Automated Generation of Understandable Contingency Plans

Max Horstmann and Shlomo Zilberstein
University of Massachusetts
Department of Computer Science
Amherst, MA 01003, U.S.A.
{horstman, zilberstein} @cs.umass.edu

Abstract

Markov Decision Processes (MDPs) and contingency planning (CP) are two widely used approaches to planning under uncertainty. MDPs are attractive because the model is extremely *general* and because many algorithms exist for deriving *optimal* plans. In contrast, CP is normally performed using heuristic techniques that do not guarantee optimality, but the resulting plans are more *compact* and more *understandable.* The inability to present MDP policies in a clear, intuitive way has limited their applicability in some important domains. We introduce an anytime algorithm for deriving contingency plans that combines the advantages of the two approaches.

1 Introduction

Two closely related decision-theoretic planning paradigms have emerged in the area of planning under uncertainty: Markov Decision Processes (MDPs) have become a general framework for planning and reinforcement learning. Contingency Planning (CP) on the other hand emphasizes compactness and understandability of the solution. In both cases, an agent is manipulating an environment by performing actions with uncertain outcomes. In each move, the agent receives some reward and the overall goal is to maximize the cumulative reward.

MDPs are solved by deriving a *policy* which maps domain states to actions, typically represented as a large table. Several existing algorithms can construct optimal policies, but the results are not easy to visualize or understand. CP is another widely used approach in stochastic domains which allows plans to include branches that may depend on arbitrary memory states.

It is clear why optimal plans are desirable, but the issue of understandability is less obvious. In fact, for some applications, a plan represented as a large table mapping states to actions may be perfectly suitable. However, the lack of clarity has limited the adoption of MDP planning in some application domains such as space exploration with unmanned rovers. In this domain, communication with the agent is restricted and due to high mission costs and associated risks, understanding and verifying plans are crucial.

We examine the relationship between policies and contingency plans and define a precise measure of complexity of contingency plans. Then, we introduce a technique for automated contingency plan generation and briefly describe our experimental results. Finally, we conclude with a summary and future research directions.

2 Formal Problem Description

A Markov Decision Process (MDP) with goal states is a tuple (S, A, P, R, s_0, G), where $S = S_1 \times \cdots \times S_n$ is a factored state space and S_i is the (finite) domain of feature *i. A* is a set of actions, *P* the transition probabilities, *R* the reward expectations, *so* a known initial state and *G* a (possibly empty) set of absorbing terminal states.

A policy $\pi : S \rightarrow A$ is a mapping from states to actions. Following the notation in iLittman et al., 1998], a Contingency Plan for an MDP is a tuple $(V, E, v_0, \rho, \delta)$, where *{V,E)* is a directed graph with start state $v_0 \in V$. $\rho : V \rightarrow A$ associates and action with each node of the graph, $\delta : E \rightarrow 2^S$ labels each edge with a set of states.

An interval label descriptor maps state sets to sets of intervals over the feature domains defined as follows: $\mathcal{D}_{int}(\delta) : E \rightarrow \{I_1, \ldots, I_k\}$ where I_j is the set: $\{[L_1; U_1], \ldots, [L_n; U_n] | 1 \leq L_i \leq U_i \leq |S_i| \forall 1 \leq i \leq n\}$.

The complexity of a contingency plan does not reflect the *computational* complexity of constructing it, but is a measure of how hard it is to understand. Formally: $Complex(\mathrm{MDP}, \mathrm{CP}, \mathcal{D}_{int}(\delta)) = |V| \cdot ABF \cdot ALDS$, where ABF is the average branching factor of the graph and ALDS is the average label descriptor size. The size of a single label descriptor is the number of constrained intervals.

The value of a policy reflects the expected discounted return when the agent selects action using *n,* starting from the initial state. It can be computed with standard algorithms such as value iteration. Similarly, the value of a contingency plan is the expected discounted return when the agent selects action using p, starting from the initial node. By considering the Markov chain induced by the state set *S* x *V,* the same methods used to evaluate MDP policies can be used to evaluate a contingency plan.

3 Automated Contingency Plan Generation

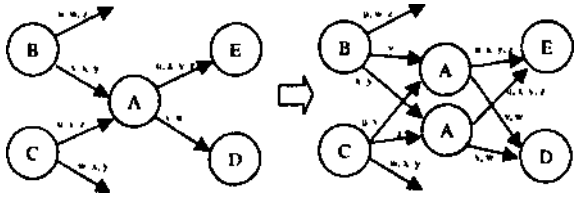
Our approach to automatically generate more understandable (less complex) contingency plans relies on solving first the underlying MDP, obtaining an optimal policy π and creating an equivalent initial contingency plan from it: $V = \{v_1, \ldots, v_{|A|}\}$, $E = V \times V$, $\rho(v_i) = \pi(a_i)$, $\delta(v_i, v_j) = \{s \in S | \pi(s) = a_j\}$. Then, the following operators are used to reduce the complexity of the plan.

By performing a depth-first search over the set of all reachable edges between the state-node pairs in *S* x *V,* a reachability analysis can be applied to remove states from label descriptors, edges from nodes and nodes from the graph.

Since a benefit of interval label descriptors is the ability to capture easily large sets of states and simplify the branch conditions, n-dimensional boxes to describe them more compactly can be generated by merging intervals along the dimensions of the state space. This is in particular very effective if the order of feature values corresponds to some natural ordering.

By trading off optimality for clarity, interval descriptors can be merged even if there are a few states that make the merge operator not admissible.

Most importantly, the node-split operator splits a node of the plan graph into two and adapts the incoming and outgoing edges according to a given partition of the state space. This gives contingency plans their real strength: The current node of the graph then represents partial information about the current state ("context information"), so less details of the actual state may be needed to make the branch decision on the next step. For instance, with a state set *S* = {u, v, *w, x, y, z)* and the partition *S\ — {u,v,w},* S_2 = {x,y,z}, the result of a split operator is illustrated below:



4 Experimental Results

It turns out that combining these operators can significantly reduce the complexity of a contingency plan. This leads to the following search problem: Given the initial contingency plan defined above, find a contingency plan with the lowest complexity, not giving up more that a certain fraction e of the optimality of the policy *n.* Unfortunately, two major obstacles make an exhaustive search intractable: The node-split operator is parameterized by a partition of the MDP state space, leading to a large number of possible splits and thus to a large branching factor. In addition, the computational complexity of reachability analysis is fairly high.

As a result, we decided to relax the requirement of minimizing complexity and developed a method to combine the operators and automatically generate some understandable plans. Performance improvement can be obtained by choosing a partition of the state space, split the nodes according to this partition, perform a reachability analysis and merge the interval descriptors subsequently. We implemented a hill-climbing algorithm that used some domain-specific heuristics for choosing a partition and performed random restarts, leading to an interruptible anytime algorithm that performed well on a simple set of test problems.

For instance, in a maze domain with stochastic move actions, the algorithm took advantage of bottlenecks, tunnels and alternative branches. In a simulated planetary rover domain with uncertain action durations, some hints for good split partitions lead to dramatically decreased complexity. The resulting plan instructed the agent to perform certain actions until a resource (one feature of the factored state space) drops below a certain threshold. This was very easy to understand for humans unlike the optimal policy given in the form of a huge lookup-table.

5 Conclusions

The main objective of this work has been to find solutions for decision-theoretic planning problems that are optimal (or near-optimal) and also compact and understandable. We defined a measure of the complexity of contingency plans, reflecting their size, branching factor and the size of the label descriptors. The results we gained by experimenting with several plan-transformation operators are encouraging and show the potential of automated generation of understandable contingency plans.

There are several interesting directions for future research. First, a better measure of the plan complexity could be developed. Second, additional operators could be added. Third, the language for representing edge labels could be enriched. Finally, methods for finding good split partitions have to be found to create more reliable algorithms. The results reported in this paper provide a good framework for future exploration of these research directions.

Acknowledgments

Support for this work was provided in part by NASA under grant NAG-2-1463. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of NASA.

References

[Bresina et al., 2002] J. Bresina, R. Dearden, N. Meulcau, S. Ramakrishnan, D. Smith, and R. Washington. Planning Under Continuous Time and Resource Uncertainty: A Challenge for AI. *Conference on Uncertainty in Artificial Intelligence,* 2002.

[Feng and Hansen, 2002] Z. Feng and E.A. Hansen. Symbolic Heuristic Search for Factored Markov Decision Processes. *Eighteenth National Conference on Artificial Intelligence,* 2002.

[Hansen and Zilberstein, 2001] E.A. Hansen and S. Zilberstein. LAO*:A Heuristic Search Algorithm that Finds Solutions with Loops. *Artificial Intelligence,* 129(l-2):35-62, 2001.

[Littman et al., 1998] M.L. Littman, J. Goldsmith, and M. Mundhenk. The Computational Complexity of Probabilistic Planning. *Journal of Artificial Intelligence Research,* 9:1-36, 1998.