Performability Optimization using Linear Bounds of Partially Observable Markov Decision Processes

Kaustubh R. Joshi[‡] Matti A. Hiltunen[§] William H. Sanders[‡]

[‡]Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
Urbana, IL, USA
{joshi1,whs}@crhc.uiuc.edu

[§]AT&T Labs Research
180 Park Ave.
Florham Park, NJ, USA
hiltunen@research.att.com

Abstract

Markov Decision Processes (MDPs) and Partially Observable MDPs (POMDPs) have been proposed as a framework for performability management. However, exact solution of even small POMDPs is very difficult because of their potentially infinite induced state spaces. In this paper, we present new lower bounds on the accumulated reward measure for MDPs and POMDPs. We describe how the bounds can be used in conjunction with heuristic search techniques in order to circumvent the state-space explosion problem in POMDPs. Our techniques can be used to choose actions that attempt to maximize performability during system recovery in self-healing systems.

1 Introduction

The goal of an autonomic self-healing system is to automatically preserve its performance despite the occurrence of faults and attacks (a.k.a. impairments). Consequently, one can intuitively view construction of such systems as a performability management problem. There have been a number of efforts over the years to utilize Markov decision processes (MDPs) as a framework for performability optimization through adaptation in general ([4], [1]), and repair in particular ([7], [2]). However, most of the work has only been exploratory, and has not progressed beyond toy examples and the basic observation that MDPs are a suitable framework for performability optimization problems.

Specifically, two important challenges that arise in most practical applications, but are not dealt with in previous work, are the well-known problem of state-space explosion, and the problem of partial observability. First, the state-space explosion problem is more severe for performability optimization than for performability evaluation because the state-space must encode *all* the possible environments and adaptation choices the system encounters, not just a single one.

Second, the assumption that the state of the system is precisely known at the decision points fails to capture the uncertainty inherent in realistic distributed systems. In self-healing systems, the occurrences of impairments are part of the system state. Due to fundamental impossibility results related to fault detection [3] and the limited fault coverage and diagnosability of realistic monitoring systems, it is not feasible to assume a precise knowledge of the system state. To address this challenge, we have proposed in [5] the use of partially-observable Markov decision processes (POMDPs) for self-healing and automatic recovery problems. However, POMDPs are even more difficult to solve than fully observable MDPs, because they possess an infinite state-space of "belief-states," which exacerbate the already problematic state-space explosion.

In this paper, we describe how we have tackled the problem of large state-spaces in POMDPs that were obtained during the construction of self-healing systems by the use of finite-depth trees of future system trajectories. We also present lower bounds for the value-functions of MDPs and POMDPs that can be used both independently and in conjunction with previously proposed upper bounds [8] to guide tree exploration and to assess risk during the choice of actions when the trajectory tree has not been fully explored. The lower bounds are solutions of a system of linear equations and are henceforth referred to as *linear bounds*. Although finite depth tree exploration has been proposed as a technique to solve POMDPs before, we believe that the proposed linear bounds and their use in risk estimation during action selection are new.

2 Definitions

In this section, we review the definitions for MDPs, POMDPs, and policies, and describe the performability optimization problem in terms of the MDP framework.

A *Markov Decision Process* is defined as a tuple $(\mathcal{S}, \mathcal{A}, p(\cdot|s,a), r(s,a))$ where S is a finite set of states, and A is a finite set of actions. $p(\cdot|s,a) \in \mathcal{S} \times \mathcal{A} \times \mathcal{S} \times [0,1]$ is a collection of state-transition probability functions, one per action, such that $p(s'|s,a)$ where $s, s' \in \mathcal{S}$, and $a \in \mathcal{A}$ denotes the probability that the MDP will transition to state

s' when action a is chosen in state s. Finally, $r(s, a) \in \mathcal{S} \times \mathcal{A} \times \mathbb{R}$ is a reward (cost) function that specifies the reward (cost) incurred when action a is chosen in state s.

When constructing self-healing systems, one is most concerned with automatic recovery from impairments with minimal interference to the system's primary mission. Hence, it is natural to view the action-space as the set of available recovery actions (e.g., rebooting or reconfiguring to mask faults). At the very least, the state-space can encode information about faults currently existing in the system. The cost measure is typically recovery time, but can vary for different applications.

A (recovery) policy ρ is then a mapping from states to probability distributions over actions such that $\rho(s, a)$ specifies the probability that action a will be chosen in state s. A deterministic policy is one that always chooses some action a with probability 1 in each state. The goal of MDP solution techniques is to construct *optimal policies* according to some optimization criteria. In this paper, we choose the *discounted reward* optimization criterion.[1] The discounted reward criterion seeks a policy ρ^* that optimizes the discounted reward accumulated by the system over its lifetime. Reward accumulated t time units in the future is discounted by β^t, where $0 \leq \beta < 1$ is the "discounting factor." Formally, $\rho^* = \text{argmax}_\rho \mathbb{E}_\rho[R] = \text{argmax}_\rho \sum_{t=0}^{\infty} \beta^t r(S_t, \rho(S_t))$, where $S_t \in \mathcal{S}$ and $\rho(S_t) \in \mathcal{A}$ are random variables representing the state of the system and the action chosen under policy ρ respectively, at time t.

Given a starting state $s \in \mathcal{S}$, the *value* of the MDP is defined as the optimal reward obtainable when starting from that state. It is known (from [6], for example) that the value function $V_m(s), \forall s \in \mathcal{S}$ is given by the dynamic programming equation:

$$V_m(s) = \max_{a \in \mathcal{A}} \left\{ r(s, a) + \beta \sum_{s' \in \mathcal{S}} p(s'|s, a) V_m(s') \right\} \qquad (1)$$

The optimal (deterministic) policy is given by choosing for each state $s \in \mathcal{S}$ the action $a^*(s)$ that maximizes the right side of Equation 1.

MDPs make the often unrealistic assumption that the state of the system is perfectly known when an action is to be chosen. Partially Observable MDPs (POMDPs) generalize MDPs by removing this assumption. A POMDP is defined as a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{O}, p(\cdot|s, a), q(\cdot|s, a), r(s, a))$ where $\mathcal{S}$, $\mathcal{A}$, p, and r are the same as for an MDP. However, the system can be observed only through a finite set of observations $\mathcal{O}$. An observation $o \in \mathcal{O}$ is generated with probability $q(o|s, a)$ whenever the system transitions to state s as a result of action a being chosen. Although optimal POMDP policies are not Markovian in terms of the observation sequence, they are Markovian in terms of the "belief state." A belief state $\pi = [\pi(1), \ldots, \pi(|S|)]$ specifies the probability with which the system is in each state $s \in \mathcal{S}$. The value of the POMDP is then the solution of an equivalent MDP defined over the state-space of belief-states ($\subseteq \mathbb{R}^{|\mathcal{S}|}$). Formally:

$$V_p(\pi) = \max_{a \in \mathcal{A}} \left\{ \pi \cdot r(a) + \beta \sum_{o \in \mathcal{O}} \gamma(o|\pi, a) V_p([T_s(\pi|o, a)]) \right\} \qquad (2)$$

$$T_s(\pi|o, a) = \frac{q(o|s, a) \sum_{s' \in \mathcal{S}} p(s|s', a)\pi(s')}{\sum_{o' \in \mathcal{O}} q(o'|s, a) \sum_{s' \in \mathcal{S}} p(s|s', a)\pi(s')} \qquad (3)$$

$$\gamma(o|\pi, a) = \sum_{s \in \mathcal{S}} q(o|s, a) \sum_{s' \in \mathcal{S}} p(s|s', a)\pi(s') \qquad (4)$$

Here, $r(a) = [r(s, a), \forall s \in \mathcal{S}]$ is the reward vector of the POMDP when action a is chosen, and $T_s(\pi|o, a)$ is the belief-state transition function that uses Bayes rule to compute the next belief-state $\pi_{t+1} = [T_s(\pi_t|o, a), s \in \mathcal{S}]$ given the current belief-state π_t, the action taken, and the observation generated by the system. Finally, $\gamma(o|\pi, a)$ computes the belief-state transition probability as the probability that observation o will be generated given the current belief-state and the action taken.

The preceding equations clearly illustrate how the state-space explosion problem of MDPs becomes even more severe in the case of POMDPs. Even when the underlying state-space of a POMDP is finite, the induced belief-state-space is in the space of probability distributions over $\mathcal{S}$ and can be infinite.

3 Heuristic Search and Bounds

In this section, we explain the motivation for the use of informed heuristic tree searches of forward MDP state-action trajectories as a technique to overcome the state-space problem and discuss the use of bounds in the search process.

3.1 The Case for Heuristic Tree Search

The two major techniques used for POMDP (and MDP) solution are value iteration and policy iteration [6]. Many variants of these techniques exist, and they use a variety of innovations, ranging from quicker convergence techniques to structured/parametric state-space and value function representations. However, all of them share the requirement that the value function for all the states of the POMDP must be computed, stored, and updated explicitly. However, the value function (or the corresponding optimal action) for a state is required by the controller only when the system is actually in that state. Hence, if the model of a system changes often (e.g., due to changes in the environment), and the system visits only a small fraction of the possible states before every change, then the non-trivial effort expended by the above techniques in computing the values of the other states is wasted.

To circumvent the problem, we use a path-based approach (see [5]). Whenever an event occurs in the system such that the controller must take some action, it uses

[1] In self-healing systems, it is more intuitive to minimize the total *undiscounted* cost accumulated by the system during its recovery. However, the discounted criterion is more common in the literature, and using it allows easier reuse of previous results. We believe that the results hold for the undiscounted case as well, but defer proofs to future work.

the POMDP model to generate a finite-depth trajectory tree $\Delta(\pi)$ rooted in its current belief-state π. $\Delta(\pi)$ can be viewed as a finite depth unrolling of the recursion of Equation 2, and consists of nodes representing actions, observations, and future belief-states. Paths in the tree represent possible future trajectories the system can take. The value of the tree $V_\Delta(\pi)$ can be computed by first assigning a value to the belief-states that are at the leaves of $\Delta(\pi)$ either using a heuristic, or using bounds as described below. Then, using Equation 2 as a template, the values at the leaves are propagated to the root through computation of weighted sums at the observation nodes, and maximums at the action nodes.

We believe that such an approach is suitable for self-healing system optimization because of the conjecture that most impairments that occur in realistic systems can be fixed in a relatively small number of steps. Hence, if a *goal state* is defined as a state in which the system has recovered from an impairment, and forward paths are terminated when a goal state is reached, the trajectory tree is likely to be fairly shallow. Second, because the number of potential impairments to a system can be huge but usually only a few of them are suspected at a time, there can be a significant number of unreachable states from any starting state. The path-based approach avoids exploring such unreachable states.

However, the path-based approach possesses a major limitation. The number of paths increases exponentially with the depth of the trajectory tree, thus necessitating termination of the tree at fairly short depths. Fortunately, it is possible to use bounds on the value of states to alleviate this problem and generate policies that are provably good. Specifically, even if a goal state has not been reached when a path is terminated, computationally cheap bounds can be constructed to estimate the remaining cost of the sub-tree rooted at the termination point.

3.2 Use of Value Function Bounds

Consider a belief-state π of a POMDP $\mathcal{P}$. Let $V_p(\pi)$ be its value (obtained using Equation 2), and let $V_p^+(\pi)$ and $V_p^-(\pi)$ be upper and lower bounds on the value. Furthermore, given a trajectory tree $\Delta(\pi)$ rooted at belief-state π, let $V_\Delta^-(\pi)$ be the value of the trajectory tree such that $V_p^-(\cdot)$ is used as an estimate of the values of the belief-states at the leaves of $\Delta(\pi)$ ($V_\Delta^+(\pi)$ can be similarly constructed using $V_p^+(\cdot)$). The bounds can be used as follows.

Branch and bound Bounds can be used in a traditional branch and bound strategy to prune useless portions of the trajectory tree. Specifically, for a belief state π and action a, let $V_a^+(\pi) = \pi \cdot r(a) + \beta \sum_{o \in \mathcal{O}} V_p^+(T(\pi|o,a))\gamma(o|\pi,a)$ be the single-step upper-bound estimate if action a is chosen ($V_a^-(\pi)$ can be similarly defined). Then, for a pair of actions a and a', if $V_a^+(\pi) < V_{a'}^-(\pi)$, then a is provably inferior to a', and the subtree rooted at it need not be explored.

Driving informed search Although any type of tree search algorithm can be used to explore the trajectory tree, if an informed search algorithm such as AO* is used, as in [8], the bounds can be used to determine both the order in which the action nodes should be traversed, and the time when tree exploration can stop because it is not possible to find a policy better than the best one found so far (see [8] for details).

Minimizing potential risk In many cases, a controller may have to take action even if an optimal strategy has not yet been found. In such a situation, the action a that maximizes $V_\Delta^-(\pi)$ can be executed. Action a minimizes risk because $V_p^-(\cdot)$ is a lower bound, and hence a (maximally) guarantees a reward of at least $V_\Delta^-(\pi)$. In contrast, an action that maximizes $V_\Delta^+(\pi)$ might prove to be too optimistic and actually yield a much lower reward.

4 Linear Lower Bounds

In this section, we present linear bounds for MDPs and POMDPs. These bounds are based on the simple observation that barring any additional information, the mean of a finite set of numbers is the tightest *linear* lower-bound of its maximum element; i.e., for any finite set $\mathcal{X}$, and function $f : \mathcal{X} \to \mathbb{R}$, $g_l(\mathcal{X}) = \frac{1}{|\mathcal{X}|}\sum_{x \in \mathcal{X}} f(x) \le \max_{x \in \mathcal{X}} f(x) = g_m(\mathcal{X})$. Furthermore, there is no other function of the form $g(\mathcal{X}) = \sum_{i=1}^{|\mathcal{X}|} a_i \cdot f(x_i)$ where x_i refers to the i^{th} element of $\mathcal{X}$ and the a_i's do not depend on $\mathcal{X}$ such that $g_l(\mathcal{X}) < g(\mathcal{X}) \le g_m(\mathcal{X})$.

Using this observation, we introduce the *linear-bound* of an MDP (V_m^-) as defined by the following set of linear equations:

$$V_m^-(s) = \frac{1}{|\mathcal{A}|}\sum_{a \in \mathcal{A}}\left\{r(s,a) + \beta\sum_{s' \in \mathcal{S}} p(s'|s,a)V_m^-(s')\right\} \quad (5)$$

Note that the above equation effectively constructs a Markov chain from the MDP by replacing the non-deterministic actions with probabilistic transitions with a uniform transition probability of $\frac{1}{|\mathcal{A}|}$. We can make the following statements, without detailed proofs, regarding the linear-bound.

Theorem 4.1 *Let $\mathcal{L} : \mathbb{R}^{|\mathcal{S}|} \to \mathbb{R}^{|\mathcal{S}|}$ be the linear transformation defined by Equation 5. Then, $\mathcal{L}$ is a contraction mapping, and therefore possesses a unique fixed-point $V_m^- \in \mathbb{R}^{|\mathcal{S}|}$.*

The existence of a unique fixed-point implies that Equation 5 has a unique solution that can be obtained simply by value iteration, or by using any standard linear system solution technique. Additionally, the linear-bound is a lower bound for the value functions of its corresponding MDP or POMDP as we state below.

Theorem 4.2 *For any Markov Decision Process $\mathcal{M} = (\mathcal{S}, \mathcal{A}, p(\cdot|s,a), r(s,a))$, the linear-bound V_m^- obtained as a solution to Equation 5 is a lower bound of the value function V_m obtained as a solution to Equation 1.*

Proof: By induction on iterations of Equations 1 and 5. □

Theorem 4.3 *For any Partially Observable Markov Decision Process $\mathcal{P} = (\mathcal{S}, \mathcal{A}, \mathcal{O}, p(\cdot|s,a), q(\cdot|s,a), r(s,a))$, define the mean linear-bound as $V_p^-(\pi) = \sum_{s\in\mathcal{S}} \pi(s) \cdot V_m^-(s)$, where V_m^- is the linear-bound of the underlying Markov Decision Process $(\mathcal{S}, \mathcal{A}, p(\cdot|s,a), r(s,a))$. Then, V_p^- is a lower bound of the value function V_p obtained as a solution to Equation 2.*

Proof: By the definition of V_p^- and induction on iterations of Equations 2 and 5. □

For the solution of POMDPs, the most straightforward use of the linear bounds presented above is the use of Theorem 4.3 to construct V_p^- as a lower-bound for the value function V_p. To bound V_p from both sides, one can use V_p^- together with an upper bound V_p^+ that is based on the solution of the underlying MDP [8] given by $V_p^+(\pi) = \sum_{s\in\mathcal{S}} \pi(s) \cdot V_m(s)$.

The importance of these bounds is significant. Because both bounds are computed on the original state-space of the POMDP as opposed to the belief-state-space (which can be infinite even for trivial problems), they can be computed quickly for problems of large size and explicitly stored. For example, in [5], we propose a POMDP model of a self-healing system in which each state of the POMDP represents a particular fault occuring in the system. For most realistic systems, we do not expect the size of this state-space to exceed hundreds of thousands of states, a number already too big for state-of-the-art POMDP techniques, but trivial for most Markov Chain/MDP solution techniques. Second, we note that constructing the V_p^- bound introduced in this paper requires solution of a single Markov chain, and is significantly cheaper than computing the V_p^+ as proposed by [8].

Model-Specific Bound Improvements Although the mean of a finite set of numbers is the tightest *general* linear lower-bound of its maximum, if the numbers are known, a better linear-bound can be found. Using this principle, it is possible to tighten the lower-bound V_p^- defined in Theorem 4.3 on a model-specific basis by using coefficients other than a uniform value of $1/|\mathcal{A}|$ for each action in the outer summation of Equation 5. As long as the coefficients sum to 1 and the weights do not depend on the state s, it can be shown that V_p^- remains a lower-bound for the value function V_p defined in Equation 2. Values of the coefficients that result in a tighter bound can be found by performing a gradient-ascent search that maximizes the min-norm of the solution of the system of linear equations defined in Equation 5.

5 Future Work and Conclusion

In this paper, we looked at the problem of solving Partially Observable Markov Decision Processes (POMDPs) that arise during the construction of self-healing systems via finite-depth tree search. We proposed new lower bounds on the maximum accumulated reward obtainable by solving both MDPs and POMDPs and described how to use them during the finite-depth search process. The proposed lower bounds are solutions to a set of linear equations defined on the state-space of the MDP/POMDP, and can be obtained easily via traditional linear system solution techniques. An implementation of a finite-depth solver that constructs the proposed bounds and uses them during searches is still under way. Although experimental results are currently lacking, we believe that use of these bounds will lead to a significantly improved quality of recovery policies in self-healing systems (such as the one described in [5]) without much increase in solution cost.

Acknowledgment This material is based on work supported in part by the National Science Foundation under Grant No. CNS-0406351. Kaustubh R. Joshi is supported by an AT&T Virtual University Research Initiative (VURI) fellowship. The authors would also like to thank Jenny Applequist for helping improve the readability of the material.

References

[1] M. Abdeen and M. Woodside. Seeking optimal policies for adaptive distributed computer systems with multiple controls. In *Proc. of Third Intl. Conf. on Parallel and Dist. Computing, Applications and Technologies*, Kanazawa, Japan, Sept. 2002.

[2] H. de Meer and K. S. Trivedi. Guarded repair of dependable systems. *Theoretical Computer Sci.*, 128:179–210, 1994.

[3] M. Fischer, N. Lynch, and M. Paterson. Impossibility of distributed consensus with one faulty process. *J. ACM*, 32(2):374–382, Apr. 1985.

[4] L. J. Franken and B. R. Haverkort. Reconfiguring distributed systems using Markov-decision models. In *Trends in Dist. Sys.*, Aachen, Oct. 1996.

[5] K. R. Joshi, M. Hiltunen, W. H. Sanders, and R. Schlichting. Automatic model-driven recovery in distributed systems. In *Proc. of Symp. on Reliable Dist. Systems (SRDS 05)*, Oct 2005. To appear.

[6] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Intersci., 1994.

[7] K. G. Shin, C. M. Krishna, and Y.-H. Lee. Optimal dynamic control of resources in a distributed system. *IEEE Trans. on Software Eng.*, 15(10):1188–1198, Oct. 1989.

[8] R. Washington. BI-POMDP: Bounded, incremental, partially-observable Markov-model planning. In S. Steel and R. Alami, editors, *Proc. of European Conf. on Planning*, volume 1348 of *Lecture Notes in Computer Sci.*, pages 440–451. Springer, 1997.