Representing Iterative Loops for Decision-Theoretic Planning* (Preliminary Report)

Liem Ngo **Peter Haddawy**
{*Liem,Haddawy*}*@cs.uwm.edu*

Department of Electrical Engineering and Computer Science
University of Wisconsin-Milwaukee
PO Box 784
Milwaukee, WI 53201

1 Introduction

Many planning problems are most naturally solved using an interative loop of actions. For example, a natural plan to unload a truckload of boxes is to repeatedly remove a box until the truck is empty. The problem of planning with loops has been investigated in the context of classical planning [Wilkins, 1988] as well as in the context of reactive planning [Musliner, 1994] but not for decision-theoretic planning. This paper discusses representing iterative loops for decision-theoretic planning. We present a representation of actions of the form "Repeat the following action until *condition* is satisfied." We address the problem of guaranteeing that such loops will terminate. Finally we discuss how to abstract iterative constructs in order to limit the number of chronicles generated in plan projection.

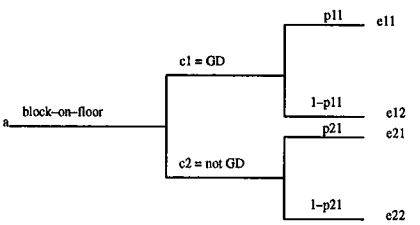

Figure 1: Example of an action tree.

2 The Representation

Our representation of actions is both conditional and probabilistic: each action has preconditions, which must be satisfied in order for the action to be meaningful, and under a certain condition an action will have a given effect with a certain probability. We can depict an action by a tree. In figure 1, the precondition of action a=pick-a-block-up-and-put-it-on-the-table is block-on-floor. The conditions c_1=GD (gripper dry) and c_2=(not GD) are mutually exclusive and exhaustive under the precondition. When c_1 is true, the execution of a results in e_{11} with probabilty p_{11} and e_{12} with probability $(1 - p_{11})$.

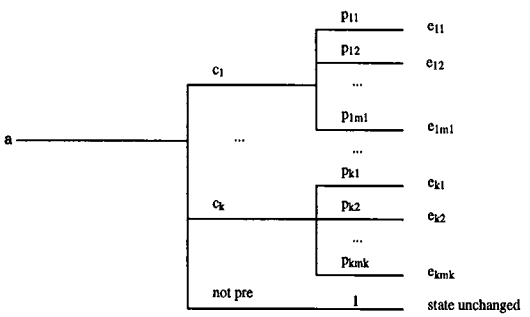

Figure 2: General action description.

If the precondition of an action must be satisfied before an action can be performed, not any arbitrary sequence of actions can be a valid plan. In a probabilistic framework, this can be problematic since the outcome of a partial plan may have a chance of satisfying a precondition of the next action in the plan and

*This work was partially supported by NSF grant #IRI-9207262.

a chance of not. We solve this problem by assuming that if the precondition of an action is not satisfied, the performance of the action will leave the world intact. We can represent this by conjoining the precondition with the probabilistic conditions and adding a dummy branch for the case in which the precondition is not satisfied, as shown in figure 2. Using this modified action description a plan is now simply any arbitrary sequence of actions. This action representation is similar to that introduced by Hanks [1990] and used in some recent planners [Haddawy and Suwandi, 1994, Kushmerick *et al.*, 1994].

We assume that changes to the world are limited to those effects explicitly specified in the action descriptions, so we do not allow exogenous events. A further assumption in our action representation is that the probability of an outcome e_{ij} of an action a is independent of any earlier actions and states of the world given the action and its corresponding condition c_i.

We call a sequence of repetitions of the same action a *repeating action*. Starting from an initial state, a plan can generate many possible final outcomes. By the above independence assumption, the probability distribution of those outcomes can be determined by probabilistic characteristics of the component actions and the starting state. When repeating an action, an outcome may occur that makes the next performance of it undesirable: either the precondition is falsified or a desirable outcome has been achieved. Such an outcome is called an *exit outcome*. The until condition of a repeat-until loop is one kind of exit outcome.

We describe the world with two kinds of variables: propositional and quantitative. Quantitative variables include resource variables like time (in cases where we have deadlines), dollars, fuel, etc.

We assume that the expected utility of a plan is a function only of the possible states of the world immediately following execution of the plan. Let $u(s)$ be the utility obtained when state s is reached after executing some plan P. We define the expected utility of plan P as

$$EU(\mathsf{P}) = \sum_S u(s) Prob(s \mid \mathsf{P}).$$

Plans are evaluated according to their expected utilities. An optimal plan is a plan with the highest expected utility.

3 Guaranteeing Loop Termination

In deterministic settings like programming languages, loops can be guaranteed to terminate if we can show that the termination condition will eventually be reached. With probabilistic actions, however, the termination condition may never be achieved. But for

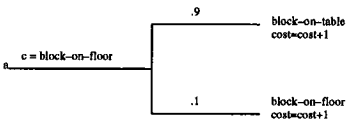

Figure 3: Pick up block action.

practical planning problems, it is undesirable to have our planner generate a plan containing an infinite loop. We can appeal to the decision-theoretic notion of optimality to avoid this situation by identifying constraints on the action descriptions and the utlity function that will guarantee that the optimal number of repetitions of a loop will be finite. Specifically, let $a^{\uparrow n}$ denote the composite action: *repeat action a at most n times.* We will provide constraints that guarantee the existence of an optimal value of n. We will use an example to first illustrate how this can be done.

Consider a blocks world with only one action a= pick-a-block-up-and-put-it-on-table, as shown in figure 3. In the figure, $cost_0$ represents the cost immediately preceding the action and *cost* represents the cost immediately following it. The precondition of a is that a block is available on the floor. By performing a, there is a probability of .9 that the action succeeds (the block is now on the table) and a probability of .1 that the action fails (the block is still on the floor). The performance of a always costs 1 unit of resource. Assume that in the starting state S_0, we have the probability distribution $P_0(block - on - floor) = 1$ and $P_0(cost = 0) = 1$. If we perform the action a until the block is on the table or at most n ($n > 0$) times, the possible final states resulting from the action $a^{\uparrow n}$ are:

- $S_n^0 = \{block - on - floor, cost = n\}$ with probability $.1^n$
- $S_n^i = \{block - on - table, cost = i\}$ with probability $.1^{(i-1)}.9$, for i from 1 to n.

Assume that the utility function is

$$u(\{block{-}on{-}table, cost = x\}) = \begin{cases} 0 & \text{if } x > 2 \\ 6 - x & \text{otherwise} \end{cases}$$

$$u(\{block{-}on{-}floor, cost = x\}) = \begin{cases} 0 & \text{if } x > 2 \\ 3 - x & \text{otherwise} \end{cases}$$

The above utility function expresses the fact that only two units of resource (cost) are available by assigning any outcome in which cost exceeds two the lower bound utility of zero. For all other outcomes, we prefer success and lower cost.

$$EU(\mathbf{a}^{\uparrow n}) = \begin{cases} .1^n(0) + (6-2).1.9 + (6-1).9 = 4.86 & \text{if } n > 2 \\ .1^2(3-2) + .1.9(6-2) + .9(6-1) = 4.87 & \text{if } n = 2 \\ .1(3-1) + .9(6-1) = 4.7 & \text{if } n = 1 \end{cases}$$

Figure 4: Expected utility of pick up block loop.

Applying the utility function to the possible outcomes of the repeating action $a^{\uparrow n}$, we obtain the expected utility of the plan consisting of only that action as shown in figure 4.

Notice that, when $n > 2$, all states S_n^0 and S_n^i, $(i > 2)$, have utility 0, and only S_n^1, S_n^2 have positive utilities, 5 and 4 respectively, with probability .9 and .09 respectively. So the optimal number for n is 2: the other composite actions $a^{\uparrow 3}$, $a^{\uparrow 4}$, etc. are suboptimal and need not be considered. Such finiteness of repetition can be obtained in a broad class of planning problems.

Proposition 1 *Under the following conditions:*

1. *the utility function depends only on the final states of chronicles;*
2. *one resource R is limited by a finite number $L > 0$;*
3. *any performance of the action a in a situation where the precondition is satisfied consumes at least $l > 0$ units of the resource R;*
4. *the smallest utility value is achieved when a resource is overconsumed. For example, if the smallest possible value of utility is 0 then the utility of a state in which the accumulative resource consumption is greater than L is 0.*

There exists a finite M such that $a^{\uparrow M}$ has the highest expected utility among $\{ a^{\uparrow n}, n > 0 \}$. Furthermore, $M \leq L/l$.

This result can be extended to apply to temporal utility functions with finite horizons, one example of which is our model of deadline goals [Haddawy and Hanks, 1992]. For such a function, time corresponds to the finite resource.

We have a similar result when incorporating a repeating action into a plan.

Proposition 2 *Given*

1. *a finite set of actions;*
2. *a utility function and a resource R such that for every action the conditions of proposition 1 are satisfied;*
3. *there is no action to replenish the resource R.*

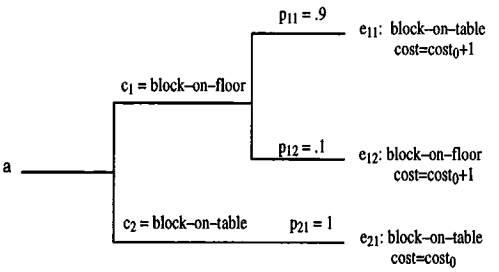

Figure 5: Augmented version of pick up block action.

There always exists (an) optimal finite plan(s) for any starting state. The plan(s) may include the finite repetition(s) of some actions.

4 Abstracting a repeating action

Since probabilistic actions can have many possible outcomes, projecting probabilistic plans can be a costly process. Previous work [Haddawy and Doan, 1994] has discussed how individual actions can be abstracted to reduce the number of chronicles that result from projection. We now examine how a repeating action *repeat action a at most n times* can be compactly represented by using abstraction.

4.1 An Example

Consider the blocks world example discussed above. In figure 5 we have augmented the action **a** by introducing a dummy condition block-on-table. When that condition occurs, the repeating action will be terminated and we have its outcome. One possible abstract description of $\mathbf{a}^{\uparrow n}$ is given in figure 6. Outcome e_{12} represents failure. The abstract outcome e_{11} represents all successful outcomes after at most n repetitions. The cost of that abstract outcome is a value within the range of the cost when the action is performed once and $n-1$ times. The probability that e_{11} is realized is $.9(\sum_{i=0}^{n-1} .1^i)$ or $1 - .1^n$, which is the probability that **a** succeeds after at most n times.

4.2 The General Case

In this section we show how to recursively construct an abstraction of a repeating action. We make several

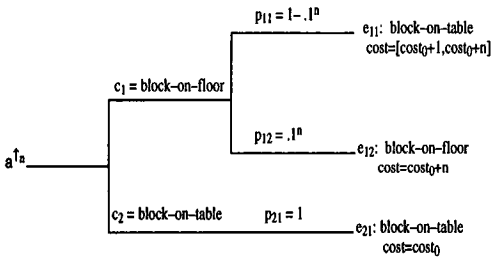

Figure 6: Abstract description of pick up block loop.

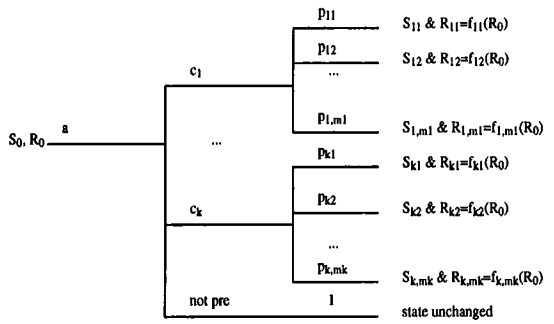

Figure 7: A general action description.

assumptions in this part. All of these assumptions are used to simplify the presentation and can be easily dropped.

We assume that there are two kinds of variables: propositional and quantitative. Consider the action a in figure 7. The condition c_i consists of only propositional variables and c_i, $i = 1, .., k$ are mutually exclusive. c_{k+1} is optional and represents the complementary condition that the precondition is not satisfied. So c_i, $i = 1..k+1$, are mutually exclusive and exhaustive. S_{ij}, $i = 1..k+1$ and $j = 1..m_{k+1}$, are propositional expressions and R_{ij}, $i = 1, .., k+1$ and $j = 1, .., m_{k+1}$, are sets of quantitative variables. Each pair (S_{ij}, R_{ij}) represents a state of the world after performing a under condition c_i. $f_{ij}()$ returns the value of the new quantitative variables after one execution of action a when condition c_i is satisfied and outcome j happens. The arguments of $f_{ij}()$ are the values of quantitative variables right before the execution of a is started. We assume that the $f_{ij}()$'s are increasingly monotonic. At the c_{k+1} branch, we have the probability 1 that the state remains the same.

We make the following assumptions:

- $\forall i, i', j$: either $S_{ij} \Rightarrow c_{i'}$ or $S_{ij} \wedge c_{i'}$ is unsatisfiable. This assumption facilitates reasoning about repeating a. After an outcome of a occurs, we can be sure which condition branch is satisfied in the next performance of a. If the pair S_{ij} and $c_{i'}$ does not satisfy the above condition, we can split the probability branch (S_{ij}, R_{ij}) into two branches $(S_{ij} \wedge c_{i'}, R_{ij})$ and $(S_{ij} \wedge \neg c_{i'}, R_{ij})$ with appropriate probability values to get a new action representation satisfying the condition.
- $\forall i : S_{ij} \wedge S_{ij'}$ is unsatisfied if $j \neq j'$. This forces the probabilistic branches in one condition branch of an action to be exclusive. Under this assumption, the state after performing a under condition c_i satisfies only one S_{ij}. This can always be achieved by splitting the probability branches S_{ij} and $S_{ij'}$ in the following way. Remove these two branches and add each of the following for which the corresponding propositional part is satisfiable: $(S_{ij} \wedge \neg S_{ij'}, R_{ij}), (\neg S_{ij} \wedge S_{ij'}, R_{ij'})$ and $(S_{ij} \wedge S_{ij'}, [min(R_{ij}, R_{ij'}), max(R_{ij}, R_{ij'})])$ with appropriate probability values. In the last case, each quantitative variable belongs to some interval of real values.
- $\forall i, i', j, j'$: if $i \neq i'$ then either $S_{ij} = S_{i'j'}$ or $S_{ij} \wedge S_{i'j'}$ is unsatisfiable. This assumption guarantees that when we augment a condition branch of an action with the dummy outcomes, the above condition is still satisfied. If this condition is violated, the above process can be applied to get a new action representation satisfying it.

Let S_k^o, $k = 1, .., r$ be all possible outcomes of a. In general, an action a can have several *exit outcomes*, the outcomes which do not allow the repeating of a. If the number of exit outcomes is zero we have an action repeated exactly n times, which we denote as $\mathbf{a}^n$.

In the first step, we augment the action by dummy probability branches. For each condition branch c_i, we add one probability branch, with probability 0, for each outcome S_k^o which is not in the set $S_{i1}, .., S_{im_i}$ (see figure 8). The function $f()$ associated with each of these states is the identity function.

After all the necessary transformations, we have a new action tree on which we construct a recursive definition of the abstracted repeating action. The tree representation of the abstracted $a^{\uparrow n}$ is like in figure 9. Because the functions $f_{ij}()$ may be in different forms, in general we can only conclude that the value of a quantitative variable is within a certain range.

The values of $p_{ij}^{(n)}, L_{ij}^{(n)}(R_0)$ and $U_{ij}^{(n)}(R_0)$ can be determined recursively by:

$n = 1$

$$p_{ij}^{(1)} = p_{ij}, \forall i, j$$

$$L_{ij}^{(1)} = U_{ij}^{(1)} = f_{ij}(R_0), \forall i, j$$

$n > 1$

$$p_{ij}^{(n)} = \sum_{S_{i'j'} = S_{ij}} p_{i'j'} (\sum_{S_{ik} \Rightarrow C_{i'}} p_{ik}^{(n-1)})$$

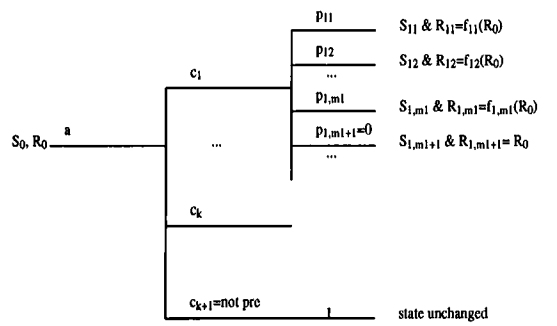

Figure 8: Augmented general action.

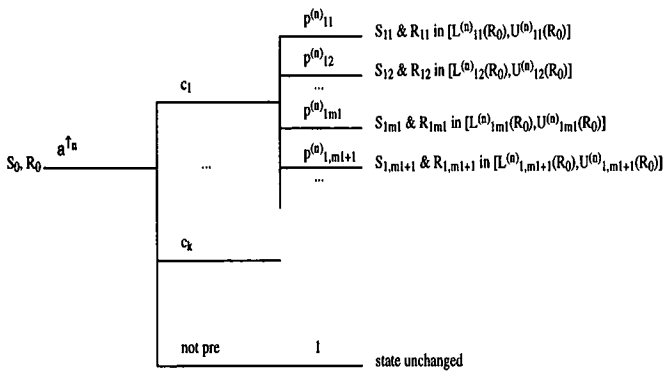

Figure 9: An abstract description of $\mathbf{a}^{\uparrow n}$.

$$L_{ij}^{(n)}(R_0) =$$

$$\min_{(S_{i'j'}=S_{ij} \wedge S_{ik} \Rightarrow C_{i'} \wedge p_{i'j'} \neq 0 \wedge p_{ik}^{(n-1)} \neq 0)} f_{i'j'}(L_{ik}^{(n-1)}(R_0))$$

$$U_{ij}^{(n)}(R_0) =$$

$$\max_{(S_{i'j'}=S_{ij} \wedge S_{ik} \Rightarrow C_{i'} \wedge p_{i'j'} \neq 0 \wedge p_{ik}^{(n-1)} \neq 0)} f_{i'j'}(U_{ik}^{(n-1)}(R_0))$$

The above abstraction process can also be applied to the case in which a is itself an abstract action or a macro-operator [Doan and Haddawy, 1995].

4.3 Declining probability of success

Some of the exit outcomes are successful outcomes and usually the repetition of a is needed when the last outcome is not yet what the planner expects. The further repetition of a may signal that a is not a good choice among the set of actions to achieve what the planner expects. In the pick-up-block example, if the block is still on the floor after the action is performed, the planner may reasonably think that other actions should be considered. We can capture this idea by decreasing the probability of success in the next performance of a, say from .9 to .8. In general, instead of the probability p_{ij} for outcome j in condition c_i, we use a function $P_{ij}(p_{ij}, n)$ where n is a positive integer indicating the number of times a has been repeated. When a is performed the first time in a loop, the probability of outcome j when c_i occurs is $P_{ij}(p_{ij}, 1) = p_{ij}$. In the n^{th} time, that probability is $P_{ij}(p_{ij}, n)$. The two position function $P_{ij}()$ must satisfy the following constraints: $\sum_j P_{ij}(., n) = 1 \forall i, n$, and $P_{ij}(x, n)$ is nonincreasing in n for successful exit outcomes and nondecreasing in n for failure branches. The above abstraction process can be easily extended to incorporate such probability varying functions.

5 Future Research

The DRIPS decision-theoretic refinement planning system [Haddawy and Suwandi, 1994] efficiently finds optimal plans by searching through a space of plans defined by an abstraction hierarchy. The hierarchy is used to prune away suboptimal classes of plans. We are investigating ways to incorporate the loop construct and its abstraction presented in this paper into the DRIPS system. It also appears that our representation of iterative loops could be incorporated into a probabilistic least-commitment planner like BURIDAN [Kushmerick *et al.*, 1994].

References

[Doan and Haddawy, 1995] A. Doan and P. Haddawy. Generating macro operators for decision-theoretic

planning. In *Working Notes of the AAAI Spring Symposium on Extending Theories of Action*, Stanford, March 1995.

[Haddawy and Doan, 1994] P. Haddawy and A.H. Doan. Abstracting probabilistic actions. In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, pages 270–277, Seattle, July 1994. Available via anonymous FTP from `pub/tech_reports` at `ftp.cs.uwm.edu`.

[Haddawy and Hanks, 1992] P. Haddawy and S. Hanks. Representations for decision-theoretic planning: Utility functions for deadline goals. In B. Nebel, C. Rich, and W. Swartout, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference (KR92)*, pages 71–82. Morgan Kaufmann, San Mateo, CA, 1992.

[Haddawy and Suwandi, 1994] P. Haddawy and M. Suwandi. Decision-theoretic refinement planning using inheritance abstraction. In *Proceedings, Second International Conference on AI Planning Systems*, pages 266–271, June 1994. Available via anonymous FTP from `pub/tech_reports` at `ftp.cs.uwm.edu`.

[Hanks, 1990] S. Hanks. *Projecting Plans for Uncertain Worlds.* PhD thesis, Yale University, January 1990.

[Kushmerick *et al.*, 1994] N. Kushmerick, S. Hanks, and D. Weld. An algorithm for probabilistic least-commitment planning. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 1073–1078, Seattle, 1994.

[Musliner, 1994] D.J. Musliner. Using abstraction and nondeterminism to plan reaction loops. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 1036–1041, Seattle, August 1994.

[Wilkins, 1988] D.E. Wilkins. *Practical Planning.* Morgan Kaufmann, San Mateo, 1988.