Constraint Relaxation in Approximate Linear Programs

Marek Petrik PETRIK@CS.UMASS.EDU
Shlomo Zilberstein SHLOMO@CS.UMASS.EDU
Department of Computer Science, University of Massachusetts Amherst, Amherst, MA 01003

Abstract

Approximate Linear Programming (ALP) is a reinforcement learning technique with nice theoretical properties, but it often performs poorly in practice. We identify some reasons for the poor quality of ALP solutions in problems where the approximation induces virtual loops. We then introduce two methods for improving solution quality. One method rolls out selected constraints of the ALP, guided by the dual information. The second method is a relaxation of the ALP, based on external penalty methods. The latter method is applicable in domains in which rolling out constraints is impractical. Both approaches show promising empirical results for simple benchmark problems as well as for a realistic blood inventory management problem.

1. Introduction

The Markov decision process (MDP) has been shown to provide an effective framework for planning under uncertainty. It offers a rich modeling language and a wide variety of solution techniques. However, the size of many real-world problems tends to be too large to be solvable using exact methods. A general approach to improve the scalability of MDP algorithms is through value function approximation and approximate dynamic programming (ADP) (Powell, 2007).

Approximate algorithms for solving MDPs are typically variations of the exact algorithms. Hence, they can be categorized as Approximate Policy Iteration (API), Approximate Value Iteration (AVI), Approximate Linear Programming (ALP) (Powell, 2007; Bertsekas & Tsitsiklis, 1996). Each one of these methods offers different advantages over the others.

ALP offers some significant theoretical advantages over other ADP algorithms (de Farias & Roy, 2003). Unlike other algorithms, ALP is guaranteed to converge to a solution with guaranteed bounds. These bounds guarantee that the solution will be as close to the optimal value function as its closest approximation in the basis. In addition, a solution of ALP is guaranteed to be an upper bound on the true value function and thus may be used as a heuristic function in search problems (Petrik & Zilberstein, 2008). However, ALP has been found to be less effective than ADP approaches (Guestrin et al., 2003). Therefore it is important to improve the practical solution quality of ALP while retaining the important favorable properties.

In this paper we propose modifications to ALP to improve the quality of the obtained solutions. First, we formally define the framework we use and show a simple example in which ALP performs poorly. We argue that the loss in quality is largely due to the presence of virtual loops in the approximate formulation. Then, we propose to improve solution quality by constraint roll-outs, based on the dual values. Next, we propose a modification of ALP that constrains the dual variables to improve solution quality. Finally, we experimentally verify the effectiveness of the approach on the mountain-car benchmark problem and a large-scale blood inventory management problem.

2. Framework

In this section, we first provide a formal definition of the terms we use. We also briefly describe the formulation of ALPs and the approximation errors involved. Finally, we show a simple example in which the solution quality of ALP is not acceptable.

A *Markov Decision Process* is a tuple $(\mathcal{S}, \mathcal{A}, P, r, \alpha)$, where $\mathcal{S}$ is the *finite* set of states, $\mathcal{A}$ is the *finite* set of actions. $P : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \mapsto [0, 1]$ is the transition function, where $P(s', s, a)$ represents the probability of transiting to state s' from state s, given action a.

Appearing in *Proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, 2009. Copyright 2009 by the author(s)/owner(s).

Function $r : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is the reward function, and $\alpha : \mathcal{S} \mapsto [0, 1]$ is the initial state distribution. To shorten the notation we assume an arbitrary ordering on the states: $s_1, s_2, \ldots$. Then we use P_a to denote the probabilistic transition matrix for action a, and r_a to denote the vector of rewards received for the transitions.

We focus on *linear value function approximation* for discounted infinite-horizon problems. In linear value function approximation the value function is represented as a linear combination of *nonlinear basis functions (vectors).* For each state s we define a row-vector $\phi(s)$ of features. The rows of the basis matrix M correspond to the $\phi(s)$, and the approximation space is generated by the columns of the matrix. The value function is represented as $v = Mx$, for any vector x.

It is well known that an infinite horizon discounted MDP problem may be formulated in terms of solving the following linear program:

$$\begin{aligned} \min_{v} \quad & \sum_{s \in \mathcal{S}} c(s) v(s) \\ \text{s.t.} \quad & v(s) \geq r(s,a) + \sum_{s' \in \mathcal{S}} \gamma P(s', s, a) v(s') \\ & \forall (s,a) \in (\mathcal{S}, \mathcal{A}) \end{aligned} \tag{2.1}$$

We use A as a shorthand notation for the constraint matrix and b for the right-hand side. The value c represents a distribution over the states, usually a uniform one. We assume in the remainder of the paper that $\sum_{s \in \mathcal{S}} c(s) = 1$. This linear program is too large to solve precisely, so it is often approximated by assuming that $v = Mx$ (de Farias & Roy, 2003). An approximate linear program is a tuple: $\mathcal{L} = (c, A, b, M)$, defined as:

$$\begin{aligned} \min_{x} \quad & c^{\mathsf{T}} Mx \\ \text{s.t.} \quad & AMx \geq b \\ & Mx \leq \|r\|_\infty / (1-\gamma) \mathbf{1} \end{aligned} \tag{2.2}$$

The standard assumption that guarantees the feasibility of the ALP is that $\mathbf{1} \in \operatorname{span}(M)$, where $\mathbf{1}$ is a vector of all ones. We implicitly assume this in the remainder of the paper. The constraint $Mx \leq \|r\|_\infty/(1-\gamma)\mathbf{1}$ ensures that the values of states are not too large. Notice that it does not limit the feasibility of the ALP, nor the quality of the solution. The number of constraints is reduced by sampling. In the following we use $\tilde{v} = Mx$ to denote the approximate value function from the optimal solution of an ALP. Its important property is that $\tilde{v} \geq v^*$. As a result, the objective of the linear program may be seen as a minimization of $\|\tilde{v} - v^*\|_{1,c}$, where $\|\cdot\|_{1,c}$ is a c-weighted L_1 norm.

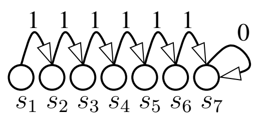

Figure 1. An example chain problem with deterministic transitions and reward denoted above the transitions.

The approximation error incurred in an ALP may be divided into three components: 1) representational error 2) sampling error and 3) transitional error. The representational error is due to the limited representation of the approximation basis M. Its value is $\epsilon_1 = \min_x \|Mx - v^*\|_\infty$. The sampling error is due to the limited number of samples. Finally the transitional error is due to the approximation method used, in this case ALP. Let $\hat{v}$ be the value for which the representational error is achieved. Then if $\tilde{v}$ is the solution of the ALP, then the transitional error is $\epsilon_2 = \|\tilde{v} - \hat{v}\|$ for some norm.

Bounds on the transitional error ϵ_2 have been extensively studied (de Farias & Roy, 2003). The basic bound on the approximation error is:

$$\epsilon_2 = \|\tilde{v} - v^*\|_{1,c} \leq \frac{2}{1-\gamma}\epsilon_1,$$

where ϵ_1 is the representational error. See (de Farias & Roy, 2003) for tighter bounds that rely on the basis having additional structure. While this bound is significantly tighter than bounds of other ADP algorithms, it is not sufficient to guarantee good solutions. Notice that this bound grows significantly as γ approaches 1. For example, for $\gamma = 0.99$, a commonly used value, $\epsilon_2 \leq 200\epsilon_1$. Such large approximation error is not acceptable in most practical applications.

Unfortunately, the looseness of the bounds is a property of the ALP solutions, not the bounds. It is possible to construct a problem in which the approximate value function has a significant error. To demonstrate this, consider the simple deterministic chain problem with a discount factor $\gamma = 0.9$, depicted in Figure 1. The optimal value function v^*, the closest approximation v_1 in terms of L_∞ norm, and the solution v_2 of an ALP are depicted in Figure 2. It is apparent that the approximation error in this problem is too large to make the value useful.

While it is possible to reduce the transitional error by requiring that an appropriate structure is present in the basis, this is not always practical. An example of such structure is the Lyapunov vectors (de Farias & Roy, 2003). The existence of such vectors in all but the simplest problem domains is typically hard to ensure.

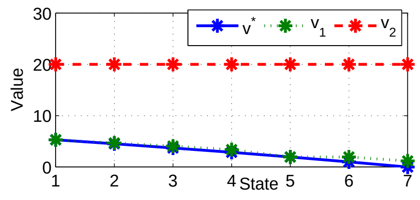

Figure 2. Approximation errors in approximate linear programming.

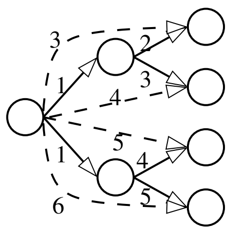

Figure 3. An example of 2-step expanded constraints (dashed) in a deterministic problem. The numbers next to the arcs represent rewards.

In this paper, we focus mostly on reducing the transitional error, ignoring the representational and sampling errors. This is reasonable, because the transitional error typically increases with decreasing sampling error. We, however, discuss the impact of the methods we present on the sampling requirements. Further, we universally assume that a generative model is available. This model can generate transitions from an arbitrary state taking arbitrary action. The approaches we propose are however applicable to problems without such a generator.

3. Constraint Roll-out

The transitional error would be zero, if the ALP constraints were in the form $v(s) \geq v^*(s)$ instead of having a constraint for each transition. This is however often impractical. We therefore consider a hybrid formulation, in which the constraints represent multi-step transitions – expanded, or rolled-out, constraints. An example of 2-step expanded constraints is depicted in Figure 3.

Definition 1. A t-step *expanded* ALP constraint for an action sequence $E = (a_1, a_2, \ldots, a_t) \in \mathcal{A}^t$ and state s_i has the following form:

$$v(s_i) \geq \sum_{s_j \in \mathcal{S}} \gamma^t P^E(s_j \mid s_i)\, v(s_j) + r^E(s_i).$$

The terms are defined as follows, using matrix notation for simplicity:

$$P^E(s_j \mid s_i) = \mathbf{1}_i^\mathsf{T} \left(\prod_{k=1}^{t} \gamma P_{a_k} \right) \mathbf{1}_j$$
$$r^E(s_i, E) = \mathbf{1}_i^\mathsf{T} \left(\sum_{l=1}^{t} \left(\prod_{k=1}^{l-1} \gamma P_{a_l} \right) r_{a_k} \right),$$

where $\mathbf{1}_i$ is the i-th unit vector. We denote such a constraint as $v(s_i) \geq \mathcal{C}(s_i, E)$. A *full* t-step *expanded* constraint is:

$$v(s_i) \geq \max_{E \in \mathcal{A}^t} \mathcal{C}(s_i, E).$$

For the sake of simplifying the analysis, we consider only *full* expanded constraints. That is, for every state we have constraints that correspond to all action sequences. For example, $a \geq \max\{b, c\}$ may be written as two constraints $a \geq b$ and $a \geq c$. Next, we describe the benefits of using expanded constraints and we address the difficulties later. The set of constraints for all states is denoted as $\mathcal{C}_t$

This idea is somewhat related to temporally extended actions, or *options*, commonly studied in reinforcement learning (Stolle & Precup, 2002). Options however typically serve a different purpose – to simplify and accelerate learning rather than to reduce the transitional error. In addition, unlike options, the expanded constraints have a fixed length.

Expanding the constraints with a larger horizon guarantees improvement in the approximation error bound, as the following generalization of the simple error bound shows.

Corollary 2. *Assume that* $\mathbf{1} \in \operatorname{span}(M)$*, and let* $\tilde{v}$ *be a solution of a* t*-step expanded approximate linear program. Then:* $\tilde{v} \geq v^*$ *and:*

$$\|\tilde{v} - v^*\|_{1,c} \leq \frac{2}{1-\gamma^t} \epsilon_1.$$

The corollary follows directly from the basic error bound on the modified MDP with actions that correspond to t-step actions in the original MDP. In fact, the approximation error is guaranteed to not increase with increased t.

Corollary 3. *Let* v *be a solution of an ALP with* t*-step expanded constraints and let* v' *be a solution of an ALP with* t'*-step expanded constraints such that* $t' = d * t$ *for some positive* $d \in \mathbb{Z}$*. Then* $\|v' - v^*\|_{1,c} \leq \|v - v^*\|_{1,c}$.

While the expanded formulation improves the solution quality, there are two main problems with its practical application. First, the number of constraints for each state is $|\mathcal{A}|^t$, which grows exponentially with t. This may however be manageable in problems with a small number of actions. We describe below how to address this problem by adaptively expanding only selected states in Section 3.2.

The second problem is that the t-step transition model is rarely available. Thus the probabilities of t-step transitions need to be obtained from sampled t-step execution traces. We show how to obtain the transition matrix using only a limited number of samples in Section 3.1.

3.1. Constraint Estimation

As described above, it may be difficult to determine an expanded constraint for a large t, even when the sequence of actions is fixed. In order to estimate such a constraint, we need to calculate the transition probabilities and rewards after taking t actions. Because a generative model typically can generate only one-step samples, the probabilities need to be calculated from sampled execution traces. We show in this section that the number of samples needed to obtain a solution is reasonably small.

An appealing property of many reinforcement learning problems is that the number of states that are targets of a transition is relatively small. This leads to a sparse transition matrix. With t-step transitions, however, the number of states to which it is possible to transit to may be large. To limit the number of necessary samples, we need a sampling bound independent of the size of the state space. The available samples for a fixed constraint, determined by an action sequence $a_1, \ldots, a_t$, are of the following form:

$$(s_1^j, \ldots, s_t^j)_{j=1}^q \quad (r_1^j, \ldots, r_t^j)_{j=1}^q.$$

The state s_i^j is the i-th state in the j-th sample, and r_i^j is the reward received in in the transition to s_i^j. The constants needed to specify the constraint are obtained as follows:

$$P_q\left(s_t \,|\, s_0\right) = \frac{1}{q}\sum_{j=1}^{q} \mathrm{I}\left\{s_t^j = s_t\right\}$$

$$R_q(s_0) = \frac{1}{q}\sum_{j=1}^{q}\sum_{k=1}^{t} \gamma^{k-1} r_k^j.$$

Here $\mathrm{I}\{\cdot\}$ denotes the indicator function. We denote the constraints that are obtained from the samples as $\mathcal{C}_t^q$ for t-step expanded constraints. These values are used to construct the empirical ALP, denoted as $\mathcal{L}_q$, based on q samples per constraint.

To derive a bound on the number of samples needed, we first analyze the sensitivity of the solution of the approximate linear program with regard to the change in the constraints. If the difference between the two linear programs is arbitrary, then even a small change in the constraints may lead to a large change in the solution. However, when the error is due to insufficient sampling, the ALP still retains the important property: $A\mathbf{1} \leq (1-\gamma)\mathbf{1}$. We exploit this property in the following lemma. We use $\|A\|_{1,\infty} = \max_i \|a_i\|_1$, that is, a maximum of L_1 norms on the matrix rows a_i.

Lemma 4. *Let $\mathcal{L}_1 = (c, A_1, b_1, M)$ and $\mathcal{L}_2 = (c, A_2, b_2, M)$ be two ALPs with optimal solutions v_1 and v_2 respectively. Also let $\epsilon_a = \|A_1 M - A_2 M\|_{1,\infty}$ and $\epsilon_b = \|b_1 - b_2\|_\infty$. Assuming that $A_1\mathbf{1} = A_2\mathbf{1} = (1-\gamma)\mathbf{1}$ then:*

$$\|\tilde{v}_1 - \tilde{v}_2\| \leq \frac{\epsilon_a \hat{x}}{1-\gamma} + \frac{\epsilon_b}{1-\gamma},$$

where $\hat{x} \geq |x(i)|$ for all i for both $\mathcal{L}_1$ and $\mathcal{L}_2$.

The lemma can be proved by constructing a solution feasible in $\mathcal{L}_2$ from the optimal solution v_1 by adding a sufficiently large constant vector.

Notice that in a t-step expanded ALP, we could use the factor $1/(1-\gamma^t)$ for tighter bounds. We ignore this for the sake of simplicity.

Lemma 4 shows that a small change in the constraints leads to only a small change in the resulting objective value. It remains to show now that given a sufficient number of samples, the probability that the empirical probability distribution significantly deviates from its true value is small.

Lemma 5. *Let $\mathcal{L} = (c, A, b, M)$ be the true ALP and let $\mathcal{L}_q = (c, A_q, b_q, M)$ be the sampled ALP. Then:*

$$\mathbf{P}\left[\|A_q M - AM\|_{1,\infty} \geq \epsilon_a\right] \leq nm \exp\left(-\frac{2q\epsilon_a^2 m^2}{\|M\|_\infty^2}\right)$$

$$\mathbf{P}\left[\|b - b_q\|_\infty \geq \epsilon_b\right] \leq n \exp\left(-\frac{2q\epsilon_b^2}{\|r\|_\infty^2}\right),$$

where n is the total number of constraints and m is the total number of state features.

The lemma follows directly from the union bound and Hoeffding's inequality. Putting the lemmas above together leads to the following theorem.

Theorem 6. *Let v_1 be the solution of the true ALP $\mathcal{L}_1$ and let v_2 be the solution of the sampled ALP $\mathcal{L}_q$. Then:*

$$\mathbf{P}\left[\|v_1 - v_2\|_{1,c} \geq \epsilon\right] \leq nm \exp\left(-\frac{2q\epsilon^2 m^2 (1-\gamma)^2}{\hat{x}^2}\right) + $$

$$+ n \exp\left(-\frac{2q\epsilon^2(1-\gamma)^2}{\|r\|_\infty^2}\right),$$

where $\hat{x} \geq |x(i)|$ for all i assuming that $\|M\|_\infty = 1$.

The size of $\hat{x}$ may be controlled by adding appropriate constraints to the ALP that limit it. For example, for piece-wise linear approximations, which we use, it is possible to constraint $\hat{x} = \|r\|_\infty/(1-\gamma)$ without loss of generality.

As with all distribution-free results, the practical use of the bounds above is limited, as they require 100s of samples in most practical circumstances, while good empirical results can be obtained with an order of 10 samples per transition. They however indicate that the number of samples required to obtain the multi-step constraints is independent of the number of states, and strengthening of the assumptions may yield practical bounds in the future.

3.2. Adaptive Selection of Expanded Constraints

Expanding all constraints may improve the solution quality, but at a steep computational cost. The number of constraints required per states scales exponentially with the number of steps for which the constraints are expanded. As a result, assuming that full constraints are added for many steps into the future is not realistic if it needs to be done for all states. In this section, we propose a scheme for selecting only some constraints for expansion.

To obtain the bounds on improvement achievable by expanding constraints, we compare solutions of an ALP with and without some constraints expanded, ignoring the approximation basis M without loss of generality. Let $A_1 v \geq b_1$ represent constraints that are *not* expanded and let $A_2 v \geq b_2$ be the expanded constraints. Then, let $\bar{A}_1 v \geq \bar{b}_1$ be a fully expanded version of the constraint $A_1 v \geq b_1$. This leads to two linear programs:

$$\begin{aligned} \min_v \quad & c^\mathsf{T} v \\ \text{s.t.} \quad & A_1 v \geq b_1 \quad A_2 v \geq b_2 \end{aligned} \tag{3.1}$$

$$\begin{aligned} \min_v \quad & c^\mathsf{T} v \\ \text{s.t.} \quad & \bar{A}_1 v \geq \bar{b}_1 \quad A_2 v \geq b_2 \end{aligned} \tag{3.2}$$

We can now state the following proposition. We use $[x]_+ = \max\{x, \mathbf{0}\}$.

Proposition 7. *Let v_1 and $\bar{v}_1$ be the* optimal *solutions of Eq. (3.1) and Eq. (3.2) respectively. Let λ_1 and λ_2 be the Lagrange multipliers that respectively correspond to constraints $A_1 v_1 \geq b_1$ and $A_2 v_1 \geq b_2$ in Eq. (3.1). Then the improvement from expanding constraints $A_1 v \geq b_1$ is at most:*

$$\|v_1 - v^*\|_{1,c} - \|\bar{v}_1 - v^*\|_{1,c} \leq \frac{\|\,[Av_1 - b_1]_+\,\|_\infty}{1-\gamma} \|\lambda_1^\mathsf{T} A_1\|_1.$$

Proof. First, let $v_2 \geq v^*$ be the optimal solution of:

$$\begin{aligned} \min_v \quad & c^\mathsf{T} v \\ \text{s.t.} \quad & \bar{A}_1 v \geq \bar{b}_1 \end{aligned}$$

From the optimality of v_1, the dual feasibility, and the complementary slackness we conclude:

$$c = A_1^\mathsf{T} \lambda_1 + A_2^\mathsf{T} \lambda_2$$
$$\lambda_2^\mathsf{T} A_2 v_1 = \lambda_2^\mathsf{T} b_2$$

The feasibility of v_2 and $\lambda_2 \geq \mathbf{0}$ imply that:

$$\lambda_2^\mathsf{T} A_2 v_1 - \lambda_2^\mathsf{T} A_2 v_2 = \lambda_2^\mathsf{T} b_2 - \lambda_2^\mathsf{T} A_2 v_2 = \lambda_2^\mathsf{T}(b_2 - A_2 v_2) \leq 0,$$

The proposition then follows from Corollary 2, and the trivial version of Holder's inequality:

$$\begin{aligned} & \|v_1 - v^*\|_{1,c} - \|\bar{v}_1 - v^*\|_{1,c} = \\ & = (\lambda_1^\mathsf{T} A_1 + \lambda_2^\mathsf{T} A_2)(v_1 - v_2) \\ & \leq \lambda_1^\mathsf{T} A_1 (v_1 - v_2) \\ & \leq \|\lambda_1^\mathsf{T} A_1\|_1 \|v_1 - v_2\|_\infty \leq \|\lambda_1^\mathsf{T} A_1\|_1 \|v_1 - v^*\|_\infty. \end{aligned}$$

The proposition follows from the Bellman residual bound on the value function approximation. □

The proposition shows that the dual variables (or Lagrange multipliers) may be used to bound the potential improvement in the approximation that can be gained from expanding some of the constraints. In the trivial case, the proposition states that expanding constraints for which $\lambda = 0$ has no effect on the solution. Thus, it is sufficient to obtain a solution of the linear program in which the sum of the Lagrange multipliers of unexpanded constraints is sufficiently small. A greedy algorithm that accomplishes this is depicted in Algorithm 1. Note that after expanding a constraint, the previous solution of the ALP may be used as the starting point, since it is a feasible solution (Corollary 3).

Algorithm 1: Iterative Expansion Algorithm

```
1 while $\|v_1 - v^*\|_{1,c} - \|\bar{v}_1 - v^*\|_{1,c} \geq \epsilon$ do
2     $v \leftarrow M \arg\min_{\{x \mid AMx \geq b\}} c^\mathsf{T} Mx$ ;
3     $\lambda_i \leftarrow$ dual variables ;
4     $a_1 \ldots a_d \leftarrow \arg\max_{a_i} \|\lambda_i a_i\|_1$ ;
5     Expand constraints $a_1 \ldots a_d$
```

We showed in this section a practical approach for obtaining expanded constraints in stochastic domains, as well as a method for selecting a subset of constraints to expand.

4. Relaxed Linear Program

In this section, we describe a method for eliminating the virtual loops without requiring constraint expansion. This method is generally inferior to constraint expansion in terms of solution quality, since it cannot use the extra multi-step transitions. However, the method is useful in problems in which rolling out constraints is impractical and in which it is possible to obtain a good approximate value function while violating only a few constraints.

As the example in Figure 2 shows, a single constraint in an ALP may degrade the quality of the whole solution. This can be addressed by allowing a small violation of a small number of constraints. One possibility of relaxing the constraints is to use: $Av \geq b - \epsilon\mathbf{1}$ for some small ϵ. This will however only lower the value function with little effect on quality. We take an alternative approach and solve the following optimization problem:

$$\min_v c^\mathsf{T} v + d^\mathsf{T} \left[b - Av\right]_+ , \tag{4.1}$$

for some vector $d \geq \mathbf{0}$. We implicitly assume that $v \in \operatorname{span}(M)$. The parameter d determines the trade-off between guaranteeing that $\tilde{v}$ is an upper bound on v^* and the closeness of the approximation. In fact, the choice of this optimization problem is motivated by its dual, which is:

$$\begin{aligned} \max_y \quad & b^\mathsf{T} y \\ \text{s.t.} \quad & A^\mathsf{T} y = c \quad y \geq \mathbf{0} \quad y \leq d \end{aligned} \tag{4.2}$$

The optimal dual solution in the exact formulation is denoted as y^* and corresponds to: $y^*(s) = \mathbf{E}_c\left[\sum_{i=0}^{\infty} \gamma^i \, \mathrm{I}\left\{S_i = s\right\}\right]$, where $S_0, S_1 \ldots$ represent the states of the MDP using the optimal policy and the initial distribution c in corresponding stages. The vector d then correspond to upper bounds on the dual values y of the linear program in Eq. (4.2). Therefore, the approach may be seen as a regularization of the linear program dual values.

The relaxed linear program formulation can be used when violating a small number of constraints significantly improves the solution. The formulation then automatically selects the appropriate constraints that need to be violated. The following proposition shows a bound on the total weight of the violated constraints.

Proposition 8. *Assume that* $\mathbf{1} \in \operatorname{span}(M)$*. Let* I_V *be the set of violated constraints and let* I_A *be the set of active constraints by the optimal solution of the relaxed ALP. Then* $d(I_V) \leq 1/(1-\gamma)$ *and* $d(I_A) + d(I_V) \geq 1/(1-\gamma)$*, where* $d(\cdot)$ *denotes the sum of the weights defined by* d *on the selected constraints.*

The proposition implies that setting $d > \frac{1}{(k+1)(1-\gamma)}\mathbf{1}$ guarantees that at most k constraints are violated. The relaxation does not guarantee an improvement on the ALP solution. It is possible to construct an instance in which the solution can be improved only by violating all constraints.

While it number of violated constraints is important, it is also necessary to bound the scale of the violation. The following theorem shows the bound on the constraint violation in terms of the Bellman error violation. It also provides guidance in selecting the constraint violation penalty d.

Theorem 9. *Let* $\min_{v \in \operatorname{span}(M)} \|v - v^*\|_\infty \leq \epsilon$ *with a minimizer* $\hat{v}$*, and let* $\tilde{v}$ *be the optimal solution of the* relaxed *linear program with weight* $d = y^* + \Delta d$*, where* $\Delta d \geq \mathbf{0}$*. Then:*

$$\|[b - A\tilde{v}]_+\|_{1,\Delta d} \leq (2 + \Delta d^\mathsf{T}\mathbf{1})\epsilon.$$

Further if the weight of the constraints violated by $\hat{v}$ *is less than* $1/(1-\gamma)$*, then* $\|[\tilde{v} - v^*]_+\|_{1,c} \leq \epsilon$*.*

Notice that the first inequality provides a lower bound on the approximate value function, while the second one provides an upper bound. The incompatibility of the bounds is due to an inherent problem with the ALP formulation that minimizes a weighted L_1, which has not been shown to provide tight error bounds on the performance.

The proof of Theorem 9 differs from the standard ALP proofs, since the approximate value function is not guaranteed to be an upper bound on the true value function v^*. It is instead based on the Lagrange function formulation of the linear program in Eq. (2.1). In particular, it relies on the fact that v^* and y^* are the optimal solutions of $\max_y \min_v c^\mathsf{T} v + y^\mathsf{T}(b - Av)$.

An important property of the formulation is that given a sufficiently large vector d, if the optimal value function v^* is representable in the approximation space d, then it will be also the optimal solution of the relaxed linear program.

Proposition 10. *Assuming that* $\mathbf{1} \in \operatorname{span}(M)$ *and* $d > \frac{1}{1-\gamma}\mathbf{1}$ *then the optimal solutions of Eq. (3.1) and Eq. (4.1) are identical.*

The proposition follows from the constraints of the dual ALP formulation.

The bounds above indicate that the values d should be upper bounds on the *optimal* visitation frequencies y^* in the MDP, which is the optimal solution of Eq. (4.2). One way of obtaining the d is to choose a value that optimizes the empirical performance, or

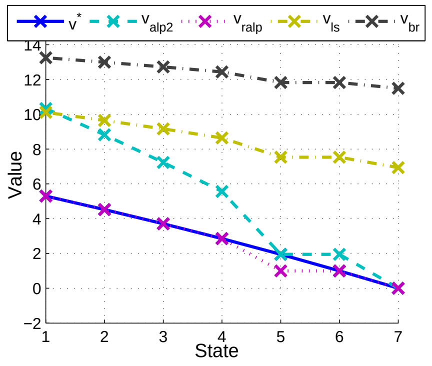

Figure 4. Benchmark results for the chain problem.

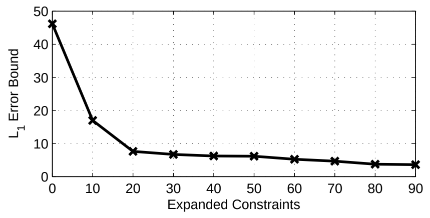

Figure 5. Bound on L_1 approximation error with regard to the number of expanded constraints on the mountain-car problem.

the value function bound. In many problems, it may be possible to use the structure to derive bounds on y^*. For example, if it is impossible to return to the same state s of the MDP in less than k steps, then $y^*(s) \leq 1/(1-\gamma^k)$. When the probability of returning is small in the first few steps, the following bound can be used.

Proposition 11. *Let p_j be the probability of returning for the first time to state s_1 in the j-th step, and assume that $p_{\hat{j}} = 1$. Let $\theta_1 = p_1$ and $\theta_j = p_j \left(1 - \sum_{k=1}^{j-1} \theta_k\right)$, then $y^*(s) = 1/\left(1 - \sum_{j=1}^{\hat{j}} \gamma^j \theta_j\right)$. Given that p_j's are upper bounds on the probabilities, this is an upper bound on $y^*(s)$.*

The result can be shown by induction on $\hat{j}$. When the probabilities are not available, they can be estimated from the domain samples.

5. Experimental Results

We experimentally evaluate the proposed approaches on three problems of increasing complexity. We first demonstrate the soundness of the main ideas on the simple chain problem described in the introduction. Then we evaluate the approach on a modified version of the mountain-car problem, a standard benchmark in reinforcement learning. Finally, we describe an application of the method to a blood-management problem, a complex multi-attribute optimization problem.

The empirical results for the chain problem are depicted in Figure 4. Here v_{br} represents the Bellman residual value function and v_{ls} is least squares minimization value function (Lagoudakis & Parr, 2003). The constraints in the problem are rolled out with 2 steps, with the value denoted by v_{alp2}. The values d in the relaxed ALP are 1, except in the last state in which it is 10. These values are upper bounds, since all states except the last one are transient. The value function of the relaxed ALP is denoted by v_{ralp}. These results show that at least in the simple problem, constraint roll-outs and relaxed formulation perform well.

In the mountain-car benchmark an underpowered car needs to climb a hill (Sutton & Barto, 1998). To do so, it first needs to back up to an opposite hill to gain sufficient momentum. In our modified version of the mountain car, the reward of 1 is received just before the top of the hill. In the traditional formulation, the task stops as soon as the reward is received. In our modified problem, the car continues for an extra 0.1 distance, but cannot receive the reward again. We used $\gamma = 0.99$.

We used piece-wise linear value function approximation, which is particularly suitable for use with ALP, because it has no constant continuous regions. The constraints were based on 3000 uniformly generated states with all 3 actions. The approximation error of the solution with regard to number of 10-step expanded constraints using Algorithm 1 is shown in Figure 5. The relaxed linear program was evaluated on this problem with $d = 0.6 * \mathbf{1}$, which is an upper bound on the dual value when all states are transient. The average $\|\cdot\|_{1,c}$ error over 10 runs was 4.474, with only 0.35% of constraints violated. The average discounted returns of the ALP policy was 0.338, of the ALP with 90 expanded constraints was 0.438, and of the relaxed ALP formulation was 0.42. Our ALP formulations outperformed our implementation of LSPI on this problem, which achieved average return of 0.38.

The blood inventory management problem, described for example in (Powell, 2007) concerns managing a blood supply inventory and determining the optimal blood-type substitution. This is a multi-dimensional resource management problem. The number of actions in the problem is essentially infinite, and the greedy step needs to be solved using a linear program. We also use in this problem a simple piece-wise linear approximation. Constraint expansion in this problem is impractical due to the complex action space and the large

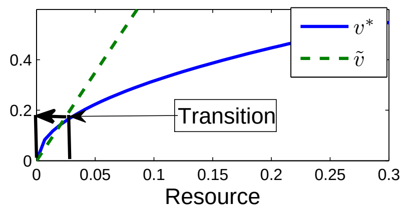

Figure 6. Illustration of the value function in the inventory management problem.

sampling error encountered in constraint estimation. A simple greedy solution in this problem achieves return of 70331, standard ALP solution achieves 19108, and the bound on the optimal solution is 94169. Relaxed ALP solution, with the assumption that a loop on any state is at least 20 states long, improves on the greedy solution with value 87055. We do not describe the details of this application due to lack of space. This is a complex optimization problem and its solution requires a number of trade-offs, which significantly influence the solution quality.

ALP performs poorly in the blood inventory management problem because the solution significantly overestimates the value of the resources. While the optimal value function in this problem is not known, we hypothesize that the function is concave. When approximating a concave function by a linear one, the ALP formulation can be seen as an approximation of the upper bound on the functions derivative. This leads to a large approximation error, as demonstrated in Figure 6. The solution of the relaxed formulation will however violate some of the constraints close to 0.

The impact of the proposed methods on the computational time depends on the tradeoffs involved in the particular domain. Solving a relaxed ALP may be in fact easier than solving the original ALP. In the case of constraint expansion, the penalty depends on the type of samples available and the cost involved in gathering them. For example, the constraint expansion method in impractical in in the Blood inventory management due to the high computational cost of gathering samples.

6. Conclusion

We identified an important drawback of approximate linear programming that may significantly impact its performance. In particular, we showed that a single constraint in an approximate linear program may significantly impact the solution quality, and proposed two methods to remedy the issue. The first one is based on constraint roll-outs, guided by the dual variables. The second one is based on allowing violation of a limited number of constraints. This approach does not ensure that the approximate solution is an upper bound on the optimal value function, but it significantly decreases the sensitivity of the approach to individual constraints. Our empirical results show that these techniques can significantly improve solution quality in comparison to plain ALP.

An important issue to be addressed future work is a deeper study of the modified ALP formulations with regard to the constraint sampling error. It is likely that the approach may also reduce the bounds on the sampling error, since it reduces the importance of the individual constraints. These results will hopefully help to establish ALP as one of leading solution methods for large stochastic optimization problems.

Acknowledgements

This work was supported by the Air Force Office of Scientific Research under agreement number FA9550-08-1-0171.

References

Bertsekas, D. P., & Tsitsiklis, J. N. (1996). *Neuro-dynamic programming*. Athena Scientific.

de Farias, D. P., & Roy, B. V. (2003). The linear programming approach to approximate dynamic programming. *Operations Research*, *51*, 850–856.

Guestrin, C., Koller, D., Parr, R., & Venkataraman, S. (2003). Efficient solution algorithms for factored MDPs. *Journal of Artificial Intelligence Research*, *19*, 399–468.

Lagoudakis, M. G., & Parr, R. (2003). Least-squares policy iteration. *Journal of Machine Learning Research*, *4*, 1107–1149.

Petrik, M., & Zilberstein, S. (2008). Learning heuristic functions through approximate linear programming. *International Conference on Automated Planning and Scheduling (ICAPS)* (pp. 248–255).

Powell, W. B. (2007). *Approximate dynamic programming*. Wiley-Interscience.

Stolle, M., & Precup, D. (2002). Learning options in reinforcement learning. *Lecture Notes in Computer Science*, *2371*, 212–223.

Sutton, R. S., & Barto, A. (1998). *Reinforcement learning*. MIT Press.