Chapter 5

Probabilistic planning

Probabilistic planning is an extension of nondeterministic planning with information on the probabilities of nondeterministic events.

Probabilites are important in quantifying the costs and success probabilities of plans when the actions are nondeterministic. In many applications it is not sufficient just to have a plan. It is important to have a plan that is efficient in the sense that the cost of the actions does not outweigh the benefits of reaching the goals. On some other problems there are no plans that are guaranteed to reach the goals. In these cases it is important to maximize the probability of reaching the goals, and hence it is vitally important to use information on the probabilities of different effects of operators.

Probabilities complicate planning, both conceptually and computationally. Whereas in the non-probabilistic of conditional planning with partial observability it is sufficient to work in a finite discrete belief space, probabilities make the belief space continuous and thereby infinite.

In this section a number of algorithms for probabilistic planning are presented. In Sections 5.1 and 5.2 we present the transition system model with probabilities that extend the definitions given in Sections 2.1 and 2.3 for non-succinct and succinct transition systems, respectively. Like in Chapter 4 we start from planning with full observability in Section 5.3. Many probabilistic planning problems with full observability are closely related to Markov decision processes [Puterman, 1994].

5.1 Probabilistic transition systems

In many types of probabilistic planning problems considered in the literature the objective is not to reach one of a set of designated goal states. Instead, the objective is to act in a way that maximizes the *rewards* or minimizes the *costs*. Planning problems with a designated set of goal states can be expressed in terms of rewards, but not vice versa.

Definition 5.1 *A* probabilistic transition system *is a 5-tuple* $\Pi = \langle S, I, O, G, P \rangle$ *where*

1. *S is a finite set of states,*
2. *I is a probability distribution over S,*
3. *O is a finite set of actions o that are partial functions that map each state to a probability distribution over S,*

4. *$G \subseteq S$ is the set of goal states, and*

5. *$P = (C_1, \ldots, C_n)$ is a partition of S to classes of observationally indistinguishable states satisfying $\bigcup\{C_1, \ldots, C_n\} = S$ and $C_i \cap C_j = \emptyset$ for all i, j such that $1 \leq i < j \leq n$.*

An action o is *applicable* in states for which $o(s)$ is defined. These states we denote by $\text{prec}(o) = \{s \in S | o(s) \text{ is defined }\}$. Below, we will denote the set of actions applicable in a state $s \in S$ by $O(S)$. We also require that $O(s)$ is non-empty for every $s \in S$.

A major difference to the definition of Markov decision processes [Puterman, 1994] is that $o \in O$ are partial functions. This means that an action does not associate every state with a probability distribution because an action is not necessarily applicable in all states.

Instead of using a designated set of goal states and have reaching a goal state or staying in the goal states as an objective, in many types of planning problems the objective is to maximize rewards or minimize costs. To formalize this we use instead of a set of goal states a cost function that associates every action and state a numerical cost.

Definition 5.2 *A* probabilistic transition system with rewards *is a 5-tuple $\Pi = \langle S, I, O, C, P\rangle$ where the components S, I, O and P are as in Definition 5.1 and $C : O \times S \to \mathcal{R}$ is a function from actions and states to real numbers, indicating the* cost *associated with an action in a given state.*

5.2 Succinct probabilistic transition systems

Probabilistic transition system can be represented exponentially more succinctly in terms of state variables and operators.

Definition 5.3 *Let A be a set of state variables. An* operator *is a pair $\langle c, e\rangle$ where c is a propositional formula over A (the* precondition*), and e is an* effect *over A. Effects over A are recursively defined as follows.*

1. *a and $\neg a$ for state variables $a \in A$ are effects over A.*

2. *$e_1 \wedge \cdots \wedge e_n$ is an effect over A if $e_1, \ldots, e_n$ are effects over A (the special case with $n = 0$ is the empty effect $\top$.)*

3. *$c \rhd e$ is an effect over A if c is a formula over A and e is an effect over A.*

4. *$p_1 e_1 | \cdots | p_n e_n$ is an effect over A if $n \geq 2$ and $e_1, \ldots, e_n$ for $n \geq 2$ are effects over A and $p_1, \ldots, p_n$ are real numbers such that $p_1 + \cdots + p_n = 1$ and $0 \leq p_i \leq 1$ for all $i \in \{1, \ldots, n\}$.*

Operators map states to probability distributions over their successor states.

Definition 5.4 (Operator application) *Let $\langle c, e\rangle$ be an operator over A. Let s be a state (a valuation of A). The operator is* applicable in *s if $s \models c$ and for every set $E \in [e]_s$ the set $\bigcup\{M | \langle p, M\rangle \in E, p > 0\}$ is consistent.*

Recursively assign each effect e a set $[e]_s$ of pairs $\langle p, M\rangle$ where p is a probability $0 \leq p \leq 1$ and M is a set of literals a and $\neg a$ where $a \in A$.

1. *$[a]_s = \{\langle 1, \{a\}\rangle\}$ and $[\neg a]_s = \{\langle 1, \{\neg a\}\rangle\}$ for $a \in A$.*

2. *$[e_1 \wedge \cdots \wedge e_n]_s = \{\langle \Pi_{i=1}^n p_i, \bigcup_{i=1}^n M_i\rangle | \langle p_1, M_1\rangle \in [e_1]_s, \ldots, \langle p_n, M_n\rangle \in [e_n]_s\}$.*

3. *$[c' \rhd e]_s = [e]_s$ if $s \models c'$ and $[c' \rhd e]_s = \{\langle 1, \emptyset\rangle\}$ otherwise.*

4. *$[p_1 e_1 | \cdots | p_n e_n]_s = \{\langle p_1 \cdot p, e\rangle | \langle p, e\rangle \in [e_1]_s\} \cup \cdots \cup \{\langle p_n \cdot p, e\rangle | \langle p, e\rangle \in [e_n]_s\}$*

For handling effects like $(0.2a|0.8b) \wedge (0.2a|0.8b)$, which produces sets $\{a\}, \{a,b\}, \{a,b\}, \{b\}$ respectively with probabilities 0.04, 0.16, 0.16 and 0.64, the sets in this definition are understood as multisets so that the probability 0.16 of $\{a,b\}$ is counted twice. Alternatively, in (4) the union of sets is defined so that for example $\{\langle 0.2, \{a\}\rangle\} \cup \{\langle 0.2, \{a\}\rangle\} = \{\langle 0.4, \{a\}\rangle\}$: same sets of changes are combined by summing their probabilities.

The successor states of s under the operator are ones that are obtained from s by making the literals in M for $\langle p, M\rangle \in [e]_s$ true and retaining the truth-values of state variables not occurring in M. The probability of a successor state is the sum of the probabilities p for $\langle p, M\rangle \in [e]_s$ that lead to it.

Definition 5.5 *A* succinct probabilistic transition system *is a 5-tuple $\Pi = \langle A, I, O, G, V\rangle$ where*

1. *A is a finite set of state variables,*

2. *I which describes a probability distribution over the possible initial states is a set of pairs $\langle p, \phi\rangle$ where p is a number such that $0 \leq p \leq 1$ and ϕ is a formula over A such that $(\sum_{s \in S, s \models \phi_1} p_1) + \cdots + (\sum_{s \in S, s \models \phi_n} p_n) = 1$ where $I = \{\langle p_1, \phi_1\rangle, \ldots, \langle p_n, \phi_n\rangle\}$,*

3. *O is a finite set of operators over A,*

4. *G is a formula over A describing the goal states, and*

5. *$V \subseteq A$ is the set of observable state variables.*

Definition 5.6 *A* succinct probabilistic transition system with rewards *is a 5-tuple $\Pi = \langle A, I, O, C, V\rangle$ where the components A, I, O and V are as in Definition 5.5 and R is a function from operators to pairs $\langle \phi, r\rangle$ where ϕ is a formula over A and r is a real number indicating the* cost *associated with an action in a given state: cost of operator $o \in O$ in state s is r if there is $\langle \phi, r\rangle \in C(o)$ such that $s \models \phi$. For this to be well defined there may be no $\{\langle \phi_1, r_1\rangle, \langle \phi_2, r_2\rangle\} \subseteq C(o)$ such that $\phi_1 \wedge \phi_2$ is satisfiable.*

We can associate a probabilistic transition system with every succinct probabilistic transition system.

5.3 Problem definition

A given plan produces infinite sequences of rewards $r_1, r_2, \ldots$. Clearly, if the planning problem has several initial states or if the actions are nondeterministic this sequence of rewards is not unique. In either case, possible plans are assessed in terms of these rewards, and there are several possibilities how good plans are defined. Because the sequences are infinite, we in general cannot simply take their sum and compare them. Instead, several other possibilities have been considered.

1. Expected total rewards over a finite horizon.

   This is a natural alternative that allows using the normal arithmetic sum of the rewards. However, there is typically no natural bound on the horizon length.

2. Expected average rewards over an infinite horizon.

   This is for many applications that involve very long actions sequences the most natural way of assessing plans. However, there are several technical complications that make average rewards difficult to use.

3. Expected discounted rewards over an infinite horizon.

   This is the most often used criterion in connection with Markov decision processes. Discounting means multiplying the ith reward by λ^{i-1} and it means that early rewards are much more important than rewards obtained much later. The discount constant λ has a value strictly between 0.0 and 1.0. The sum of the geometrically discounted rewards is finite. Like with choosing the horizon length when evaluating plans with respect to their behavior within a finite horizon, it is often difficult to say why a certain discount constant λ is used.

For the latter two infinite horizon problems there always is an optimal plan that is a mapping from states to actions. For finite horizon problems the optimal actions in a given state at different time points may be different. The optimal plans are therefore time-dependent.

5.4 Algorithms for finding finite horizon plans

Conceptually the simplest probabilistic planning is when plan executions are restricted to have a finite horizon of length N. We briefly describe this problem to illustrate the techniques that are used in connection with the infinite horizon planning problems.

The optimum values $v_i(s)$ that can be obtained in state $s \in S$ at time point $i \in \{1, \dots, N\}$ fulfill the following equations.

$$\begin{aligned} v_N(s) &= \max_{a \in O(s)} R(s,a) \\ v_i(s) &= \max_{a \in O(s)} \left(R(s,a) + \sum_{s' \in S} p(s'|s,a) v_{i+1}(s') \right), \text{ for } i \in \{1, \dots, N-1\} \end{aligned}$$

The value at the last stage N is simply the best immediate reward that can be obtained, and values of states for the other stages are obtained in terms of the values of states for the later stages.

These equations also directly yield an algorithm for computing the optimal values and optimal plans: first compute v_N, then v_{N-1}, v_{N-2} and so on, until v_1 is obtained. The action to be taken in state $s \in S$ at time point i is $\pi(s,i)$ defined by

$$\begin{aligned} \pi(s,N) &= \arg\max_{a \in O(s)} R(s,a) \\ \pi(s,i) &= \arg\max_{a \in O(s)} \left(R(s,a) + \sum_{s' \in S} p(s'|s,a) v_{i+1}(s') \right), \text{ for } i \in \{1, \dots, N-1\} \end{aligned}$$

5.5 Algorithms for finding plans under discounted rewards

The value $v(s)$ of a state $s \in S$ is the discounted sum of the expected rewards that can be obtained by choosing the best possible action in s and assuming that the best possible actions are also chosen in all the possible successor states. The following equations, one for each state $s \in S$, characterize the relations between the values of states of a stochastic transition system under an optimal plan and geometrically discounted rewards with discount constant λ.

$$v(s) = \max_{a \in O(s)} \left(R(s,a) + \sum_{s' \in S} \lambda p(s'|s,a) v(s') \right) \tag{5.1}$$

These equations are called the optimality equations or the Bellman equations, and they are the basis of the most important algorithms for finding optimal plans for probabilistic planning problems with full observability.

5.5.1 Evaluating the value of a given plan

Given a plan π its value under discounted rewards with discount constant λ satisfies the following equation for every $s \in S$.

$$v(s) = R(s,\pi(s)) + \sum_{s' \in S} \lambda p(s'|s,\pi(s)) v(s') \tag{5.2}$$

This yields a system of linear equation with $|S|$ equations and unknowns. The solution of these equations yields the value of the plan in each state.

5.5.2 Value iteration

The value iteration algorithm finds an approximation of the value of the optimal λ-discounted plan within a constant ϵ, and a plan with at least this value.

1. $n := 0$
2. Assign (arbitrary) initial values to $v^0(s)$ for all $s \in S$.
3. For each $s \in S$, assign

   $$v^{n+1}(s) := \max_{a \in O(s)} \left(R(s,a) + \sum_{s' \in S} \lambda p(s'|s,a) v^n(s') \right)$$

   If $|v^{n+1}(s) - v^n(s)| < \frac{\epsilon(1-\lambda)}{2\lambda}$ for all $s \in S$ then go to step 4.

   Otherwise, set $n := n + 1$ and go to step 3.
4. Assign

   $$\pi(s) := \arg\max_{a \in O(s)} \left(R(s,a) + \sum_{s' \in S} \lambda p(s'|s,a) v^{n+1}(s') \right)$$

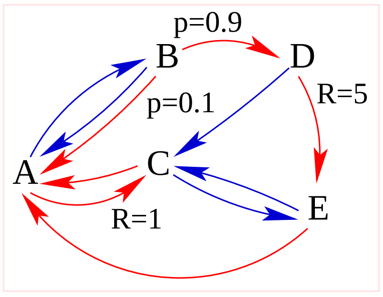

Figure 5.1: A stochastic transition system

Theorem 5.7 *Let v_π be the value function of the plan produced by the value iteration algorithm, and let v^* be the value function of an optimal plan. Then $|v^*(s) - v_\pi(s)| \leq \epsilon$ for all $s \in S$.*

Notice that unlike in partially observable planning problems, under full observability there is never a trade-off between the values of two states: if the optimal value for state s_1 is r_1 and the optimal value for state s_2 is r_2, then there is one plan that achieves these both.

Example 5.8 Consider the stochastic transition system in Figure 5.1. Only one of the actions is nondeterministic and only in state B, and all the other actions and states have zero reward except one of the actions in states A and D, with rewards 1 and 5, respectively. ■

5.5.3 Policy iteration

The second, also rather widely used algorithm for finding plans, is policy iteration[1]. It is slightly more complicated to implement than value iteration, but it typically converges after a smaller number of iterations, and it is guaranteed to produce an optimal plan.

The idea is to start with an arbitrary plan (assignment of actions to states), compute its value, and repeatedly choose for every state an action that is better than its old action.

1. Assign $n := 0$.
2. Let π^0 be any mapping from states to actions.
3. Compute the values $v^n(s)$ of all $s \in S$ under π^n.
4. Let $\pi^{n+1}(s) = \arg\max_{a \in O(s)} \left(R(s,a) + \sum_{s' \in S} \lambda p(s'|s,a) v^n(s')\right)$.
5. Assign $n := n + 1$.
6. If $n = 1$ or $v^n \neq v^{n-1}$ then go to 3.

Theorem 5.9 *The policy iteration algorithm terminates after a finite number of steps and returns an optimal plan.*

Proof: Outline: There is only a finite number of different plans, and at each step the new plan assigns at least as high a value to each state as the old plan. □

[1] In connection with Markov decision processes the word *policy* is typically used instead of the word *plan*.

It can be shown that the convergence rate of policy iteration is always at least as fast as that of value iteration [Puterman, 1994], that is, the number of iterations needed for finding an ϵ-optimal plan for policy iteration is never higher than the number of iterations needed by value iteration.

In practise policy iteration often finds an optimal plan after just a few iterations. However, the amount of computation in one round of policy iteration is substantially higher than in value iteration, and value iteration is often considered more practical.

5.5.4 Implementation of the algorithms with ADDs

Similarly to the techniques in Section 4.2 that allow representing state sets and transition relations as formulae or binary decision diagrams, also probabilistic planning algorithms can be implemented with data structures that allow the compact representation of probability distributions.

A main difference to the non-probabilistic case (Sections 4.4.1 and 4.4.2) is that for probabilistic planning propositional formulae and binary decision diagrams are not suitable for representing the probabilities of nondeterministic operators nor the probabilities of the value functions needed in the value and policy iteration algorithms. *Algebraic decision diagrams* ADDs are a generalization of BDDs can represent probability distributions. (Section 2.2.3).

In Section 4.1.2 we gave a translation from nondeterministic operators to propositional formulae. The definition of nondeterministic operators and the translation does not use probabilities.

Next we define a similar translation from nondeterministic operators to ADDs that represents the probabilities. The translation is based on a function $\tau_B^{prob}(e)$ that translates an effect e with that possibly affects state variables in B to an ADD.

$$\begin{aligned}
\tau_B^{prob}(e) &= \tau_B(e) \text{ when } e \text{ is deterministic} \\
\tau_B^{prob}(p_1 e_1 | \cdots | p_n e_n) &= p_1 \cdot \tau_B^{prob}(e_1) + \cdots + p_n \cdot \tau_B^{prob}(e_n) \\
\tau_B^{prob}(e_1 \wedge \cdots \wedge e_n) &= \tau_{B\backslash(B_2 \cup \cdots \cup B_n)}^{prob}(e_1) \cdot \tau_{B_2}^{prob}(e_2) \cdot \ldots \cdot \tau_{B_n}^{prob}(e_n) \\
&\quad \text{where } B_i = \textit{changes}(e_i) \text{ for all } i \in \{1, \ldots, n\}
\end{aligned}$$

The first part of the translation $\tau_B^{prob}(e)$ for deterministic e is the translation of deterministic effects we presented in Section 3.6.2, but restricted to state variables in B. The result of this translation is a normal propositional formula, which can be further transformed to a BDD and an ADD with only two terminal nodes 0 and 1. The other two cases cover all nondeterministic effects in normal form.

The translation of an effect e in normal form into an ADD is $\tau_A^{prob}(e)$ where A is the set of all state variables. Translating an operator $o = \langle c, e \rangle$ to an ADD representing its incidence matrix is as $T_o = c \cdot \tau_A^{prob}(e)$, where c is the ADD representing the precondition.

Example 5.10 Consider effect $(0.2\neg A|0.8A) \wedge (0.5(b \rhd \neg b)|0.5\top)$. The two conjunct translated to functions

aa'	f_a
00	0.2
01	0.8
10	0.2
11	0.8

bb'	f_b
00	1.0
01	0.0
10	0.5
11	0.5

Notice that the sum of the probabilities of the successor states is 1.0. These functions are below depicted in the same table. Notice that the third column, with the two functions componentwise multiplied, has the property that the sum of successor states of each state is 1.0.

$aba'b'$	f_a	f_b	$f_a \cdot f_b$
0000	0.2	1.0	0.2
0001	0.2	0.0	0.0
0010	0.8	1.0	0.8
0011	0.8	0.0	0.0
0100	0.2	0.5	0.1
0101	0.2	0.5	0.1
0110	0.8	0.5	0.4
0111	0.8	0.5	0.4
1000	0.2	1.0	0.2
1001	0.2	0.0	0.0
1010	0.8	1.0	0.8
1011	0.8	0.0	0.0
1100	0.2	0.5	0.1
1101	0.2	0.5	0.1
1110	0.8	0.5	0.4
1111	0.8	0.5	0.4

■

We represent the rewards produced by operator $o = \langle c, e \rangle \in O$ in different states compactly as a list $R(o) = \{\langle \phi_1, r_1 \rangle, \ldots, \langle \phi_n, r_n \rangle\}$ of pairs $\langle \phi, r \rangle$, meaning that when o is applied in a state satisfying ϕ the reward r is obtained. In any state only one of the formulae ϕ_i may be true, that is $\phi_i \models \neg\phi_j$ for all $\{i, j\} \subseteq \{1, \ldots, n\}$ such that $i \neq j$. If none of the formula is true in a given state, then the reward is zero. Hence R_o is simply a mapping from states to a real numbers.

The reward functions $R(o)$ can be easily translated to ADDs. First construct the BDDs for $\phi_1, \ldots, \phi_n$ and then multiply them with the respective rewards as

$$R_o = r_1 \cdot \phi_1 + \cdots + r_n \cdot \phi_n - \infty \cdot \neg c.$$

The summand $\infty \cdot \neg c$ handles the case in which the precondition of the operator is not satisfied: application yields immediate reward minus infinity. This prevent using the operator in any state.

Similarly, the probability distribution on possible initial states can be represented as $I = \{\langle \phi_1, p_1 \rangle, \ldots, \langle \phi_n, p_n \rangle\}$ and translated to an ADD.

Now the value iteration algorithm can be rephrased in terms of ADD operations as follows.

1. Assign $n := 0$ and let v^n be an ADD that is constant 0.
2. $$v^{n+1} := \max_{\langle c,e \rangle = o \in O} \left(R_o + \lambda \cdot \exists A'.(T_o \cdot (v^n[A'/A])\right) \text{ for every } s \in S$$

   If all terminal nodes of ADD $|v^{n+1} - v^n|$ are $< \frac{\epsilon(1-\lambda)}{2\lambda}$ then stop.

   Otherwise, set $n := n + 1$ and repeat step 2.

5.6 Literature

A comprehensive book on (fully observable) Markov decision processes has been written by Puterman [1994], and our presentation of the algorithms in Section 5.5 (5.5.2 and 5.5.3) follows that of Puterman. The book represents the traditional research on MDPs and uses exclusively enumerative representations of state spaces and transition probabilities. The book discusses all the main optimality criteria as well as algorithms for solving MDPs by iterative techniques and linear programming. There are also many other books on solving MDPs.

A planning system that implements the value iteration algorithm with ADDs is described by Hoey et al. [1999] and is shown to be capable of solving problems that could not be efficiently solved by conventional implementations of value iteration.

The best known algorithms for solving partially observable Markov decision processes were presented by Sondik and Smallwood in the early 1970's [Sondik, 1978; Smallwood and Sondik, 1973] and even today most of the work on POMDPs is based on those algorithms [Kaelbling *et al.*, 1998]. In this section we have presented the standard value iteration algorithm with the simplification that there is no sensing uncertainty, that is, for every state the same observation, dependent on the state, is always made.

The most general infinite-horizon planning problems and POMDP solution construction are undecidable [Madani *et al.*, 2003]. The complexity of probabilistic planning has been investigated for example by Mundhenk et al. [2000] and Littman [1997].

Bonet and Geffner [2000] and Hansen and Zilberstein [2001] have presented algorithms for probabilistic planning with Markov decision processes that use heuristic search.

5.7 Exercises

5.1 Prove that on each step of policy iteration the policy improves.