Bounding the Suboptimality of Reusing Subproblems

Michael Bowling   Manuela Veloso
mhb@cs.cmu.edu   veloso@cs.cmu.edu

Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213-3890

Abstract

We are interested in the problem of determining a course of action to achieve a desired objective in a non-deterministic environment. Markov decision processes (MDPs) provide a framework for representing this action selection problem, and there are a number of algorithms that learn optimal policies within this formulation. This framework has also been used to study state space abstraction, problem decomposition, and policy reuse. These techniques sacrifice optimality of their solution for improved learning speed. In this paper we examine the suboptimality of reusing policies that are solutions to subproblems. This is done within a restricted class of MDPs, namely those where non-zero reward is received only upon reaching a goal state. We introduce the definition of a subproblem within this class and provide motivation for how reuse of subproblem solutions can speed up learning. The contribution of this paper is the derivation of a tight bound on the loss in optimality from this reuse. We examine a bound that is based on Bellman error, which applies to all MDPs, but does not provide us with a tight bound. We contribute our own theoretical result that gives an empirically tight bound on this suboptimality.

1 Introduction

Selecting a course of action in a complex environment is a difficult task. It has been examined in a number of frameworks, including classical planning, probabilistic planning, and Markov decision processes. In all of these formulations it is a goal to solve difficult problems through simplifying techniques, such as finding and exploiting levels of state abstraction [3, 5], decomposing into smaller subproblems [2], reusing previously discovered solutions [8, 7]. All of these techniques make locally optimal "decisions" and therefore will almost certainly find a suboptimal solution. This is the tradeoff: sacrificing optimality of the solution for efficiency in finding it.

In this paper we examine this tradeoff in the context of Markov decision processes (MDPs). We assume the reader has a basic understanding of MDPs [4], though we will give a brief overview emphasizing the theory that is needed for this paper. We will formalize a notion of a subproblem and what it means to use a subproblem. We will then provide examples of the use of subproblem solutions, and also the tradeoff between suboptimality and efficiency. The focus of the paper is the introduction of a quantitative bound on this loss of optimality. We will examine a bound based on Bellman error, which does not give us a tight bound. We contribute our own theoretical result and show empirical results that this bound is tight using our presented examples.

2 Markov Decision Processes

A *Markov Decision Process*, $\mathcal{P}$ is a tuple (S, A, T, R), where S is a set of states, A is a set of actions, T is a transition function $S \times A \times S \rightarrow [0, 1]$, and R is a reward function $S \times A \rightarrow \mathbb{R}$. The transition function defines a probability distribution on next states as a function of the current state and the agent's action. The reward function defines the reward received when selecting an action from the given state. The goal is to find a policy, $\pi : S \rightarrow A$, which determines the agent's actions so as to maximize discounted future reward, with discount factor γ.

For any policy π the value of a state, V^π, is the expected discounted future reward of following the policy π from that state. We say π^* is the optimal policy and V^* is the value function of the optimal policy. A standard result from the theory of dynamic programming [1] states that the optimal value function is the unique solution to the simultaneous equations for all states, s,

$$V^*(s) = \max_{a \in A} \left(R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V^*(s') \right)$$

These equations are called the Bellman equations. For any value function, V, not necessarily optimal, we can define the *Bellman residual* or just *Bellman error* as,

$$\max_{s \in S} \left| V(s) - \max_{a \in A} \left(R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V(s') \right) \right| .$$

So the optimal value function has a Bellman error of 0, and reinforcement learning techniques, in general, can be viewed as trying to minimize the Bellman error.

In this paper we will examine a restricted class of MDPs, called *goal state MDPs*. MDPs in this class have a set of goal states. Reaching a goal state ends the trial, so all actions from a goal state simply transition to the same state with zero reward. Other transitions into goal states receive a reward of 1, and all other transitions receive a reward of 0. Using this class we can replace the reward function R with the goal set $G \subseteq S$ since this uniquely defines the reward function. Throughout this paper we will refer to MDPs in this class as the tuple (S, A, T, G).

3 Subproblems

Before we can proceed we need to formalize the notion of a subproblem within the frameworks of our restricted class of MDPs. We also need to define what it means to *use* a subproblem. These notions hinge on the definition of the boundary of a set of states.

Definition 1 *For a set of states, S, let the* boundary *of S, denoted $B(S)$, be the set of all states not in S with a non-zero transition from a state in S.*

In other words the boundary of S are all the states not in S that are reachable from S in a single time step.

We can now define our notion of a subproblem.

Definition 2 *A subproblem, $\mathcal{S}$, of an MDP, $\mathcal{P} = (S, A, T, G)$, is a pair (S', G') satisfying:*

- $S', G' \subseteq S$.
- $S' \cap G = \emptyset$ *and* $B(S') \cap G = \emptyset$.
- $G' \subseteq B(S')$.

This definition induces a new MDP over the states S'. The state transitions remain the same as in $\mathcal{P}$, but the reward function is zero for all transitions except those entering states in G', where the reward is one.

Given this notion of a subproblem we want to investigate the effects of using a subproblem solution. So we define a formal notion of *using* a subproblem.

Definition 3 *For a subproblem, $\mathcal{S} = (S', G')$, of a problem $\mathcal{P}$, we say that a policy, π for $\mathcal{P}$* uses *$\mathcal{S}$ if and only if π restricted to states in S' is optimal for the subproblem, $\mathcal{S}$. Additionally, the optimal policy for $\mathcal{P}$ that* uses *$\mathcal{S}$ is denoted $\pi^*_{\mathcal{S}}$.*

4 Examples

Given these definitions of subproblem and how it is used, we want to examine the tradeoffs of reusing subproblem solutions. We explored this in the context of some simple stochastic maze problems, which are within our class of goal state MDPs.

4.1 Hallway Example

Figure 1 depicts a simple grid environment that was first investigated in [7] within the context of sharing policies. The agent has actions that move it in any of the eight grid directions, but with probability 0.1 its action is ignored and it is moved to a random neighbor state. There is a single goal state labeled ’G’ in the diagram. The agent receives a reward of 1 for a transition into the goal state, and a reward of 0 otherwise. The agent is trying to maximize its discounted future reward with a discount factor, γ, of 0.9.

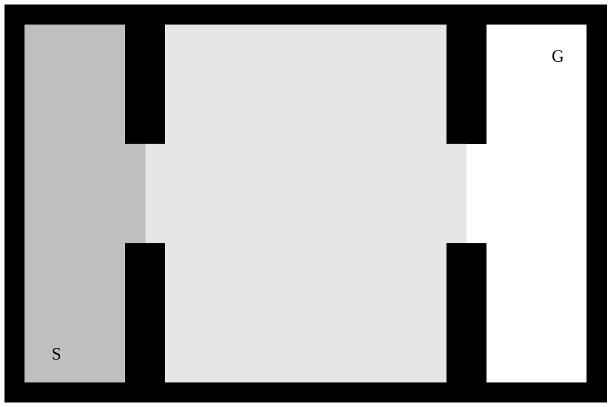

Figure 1: Hallway example. Each grid position represents a state. ‘S’ denotes the start state, and ‘G’ the goal state. The shaded regions mark off two different subproblems.

The shaded regions in the diagram denote subproblems. For example the subproblem of exiting the left room has the dark shaded region as its state space, S'. The five states in the eastern half of the doorway into the middle room are the boundary of S', and in this case $G' = B(S')$. Another subproblem is exiting both the left and middle rooms. In this case the states, S', would be both shaded regions, and the goal states would be the five eastern states of the doorway to the room on the right.

We explored how using the solution to these subproblems might speed learning of the complete problem. We fixed the policy to always select the subproblem’s optimal action while within the subproblem’s state space (i.e. the shaded regions.) We then proceed to learn the optimal policy given this constraint.

The results of this learning can be seen in figure 2. When we use the subproblem of exiting the left and mid-

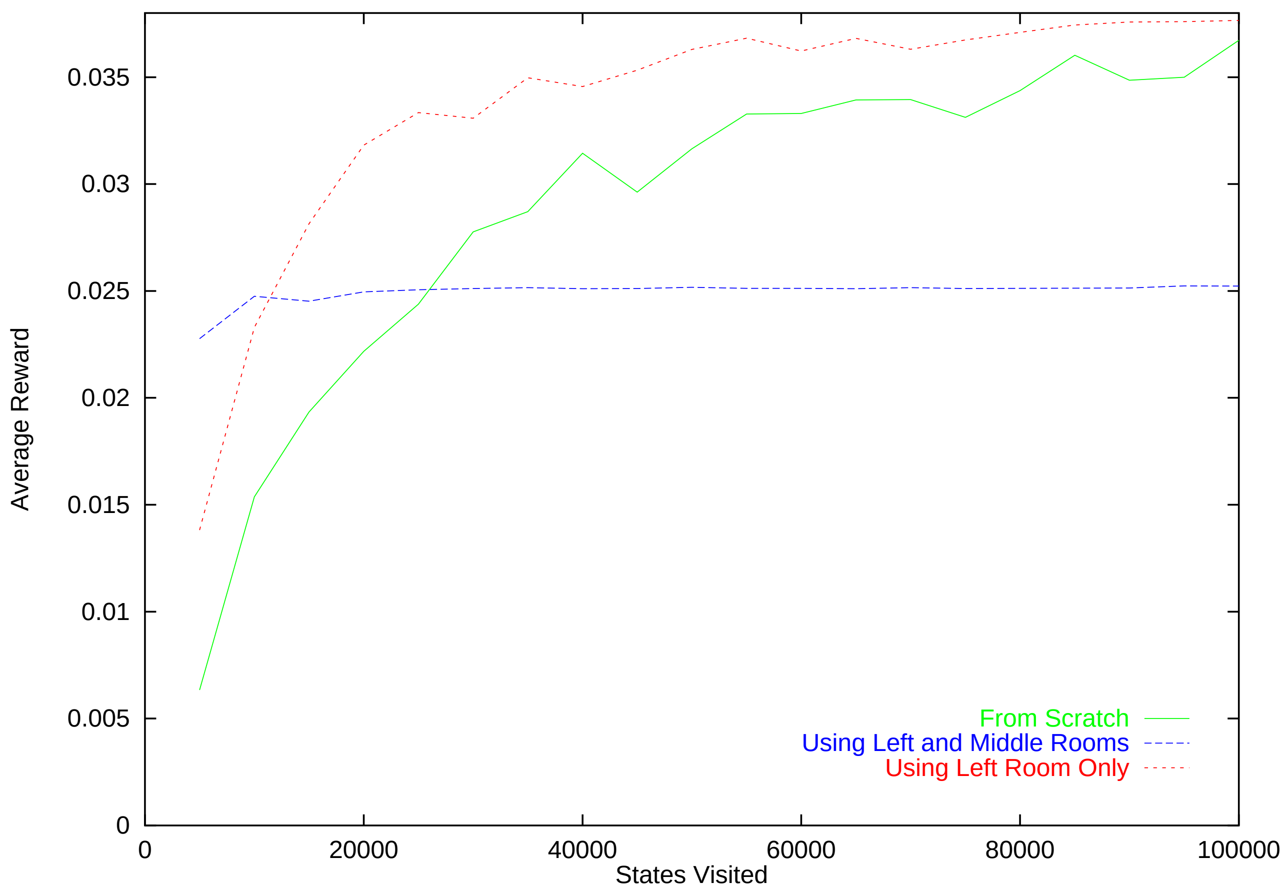

Figure 2: Results from the hallway example.

dle rooms, we have a huge boost in learning speed. After visiting 5000 states, it is doing approximately six times better than learning from scratch. Notice that this sacrifices convergence to the optimal solution. The more conservative approach of using the subproblem of exiting just the left room has a more conservative learning improvement. After visiting 5000 it is doing almost three times better than learning from scratch. Although the improvement isn't as pronounced, it does not appear that it converges to a less than optimal solution.

4.2 Doorways Example

A second example that we explored consisted of a similar grid world but with multiple doorways. This is depicted in figure 3. The actions and transitions remain the same as was described for the hallways example. Like that example, this also has intuitive subproblems, such as reaching a subset of the three doorways. We explored this domain using a number of these subproblems. Results are shown in figure 4 for learning when using the subproblem of reaching the right doorway, reaching either the right or middle doorway, and learning from scratch.

The results are very similar to the previous example.

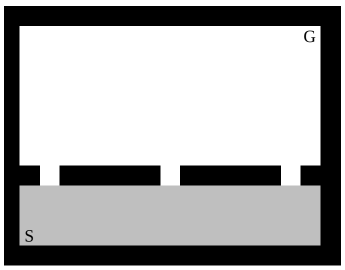

Figure 3: Doorways example. 'S' denotes the start state, and 'G' the goal state. The shaded region marks off the state space of the subproblem.

Subproblems speed learning but are likely to converge on a suboptimal solution. These results also motivate our goal, to be able to make some claim as to the amount of optimality sacrificed in order to gain the improved learning speed.

5 Bounding the Suboptimality

In order to better judge this tradeoff it is important to know the suboptimality of the solution obtained. For-

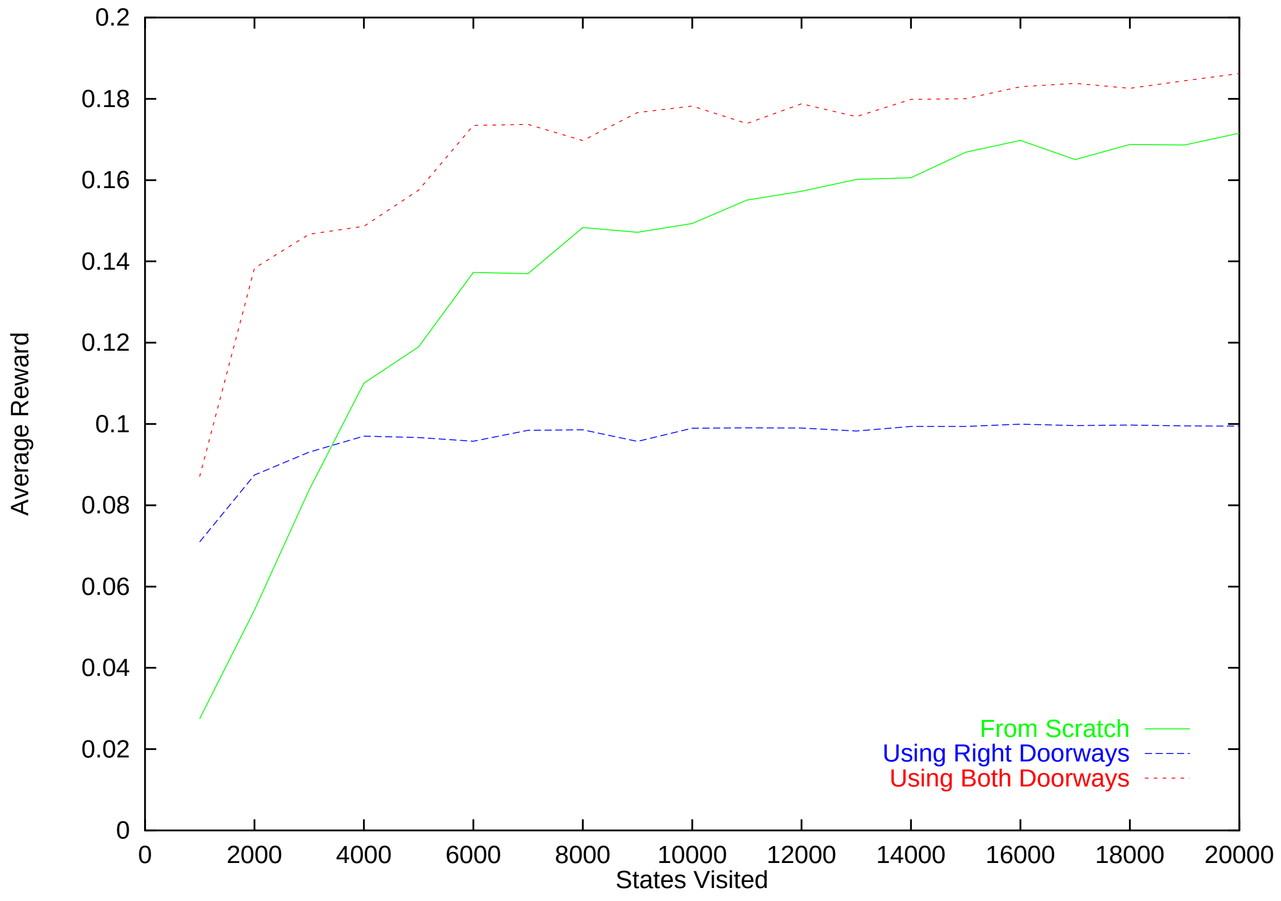

Figure 4: Results from the doorways example.

mally, we want to find a bound on the suboptimality of $\pi^*_{\mathcal{S}}$. This would provide us with a quantitative measure of the trade-off of using $\mathcal{S}$ as a subproblem. Before we discuss possible bounds we need a notion of suboptimality.

Definition 4 *A policy, π, is α-*optimal *if and only if* $\forall s\ V^{\pi}(s) \geq \alpha V^*(s)$.

Using this definition, α values are in the range $[0,1]$, where large values are closer to the optimal solution. A summary of the optimality for our examples (figures 2 and 4) is given in the first column of table 1. These values were computed by taking the minimum over all states of the ratio between the value of the state when using the subproblem and the optimal value of the state. So we can quantify our goal as finding a guaranteed lower bound for these values without requiring knowledge of the value of the optimal policy or value.

We will first examine a bound based on Bellman error and give some empirical results on our examples that show that this bound is not satisfactorily tight. We will then introduce a new bound based on boundary values and show its results on these same examples.

5.1 Bound Using Bellman Error

The use of the Bellman error to bound the suboptimality of a policy was developed by Williams and Baird [9]. Bellman error, as mentioned earlier, is simply the maximum error in the Bellman equations, and the optimal policy would have a Bellman error of zero. They proved the following for all MDPs,

Theorem 1 *For any value function, V, with Bellman error, ϵ. Let π be a policy greedy for V. Then for all states, x,*

$$V^{\pi}(x) \geq V^*(x) - \frac{2\gamma\epsilon}{1-\gamma}$$

Qualitatively, this means we can bound the suboptimality of a policy by examining the Bellman error of a value function that induces that policy.

We can use this to compute a bound by simply calculating the Bellman error of $V^*_{\mathcal{S}}$. We define a value function that induces $\pi^*_{\mathcal{S}}$ and satisfies the Bellman equations for states in S (because we are using an optimal solution to the subproblem) and also for states outside of S (because we are acting optimally outside of S.) Then the only states with a non-zero Bellman error would be on the boundary of S.

Example	Subproblem	Optimality		
		Actual	Bellman Bound	Boundary Bound
Hallway	Left and Middle Rooms	0.66	0.01	0.65
	Left Room	0.99	0.22	0.98
Doorways	Right and Middle Doorway	0.99	0.38	0.77
	Right Doorway	0.52	0.04	0.16

Table 1: Comparison of the suboptimality bounds computed from the Bellman error and from our method, against the actual suboptimality of using the subproblem. Larger values signify a more optimal solution (or a better guarantee.)

Results from the Examples. Table 1 gives the results of using this bound in the subproblems in section 4. These values were computed by finding the optimal solution using the subproblem, calculating the Bellman error for that value function, and applying the equation. The results are quite disappointing. Even for subproblems that are nearly optimal (e.g. the left and middle rooms of the hallway problem) result in a weak constraint on the suboptimality. The computed bound does not help identify when a subproblem results in a good solution.

5.2 Bound Using Boundary Values

The following two definitions play a crucial role in our bound on suboptimality. The first provides a quantitative measure of the stochasticity of a problem. The second provides a measure for comparing values of sets of states.

Definition 5 *Let ρ_s be the probability that following the optimal policy at s will transition to a state, s', where $V^*(s') < V^*(s)$. Let $\rho = \max_{s \in S} \rho_s$. This captures the stochasticity of the problem, where $\rho = 0$ for a deterministic problem.*

Definition 6 *For a set of states S_1 and S_2, and a policy π, we say $S_1 \geq^{\pi}_{\alpha} S_2$ if and only if,*

$$\forall s_1 \in S_1, s_2 \in S_2 \qquad V^{\pi}(s_1) \geq \alpha V^{\pi}(s_2)$$

If $S \leq^{\pi}_{\alpha} S$ then S is α -equivalent with respect to π, i.e. the values of all the states in the set are within α.

The following theorem provides a bound on the suboptimality of using a subproblem where the goal states for the subproblem are its entire boundary.

Theorem 2 *Given a subproblem $\mathcal{S} = (S', B(S'))$ of $\mathcal{P}$, if $B(S')$ is α-equivalent with respect to $\pi^*_{\mathcal{S}}$, then,*

$$\pi^*_{\mathcal{S}} \text{ is } \alpha\beta\text{-optimal for } \mathcal{P},$$

where $\beta = \frac{1-\sigma}{1-\sigma\alpha}$ and $\sigma = \frac{\rho\gamma^2}{1-(1-\rho)\gamma}$.

The σ value may appear very cryptic but it is simply the discounted probability of ever reaching a state of lower value. The value emerges from the proof which is presented in appendix A.

We have attempted to generalize the theorem to all subproblems with the following conjecture,

Conjecture 1 *Given a subproblem $\mathcal{S} = (S', G')$ of $\mathcal{P}$, let $\bar{G}' = B(S') \backslash G'$. If $G' \geq^{\pi^*_{\mathcal{S}}}_{\alpha_1} (S \cup G')$ and $\bar{G}' \leq^{\pi^*_{\mathcal{S}}}_{\alpha_2} (S \cup \bar{G}')$, then*

$$\pi_{\mathcal{S}} \text{ is } \alpha_1\alpha_2\beta\text{-optimal for } \mathcal{P},$$

where $\beta = \frac{1-\sigma}{1-\sigma\alpha_1\alpha_2}$ and $\sigma = \frac{\rho\gamma^2}{1-(1-\rho)\gamma}$.

Theorem 3 *Conjecture 1 is true if $\mathcal{P}$ is deterministic (i.e. if $\rho = 0$.)*

Theorem 3 proves the conjecture true for deterministic MDPs and the β factor should provide the necessary adjustment for the non-determinism as it did in theorem 2. The proof for theorem 3 is given in appendix B.

Results from the Examples. Table 1 gives the results of using this bound in our example subproblems. The values were computed much like those for the Bellman error; an optimal solution was learned using the subproblem, the α values were calculated, and the bound computed. The results show a reasonably tight bound, especially for the single exit subproblems. Thought it's not as tight on the multiple exit subproblems it is still much improved over the bound computed from Bellman error.

This bound provides considerable information as to the loss in optimality of using a subproblem. In addition to this guarantee on the resulting policy it also helps to identify a possibly better subproblem. This is done by examining the α values. For example, in the doorways problem using the right doorway, the guaranteed optimality is fairly weak. But, by examining the α values it becomes obvious that this bound could be improved if the middle door was in the subproblem's goal states. By using the "new" subproblem we achieve a better policy and a better bound on its suboptimality.

6 Related Work

The definitive performance bound for *general* MDPs is the one based on Bellman error [9]. It is provably tight for the general case of any value function of any MDP. Although, it is not as strong at bounding subproblem use in our restricted class of MDPs.

There has been a great deal of recent work in finding and exploiting layers of abstraction in MDPs. This work is very related to subproblem reuse since the operators at a high layer of abstraction can be thought of as subproblem solutions of the lower layers. Theoretical bounds on the suboptimality of subproblem reuse also help to understand the suboptimality of using abstractions (and vice versa.)

Sutton and his colleagues provided a theoretical framework for dealing with abstraction in reinforcement learning, through a *multi-time model*. [6] This framework defines *temporally abstract options* or *macro-actions*, which are local policies that act like subroutines. Options, unlike primitive actions, can control the agent over multiple time steps, and so can work at different levels of abstraction. The theoretical model of options allows them to be treated as actions in the MDP, and can be used to possibly speed planning and still find an optimal solution.

Hauskrecht, et al. [3] extends Sutton's original theoretical results on planning with macro-actions to develop a hierarchical model of MDPs. The motivation was to use macro-actions to reduce the state and action space of large MDPs, through abstraction. This is done by creating a state partition of the MDP and restricting a macro to be a local policy over one of these regions. This partition and associated macros defines a new *abstract MDP* whose states are the boundaries of the regions, and whose actions are the macros that move between regions.

The solution to the abstract MDP has no guarantee of being a good solution to the original MDP. In order to make some guarantees on the quality of the solution requires a macro set that "covers" the range of values of a region's exit states. This may be intractable since the number of required macros grows exponential in the number of exit states for the region and the desired bound on suboptimality.

Parr [5] developed a number of improved methods for constructing macro sets, or policy caches, for an abstract MDP. He presents two algorithms that give optimality guarantees on the original MDP, though are not exponential in either the error bound or the size of the region's boundary.

7 Conclusion

In this paper we examined the reuse of subproblem solutions in the context of MDPs. We provided examples of the tradeoff between learning speed and loss in optimality. In order to quantify this tradeoff we examined quantitative bounds on the suboptimality of these techniques. Although generally applicable, a bound based on the Bellman error is often not tight enough to be useful. We introduced an empirically tight bound that applies specifically subproblems in our restricted class of MDPs.

There is still many questions of extending and using this bound. Relaxing the restriction on the applicable class of MDPs would be useful. Another critical step is examining the loss in optimality of using multiple subproblem solutions simultaneously. There are also questions on how this bound may help identify good and bad subproblems *before* using them, especially if there is knowledge about the problem or problems using them.

A Proof of Theorem 2

Proof. Let $s_m = \text{argmax}_{s \in B(S)} V^*(s)$. We will first show that π is β-optimal for s_m. From this we will be able to show that for all other states it is $\alpha\beta$-optimal.

We can write $V^*(s_m)$ as,

$$V^*(s_m) = \sum_{\text{all paths}} \text{Pr}(path)\text{Value}(path)$$

We can then split the sum into the paths containing a state in S' and those that don't contain a state in S'. Let p be the probability a path from s_m, following the optimal policy, contains a state in S'. Then,

$$\begin{aligned} V^*(s_m) = \quad & (1-p) \quad E(\text{Value}(path) | S' \cap path = \emptyset) + \\ & p \quad E(\text{Value}(path) | S' \cap path \neq \emptyset) \end{aligned}$$

Let $\nu = (1-p)E(\text{Value}(path)|S' \cap path = \emptyset)$. Notice that any path that contains a state in S' must also contain a state in $B(S')$. So we can write the second term as a value of that state discounted by the length of the path before the state is reached. Let $\ell(path, s)$ be the number of steps in $path$ until s is reached. So,

$$V^*(s_m) = \nu + \sum_{\substack{\text{paths with} \\ s \in B(S')}} \text{Pr}(path)\gamma^{\ell(path,s)}V^*(s)$$

Since $V^*(s_m) \geq V^*(s)$, we can substitute $V^*(s_m)$ for $V^*(s)$ and pull it outside of the sum. The resulting sum is just the expected discount between s_m and s when following the optimal policy. Let this value be δ. So we get,

$$V^*(s_m) \leq \nu + p\delta V^*(s_m)$$

This recurrence can be written as,

$$\begin{aligned} V^*(s_m) & \leq \nu \left(\sum_{i=0}^{\infty} (p\delta)^i \right) \\ & \leq \nu \left(\frac{1}{1 - p\delta} \right) \end{aligned}$$

We can also achieve a similar result for $V^*_{\mathcal{S}}(s_m)$. Since π acts optimally for $s \notin S'$, then it can achieve the same value for paths not containing $s \in S'$. As before, paths containing a state in S', must contain a state in $B(S')$. So we can write,

$$V^*_{\mathcal{S}}(s_m) = \nu + \sum_{\substack{\text{paths with} \\ s \in B(S')}} \Pr(path)\gamma^{\ell(path,s)} V^*_{\mathcal{S}}(s)$$

By the α-equivalence of $B(S)$ we know that $V^*_{\mathcal{S}}(s) \geq \alpha V^*_{\mathcal{S}}(s_m)$. So, as before, we can substitute $\alpha V^*_{\mathcal{S}}(s_m)$ for $V^*_{\mathcal{S}}(s)$ and pull it outside of the sum. The resulting sum is the expected discount between s_m and s when following $\pi^*_{\mathcal{S}}$. Let this value be δ'. So we get,

$$V^*_{\mathcal{S}}(s_m) \geq \nu + p\delta'\alpha V^*_{\mathcal{S}}(s_m)$$

Since $\pi^*_{\mathcal{S}}$ is optimal over the subproblem, it will reach a state in $B(S')$ with less expected discount than π^*. Hence, $\delta' \geq \delta$, and we can substitute δ for δ' and maintain the inequality. As above, we can rewrite the recurrence as,

$$\begin{aligned} V^*_{\mathcal{S}}(s_m) &\leq \nu\left(\sum_{i=0}^{\infty}(p\delta\alpha)^i\right) \\ &\leq \nu\left(\frac{1}{1-p\delta\alpha}\right) \end{aligned}$$

Now we examine the ratio of the two values,

$$\begin{aligned} \frac{V^*_{\mathcal{S}}(s_m)}{V^*(s_m)} &\geq \frac{\nu\left(\frac{1}{1-p\delta\alpha}\right)}{\nu\left(\frac{1}{1-p\delta}\right)} \\ &\geq \frac{1/(1-p\delta\alpha)}{1/(1-p\delta)} \\ &\geq \frac{1-p\delta}{1-p\delta\alpha} \end{aligned}$$

The value, $p\delta$, is the probability of returning to a state in S', discounted by the distance to that state and the distance from that state to a state in $B(S')$. Since all states in S' have a smaller value than t_m, then we can bound this by the discounted probability of ever reaching a state with smaller value. But this can be computed from ρ and is the following infinite sum,

$$\gamma(\rho\gamma+\rho(1-\rho)\gamma^2+\rho(1-\rho)^2\gamma^3\ldots) = \frac{\rho\gamma^2}{1-(1-\rho)\gamma} = \sigma$$

So, $p\delta \leq \sigma$, hence,

$$\frac{V^\pi(t_m)}{V^*(t_m)} \geq \frac{1-\sigma}{1-\sigma\alpha} \geq \beta$$

We can now show that $\pi^*_{\mathcal{S}}$ is $\alpha\beta$-optimal for any state, s. We can decompose the value of s just as was done for s_m. Let p be the probability that a path from s, following π^*, contains a state in S'. Notice that if $s \in S'$, then $p = 1.0$. We let ν and δ represent the same values with respect to s. Then we can bound $V^*(s)$, $V^*_{\mathcal{S}}(s)$, and their ratio as follows:

$$\begin{aligned} V^*(s) &\leq \nu + p\delta V^*(s_m) \\ V^*_{\mathcal{S}}(s) &\geq \nu + p\delta\alpha V^*_{\mathcal{S}}(s_m) \\ \frac{V^*_{\mathcal{S}}(s)}{V^*(s)} &\geq \frac{\nu + p\delta\alpha V^*_{\mathcal{S}}(s_m)}{\nu + p\delta V^*(s_m)} \\ &\geq \frac{p\delta\alpha V^*_{\mathcal{S}}(s_m)}{p\delta V^*(s_m)} \\ &\geq \alpha\frac{V^*_{\mathcal{S}}(s_m)}{V^*(s_m)} \\ &\geq \alpha\beta \end{aligned}$$

So, $\forall s \quad V^*_{\mathcal{S}}(s) \geq \alpha\beta V^*(s)$, therefore $\pi^*_{\mathcal{S}}$ is $\alpha\beta$-optimal for $\mathcal{P}$. ◇

B Proof of Theorem 3

Proof. Since $\mathcal{P}$ is deterministic then $\rho = 0$ and $\beta = 1$. Also, for any state, s, and a policy there is a unique path of states to a state in G from following that policy. Trivially if the optimal path (path of states from following the optimal policy) from s does not contain a state in S' then $V^*_{\mathcal{S}}(s) = V^*(s)$.

Consider the case where the optimal path from s contains a state in S'. Then it must contain a state in $B(S')$. Let g be the last state in the optimal path where $g \in B(S')$. So the optimal path from g cannot contain a state in S', and therefore,

$$V^*_{\mathcal{S}}(g) = V^*(g)$$

Now, there are two cases:

1. $g \in G'$. The path from following $\pi^*_{\mathcal{S}}$ will reach a state $g' \in G'$ in fewer steps (because $\pi^*_{\mathcal{S}}$ is optimal at reaching goal states of the subproblem.) Since $G' \geq^{\pi^*_{\mathcal{S}}}_{\alpha_1} (S \cup G')$, then

   $$V^*_{\mathcal{S}}(g') \geq \alpha_1 V^*_{\mathcal{S}}(g) = \alpha_1 V^*(g)$$

   And since g' is reached in fewer steps,

   $$V^*_{\mathcal{S}}(s) \geq \alpha_1 V^*(s)$$

2. $g \notin G'$. Since, $\bar{G}' \leq^{\pi^*_{\mathcal{S}}}_{\alpha_2} (S \cup \bar{G}')$, then

   $$V^*_{\mathcal{S}}(s) \geq \alpha_2 V^*_{\mathcal{S}}(g) = \alpha_2 V^*(g) \geq \alpha_2 V^*(s)$$

So, $\pi^*_{\mathcal{S}}$ is $\alpha_1\alpha_2$-optimal. ◇

References

[1] D. P. Bertsekas. *Dynamic Programming: Deterministic and Stochastic Models*. Prentice-Hall, Englewood Cliffs, NJ, 1987.

[2] T. Dean and S.-H. Lin. Decomposition techniques for planning in stochastic domains. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, 1995.

[3] M. Hauskrecht, N. Meuleau, L. P. Kaelbling, T. Dean, and C. Boutilier. Hierarchical solution of markove decision processes using macro-actions. In *Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-98)*, 1998.

[4] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

[5] R. Parr. Flexible decomposition algorithms for weakly coupled markov decision problems. In *Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-98)*, 1998.

[6] R. Sutton, D. Precup, and S. Singh. Between MDPs and semi-MDPs: Learning, planning, and representing knowledge at multiple temporal scales. Technical report, Department of Computer Science, University of Massachusets, Amherst, 1998.

[7] S. Thrun and A. Schwartz. Finding structure in reinforcement learning. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems*, volume 7, pages 385–392. The MIT Press, 1995.

[8] M. M. Veloso. Flexible strategy learning: Analogical replay of problem solving episodes. In *Proceedings of AAAI-94, the Twelfth National Conference on Artificial Intelligence*, pages 595–600, Seattle, WA, August 1994. AAAI Press.

[9] R. J. Williams and L. C. Baird. Tight performance bounds on greedy policies based on imperfect value functions. Technical report, College of Computer Science, Northeastern University, 1993.