Proceedings of the
47th IEEE Conference on Decision and Control
Cancun, Mexico, Dec. 9-11, 2008

Choosing the Cost Vector of the Linear Programming Approach to Approximate Dynamic Programming

Daniela Pucci de Farias
Department of Mechanical Engineering
MIT
pucci@mit.edu

Théophane Weber
Operations Research Center
MIT
theo_w@mit.edu

Abstract— **We consider the linear programming approach to approximate dynamic programming. In the general case of linear combination of features, we prove the existence of a solution which can be used to generate a policy with performance proportional to the strength of the architecture. In the special case of features defined on a partition of the state space, we give a simpler method to find the solution, as well as a stronger performance bound.**

I. Introduction

The linear programming approach to approximate dynamic programming was introduced to tackle the curse of dimensionality in Markov Decision Processes (MDP), combining the formulation of dynamic programming (DP) as a linear program (see [11]), with ideas from value function approximation. Initially introduced by [12], formulation and results have been improved by various work, including [3,4,5,6,13,14]. In the general case, the solution of the approximate linear program (ALP) depends on the cost vector used, a result which contrasts sharply with the case of linear programming for exact dynamic programming. Therefore, an important question to address is how to choose the cost vector which will be used to generate a solution, and hence, a policy.
In this paper, we first prove the existence of cost vectors for which solutions have a performance loss proportionally bounded by the distance between the optimal cost-to-go function and the space spanned by the approximating architecture, for a norm that scales gracefully with the size of the state-space. These cost vectors are solutions of a family of fixed-point equations, and we show numerical experiments where we find the fixed point using a simple stochastic approximation algorithm.
These results hold without any specific structural assumption concerning the Markov Decision Process or the approximating space, making the methods very general. In the special case of features defined on a partition of the state space (generalization of indicator functions), it can be seen that the solution of the linear program does not depend on the cost vector used. Then, one can directly obtain an approximation to the cost-to-go which is the best lower bound possible attainable in the approximating architecture, in a ”pointwise” meaning, and this allows to write a stronger performance bound for the corresponding policy.

II. ALP and state-relevance weights

A. The approximate linear program

Let $\{S, U, g, P\}$ be a finite decision Markov Decision Process with state space $S = \{1, 2, ..n\}$ and action space U_x for each $x \in S$. Transition probabilities $P_a(x, y)$ for $x, y \in S$ and $a \in U(x)$ denote the probability of transiting from x to y if action a is chosen, while $g_a(x)$ denotes the cost of taking action a when in state x.

We consider stationary, randomized policies and each policy u is identified with a function from the state-space to the set of distributions on the action spaces. If we let $\Delta(A)$ be the set of probability distributions (simplex) over the elements of the finite set A, then a policy u can be written

$$u : x \to u(x) = [u(x, a_1), .., u(x, a_{|U_x|})] \in \Delta(U_x)$$

The set of all randomized policies is $\Delta(U)$, which is compact and convex. For any policy u, we denote $g_u(x) = \sum_{a \in U(x)} u(x, a)g(x, a)$ the cost of the randomized action $u(x)$ in state x.

The objective of the controller is to minimize the α-discounted cost

$$J^*(x_0) = \min_{u \in \Delta(U)} E[\sum_{t=0}^{\infty} \alpha^t g_u(x_t)|x_0] = (I - \alpha P_u)^{-1} g_u.$$

The above quantity is denoted $J^(x_0)$ and called the optimal value function (or cost-to-go function). We also denote T be the traditional dynamic programming operator: for any $J \in \mathbb{R}^{|S|}$, TJ is also in $\mathbb{R}^{|S|}$ and defined by:

$$\begin{aligned} (TJ)(x) &= \min_{a \in U(x)} (T_a J)(x) \\ &\text{where} \\ (T_a J)(x) &= \big(g_a(x) + \sum_y P_a(x, y)J(y)\big) \end{aligned}$$

The optimal value function J^* verifies the Bellman equation $TJ^* = J^*$.
One approach to solve this equation is to solve the following linear program, for a positive cost vector c:

$$\begin{aligned} &\max_J \; c'J \\ &\text{such that} \\ &TJ \geq J \qquad (1) \end{aligned}$$

Since $TJ \geq J \Leftrightarrow \forall x, a,\ (T_a J)(x) \geq J(x)$, the seemingly nonlinear constraint $TJ \geq J$ can be easily converted into a family of linear constraints as follows:

$$\forall x \in S, a \in U(x),\ g_a(x) + \sum_y P_a(x,y)J(y) \geq J(x) \quad (2)$$

which is linear in J. As long as the cost vector c is positive and the state-space finite, the unique solution to equation (1) is the optimal cost vector J^*.

The purpose of approximate dynamic programming is to fit J^* with an approximation architecture. Most commonly used are linear architectures, which approximate the value function J^* by a linear combination of "features" $\phi_i, i = 1,\ldots,r$. Each feature ϕ_i is an arbitrary (possibly nonlinear) mapping from S to $\mathbb{R}$. The approximation consists in fitting linear parameters $r_i \in \mathbb{R}$, so that $J^*(x)$ is "close" to the approximation $\sum_i r_i\phi_i(x)$ (or, in compact form, $\Phi(x)r$, where $\Phi(x) = [\phi_1(x)|\phi_2(x)|\ \phi_3(x)|\ldots|\phi_r(x)]$). In this context, it is natural to consider the so-called approximate LP:

$$\max_{r \in \mathbb{R}^r} c'\Phi r$$
$$\text{such that}$$
$$T\Phi r \geq \Phi r \quad (3)$$

As in (2), the nonlinear constraint $T\Phi r \geq \Phi r$ can be rewritten into a family of linear constraints indexed by a state x and action a:

$$g_a(x) + \alpha \sum_i \big(\sum_{y \in S} P_a(x,y)\phi_i(y)\big) r_i \geq \sum_i \phi_i(x) r_i$$

The ALP has far fewer variables than the LP formulation. However, it has a very large number of constraints, and large-scale optimization methods have to be devised to solve the problem (see [5], [13]). Also, in general, the solution r (and the corresponding approximation Φr) will depend on the choice of the nonnegative vector c. Since c is non negative and can be rescaled without changing the solution of the ALP, c can be considered to be a distribution over the state-space, and will be called vector of "state relevance weights".

B. Known performance bounds

Approximate LP offers two advantages as an approximate dynamic programming method: First, it always returns an approximation to the optimal cost-to-go function in finite time (polynomial if using constraint sampling) , and has therefore no convergence issues; second, the approximation returned by the ALP is "close" to the optimal cost-to-go function, in a sense detailed by the following definition and theorem.

Definition 1 (Lyapunov combination): A feature vector $v \in \mathbb{R}^m$ is called Lyapunov with rate β_v if :

$$\forall x,\ \Phi(x)v \geq 0$$

$$\alpha \max_{u \in U(x)} \sum_{y \in \mathcal{S}} P_u(x,y)\Phi(y)v \leq \beta_v \Phi(x)v$$

Note that the unit vector $e = (1,1,1,\ldots,1) \in \mathbb{R}^n$ is always a Lyapunov function with rate α. Also for any positive vector $c \in \mathbb{R}^n$ and vector $J \in \mathbb{R}^n$, denote $\|J\|_{1,c} = \sum_x c(x)|J(x)|$ (weighted L_1 norm).

Theorem 1 (Theorem 4.2 in [3]): Let $r(c)$ be any optimal solution of the approximate linear program with the state relevance weight c, and let v be any Lyapunov combination with rate β_v. Then:

$$\|J^* - \Phi r(c)\|_{1,c} \leq \frac{2c^T\Phi v}{1-\beta_{\Phi v}} \min_{r \in \mathbb{R}^m} \|J^* - \Phi r\|_{\infty,1/\Phi v} \quad (4)$$

The theorem above states that ALP provides a good approximation to the optimal cost J^*. Intuitively, if J^* is well approximated by some point of the approximation architecture, then it will be decently approximated by the solution Φr of the LP. It is therefore natural to consider as a candidate policy the *greedy* policy with respect to Φr, i.e. the policy which, for any state x, chooses for $u(x)$ the action a which minimizes:

$$g_a(x) + \sum_{i,y} P_a(x,y)\phi_i(y)r_i$$

In the rest of the paper, we denote $u(J)$ the policy which is greedy with some approximation J of the cost-to-go J^*.

The question that naturally arises then is whether a policy greedy with respect to Φr will perform well. Indeed, in MDP theory J^* is both the performance of the optimal policy and the cost-to-go function used to compute that optimal policy. However, we only know Φr approximates well J^*, without particular knowledge of the performance of the policy induced by Φr. This question is partially answered by the following bound:

Theorem 2 (Theorem 3.1 in [3]): Let $J \in \mathbb{R}^n$ such that $TJ \geq J$, u_J be the policy greedy with respect to J, and let ν be a distribution. Then,

$$\|J_{u(J)} - J^*\|_{1,\nu} \leq \frac{1}{1-\alpha}\|J - J^*\|_{1,\mu_{\nu,u(J)}} \quad (5)$$

where $\mu_{\nu,u(J)} := (1-\alpha)\nu^T(I - \alpha P_{u(J)})^{-1}$ is a distribution over the state space.

For α close to 1, $\mu_{\nu,u(J)}$ is close to the steady-state distribution $\pi_{u(J)}$ of $P_{u(J)}$.

The caveat is that a norm mismatch prevents us from concluding anything about the performance of the policy which is greedy with respect to the cost-to-go approximation returned by the ALP. More precisely, if we solve the ALP with state relevance weights c, and use policy $u(\Phi r)$, the performance bound (5) becomes

$$\|J_{u(\Phi r)} - J^*\|_{1,\nu} \leq \frac{1}{1-\alpha}\|\Phi r(c) - J^*\|_{1,\mu_{\nu,u(\Phi r)}} \quad (6)$$

but the right hand side of this equation is a 1-norm with weights vector $\mu_{\nu,u(\Phi r)}$, unlike equation 4 where the left hand side has weight vector c. We will see that with a special choice of state-relevance weight and using exploration in a fashion similar to [4], the norm mismatch disappears, resulting in the first general performance bound for approximate LP. The appropriate cost vectors will be solutions of a family of fixed point equations we detail next.

III. Performance bounds for approximate LP

A. Operators on the space of state-relevance weights

If one could ensure that for some c, its ALP solution r had a corresponding $\mu_{\nu,u(\Phi r)}$ vector equal to c, then we would be able to obtain a performance bound. Indeed, in this case, the left hand side of equation 6 would be equal (multiplicative constant aside) to the left hand side of equation 4. We would then obtain that the performance loss is bounded by the strength of the approximation architecture. However, for almost all $c > 0$ the solution to the ALP is one of finitely many extreme points of the feasible polyhedron $R = \{r \in \mathbb{R}^r | T\Phi r \geq \Phi r\}$, and therefore the corresponding $\mu_{\nu,u(\Phi r)}$ vector is one of finitely many vectors. If one of these vectors is image of itself by the mapping $c \longrightarrow r(c) \longrightarrow \mu_{\nu,u(\Phi r(c))}$, then we can obtain a performance bound. However, this is not true in general, and in order to conclude, we need to incorporate exploration as follows.

The first level of exploration is to include some level of randomization on the cost vector c, in order to allow the ALP to return vectors which are almost solutions, but not necessarily extreme points.

Let $r_1, r_2, \ldots, r_k$ be the extreme points of P. We denote $R_i = \{c \in \Delta(S) | c^T\Phi r_i \geq c^T\Phi r_j, \text{ for all } j\}$.

Intuitively, R_i is the set of state-relevance weights which return r_i as solution to the ALP.

Definition 2 (smoothed r function): For a given $\mu > 0$, the smoothed r function with noise μ is defined as

$$r^\mu(c) = \sum_{1\leq i\leq k} P(c + \widehat{\mu} \in R_i).r_i$$

where $\widehat{\mu}$ is a gaussian vector with i.i.d. components of variance μ. μ is called noise of the smoothed function r.

Proposition 1: For any $\mu > 0$, $r^\mu(c)$ is a well-defined, smooth ($\mathcal{C}^\infty$) function.

Then, we include exploration in the policies:

Definition 3 (δ -greedy policy): For a given positive number δ, the δ-greedy policy with respect to J is the policy which in state x chooses action a with probability proportional to

$$\exp\left(\frac{-g_a(x) - \sum_y P_a(x \to y).J(y)}{\delta}\right)$$

The policy will be denoted $u^\delta(J)$, and δ is called *temperature* of the δ-greedy policy.

Similarly as proposition 1, we have:

Proposition 2: For any $\delta > 0$, the δ -greedy policy $u^\delta(\Phi r)$ is well defined and smooth (as a function of r).

Combining propositions (1) and (2), we can now state the following :

Proposition 3: The composed function:

$$f : (\mu, \delta, c) \to (1-\alpha)\nu^T(I - \alpha P_{u^\delta(\Phi r^\mu(c))})^{-1}$$

is smooth. Furthermore, for any fixed $\mu, \delta > 0$, its restriction $c \to f^{\mu,\delta}(c)$ is a smooth mapping of $\Delta(S)$ (i.e. a smooth function from $\Delta(S)$ to $\Delta(S)$).

The second part of the proposition is essential since it will allow us to use Brouwer's theorem to assert the existence of a fixed point.

B. Fixed point and Performance bound

For fixed $\alpha > 0, \nu \in \mathbb{R}^n, \mu > 0, \delta > 0$ We define the smooth mapping

$$\begin{array}{rcl} W^{\mu,\delta} & : & \\ \Delta(S) \times R & \to & \Delta(S) \times R \\ (c, r) & \to & (f^{\mu,\delta}(c), r^\delta(c)) \end{array}$$

By Brouwer's fixed point theorem, it has at least one fixed point $(c^{\mu,\delta}, r^{\mu,\delta})$, temporarily denoted $(\bar{c}, \bar{r})$ for notational simplicity. We are now able to write a performance bound for the δ-greedy policy with respect to Φr.

Theorem 3 (Performance bound with exploration): The δ-greedy policy $\bar{u}$ with respect to $\Phi\bar{r}$ has the following performance bound:

$$\begin{aligned} \|J^* - J_{\bar{u}}\|_{1,\nu} \leq \quad & \frac{\bar{c}^T\Phi v}{(1-\alpha)(1-\beta_{\Phi v})} \min \|\Phi r - J^*\|_{\infty,\frac{1}{\Phi v}} \\ & + O(\mu) + O(\delta) \end{aligned} \tag{7}$$

where the O functions are independent of r.

Assuming the unit vector e belongs to the span of the feature space, we obtain as a corollary:

$$\|J^* - J_{\bar{u}}\|_{1,\nu} \leq \frac{2}{(1-\alpha)^2} \min \|\Phi r - J^*\|_\infty + O(\mu) + O(\delta) \tag{8}$$

Using exploration was essential to be able to relate the performance bound to the approximation bound, resulting in performance loss bound for all possible values of the noise and temperature $\mu, \delta > 0$. The dependency on the noise and temperature values seems to vanish as these quantities go to zero, and therefore, one may wonder if it is possible to find a feature weights vector r and cost vector c such that the performance bound holds without noise, without necessarily having a fixed point. It turns out this is possible, as follows:

Theorem 4 (General Performance Bound): There exists a state relevance weights vector c, and a feature weights vector r optimal for the ALP with cost c, such that the greedy policy u with respect to Φr has the following performance bound:

$$\|J^* - J_u\|_{1,\nu} \leq 2\frac{c^T\Phi v}{(1-\alpha)(1-\beta_{\Phi v})} \min \|\Phi r - J^*\|_{\infty,\frac{1}{\Phi v}} \tag{9}$$

An important feature of theorem 4 is that it holds without any specific assumptions, neither on the structure of the Markov Decision Process, nor on the approximation architecture (the assumption of the existence of a Lyapunov function can be relaxed and the theorem extended to the formulation of ALP with cost-shaping function, as seen in [6]). Note also that the feature vector r mentioned in (9) is not any solution to the ALP with cost vector c, and for some situations, r will not be an extreme point of R, and the cost vector c will belong to a interior of the feasible polyhedron R. Pratically, that implies that one should solve the fixed point problem for small values of μ, δ, and use limit values of r as μ and δ go to zero.

Also, while it is easy to simulate a δ- greedy policy with respect to Φr, one may object that computing the expected value of the solution vector with respect to a perturbed cost vector is too computationally cumbersome: solving LPs takes time, and one certainly does not want to solve the many variations of the same LP necessary to get a good estimate of $r^{\mu}(c)$. It turns out one can also smooth the function $c \to r(c)$ by using a barrier (interior point) method without taking the barrier parameter all the way to zero. It is therefore possible to compute $r^{\mu}(c)$ by solving only one LP. The exposition being more technical but intuitively the same, we opted to present results with the randomized cost function.

C. *Features defined on a partition of the space*

While many approximate DP methods may encounter convergence issues when faced with general MDPs or architecture spaces, most methods have good guarantees for convergence and quality of the solution when the features are indicator functions (see [7]). It is therefore interesting to see how does ALP perform in a slight generalization of this special case.

We assume now that the state space is partitioned into subspaces denoted $S_i, i = 1 \ldots r$, where $\bigcup_{i=1\ldots r} S_i = S$, and $S_i \bigcap S_j = \emptyset$, for all $i \neq j$. Assume there are r non negative features $\phi_i, i = 1 \ldots r$, such that $\phi_i(x) > 0$ only if $x \in S_i$. Note that the non negative assumption is done without loss of generality: if features are defined over a partition of the state-space, but are not constrained in sign, then every feature ϕ_i can be separated into two features $\phi_i^+(x) = \sup(0, \phi_i(x))$ (with corresponding set $S_i^+ = S_i \bigcap \{x \in S | \phi_i(x) \geq 0\}$) and $\phi_i^-(x) = \sup(0, -\phi_i(x))$ (set $S_i^- = S_i \bigcap \{x \in S | \phi_i(x) < 0\}$). This features typically arise when discretizing a continuous state-space, or when performing state aggregation.

Under that specific assumption, the feasible polyhedron R takes a particular structure, similar to the one of the feasible polyhedron for the exact linear program $\{J \in \mathbb{R}^n | TJ \geq J\}$: more precisely, the solution of the LP is always the same as long as the cost vector is positive.

Proposition 4: There exists an extreme point r^* such that for all $c > 0$, r is the unique solution to the ALP with cost vector c. Equivalently, the pointwise maximum of all $r \in R$ is a feasible point r^*.

Combining proposition (4) and theorem (4), we can then prove the following:

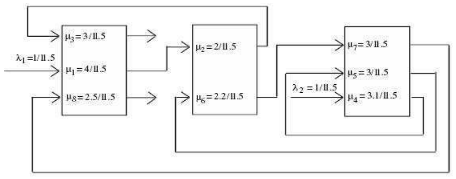

Fig. 1. Eight-dimensional queueing network

Proposition 5: The policy u obtained by using ALP with features defined on a partition of the state-space satisfies:

$$\|J^* - J_u\|_{\infty, \frac{1}{\Phi v}} \leq \frac{2}{(1 - \beta_{\Phi v})^2} \min \|\Phi r - J^*\|_{\infty, \frac{1}{\Phi v}} \quad (10)$$

For instance, if the ϕ_i are indicator functions of a partition of the state-space, the performance bound becomes:

$$\|J^* - J_u\|_{\infty} \leq \frac{2}{(1 - \alpha)^2} \min \|\Phi r - J^*\|_{\infty} \quad (11)$$

A notable feature of the above is the use of weighted infinite norms, and more importantly, that one does not need to worry about the existence of a fixed point, or convergence to the said fixed point. To our knowledge, such assumptions are required by most, if not all, approximate DP bounds. In the case of ALP defined on a partition of the state-space, we need to solve only one LP, and the policy associated to the solution immediately satisfies a performance bound. However, our bound scales roughly as $\frac{1}{(1-\alpha)^2}$, which means in the limit $\alpha \to 1$ the bound becomes trivial unless more features are added. This is in contrast with the powerful bound of [7], the only approximate DP bound we know of which scales as $\frac{1}{1-\alpha}$ and therefore is useful for all values of α (for a longer discussion about how each side of the equations should scale, see [7]).

IV. NUMERICAL EXPERIMENTS

In this section we show some numerical experiments aiming at finding the desired vectors of state relevance weights, for a high dimensional example with very large number of states. Consider the queueing control problem of figure 1, with arrival rates indicated by $\delta_i, i = 1 \ldots 8$, service rates by $\mu_i, i = 1 \ldots 8$. The cost function is the sum of customers in system, discount factor is $\alpha = 0.995$, the features are all polynomials of degree less than 2 (total number of features: 46). The initial vector of state relevance weight is geometric, $c(x) = (1 - \xi)^8 \xi^{|x|}$, and so is the initial distribution $\nu(x) = (1 - \nu)^8 \nu^{|x|}$, and the sampling distribution (equal to ν). We then generate a sequence of cost vectors c_k as follows:

∘ $c_0 = \nu$
∘ $c_{k+1} = (1 - h).c_k + h f^{\mu,\delta}(c)$, for $k \geq 1$

For this particular experiment, h was set to 0.8, ν was set to 0.9, ξ to various values, the noise and temperature levels μ and δ were set to zero, and the distribution μ was generated by monte-carlo simulation. Also, it is impossible

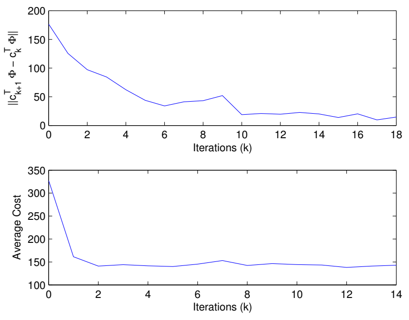

Fig. 2. Convergence to a fixed point and corresponding performance

to store the vectors c_k (since they have very high dimension), but it is equivalent to store the vectors $c_k^T\Phi \in \mathbb{R}^r$, which was done for all numerical implementations of the algorithm. The example shows the importance of an appropriate choice of c, and validates the insight that the performance of the ALP policy is related to whether c is a fixed point of f or not. Indeed, the average number of customers in systems showed to be highly dependent on the cost vector used (there was difference of more than 50% between the best choice of c and the initial c_0). Moreover, while this simple algorithm has no convergence guarantees, the norm of the difference between c_k and c_{k+1} does go down to small values, and as the norm difference goes down, so does the average number of customers in system: Corresponding policies have a performance of under 150 customers in queue on average. By comparison, the best ALP performance for the same problem (obtained by search on the parameters) goes down to 134; heuristics get systematically worse results (LBFS obtains 153, FIFO 163, LONGEST 168, see [3]).

V. Proofs of results in section 3.1 and 3.2

We first note that $r^\mu(c)$ is well defined, since for all $c \in \mathbb{R}^n$ and $\mu > 0$, $\sum P(c + \widehat{\mu} \in R_i) = 1$ and therefore, $r^\mu(c) = \sum\limits_{1\le i\le k} P(c + \widehat{\mu} \in R_i) r_i$ is a convex combination of extreme points of R, and therefore belongs to R.
Proposition 1 is a consequence of the following lemma: let f be a piecewise constant function from $\mathbb{R}^n$ to a compact set C, and g be a non-negative, smooth ($\mathcal{C}^\infty$) function from R^n to $\mathbb{R}$. Then, the function defined by: $f(x) \rightarrow \int\limits_{\mathbb{R}^n} f(x+y)g(y)\,dy$ is smooth. We then choose for f the function which assigns to a cost vector c any solution of the ALP with cost c, and choose for g the density of a gaussian random vector with i.i.d. components. Proposition 2 comes directly from the definition of the $u^\delta(J)$ policy, which for every state is a convex combination of actions, each coefficient of the convex combination being a smooth function of J. Finally, proposition 3 is derived from propositions 1 and 2, and from the fact that $P_u \longrightarrow \nu^T(I-\alpha P_u)^{-1}$ is a smooth function of P_u (when P_u is a stochastic matrix). The existence of fixed points for all values of $\alpha < 1$, $\mu > 0$ and $\delta > 0$, and distributions ν is an application of Brouwer's fixed point theorem to proposition 3. Before we can prove our main theorem, we first need some properties of δ-greedy policies. Recall that a δ-greedy policy $u^\delta(J)$ with respect to $J \in \mathbb{R}^{|\mathcal{S}|}$ chooses decision a in state x with probability

$$u^\delta(J)(a,x) = \frac{\exp[-(g_a(x) + \alpha P_a(x)J)/\delta]}{\sum_{a' \in U(x)} \exp[-(g_{a'}(x) + \alpha P_{a'}(x)J)/\delta]}. \tag{12}$$

Lemma 1: Assume that $\underline{A} = \{a_1, \ldots, a_q\} \subset U(x)$ is the set of minimizers of $g_a(x) + \alpha P_a(x)J$. Then,

$$\lim_{\delta\downarrow 0} u^\delta(J)(a,x) = u^0(J)(a,x) = \begin{cases} 1/q \text{ if } a \in \underline{A} \\ 0 \text{ otherwise} \end{cases}$$

The meaning of this lemma is that a δ-greedy policy is essentially the same as a greedy-policy if the noise level δ is very small. Our next lemma, from [3], states a similar result, but with a better control on the difference between δ-greedy and greedy:

Lemma 2: Let J be a vector in $\mathbb{R}^n$, and let $u^0(J)$ be a greedy policy with respect to J, and $u^\delta(J)$ be the δ-greedy policy with respect to J. Define the δ-greedy bellman operator $T^\delta J = g_{u^\delta(J)} + \alpha P_{u^\delta(J)} J$.
Then,

$$\left\|TJ - T^\delta J\right\|_\infty \le \frac{\delta m}{e} \tag{13}$$

where m is the maximum number of actions available in a state, and e is the base of the natural logarithm.

Note that the bound in the previous lemma does not depend on J. We prove two intermediate lemmas before finally proving theorem (3).

Lemma 3: For a given feasible vector of features weights r, the δ-greedy policy with respect to Φr has the following performance bound:

$$\|J^* - J_{u^\delta}\|_{1,\nu} \le \frac{1}{1-\alpha}\|J^* - \Phi r\|_{1,\mu(u^\delta(\Phi r))} + O(\delta)$$

and the O function is independent of r.

Proof: Since $T\Phi r \ge \Phi r$, we have by monotonicity of the bellman operator $\Phi r \le J^*$. By optimality of J^*, we also have: $J^* \le J_{u^\delta}$

$$\begin{aligned}
\|J^* - J_{u^\delta}\|_{1,\nu} &= \nu^T(J_{u^\delta} - J^*) \\
&\le \nu^T(J_{u^\delta} - \Phi r) \\
&\le \nu^T(I - \alpha P_{u^\delta})^{-1}(T^\delta\Phi r - T\Phi r) \\
&+ \nu^T(I - \alpha P_{u^\delta})^{-1}(T\Phi r - \Phi r) \\
&\le \nu^T(I - \alpha P_{u^\delta})^{-1} e \left\|T^\delta\Phi r - T\Phi r\right\|_\infty \\
&+ \nu^T(I - \alpha P_{u^\delta})^{-1}(T\Phi r - \Phi r) \\
&\le \frac{1}{1-\alpha}\frac{\delta}{me} \\
&+ \nu^T(I - \alpha P_{u^\delta})^{-1}(T\Phi r - \Phi r) \\
&\le O(\delta) + \frac{1}{1-\alpha}\|J^* - \Phi r\|_{1,\mu_{\nu,u^\delta}}
\end{aligned}$$

■

Lemma 4: For any cost vector c

$$\|J^* - \Phi r^\mu(c)\|_{1,c} \le 2\frac{c^T\Phi v}{(1-\beta_{\Phi v})}\min\|\Phi r - J^*\|_{\infty,\frac{1}{\Phi v}} + O(\mu)$$

and the O function is independent of c.

Proof: For any realization $\widehat{\mu}(\omega)$ of $\widehat{\mu}$ such that $c+\widehat{\mu}(\omega)$ belongs to R_i, we have (theorem 4.2 in [3])

$$(c+\widehat{\mu}(\omega))^T(J^*-\Phi r_i) \leq 2\frac{(c+\widehat{\mu}(\omega))^T\Phi v}{(1-\beta_{\Phi v})}\min\|\Phi r - J^*\|_{\infty,\frac{1}{\Phi v}}$$

Therefore:

$$\begin{aligned} c^T(J^*-\Phi r_i) &\leq 2\frac{c^T\Phi v}{(1-\beta_{\Phi v})}.\min\|\Phi r - J^*\|_{\infty,\frac{1}{\Phi v}} \\ &+ O(\mu) \end{aligned}$$

where O is independent from c.

By summing:

$$\begin{aligned} \|J^*-\Phi r^{\mu}c\|_{1,c} &= \sum_i P(c+\widehat{\mu}\in R_i)c^T(J^*-\Phi r_i) \\ &\leq 2\frac{c^T\Phi v}{(1-\beta_{\Phi v})}\min\|\Phi r - J^*\|_{\infty,\frac{1}{\Phi v}} \\ &+ O(\mu) \end{aligned}$$

■

Theorem 3 follows from immediately lemmas 5 and 6.

VI. Proofs of the results in section 3.3

We now proceed to the proofs of the bounds and results in the special case of nonnegative features defined on a partition of the state-space. Recall the state-space is partitioned as $S = \bigcup_{i=1\ldots r} S_i$, and we have $S_i \bigcap S_j = \emptyset$, for all $i \neq j$. There are r nonnegative features $\phi_i, i = 1\ldots d$, each defined on a partition: $\phi_i(x) > 0$ only if $x \in S_i$. For any vector of feature weights $r \in \mathbb{R}^d$, denote r_i its i^{th} component. All results in section 3.3 stem from the following lemma, which is equivalent to proposition 4:

Lemma 5: Define the pointwise maximum r^* of R as follows:

$$\forall i \in 1\ldots d,\ r^*_i = \max_{r\in P} r_i$$

Then r^* is feasible

Proof: Let us evaluate $T\Phi r^*$ in some point x:

$$\begin{aligned} T\Phi r^*(x) &= T_a\Phi r^*(x) \text{ for some action } a \\ &= g_a(x) + \alpha\sum_y P_a(x,y)\Phi(y)r^* \end{aligned}$$

Since $\Phi(y) \geq 0$ and $r^* \geq r$ for any feasible r , this implies:

$$\begin{aligned} T\Phi r^*(x) &\geq g_a(x) + \alpha\sum_y P_a(x,y)\Phi(y)r, \text{ for any feasible } r \\ &\geq \Phi(x)r, \text{ for any feasible } r \end{aligned}$$

Let's denote $i(x)$ the index of the set S_i such that $x \in S_i$. For any x, and for any r, we have $\Phi(x)r = \phi_{i(x)}(x)r_{i(x)}$. Let us then consider the feasible $\bar{r}$ which achieves maximum r_i, i.e.: $\bar{r}_{i(x)} = r^*_i$. We conclude:

$$T\Phi r^*(x) \geq \Phi(x)\bar{r} = \phi_{i(x)}\bar{r}_{i(x)} = \phi_{i(x)}r^*_i = \Phi(x)r^*$$

■

The approach we took to prove the previous lemma was to define a priori r^* and prove it is feasible. One other way is to prove that for any system of features, the pointwise maximum of Φr over the entire polyhedron R is always feasible (for the exact LP for DP), but only with partitioned features this pointwise maximum can be represented as Φr^* for some particular r^*.

Proposition 4 and lemma 7 are easily seen to be equivalent, and imply proposition 5.

VII. Conclusions

Many approximate dynamic programming methods with cost-to-go function approximation either fail to provide guarantees on convergence or fail to ensure high quality of the resulting policy in the case of a general feature space. We prove here that, with a careful choice of the state-relevance weights vector, the solution of the ALP is always bounded, and has good performance guarantee. Future work includes looking into generalizations of the "pointed polyhedron" result obtained for features defined on a partition of the state-space, strengthening of the bound, and development of an algorithm to find the fixed point for the general theorem. Preliminary results with homotopy-based methods are very encouraging.

References

[1] Dimitris Bertsimas and John N. Tsitsiklis, *Introduction to Linear Optimization*, Athena Scientific, 1997.

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

[3] Daniela P. de Farias and Ben Van Roy, *The Linear Programming Approach to Approximate Dynamic Programming*, Operations Research, Vol. 51, No. 6, 2003.

[4] Daniela P. de Farias and Ben Van Roy, *On the Existence of Fixed Points for Approximate Value Iteration and Temporal-Difference Learning*, Journal of Optimization Theory and Applications, Vol. 105, No. 3, June, 2000.

[5] Daniela P. de Farias and Ben Van Roy, *On Constraint Sampling for the Linear Programming Approach to Approximate Dynamic Programming*, Mathematics of Operations Research, August, Vol.29, No.3, August 2004.

[6] Daniela P. de Farias and Ben Van Roy, *A Cost-Shaping Linear Program for Average-Cost Approximate Dynamic Programming with Performance Guarantees*, Mathematics of Operations Research, Vol. 31, No. 3, pp. 597-620, 2006.

[7] Ben Van Roy, *Performance Loss Bounds for Approximate Value Iteration with State Aggregation*, Mathematics of Operations Research, Vol. 31, No. 2, pp. 234-244, 2006.

[8] Alan S. Manne, *Linear Programming and Sequential Decisions*, Management Science, Vol.6, 1960.

[9] Paul J. Schweitzer and Abraham Seidmann, *Generalized Polynomial Approximations in Markovian Decision Processes*, Math. Anal. App., Vol.110, No.2, 1985.

[10] Milos Hauskrecht and Branislav Kveton, *Linear Program Approximations for Factored Continuous-State Markov Decision Processes*, Advances in Neural Information Processing Systems, 2004.

[11] Csaba Szepesvari and Remi Munos, *Finite Time Bounds for Sampling Based Fitted Value Iteration*, International Conference on Machine Learning, 2005.