etudes de Recherche Opérationnelle*, 2:161–179, 1960.

[13] E.V. Denardo. On linear programming in a Markov decision problem. *Management Science*, 16(5):282–288, 1970.

[14] F. D'Epenoux. A probabilistic production and inventory problem. *Management Science*, 10(1):98–108, 1963.

[15] G. Gordon. *Approximate Solutions to Markov Decision Processess.* PhD thesis, Carneggie Mellon University, 1999.

[16] M. Grőtschel and O. Holland. Solution of large-scale symmetric travelling salesman problems. *Mathematical Programming*, 51:141–202, 1991.

[17] C. Guestrin, D. Koller, and R. Parr. Efficient solution algorithms for factored MDPs. Submitted to Journal of Artificial Intelligence Research, 2001.

[18] S. Haykin. *Neural Networks: A Comprehensive Formulation*. McMillan, 1994.

[19] A. Hordijk and L.C.M. Kallenberg. Linear programming and Markov decision chains. *Management Science*, 25:352–362, 1979.

[20] P.R. Kumar and T.I. Seidman. Dynamic instabilities and stabilization methods in distributed real-time scheduling of manufacturing systems. *IEEE Transactions on Automatic Control*, 35(3):289–298, 1990.

[21] F. Longstaff and E.S. Schwartz. Valuing American options by simulation: A simple least squares approach. *The Review of Financial Studies*, 14:113–147, 2001.

[22] A.S. Manne. Linear programming and sequential decisions. *Management Science*, 6(3):259–267, 1960.

[23] J.R. Morrison and P.R. Kumar. New linear program performance bounds for queueing networks. *Journal of Optimization Theory and Applications*, 100(3):575–597, 1999.

[24] I.C. Paschalidis and J.N Tsitsiklis. Congestion-dependent pricing of network services. *IEEE/ACM Transactions on Networking*, 8(2):171–184, 2000.

[25] A.N. Rybko and A.L. Stolyar. On the ergodicity of stochastic processes describing the operation of open queueing networks. *Problemy Peredachi Informatsii*, 28:3–26, 1992.

[26] D. Schuurmans and R. Patrascu. Direct value-approximation for factored MDPs. In *Advances in Neural Information Processing Systems*, volume 14, 2001.

[27] P. Schweitzer and A. Seidmann. Generalized polynomial approximations in Markovian decision processes. *Journal of Mathematical Analysis and Applications*, 110:568–582, 1985.

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

[29] R.S. Sutton and A.G. Barto. *Learning to Predict by the Methods of Temporal Differences*. MIT Press, 1998.

[30] G.J. Tesauro. Temporal difference learning and TD-gammon. *Communications of the ACM*, 38:58–68, 1995.

[31] M. Trick and S. Zin. A linear programming approach to solving dynamic programs. Unpublished manuscript, 1993.

[32] M. Trick and S. Zin. Spline approximations to value functions: A linear programming approach. *Macroeconomic Dynamics*, 1, 1997.

[33] J.N. Tsitsiklis and B. Van Roy. Regression methods for pricing complex American-style options. *IEEE Transactions on Neural Networks*, 12(4):694–703, 2001.

[34] 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.

[35] B. Van Roy. *Learning and Value Function Approximation in Complex Decision Processes*. PhD thesis, Massachusetts Institute of Technology, 1998.

[36] B. Van Roy. Neuro-dynamic programming: Overview and recent trends. In E. Feinberg and A. Schwartz, editors, *Markov Decision Processes: Models, Methods, Directions, and Open Problems*. Kluwer, 2000.

[37] W. Zhang and T.G. Dietterich. High-performance job-shop scheduling with a time-delay TD(λ) network. In *Advances in Neural Information Processing Systems*, volume 8, 1996.

A Proofs

Lemma 3.1: A vector $\tilde{r}$ solves

$$\begin{array}{ll} \max & c'\Phi r \\ \text{s.t.} & T\Phi r \geq \Phi r, \end{array}$$

if and only if it solves

$$\begin{array}{ll} \min & \|J^* - \Phi r\|_{1,c} \\ \text{s.t.} & T\Phi r \geq \Phi r. \end{array}$$

Proof: It is well known that the dynamic programming operator T is monotonic. From this and the fact that T is a contraction with fixed point J^*, it follows that, for any J with $J \leq TJ$, we have

$$J \leq TJ \leq T^2 J \leq ... \leq J^*.$$

Hence, any r that is a feasible solution to the optimization problems of interest satisfies $\Phi r \leq J^*$. It follows that

$$\|J^* - \Phi r\|_{1,c} = \sum_{x \in \mathcal{S}} c(x)|J^*(x) - (\Phi r)(x)| = c'J^* - c'\Phi r,$$

and maximizing $c'\Phi r$ is therefore equivalent to minimizing $\|J^* - \Phi r\|_{1,c}$. □

Lemma 3.2: $\mu_{u,\nu}$ is a probability distribution.
Proof: Let e be the vector of all ones. Then we have

$$\begin{aligned} \sum_{x \in \mathcal{S}} \mu_{u,\nu}(x) &= (1-\alpha)\nu^T \sum_{t=0}^{\infty} \alpha^t P_u^t e \\ &= (1-\alpha)\nu^T \sum_{t=0}^{\infty} \alpha^t e \\ &= (1-\alpha)\nu^T (1-\alpha)^{-1} e \\ &= 1, \end{aligned}$$

and the claim follows. □

Theorem 3.1: Let $J : \mathcal{S} \mapsto \Re$ be such that $TJ \geq J$. Then

$$\|J_{u_J} - J^*\|_{1,\nu} \leq \frac{1}{1-\alpha} \|J - J^*\|_{1,\mu_{u_J,\nu}}.$$

Proof: We have

$$\begin{aligned} J_{u_J} - J &= (I - \alpha P_{u_J})^{-1} g_{u_J} - J \\ &= (I - \alpha P_{u_J})^{-1} \left[g_{u_J} - (I - \alpha P_{u_J}) J \right] \\ &= (I - \alpha P_{u_J})^{-1} (g_{u_J} + \alpha P_{u_J} J - J) \\ &= (I - \alpha P_{u_J})^{-1} (TJ - J). \end{aligned}$$

Since $J \leq TJ$, we have $J \leq TJ \leq J^* \leq J_{u_J}$. Hence

$$\|J_{u_J} - J^*\|_{1,\nu} = \nu^T (J_{u_J} - J^*)$$

$$
\begin{aligned}
&\leq \nu^T(J_{u_J} - J) \\
&= \nu^T(I - \alpha P_{u_J})^{-1}(TJ - J) \\
&= \frac{1}{1-\alpha}\mu^T_{u_J,\nu}(TJ - J) \\
&\leq \frac{1}{1-\alpha}\mu^T_{u_J,\nu}(J^* - J) \\
&= \frac{1}{1-\alpha}\|J^* - J\|_{1,\mu^T_{u_J,\nu}},
\end{aligned}
$$

and the claim follows. □

Lemma 4.1: For any J and $\overline{J}$,

$$|TJ - T\bar{J}| \leq \alpha \max_u P_u |J - \bar{J}|.$$

Proof: Note that, for any J and $\bar{J}$,

$$
\begin{aligned}
TJ - T\bar{J} &= \min_u \{g_u + \alpha P_u J\} - \min_u \left\{g_u + \alpha P_u \bar{J}\right\} \\
&= g_{u_J} + \alpha P_{u_J} J - g_{u_{\bar{J}}} - \alpha P_{u_{\bar{J}}} \bar{J} \\
&\leq g_{u_{\bar{J}}} + \alpha P_{u_{\bar{J}}} J - g_{u_{\bar{J}}} - \alpha P_{u_{\bar{J}}} \bar{J} \\
&\leq \alpha \max_u P_u (J - \bar{J}) \\
&\leq \alpha \max_u P_u |J - \bar{J}|,
\end{aligned}
$$

where u_J and $u_{\bar{J}}$ denote greedy policies with respect to J and $\bar{J}$, respectively. An entirely analogous argument gives us

$$T\bar{J} - TJ \leq \alpha \max_u P_u |J - \bar{J}|,$$

and the result follows. □

Lemma 4.2: For any vector V with positive components and any vector J,

$$TJ \geq J - (\alpha HV + V)\,\|J^* - J\|_{\infty,1/V}.$$

Proof: Note that

$$|J^*(x) - J(x)| \leq \|J^* - J\|_{\infty,1/V} V(x).$$

By Lemma 4.1,

$$
\begin{aligned}
|(TJ^*)(x) - (TJ)(x)| &\leq \alpha \max_a \sum_{y \in \mathcal{S}} P_a(x,y)|J^*(y) - J(y)| \\
&\leq \alpha \|J^* - J\|_{\infty,1/V} \max_{a \in \mathcal{A}_x} \sum_{y \in \mathcal{S}} P_a(x,y) V(y) \\
&= \alpha \|J^* - J\|_{\infty,1/V} (HV)(x).
\end{aligned}
$$

Letting $\epsilon = \|J^* - J\|_{\infty,1/V}$, it follows that

$$
\begin{aligned}
(TJ)(x) &\geq J^*(x) - \alpha\epsilon(HV)(x) \\
&\geq J(x) - \epsilon V(x) - \alpha\epsilon(HV)(x).
\end{aligned}
$$

The result follows. □

Lemma 4.3: Let v be a vector such that Φv is a Lyapunov function, r be an arbitrary vector, and

$$\overline{r} = r - \|J^* - \Phi r\|_{\infty,1/\Phi v} \left(\frac{2}{1-\beta_{\Phi v}} - 1\right) v.$$

Then,

$$T\Phi\bar{r} \geq \Phi\bar{r}.$$

Proof: Let $\epsilon = \|J^* - \Phi r\|_{\infty,1/\Phi v}$. By Lemma 4.1,

$$\begin{aligned}
|(T\Phi r)(x) - (T\Phi\overline{r})(x)| &= \left|(T\Phi r)(x) - \left(T\left[(\Phi r - \epsilon\left(\frac{2}{1-\beta_{\Phi v}} - 1\right)\Phi v\right]\right)(x)\right| \\
&\leq \alpha \max_a \sum_{y\in\mathcal{S}} P_a(x,y)\epsilon\left(\frac{2}{1-\beta_{\Phi v}} - 1\right)(\Phi v)(y) \\
&= \alpha\epsilon\left(\frac{2}{1-\beta_{\Phi v}} - 1\right)(H\Phi v)(x),
\end{aligned}$$

since Φv is a Lyapunov function and therefore $2/(1-\beta_{\Phi v}) - 1 > 0$. It follows that

$$T\Phi\overline{r} \geq T\Phi r - \alpha\epsilon\left(\frac{2}{1-\beta_{\Phi v}} - 1\right) H\Phi v.$$

By Lemma 4.2,

$$T\Phi r \geq \Phi r - \epsilon\left(\alpha H\Phi v + \Phi v\right),$$

and therefore

$$\begin{aligned}
T\Phi\overline{r} &\geq \Phi r - \epsilon\left(\alpha H\Phi v + \Phi v\right) - \alpha\epsilon\left(\frac{2}{1-\beta_{\Phi v}} - 1\right) H\Phi v \\
&= \Phi\overline{r} - \epsilon\left(\alpha H\Phi v + \Phi v\right) + \epsilon\left(\frac{2}{1-\beta_{\Phi v}} - 1\right)(\Phi v - \alpha H\Phi v) \\
&\geq \Phi\overline{r} - \epsilon\left(\alpha H\Phi v + \Phi v\right) + \epsilon(\Phi v + \alpha H\Phi v) \\
&= \Phi\overline{r},
\end{aligned}$$

where the last inequality follows from the fact that $\Phi v - \alpha H\Phi v > 0$ and

$$\begin{aligned}
\frac{2}{1-\beta_{\Phi v}} - 1 &= \frac{2}{1 - \max_x \frac{\alpha(H\Phi v)(x)}{(\Phi v)(x)}} - 1 \\
&= \max_x \frac{2(\Phi v)(x)}{(\Phi v)(x) - \alpha(H\Phi v)(x)} - 1 \\
&= \max_x \frac{(\Phi v)(x) + \alpha(H\Phi v)(x)}{(\Phi v)(x) - \alpha(H\Phi v)(x)}.
\end{aligned}$$

□