Dual Representations for Dynamic Programming and Reinforcement Learning

Tao Wang Michael Bowling Dale Schuurmans
Department of Computing Science
University of Alberta
Edmonton, Canada
Email: {trysi,bowling,dale}@cs.ualberta.ca

Abstract— **We investigate the dual approach to dynamic programming and reinforcement learning, based on maintaining an explicit representation of stationary distributions as opposed to value functions. A significant advantage of the dual approach is that it allows one to exploit well developed techniques for representing, approximating and estimating probability distributions, without running the risks associated with divergent value function estimation. A second advantage is that some distinct algorithms for the average reward and discounted reward case in the primal become unified under the dual. In this paper, we present a modified dual of the standard linear program that guarantees a globally normalized state visit distribution is obtained. With this reformulation, we then derive novel dual forms of dynamic programming, including policy evaluation, policy iteration and value iteration. Moreover, we derive dual formulations of temporal difference learning to obtain new forms of Sarsa and Q-learning. Finally, we scale these techniques up to large domains by introducing approximation, and develop new approximate off-policy learning algorithms that avoid the divergence problems associated with the primal approach. We show that the dual view yields a viable alternative to standard value function based techniques and opens new avenues for solving dynamic programming and reinforcement learning problems.**

I. Introduction

Algorithms for dynamic programming (DP) and reinforcement learning (RL) are usually formulated in terms of *value functions*—representations of the long run expected value of a state or state-action pair [1]. The concept of value is so pervasive in DP and RL, in fact, that it is hard to imagine that a value function representation is not a necessary component of any solution approach. Yet, linear programming (LP) methods clearly demonstrate that the value function is *not* a necessary concept for solving DP/RL problems. In LP methods, value functions only correspond to the primal formulation of the problem, and do not appear at all in the dual. Rather, in the dual, value functions are replaced by the notion of state (or state-action) *visit distributions* [2], [3], [4]. It is entirely possible to solve DP and RL problems in the dual representation, which offers an equivalent but different approach to solving DP/RL problems without any reference to value functions.

Despite the well known LP duality, dual representations have not been widely explored in DP and RL. In fact, they have only been anecdotally and partially treated in the RL literature [5], [6], and not in a manner that acknowledges any connection to LP duality. Nevertheless, as we will show, there exists a dual form for every standard DP and RL algorithm, including policy evaluation, policy iteration, Bellman iteration, temporal difference (TD) estimation, Sarsa learning, and Q-learning, and for variants of these algorithms that use linear approximation.

In this paper, we offer a systematic investigation of dual solution techniques based on representing state visit and state-action visit distributions instead of value functions. Although many of our results show that the dual approach yields equivalent results to the primal approach—as one would expect—we also uncover some potential advantages for the dual approach, including the ability to use well developed methods for estimating probability distributions, automatically avoiding the risk of divergence associated with linear approximators, and unifying some distinct algorithms for the average reward and discounted reward cases. The dual view offers a coherent and comprehensive perspective on optimal sequential decision making problems, just as the primal view, but offers new algorithmic insight and new opportunities for developing algorithms that exploit alternative forms of prior knowledge and constraints. In fact, there is the opportunity to develop a joint primal-dual view of RL and DP, where combined algorithms might be able to exploit the benefits of both approaches in theoretically justified ways.

II. Preliminaries

We are concerned with the problem of optimal sequential decision making, and in particular, the problem of computing an optimal behavior strategy in a *Markov decision process* (MDP). An MDP is defined by:

- a set of actions A;
- a set of states S;
- a transition model, which we will represent by an $|S||A| \times |S|$ *matrix* P, whose entries $P_{(sa,s')}$ specify the conditional probability of transitioning to state s' starting from state s and taking action a (hence P is nonnegative and *row* normalized);
- a reward model, which we will represent by an $|S||A| \times 1$ *vector* $\mathbf{r}$, whose entries $\mathbf{r}_{(sa)}$ specify the reward obtained when taking action a in state s.

A *behavior strategy* is a rule for selecting actions based on observed states. In an MDP there is an obvious tradeoff between obtaining immediate rewards and guiding the process toward future states that yield potentially greater rewards.

Generally, the goal is to determine a behavior strategy that maximizes the rewards obtained over the long run. However, there are different ways to define long run reward in an MDP and these can affect the identity of the optimal behavior strategy. In this paper we will focus on the two standard criteria: (1) maximizing the infinite horizon *discounted* reward $r_0 + \gamma r_1 + \gamma^2 r_2 + \cdots = \sum_{t=0}^{\infty} \gamma^t r_t$ obtained over an infinite run of the system, given a discount factor $0 < \gamma < 1$; or (2) maximizing the infinite horizon asymptotic rate of return per time step, $\lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} r_t$. Most attention will be paid to discounted rewards in this paper.

In either case, it is known that an optimal behavior strategy can always be expressed by a stationary *policy*. In this paper, we will represent a stationary policy by an $|S||A|\times 1$ *vector* $\boldsymbol{\pi}$, whose entries $\boldsymbol{\pi}_{(sa)}$ specify the probability of taking action a in state s; that is $\sum_a \boldsymbol{\pi}_{(sa)} = 1$ for all s. Stationarity refers to the fact that the action selection probabilities do not change over time. In addition to stationarity, it is known that furthermore there always exists a *deterministic* policy that gives the optimal action in each state (i.e., simply a policy with probabilities of 0 or 1) [3].

The main problem is to compute an optimal policy given either (1) a complete specification of the environmental variables P and $\mathbf{r}$ (the "*planning problem*"), or (2) limited access to the environment through observed states and rewards and the ability to select actions to cause further state transitions (the "*learning problem*"). The first version of the problem is normally tackled by LP or DP methods, and the latter by RL methods (although RL techniques can also be applied when the environment is known).

III. Linear Programming

To establish the dual form of representation, we begin by briefly reviewing the LP approach for solving MDPs in the discounted reward case. Here we assume we are given the environmental variables P and $\mathbf{r}$, the discount factor γ, and the initial distribution over states, expressed by an $|S| \times 1$ vector $\boldsymbol{\mu}$.

A standard LP for solving the planning problem can be expressed as

$$\begin{aligned} &\min_{\mathbf{v}} (1-\gamma)\boldsymbol{\mu}^\top \mathbf{v} \quad \text{subject to} \\ &\mathbf{v}_{(s)} \geq \mathbf{r}_{(sa)} + \gamma P_{(sa,:)}\mathbf{v} \qquad \forall s, a \end{aligned} \tag{1}$$

It is known that the optimal solution $\mathbf{v}^*$ to this LP corresponds to the *value function* for the optimal policy [3], [4]. In particular, given $\mathbf{v}^*$, the optimal policy can be recovered by

$$\begin{aligned} a^*(s) &= \arg\max_a \mathbf{r}_{(sa)} + \gamma P_{(sa,:)}\mathbf{v}^* \\ \boldsymbol{\pi}^*_{(sa)} &= \begin{cases} 1 & \text{if } a = a^*(s) \\ 0 & \text{if } a \neq a^*(s) \end{cases} \end{aligned} \tag{2}$$

Note that $\boldsymbol{\mu}$ and $(1-\gamma)$ behave as an arbitrary positive vector and positive constant in the LP above and do not affect the solution, provided $\boldsymbol{\mu} > 0$ and $\gamma < 1$ [7]. However, both play an important and non-arbitrary role in the dual LP below (as we will see) and we have chosen the objective in (1) in a specific way to obtain the result below.

To derive the particular form of the dual LP we will exploit below, first introduce a $|S||A|\times 1$ vector of Lagrange multipliers $\mathbf{d}$, and then form the Lagrangian of (1)

$$L(\mathbf{v}, \mathbf{d}) = (1-\gamma)\boldsymbol{\mu}^\top \mathbf{v} + \mathbf{d}^\top(\mathbf{r} + \gamma P\mathbf{v} - \Xi^\top \mathbf{v}), \quad \mathbf{d} \geq 0$$

Here, Ξ is the $|S| \times |S||A|$ marginalization matrix. That is, Ξ is constructed to simply ensure that the constraint $\Xi^\top \mathbf{v} \geq \mathbf{r} + \gamma P\mathbf{v}$ corresponds to the system of inequalities given in the primal LP (1). In particular, Ξ is a sparse matrix built by placing $|S|$ row blocks of length $|A|$ in a block diagonal fashion, where each row block consists of all 1s.

Next, taking the gradient of the Lagrangian with respect to $\mathbf{v}$ and setting the resulting vector to equal zero yields

$$\Xi\mathbf{d} = (1-\gamma)\boldsymbol{\mu} + \gamma P^\top \mathbf{d}$$

Substituting this constraint back into the Lagrangian eliminates the $\mathbf{v}$ variable and results in the dual LP

$$\begin{aligned} &\max_{\mathbf{d}} \mathbf{d}^\top \mathbf{r} \quad \text{subject to} \\ &\mathbf{d} \geq 0, \;\; \Xi\mathbf{d} = (1-\gamma)\boldsymbol{\mu} + \gamma P^\top \mathbf{d} \end{aligned} \tag{3}$$

Interestingly, the following lemma establishes that any feasible vector in (3) is guaranteed to be normalized, and therefore the solution $\mathbf{d}^*$ is always a joint *probability distribution* over state-action pairs.

Lemma 1: If $\mathbf{d}$ satisfies the constraint in (3) then $\mathbf{1}^\top \mathbf{d} = 1$.

Proof: First note that $\mathbf{1}^\top \mathbf{d} = \mathbf{1}^\top \Xi \mathbf{d}$, where the first $\mathbf{1}$ is $|S||A|\times 1$ and the second is $|S|\times 1$. Then one can determine that $\mathbf{1}^\top \Xi\mathbf{d} = (1-\gamma)\mathbf{1}^\top \boldsymbol{\mu} + \gamma \mathbf{1}^\top P^\top \mathbf{d} = (1-\gamma)1 + \gamma 1 = 1$. ■

By strong duality, we know that the optimal objective value of this dual LP equals the optimal objective value of the primal LP. Furthermore, given a solution to the dual $\mathbf{d}^*$, the optimal policy can be directly recovered by the much simpler transformation [8]

$$\boldsymbol{\pi}^*_{(sa)} = \frac{\mathbf{d}^*_{(sa)}}{\sum_a \mathbf{d}^*_{(sa)}} \tag{4}$$

A careful examination of (3) shows that the joint distribution $\mathbf{d}^*$ does *not* actually correspond to the stationary state-action visit distribution induced by $\boldsymbol{\pi}^*$ (unless $\gamma = 1$), but it does correspond to a distribution of discounted state-action visits beginning in the initial state distribution $\boldsymbol{\mu}$.

What this dual LP formulation establishes is that the optimal policy $\boldsymbol{\pi}^*$ for an MDP can be recovered without any direct reference whatsoever to the *value* function. Instead, one can work in the dual, and bypass value functions entirely, while working instead with *normalized* probability distributions over state-action pairs. Although this observation seems limited to the LP approach to solving the MDP planning problem, in fact, we find that explicit representations of probability distributions over state and state-action pairs can be used as a dual alternative to classical DP methods, classical RL methods, and even classical approximation methods.

IV. Dynamic Programming

Dynamic programming methods for solving MDP evaluation and planning problems are typically expressed in terms of the primal value function. Here we demonstrate that all of these classical algorithms have natural duals expressed in terms of state and state-action probability distributions.

A. Policy Evaluation

First consider the problem of policy evaluation. Here we assume we are given a fixed policy $\boldsymbol{\pi}$, and wish to compute either its value function or its distribution of discounted state visits. Below we will find it convenient to re-express a policy $\boldsymbol{\pi}$ by an equivalent representation as an $|S| \times |S||A|$ matrix Π where

$$\Pi_{(s,s'a)} = \begin{cases} \boldsymbol{\pi}_{(sa)} & \text{if } s' = s \\ 0 & \text{if } s' \neq s \end{cases}$$

That is, Π is a sparse matrix built by placing $|S|$ row blocks of length $|A|$ in a block diagonal fashion, where each row block gives the conditional distribution over actions specified by $\boldsymbol{\pi}$ in a particular state s. Although this representation of Π might appear unnatural, it is in fact extremely convenient: from this definition, one can quickly verify that the $|S| \times |S|$ matrix product ΠP gives the *state to state* transition probabilities induced by the policy $\boldsymbol{\pi}$ in the environment P, and the $|S||A| \times |S||A|$ matrix product $P\Pi$ gives the *state-action to state-action* transition probabilities induced by policy $\boldsymbol{\pi}$ in P. We will make repeated use of these two matrix products below.

In the primal view, the role of policy evaluation is to recover the *value function*, which is defined to be the expected sum of future discounted rewards

$$\mathbf{v} = \sum_{i=0}^{\infty} \gamma^i (\Pi P)^i \Pi \mathbf{r} \tag{5}$$

As is well known and easy to verify, this infinite series satisfies a recursive relationship that allows one to recover $\mathbf{v}$ by solving a linear system of $|S|$ equations on $|S|$ unknowns.

Lemma 2: $\mathbf{v} = \Pi(\mathbf{r} + \gamma P \mathbf{v})$, hence $(I - \gamma \Pi P)\mathbf{v} = \Pi \mathbf{r}$

Proof:
$$\begin{aligned} \mathbf{v} &= \Pi \mathbf{r} + \gamma \Pi P \mathbf{v} \\ &= \Pi \mathbf{r} + \gamma \Pi P (\Pi \mathbf{r} + \gamma \Pi P \mathbf{v}) \\ &= \Pi \mathbf{r} + \gamma \Pi P \Pi \mathbf{r} + \gamma^2 (\Pi P)^2 \mathbf{v} \\ &\vdots \\ &= \textstyle\sum_{i=0}^{\infty} \gamma^i (\Pi P)^i \Pi \mathbf{r} \end{aligned}$$

■

In the dual form of policy evaluation, one needs to recover a probability distribution over states that has a meaningful correspondence to the long run discounted reward achieved by the policy. Such a correspondence can be achieved by recovering the following probability distribution over states implicitly defined by a linear system of $|S|$ equations on $|S|$ unknowns.

$$\mathbf{c}^\top = (1-\gamma)\boldsymbol{\mu}^\top + \gamma \mathbf{c}^\top \Pi P \tag{6}$$

It can be easily verified that this defines a probability distribution.

Lemma 3: If $\mathbf{c}$ satisfies (6) then $\mathbf{c}^\top \mathbf{1} = 1$

Proof: Unrolling the recursion as in Lemma 2 yields

$$\mathbf{c}^\top = (1-\gamma)\boldsymbol{\mu}^\top \sum_{i=0}^{\infty} \gamma^i (\Pi P)^i \tag{7}$$

The result then follows from noting that $(\Pi P)^i \mathbf{1} = \mathbf{1}$ since ΠP is row normalized. ■

Not only is $\mathbf{c}$ a proper probability distribution over states, however, it also allows one to easily compute the expected discounted return of the policy $\boldsymbol{\pi}$.

Lemma 4: $(1-\gamma)\boldsymbol{\mu}^\top \mathbf{v} = \mathbf{c}^\top \Pi \mathbf{r}$

Proof: Immediate by plugging in the series expressions for $\mathbf{v}$ and $\mathbf{c}$ given in (5) and (7) respectively. ■

Thus, a dual form of policy evaluation can be conducted by recovering $\mathbf{c}$ from (6). The expected discounted reward obtained by policy $\boldsymbol{\pi}$ starting in the initial state distribution $\boldsymbol{\mu}$ can then be computed by $\mathbf{c}^\top \Pi \mathbf{r}/(1-\gamma)$ (Lemma 4). In principle, this gives a valid form of policy evaluation in a dual representation. However, below we will find that merely recovering the state distribution $\mathbf{c}$ is inadequate for *policy improvement*, since there is no apparent way to improve $\boldsymbol{\pi}$ given access to $\mathbf{c}$. Thus, we are compelled to extend the dual representation to a richer representation that avoids an implicit dependence on the initial distribution $\boldsymbol{\mu}$.

Consider the following definition for an $|S| \times |S|$ matrix

$$M = (1-\gamma)I + \gamma M \Pi P \tag{8}$$

The matrix M that satisfies this linear relation is similar to $\mathbf{c}^\top$, in that each row is a probability distribution (Lemma 5 below) and the entries $M_{(s,s')}$ correspond to the probability of discounted state visits to s' for a policy $\boldsymbol{\pi}$ starting in state s. Unlike $\mathbf{c}^\top$ however, M drops the dependence on $\boldsymbol{\mu}$ and obtains a close relationship with $\mathbf{v}$ (Theorem 1 below).

Lemma 5: $M\mathbf{1} = \mathbf{1}$ and $\mathbf{c}^\top = \boldsymbol{\mu}^\top M$

Proof: Unrolling the recursion in (8) yields

$$M = (1-\gamma)\sum_{i=0}^{\infty} \gamma^i (\Pi P)^i \tag{9}$$

The first result then follows from the fact that $(\Pi P)^i \mathbf{1} = \mathbf{1}$. The second result is immediate from (6) and (8). ■

Interestingly, Lemma 5 shows that M is a variant of Dayan's "successor representation" proposed in [5], but here extended to the infinite horizon discounted case. Moreover, not only is M a matrix of probability distributions over states, it allows one to easily recover the state values of the policy $\boldsymbol{\pi}$.

Theorem 1: $(1-\gamma)\mathbf{v} = M \Pi \mathbf{r}$

Proof: The result follows easily from (5) and (9).

$$\begin{aligned} (1-\gamma)\mathbf{v} &= (1-\gamma)\sum_{i=0}^{\infty} \gamma^i (\Pi P)^i \Pi \mathbf{r} \\ &= \left[(1-\gamma)\sum_{i=0}^{\infty} \gamma^i (\Pi P)^i \right] \Pi \mathbf{r} \\ &= M \Pi \mathbf{r} \end{aligned}$$

■

As with $\mathbf{c}$ above, a dual form of policy evaluation can be conducted by recovering M from (8). Then at any time, an equivalent representation to $\mathbf{v}$ can be recovered by $M\Pi\mathbf{r}/(1-\gamma)$, as shown in Theorem 1.

B. State-Action Policy Evaluation

Although state based policy evaluation methods like those outlined above are adequate for assessing a given policy, and eventually for formulating DP algorithms, when we consider RL algorithms below we will generally need to maintain joint *state-action* based evaluations.

In the primal representation, the policy state-action value function can be specified by an $|S||A| \times 1$ vector

$$\mathbf{q} \;=\; \sum_{i=0}^{\infty} \gamma^i (P\Pi)^i \mathbf{r} \tag{10}$$

This state-action value function satisfies a similar recursive relation and is closely related to the previous state value function.

Lemma 6: $\mathbf{q} = \mathbf{r} + \gamma P\Pi\mathbf{q}$

Proof: Unroll the recursion, as in Lemma 2. ■

Lemma 7: $\mathbf{v} = \Pi\mathbf{q}$

Proof:

$$\begin{aligned} \Pi\mathbf{q} \;&=\; \Pi \sum_{i=0}^{\infty} \gamma^i (P\Pi)^i \mathbf{r} \\ &=\; \sum_{i=0}^{\infty} \gamma^i (\Pi P)^i \Pi\mathbf{r} \;=\; \mathbf{v} \end{aligned}$$

■

To develop a *dual* form of state-action policy evaluation, we first introduce a probability distribution over state-action pairs that has a useful correspondence to the long run expected discounted rewards achieved by the policy. Consider the linear system of $|S||A|$ constraints on $|S||A|$ unknowns

$$\mathbf{d}^\top \;=\; (1-\gamma)\boldsymbol{\mu}^\top\Pi + \gamma \mathbf{d}^\top P\Pi \tag{11}$$

It can be verified that this defines a probability distribution.

Lemma 8: If $\mathbf{d}$ satisfies (11) then $\mathbf{d}^\top\mathbf{1} = 1$

Proof: Unrolling the recursion as in Lemma 2 yields

$$\mathbf{d}^\top \;=\; (1-\gamma)\boldsymbol{\mu}^\top\Pi \sum_{i=0}^{\infty} \gamma^i (P\Pi)^i \tag{12}$$

The result then follows from noting that $(P\Pi)^i\mathbf{1} = \mathbf{1}$ since $P\Pi$ is row normalized, and also $\Pi\mathbf{1} = \mathbf{1}$. ■

Not only is $\mathbf{d}$ a proper probability distribution over state-action pairs, it also allows one to easily compute the expected discounted return of the policy $\boldsymbol{\pi}$.

Lemma 9: $(1-\gamma)\boldsymbol{\mu}^\top\Pi\mathbf{q} = \mathbf{d}^\top\mathbf{r}$

Proof: Immediate by plugging in the series expressions for $\mathbf{q}$ and $\mathbf{d}$ given in (10) and (12) respectively. ■

Thus, a dual form of state-action policy evaluation can be conducted by recovering $\mathbf{d}$ from (11) and computing the expected discounted reward obtained by policy $\boldsymbol{\pi}$ starting in the initial state distribution $\boldsymbol{\mu}$ by $\mathbf{d}^\top\mathbf{r}/(1-\gamma)$ (Lemma 9). However, once again we will find that merely recovering the state-action distribution $\mathbf{d}$ is inadequate for *policy improvement*, since there is no apparent way to improve $\boldsymbol{\pi}$ given access to $\mathbf{d}$. Thus, again, we have to extend the dual representation to a richer representation that avoids an implicit dependence on the initial distribution $\boldsymbol{\mu}$.

Consider the following definition for an $|S||A| \times |S||A|$ matrix

$$H \;=\; (1-\gamma)I + \gamma HP\Pi \tag{13}$$

The matrix H that satisfies this linear relation is similar to $\mathbf{d}^\top$, in that each row is a probability distribution (Lemma 10 below) and the entries $H_{(sa,s'a')}$ correspond to the probability of discounted state-action visits to $(s'a')$ for a policy $\boldsymbol{\pi}$ starting in state-action pair (sa). Unlike $\mathbf{d}^\top$ however, H drops the dependence on $\boldsymbol{\mu}$ and obtains a close relationship with $\mathbf{q}$ (Theorem 2 below).

Lemma 10: $H\mathbf{1} = \mathbf{1}$ and $\mathbf{d}^\top \;=\; \boldsymbol{\mu}^\top\Pi H$

Proof: Unrolling the recursion in (13) yields

$$H \;=\; (1-\gamma)\sum_{i=0}^{\infty} \gamma^i (P\Pi)^i \tag{14}$$

The first result then follows from the fact that $(P\Pi)^i\mathbf{1} = \mathbf{1}$. The second result is immediate from (11) and (13). ■

Not only is H a matrix of probability distributions over state-action pairs, it allows one to easily recover the state-action values of the policy $\boldsymbol{\pi}$.

Theorem 2: $(1-\gamma)\mathbf{q} = H\mathbf{r}$

Proof: The result follows easily from (10) and (14).

$$\begin{aligned} (1-\gamma)\mathbf{q} \;&=\; (1-\gamma)\sum_{i=0}^{\infty} \gamma^i (P\Pi)^i \mathbf{r} \\ &=\; \left[(1-\gamma)\sum_{i=0}^{\infty} \gamma^i (P\Pi)^i\right]\mathbf{r} \;=\; H\mathbf{r} \end{aligned}$$

■

As with $\mathbf{d}$ above, a dual form of state-action policy evaluation can be conducted by recovering H from (13). Then at any time, an equivalent representation to $\mathbf{q}$ can be recovered by $H\mathbf{r}/(1-\gamma)$, as shown in Theorem 2.

Finally, one can relate the state and state-action matrices defined above to each other.

Lemma 11: $M\Pi = \Pi H$

Proof:

$$\begin{aligned} M\Pi \;&=\; (1-\gamma)\sum_{i=0}^{\infty} \gamma^i (\Pi P)^i \Pi \\ &=\; (1-\gamma)\sum_{i=0}^{\infty} \gamma^i \Pi (\Pi P)^i \\ &=\; \Pi(1-\gamma)\sum_{i=0}^{\infty} \gamma^i (\Pi P)^i \;=\; \Pi H \end{aligned}$$

■

Thus, to this point, we have developed new dual representations that can form the basis for state based and state-action based policy evaluation, respectively. These are defined in terms of state distributions and state-action distributions, and do not require value functions to be computed.

C. Policy Improvement

The next step is to consider mechanisms for policy improvement, which combined with policy evaluation form policy iteration algorithms capable of solving MDP planning problems.

The standard primal policy improvement update is well known. Given a current policy $\boldsymbol{\pi}$, whose state value function $\mathbf{v}$ or state-action value function $\mathbf{q}$ have already been determined, one can derive an improved policy $\boldsymbol{\pi}'$ via the update

$$\begin{aligned} a^*(s) &= \arg\max_a \mathbf{q}_{(sa)} \\ &= \arg\max_a \mathbf{r}_{(sa)} + \gamma P_{(sa,:)}\mathbf{v} \qquad (15) \end{aligned}$$

$$\boldsymbol{\pi}'_{(sa)} = \begin{cases} 1 & \text{if } a = a^*(s) \\ 0 & \text{if } a \neq a^*(s) \end{cases} \qquad (16)$$

The subsequent "policy improvement theorem" verifies that this update leads to an improved policy.

Theorem 3: $\Pi\mathbf{q} \leq \Pi'\mathbf{q}$ implies $\mathbf{v} \leq \mathbf{v}'$

Proof:

$$\begin{aligned} \mathbf{v} &= \Pi(\mathbf{r} + \gamma P\mathbf{v}) \\ &\leq \Pi'(\mathbf{r} + \gamma P\mathbf{v}) \\ &= \Pi'\mathbf{r} + \gamma\Pi' P\mathbf{v} \\ &= \Pi'\mathbf{r} + \gamma\Pi' P\Pi(\mathbf{r} + \gamma P\mathbf{v}) \\ &\leq \Pi'\mathbf{r} + \gamma\Pi' P\Pi'(\mathbf{r} + \gamma P\mathbf{v}) \\ &= \Pi'\mathbf{r} + \gamma\Pi' P\Pi'\mathbf{r} + \gamma^2(\Pi' P)^2\mathbf{v} \\ &\vdots \\ &= \sum_{i=0}^{\infty} \gamma^i (\Pi' P)^i \Pi'\mathbf{r} \;=\; \mathbf{v}' \end{aligned}$$

■

This development can be parallelled in the dual by first defining and analogous policy update and proving an analogous policy improvement theorem. Given a current policy $\boldsymbol{\pi}$, the dual form of the policy update can be expressed in terms of the state-action matrix H for $\boldsymbol{\pi}$

$$\begin{aligned} a^*(s) &= \arg\max_a H_{(sa,:)}\mathbf{r} \\ &= \arg\max_a (1-\gamma)\mathbf{r}_{(sa)} + \gamma P_{(sa,:)} M\Pi\mathbf{r} \qquad (17) \end{aligned}$$

$$\boldsymbol{\pi}'_{(sa)} = \begin{cases} 1 & \text{if } a = a^*(s) \\ 0 & \text{if } a \neq a^*(s) \end{cases} \qquad (18)$$

In fact, by Theorem 2, the two policy updates given in (15) and (17) respectively, must lead to the same resulting policy $\boldsymbol{\pi}'$. Therefore, not surprisingly, we have an analogous policy improvement theorem in this case.

Theorem 4: $\Pi H\mathbf{r} \leq \Pi' H\mathbf{r}$ implies $M\Pi\mathbf{r} \leq M'\Pi'\mathbf{r}$

Proof:

$$\begin{aligned} & M\Pi\mathbf{r} \qquad (19) \\ &= \Pi H\mathbf{r} \\ &\leq \Pi' H\mathbf{r} \\ &= \Pi'\left[(1-\gamma)I + \gamma PM\Pi\right]\mathbf{r} \\ &= (1-\gamma)\Pi'\mathbf{r} + \gamma\Pi' PM\Pi\mathbf{r} \\ &= (1-\gamma)\Pi'\mathbf{r} + \gamma\Pi' P\Pi H\mathbf{r} \\ &\leq (1-\gamma)\Pi'\mathbf{r} + \gamma\Pi' P\Pi' H\mathbf{r} \\ &= (1-\gamma)\Pi'\mathbf{r} + \gamma\Pi' P\Pi'\left[(1-\gamma)I + \gamma PM\Pi\right]\mathbf{r} \\ &= (1-\gamma)\Pi'\mathbf{r} + (1-\gamma)\gamma\Pi' P\Pi'\mathbf{r} + \gamma^2(\Pi' P)^2 M\Pi\mathbf{r} \\ &= (1-\gamma)\Pi'\mathbf{r} + (1-\gamma)\gamma\Pi' P\Pi'\mathbf{r} + \gamma^2(\Pi' P)^2 \Pi H\mathbf{r} \\ &\leq (1-\gamma)\Pi'\mathbf{r} + (1-\gamma)\gamma\Pi' P\Pi'\mathbf{r} + \gamma^2(\Pi' P)^2 \Pi' H\mathbf{r} \\ &\vdots \\ &= (1-\gamma)\sum_{i=0}^{\infty} \gamma^i (\Pi' P)^i \Pi'\mathbf{r} \;=\; M'\Pi'\mathbf{r} \end{aligned}$$

■

Thus, a dual policy iteration algorithm can be completely expressed in terms of the dual representation M, incorporating both dual policy evaluation (8) and dual policy improvement (17) (see Algorithm 2) leading to an equivalent result to the standard primal policy iteration algorithm based on (Lemma 2) and primal policy improvement (15)—see Algorithm 1.

D. Bellman Iteration

Finally, direct Bellman iteration algorithms can be developed based on the dual representations introduced above. These iterations bypass the explicit representation of a policy $\boldsymbol{\pi}$, and instead attempt to update the evaluation of the optimal policy implicitly.

In the primal case, Bellman iteration corresponds to the well known state value iteration update

$$\begin{aligned} \mathbf{v}'_{(s)} &= \max_a \mathbf{q}_{(sa)} \\ &= \max_a \mathbf{r}_{(sa)} + \gamma P_{(sa,:)}\mathbf{v} \qquad (20) \end{aligned}$$

In the dual representation, an analogous Bellman update can be derived with respect to the state distribution matrix

$$M'_{(s,:)} = (1-\gamma)\mathbf{1}_s^\top + \gamma P_{(sa^*,:)} M \qquad (21)$$

where $a^* = \arg\max_a P_{(sa,:)} M\Pi\mathbf{r}$ and $\mathbf{1}_s$ is the vector of all zeros except for a 1 in the sth position. Thus a dual form of Bellman update need not refer to the primal value functions at all. Nevertheless, these two updates (20) and (21) behave equivalently by Theorem 1.

V. Temporal Difference Learning

Beyond demonstrating novel dual representations for DP, we can also show how these representations can be used to derive novel forms of RL algorithms as well. In this section we don not assume the environmental variables P and $\mathbf{r}$ are known, but instead assume that our access to the environment is limited to the selection of actions and observation of state transitions and rewards.

A. TD Evaluation

First, we address TD prediction methods for policy evaluation. In the primal case, the value of a given policy can be estimated by the standard TD evaluation algorithm (see Algorithm 3), with the update step given by

$$\mathbf{v}_{(s)} \;\leftarrow\; (1-\alpha)\mathbf{v}_{(s)} + \alpha\left[r + \gamma\mathbf{v}_{(s')}\right] \qquad (22)$$

In the dual representation, an analogous TD evaluation algorithm (see Algorithm 4) can be derived with respect to the state distribution matrix. In this case, the update step is

$$M_{(s,:)} \;\leftarrow\; (1-\alpha)M_{(s,:)} + \alpha\left[(1-\gamma)\mathbf{1}_s^\top + \gamma M_{(s',:)}\right] \tag{23}$$

B. Sarsa: On-policy TD Control

Extending these methods, one can then consider the on-policy control problem. In the primal case, the $Sarsa$ algorithm (see Algorithm 5) approximates the action-value function of the policy being followed, and interleaves this with policy improvement. The action-value update step is

$$\mathbf{q}_{(sa)} \;\leftarrow\; (1-\alpha)\mathbf{q}_{(sa)} + \alpha\left[r + \gamma\mathbf{q}_{(s'a')}\right]$$

In the dual representation, an analogous $Sarsa$ algorithm (see Algorithm 6) can be derived with respect to the state-action distribution matrix. In this case, the distribution update can be given by

$$H_{(sa,:)} \;\leftarrow\; (1-\alpha)H_{(sa,:)} + \alpha\left[(1-\gamma)\mathbf{1}_{sa}^\top + \gamma H_{(s'a',:)}\right]$$

where $\mathbf{1}_{sa}$ is the vector of all zeros except for a 1 in the sa^{th} position.

C. Q-Learning: Off-policy TD Control

Finally, we consider the off-policy control problem. In the primal case, the Q-learning algorithm (see Algorithm 7) directly approximates $\mathbf{q}^*$, the optimal action-value function. Here the state-action value update is

$$\mathbf{q}_{(sa)} \;\leftarrow\; (1-\alpha)\mathbf{q}_{(sa)} + \alpha\left[r + \gamma\max_{a'}\mathbf{q}_{(s'a')}\right]$$

In the dual representation, an analogous Q-Learning algorithm (see Algorithm 8) can be derived with respect to the state-action distribution matrix, by using the distribution update

$$H_{(sa,:)} \;\leftarrow\; (1-\alpha)H_{(sa,:)} + \alpha\left[(1-\gamma)\mathbf{1}_{sa}^\top + \gamma H_{(s'a'^*,:)}\right]$$

where $a'^* = \arg\max_{a'} H_{(s'a',:)}\mathbf{r}$.

VI. Scaling Up via Linear Approximation

Scaling up LP, DP and RL algorithms to large sequential decision making problem domains has received a great deal of attention in recent years. The general goal is to generalize across state or state-action space by exploiting the structure of the problem with *function approximation*.

In the standard primal approach, usually referred to as "value function approximation", one approximates the desired value function as a simple parameterized function of a set of features (or basis functions). A common choice is to represent the value function as a linear function of k basis functions

$$\hat{\mathbf{v}} = \Phi\mathbf{w} \tag{24}$$

where Φ is a $|S|\times k$ matrix, and $\mathbf{w}$ is a $k\times 1$ vector of adjustable weights.

Linear value function approximation has been applied to derive approximate forms of LP, DP and RL algorithms. For

```
1. Initialization
    $\mathbf{v} \leftarrow$ arbitrary value
    $\Pi \leftarrow$ arbitrary policy $\boldsymbol{\pi}$
2. Policy Evaluation
    Solve for $\mathbf{v}$ in $\mathbf{v} = \Pi(\mathbf{r} + \gamma P\mathbf{v})$
3. Policy Improvement
    policy-stable $\leftarrow$ true
    For each $s \in \mathcal{S}$
        best-action $\leftarrow a$ s.t. $\boldsymbol{\pi}_{(sa)} = 1$
        $a^*(s) = \arg\max_a \mathbf{r}_{(sa)} + \gamma P_{(sa,:)}\mathbf{v}$
        $\boldsymbol{\pi}_{(sa)} = \begin{cases} 1 & \text{if } a = a^*(s) \\ 0 & \text{if } a \neq a^*(s) \end{cases}$
        If best-action $\neq a^*(s)$, then policy-stable $\leftarrow$ false
    If policy-stable, then stop; else go to 2
```

Algorithm 1: The policy iteration algorithm

```
1. Initialization
    $M \leftarrow$ a matrix with rows that are probability distributions
    $\Pi \leftarrow$ arbitrary policy $\boldsymbol{\pi}$
2. Policy Evaluation
    Solve for $M$ in $M = (1-\gamma)I + \gamma M\Pi P$
3. Policy Improvement
    policy-stable $\leftarrow$ true
    For each $s \in \mathcal{S}$
        best-action $\leftarrow a$ s.t. $\boldsymbol{\pi}_{(sa)} = 1$
        $a^*(s) = \arg\max_a (1-\gamma)\mathbf{r}_{(sa)} + \gamma P_{(sa,:)}M\Pi\mathbf{r}$
        $\boldsymbol{\pi}_{(sa)} = \begin{cases} 1 & \text{if } a = a^*(s) \\ 0 & \text{if } a \neq a^*(s) \end{cases}$
        If best-action $\neq a^*(s)$, then policy-stable $\leftarrow$ false
    If policy-stable, then stop; else go to 2
```

Algorithm 2: The dual policy iteration algorithm

```
Initialize $\mathbf{v}$ arbitrarily and $\boldsymbol{\pi}$ to the policy to be evaluated
Repeat (for each episode):
    Initialize $s$ arbitrarily
    Repeat (for each step of episode):
        $a \leftarrow$ action given by $\boldsymbol{\pi}$ for $s$
        Take action $a$ and observe the reward $r$ and next state $s'$
        $\mathbf{v}_{(s)} \leftarrow (1-\alpha)\mathbf{v}_{(s)} + \alpha\left[r + \gamma\mathbf{v}_{(s')}\right]$
        $s \leftarrow s'$
    until $s$ is terminal
```

Algorithm 3: The TD(0) algorithm

```
Initialize $M$ arbitrarily and $\boldsymbol{\pi}$ to the policy to be evaluated
Repeat (for each episode):
    Initialize $s$ arbitrarily
    Repeat (for each step of episode):
        $a \leftarrow$ action given by $\boldsymbol{\pi}$ for $s$
        Take action $a$ and observe the reward $r$ and next state $s'$
        $M_{(s,:)} \leftarrow (1-\alpha)M_{(s,:)} + \alpha\left[(1-\gamma)\mathbf{1}_s^\top + \gamma M_{(s',:)}\right]$
        $s \leftarrow s'$
    until $s$ is terminal
```

Algorithm 4: The dual TD(0) algorithm

```
Initialize $\mathbf{q}$ arbitrarily
Repeat (for each episode):
    Initialize $s$ arbitrarily
    Choose $a$ from $s$ using policy derived from $\mathbf{q}_{(sa)}$
    (e.g. $\epsilon$-greedy)
    Repeat (for each step of episode):
        Take action $a$ and observe the reward $r$ and next state $s'$
        Choose $a'$ from $s'$ using policy derived from $\mathbf{q}_{(s'a')}$
        (e.g. $\epsilon$-greedy)
        $\mathbf{q}_{(sa)} \leftarrow (1-\alpha)\mathbf{q}_{(sa)} + \alpha\left[r + \gamma\mathbf{q}_{(s'a')}\right]$
        $s \leftarrow s'$ $a \leftarrow a'$
    until $s$ is terminal
```

Algorithm 5: The Sarsa algorithm

```
Initialize $H$ with rows that are probability distributions
Repeat (for each episode):
    Initialize $s$ arbitrarily
    Choose $a$ from $s$ using policy derived from $H_{(sa,:)}$
    (e.g. $\epsilon$-greedy where $a_{greedy} = \arg\max_a H_{(sa,:)}\mathbf{r}$)
    Repeat (for each step of episode):
        Take action $a$ and observe the reward $r$ and next state $s'$
        Choose $a'$ from $s'$ using policy derived from $H_{(s'a',:)}$
        (e.g. $\epsilon$-greedy)
        $H_{(sa,:)} \leftarrow (1-\alpha)H_{(sa,:)} + \alpha\left[(1-\gamma)\mathbf{1}_{sa}^{\top} + \gamma H_{(s'a',:)}\right]$
        $s \leftarrow s'$ $a \leftarrow a'$
    until $s$ is terminal
```

Algorithm 6: The dual Sarsa algorithm

```
Initialize $\mathbf{q}$ arbitrarily
Repeat (for each episode):
    Initialize $s$ arbitrarily
    Repeat (for each step of episode):
        Choose $a$ from $s$ using policy derived from $\mathbf{q}_{(sa)}$
        (e.g. $\epsilon$-greedy)
        Take action $a$ and observe the reward $r$ and next state $s'$
        $\mathbf{q}_{(sa)} \leftarrow (1-\alpha)\mathbf{q}_{(sa)} + \alpha\left[r + \gamma\max_{a'}\mathbf{q}_{(s'a')}\right]$
        $s \leftarrow s'$
    until $s$ is terminal
```

Algorithm 7: The Q-learning algorithm

```
Initialize $H$ with rows that are probability distributions
Repeat (for each episode):
    Initialize $s$ arbitrarily
    Repeat (for each step of episode):
        Choose $a$ from $s$ using policy derived from $H_{(sa,:)}$
        (e.g. $\epsilon$-greedy where $a_{greedy} = \arg\max_a H_{(sa,:)}\mathbf{r}$)
        Take action $a$ and observe the reward $r$ and next state $s'$
        $H_{(sa,:)} \leftarrow (1-\alpha)H_{(sa,:)} + \alpha\left[(1-\gamma)\mathbf{1}_{sa}^{\top} + \gamma H_{(s'a'^*,:)}\right]$
        where $a'^* = \arg\max_{a'} H_{(s'a',:)}\mathbf{r}$
        $s \leftarrow s'$
    until $s$ is terminal
```

Algorithm 8: The dual Q-learning algorithm

the LP approach, one can obtain an approximate primal LP by plugging (24) into the exact primal LP (1):

$$\min_{\mathbf{w}}(1-\gamma)\boldsymbol{\mu}^{\top}\Phi\mathbf{w} \quad \text{subject to}$$
$$(\Phi\mathbf{w})_{(s)} \geq \mathbf{r}_{(sa)} + \gamma P_{(sa,:)}\Phi\mathbf{w} \qquad \forall s, a \qquad (25)$$

Solving this LP yields set of combination weights $\mathbf{w}^*$. The approximate optimal solution $\mathbf{v}^*$ can be extracted from $\Phi\mathbf{w}^*$.

Linear value function approximation can also be applied to the various TD algorithms for RL. For example, for the TD estimate at the core of a significant class of RL algorithms (22), one can compute a gradient based update to the linear approximation

$$\mathbf{w} \quad \leftarrow \quad \mathbf{w} + \alpha\left(\mathbf{r}_{(s)} + \gamma\hat{\mathbf{v}}_{(s')} - \hat{\mathbf{v}}_{(s)}\right)\Phi_{(s,:)}^{\top} \qquad (26)$$

where α is a step size parameter.

Similarly, we could apply the idea of function approximation in the dual representation. Here we combine k basis *distributions* instead of basis functions, and simply add the constraint that the combination weights $\boldsymbol{\omega}$ be a normalized probability distribution, which ensures that the convex combination of basis distributions yields a valid distribution. For example, a joint probability distribution over state-action pairs can be approximated as a convex combination of the basis distributions via

$$\hat{\mathbf{d}} \quad = \quad \Psi\boldsymbol{\omega} \quad \text{subject to } \boldsymbol{\omega} \geq 0,\ \boldsymbol{\omega}^{\top}\mathbf{1} = 1 \qquad (27)$$

where Ψ is a $|S||A| \times k$ nonnegative *row* normalized matrix, and $\boldsymbol{\omega}$ is a $k \times 1$ vector of nonnegative, normalized, adjustable weights.

Just as in the primal case, linear distribution approximation can be applied to derive approximate forms of *dual* LP, DP and RL algorithms. For the dual LP approach, one can easily obtain an approximate dual LP by plugging (27) into the dual LP (3):

$$\max_{\boldsymbol{\omega}}\boldsymbol{\omega}^{\top}\Psi^{\top}\mathbf{r} \quad \text{subject to} \quad \boldsymbol{\omega} \geq 0,\ \boldsymbol{\omega}^{\top}\mathbf{1} = 1$$
$$\Xi\Psi\boldsymbol{\omega} \leq (1-\gamma)\boldsymbol{\mu} + \gamma P^{\top}\Psi\boldsymbol{\omega} \qquad (28)$$

Similarly, the idea of exploiting the structure of the problem with function approximation could be applied to both the primal and dual forms of the temporal difference RL algorithms TD evaluation, Sarsa, and Q-learning. In these cases, the primal approximation parameters are normally adjusted using gradient principles. In the dual, one proceeds similarly by maintaining linear combinations of basis distributions, but under the additional constraint of maintaining normalization. Thus, the updates in the dual case are generally gradient-projection updates, where one first determines the desired gradient, but then projects this back to the subspace of normalized vectors before taking a convex-combination update step. Furthermore, as we saw for the dual TD algorithms above, we need to approximate a *matrix* of (row-wise) probability distributions, rather than just a single probability distribution in a vector, as in (27). A matrix approximation that maintains nonnegativity and row normalization can be achieved by introducing two fixed matrices of basis distributions, Υ and Γ, and one square combination matrix of adjustable weights, W, combined in a product

$$\hat{M} \quad = \quad \Upsilon W \Gamma \quad \text{ subject to } W \geq 0, \;\; W\mathbf{1} = \mathbf{1} \qquad (29)$$

where Υ is $|S|{\times}k$, W is $k{\times}k$, Γ is $k{\times}|S|$, and all matrices are nonnegative and *row* normalized, which is sufficient to ensure that $\hat{M}$ remains a nonnegative, row normalized approximation to M.

Given this linear approximation architecture for the dual representation M, a gradient based update for the dual TD estimate (23) can be derived as

$$W \quad \leftarrow \quad W + \alpha\delta\,\Delta \qquad (30)$$

where α is a step size parameter; $\delta = 1 - \gamma + \gamma\hat{M}_{(s',s'')} - \hat{M}_{(s,s')}$ and the $k{\times}k$ update matrix Δ is the projection of the gradient matrix onto the space of row normalized matrices, obtained by solving the small auxiliary quadratic program (QP)

$$\min_{\Delta} \sum_{ij} \left(\Delta_{(i,j)} - D_{(i,j)}\right)^2 \text{ subject to}$$

$$\Delta\mathbf{1} = \mathbf{0}, \;\; D = \Upsilon^{\top}_{(s,:)}\Gamma^{\top}_{(:,s')} \qquad (31)$$

Note that the $k \times k$ matrix D gives the gradient update direction for W, but since D is not necessarily row normalized, we require the auxiliary QP to compute the nearest update matrix Δ that preserves row normalization. The update equation (30) is guaranteed to maintain the row normalization of W (although the step size α may need to be adjusted to maintain nonnegativity), and this in turn is sufficient to guarantee that the matrix approximation $\hat{M}$ remains a valid matrix of row-wise probability distributions.

Fortunately, the small quadratic program (31) has a closed form solution. In fact, the solution to (31) can be written

$$\Delta \quad = \quad D - \frac{1}{k}D(\mathbf{1}\mathbf{1}^{\top}) \qquad (32)$$

where $D = \Upsilon^{\top}_{(s,:)}\Gamma^{\top}_{(:,s')}$. Note that Δ minimizes the objective in QP (31), because each row in is the projection of D onto the constraint.

Lemma 12: $\Delta\mathbf{1} = \mathbf{0}$

Proof:

$$\begin{aligned} \Delta\mathbf{1} &= D\mathbf{1} - \frac{1}{k}D(\mathbf{1}\mathbf{1}^{\top})\mathbf{1} \\ &= D\mathbf{1} - \frac{1}{k}D\mathbf{1}k \\ &= D\mathbf{1} - D\mathbf{1} \\ &= \mathbf{0} \end{aligned} \qquad (33)$$

■

Although the approximate dual TD update (30) is computationally more expensive than the primal counterpart (26) because of the projection, it has the advantage of keeping a bounded representation, which automatically avoids the risks of divergence that exist for primal approximation algorithms [1], [9], [10].

VII. Conclusion

We investigated new dual representations for LP, DP and RL algorithms based on maintaining probability distributions, and explored connections to their primal counterparts based on maintaining value functions. In particular, we derived the original dual form representations from basic LP duality, extended these representations to derive new forms of DP algorithms and new forms of RL algorithms (TD evaluation, Sarsa, and Q-learning), and furthermore demonstrated how these dual representations can be scaled up to large domains by introducing normalized linear approximations. Although many of the results demonstrate equivalence between the primal and dual approaches, some advantages seem apparent for the dual approach, including an intrinsic robustness against divergence, and the contribution of a novel perspective that yields new forms of prior knowledge that can be exploited in large domains.

References

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

[2] M. Puterman, *Markov Decision Processes: Discrete Dynamic Programming*. Wiley, 1994.

[3] D. Bertsekas, *Dynamic Programming and Optimal Control*. Athena Scientific, 1995, vol. 2.

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

[5] P. Dayan, “Improving generalisation for temporal difference learning: The successor representation,” *Neural Computation*, vol. 5, pp. 613–624, 1993.

[6] A. Ng, R. Parr, and D. Koller, “Policy search via density estimation,” in *Proceedings NIPS*, 1999.

[7] D. de Farias and B. Van Roy, “The linear programming approach to approximate dynamic programming,” 2001.

[8] S. Ross, *Introduction to Probability Models*, 6th ed. Academic Press, 1997.

[9] J. Tsitsiklis and B. Van Roy, “An analysis of temporal-difference learning with function approximation,” *IEEE Transactions on Automatic Control*, vol. 42, no. 5, pp. 674–690, 1997.

[10] L. Baird, “Residual algorithms: Reinforcement learning with function approximation,” in *Proceedings ICML*, 1995.