9

Graphical Languages for Specification of Decision Problems

A Bayesian network serves as a model for a part of the world, and the relations in the model reflect causal impact between events. The reason for building these computer models is to use them in taking decisions. In other words, the probabilities provided by the network are used to support some kind of decision making. In principle, there are two kinds of decisions, namely *test decisions* and *action decisions*.

A test decision is a decision to look for more evidence to be entered into the model, and an action decision is a decision to change the state of the world. In real life, this distinction is not very sharp; tests may have side effects, and by performing a treatment against a disease, evidence on the diagnosis may be acquired. In order to be precise, we should say that decisions have two *aspects*, namely a test aspect and an action aspect. The two aspects are handled differently in connection with Bayesian networks, and accordingly we treat them separately.

Although both observations and actions may change the probability distributions in the model, they are fundamentally different. To highlight this, consider the example in Figure 9.1.

A wheat type may be genetically resistant to mildew. If so, there will be no attack, and this has an impact on the quality of the crop. If you *observe* that there is no attack, the probabilities for *Resistance* and *Crop* are changed. If you, on the other hand, prevent an attack through spraying and thereby fix the state of *Attack* to *no*, then it has no impact on your belief about *Resistance*. That is, *the impact of actions can only follow the direction of the causal links*.

The example stresses the important point already made in Section 3.2.6 concerning the use of Bayesian networks. Using Bayes' theorem, it is easy to establish the model in Figure 9.2, which reflects a kind of diagnostic reasoning.

From the point of view of entering evidence and propagating probabilities, the two Bayesian networks in Figure 9.1 and Figure 9.2 represent the same joint probability distribution, so why bother emphasizing that the links in the network should be causal links? The difference becomes apparent when one

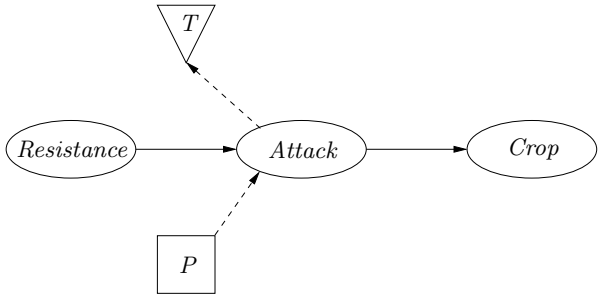


Fig. 9.1. A simple Bayesian network with an action and a test attached. The decision (Prevention) can by spraying fix the state of *Attack* to *no*. The test T can determine the state of *Attack*.

Fig. 9.2. A Bayesian network equivalent to the one in Figure 9.1.

sprays. In Figure 9.2, spraying will change the probability of resistance but it will have no impact on the crop.

In Section 9.1 we show how to extend a Bayesian network to cope with a single decision, and in Section 9.2 we describe fundamentals of rational decision making. Sections 9.3–9.5 present various graphical frameworks for modeling decision problems with several decisions involved, and in Section 9.6 we deal with problems that have an unbounded time horizon.

9.1 One-Shot Decision Problems

A Bayesian network provides a model of the world that can be used in making decisions. The typical situation is that we have observed some of the variables in the domain and based on these observations we make an inquiry to the Bayesian network about some other set of variables (probability updating). The result of the inquiry is in turn used in the subsequent decision-making process.

This type of application of Bayesian networks can be taken one step further, so that rather than keeping the model separated from the decision-making process, you could combine these two parts. That is, not only does the final model reveal the structure of the decision problem, but it can also be used to give advice about the decisions. In the simple situation in which only a single decision is to be made, the Bayesian network can readily be extended to reflect the structure of the decision problem.

9.1.1 Fold or Call?

Consider the poker example in Section 3.1.4 as extended in Exercise 3.13 with the variables *MH* ("my hand" having the same states as *OH2*) and *BH* ("best hand" with the states *me*, *opponent*, and *draw*), see Figure 9.3. The conditional probability distribution for *BH* is a deterministic function of *OH2* and *MH*.

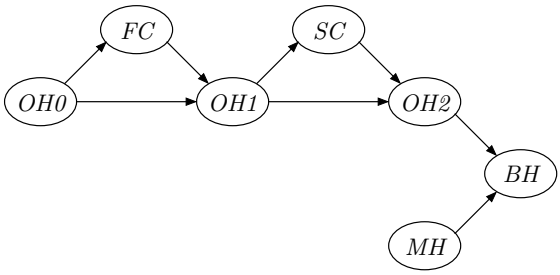
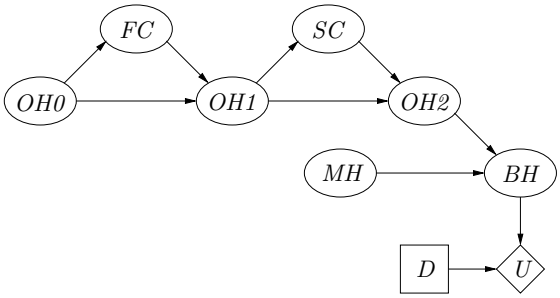


Fig. 9.3. The poker model extended with variables for my hand and best hand.

The reason I am interested in knowing which hand is best is that I shall take a decision on an action. For this game, the rules are that we both placed \$1 on the table to get the initial hand, and after the rounds of card changing, my opponent places \$1 extra (in this game she is forced to place \$1 regardless of her hand). Now, I may either *fold* or *call*. If I fold, my opponent takes the pot, and if I call, I place \$1 on the table, and we compare the hands. The player with the best hand takes the pot (in case of a draw we share).

My decision problem in deciding to fold or to call can be represented graphically by extending the Bayesian network with a couple of extra nodes. The decision options are represented by a rectangular node *D* with states *fold* and *call*. Another type of node, *U*, represents the possible outcomes in dollars. The node *U* is called a *utility node*, and the outcomes are called *utilities*. The variables determining the outcomes are *BH* and *D*, and this is shown graphically through directed links from *BH* and *D* to the diamond-shaped node *U*. See Figure 9.4. Note that in this example the utilities also include the initial \$1 that I was forced to put on the table.

When I have extended the Bayesian network to the model in Figure 9.4, I can use the model to give advice on the decision *D*. I have observed my opponent's change of cards (for example, two cards and one card), and I know my own hand (for example, a flush). The probability for *BH* (best hand) is calculated, and it is used to calculate EU(*call*), the *expected utility* of calling: the sum of the various wins and losses weighted by their probability. The formula for EU(*call*) is



		D	
		fold	*call*
BH	*me*	−1	2
	opponent	−1	−2
	draw	−1	0

Fig. 9.4. Graphical representation of my decision problem of whether to fold or call. The variable D is a decision variable. The variable U represents the outcome in \$ (shown in the table), and the links into U indicate that the outcomes of the game (only) depend on D and BH.

$$\begin{aligned}\mathrm{EU}(\mathit{call}) &= \sum_{BH} U(BH, \mathit{call}) P(BH \,|\, \text{evidence}) \\ &= P(BH = \mathit{me} | FC = \mathit{two}, SC = \mathit{one}, MH = \mathit{flush}) U(BH = \mathit{me}, \mathit{call}) \\ &\quad + P(BH = \mathit{draw} | FC = \mathit{two}, SC = \mathit{one}, MH = \mathit{flush}) \\ &\qquad U(BH = \mathit{draw}, \mathit{call}) \\ &\quad + P(BH = \mathit{opponent} | FC = \mathit{two}, SC = \mathit{one}, MH = \mathit{flush}) \\ &\qquad U(BH = \mathit{opponent}, \mathit{call}).\end{aligned}$$

If you use the probabilities found in Section 3.2.3, the expected utility of calling is

$$\mathrm{EU}(\mathit{call}) = 0.4 \cdot 2 + 0.054 \cdot (-2) + 0.546 \cdot 0 = 0.692,$$

and since the expected utility of folding is −1, I should call.

9.1.2 Mildew

Two months before the harvest of a wheat field, the farmer observes the state Q of the crop, and he observes whether it has been attacked by mildew, M. If there is an attack, he will decide on a treatment with fungicides.

There are five variables:

- Q with states fair (f), not too bad (n), average (a), and good (g);
- M with states *no*, little (l), moderate (m), and severe (s);
- H (state of the crop at time of harvest) with the states from Q plus rotten (r), bad (b), and poor (p) (farmers in all countries tend to describe their harvests in pessimistic terms);
- OQ (observation of Q) with the same states as Q;
- OM (observation of M) with the same states as M.

Furthermore, there is a decision node A with decision options *no*, light (l), moderate (m), and heavy (h) and a variable M' describing the mildew attack after the decision. We define a utility function $U(H)$ giving the utility of the outcome of the harvest for each state of the crop. The cost of the decisions is modeled as a utility function C attached to A (the values of C are either negative or zero). The total utility is $U + C$. Figure 9.5 gives a model.

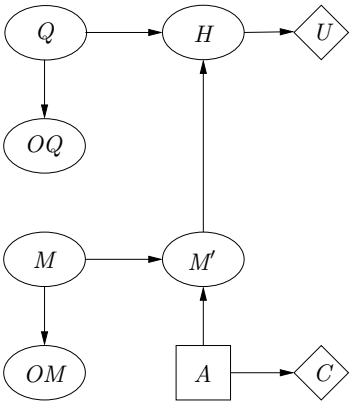


Fig. 9.5. A decision model for mildew.

With evidence e (statements on OQ and OM), the farmer wishes to determine an optimal decision (a decision of maximal expected utility). To do this, he needs to calculate the expected utility of the various options. That is, for each state a of A, we first calculate $P(H \mid A = a, e)$, and then

$$\mathrm{EU}(A \mid e) = C(A) + \sum_{H} U(H)P(H \mid A, e).$$

9.1.3 One Decision in General

The general situation with one decision variable is as described in Figure 9.6. There is a Bayesian network structure with chance nodes and directed links. The network is extended with a single decision node D that may have an impact on the variables in the structure. In other words, there may be a link from D to some chance nodes. Furthermore, there is a set of utility functions, $U_1, \ldots, U_n$, over domains $\mathcal{X}_1, \ldots, \mathcal{X}_n$.

The task is to determine the decision that yields the highest expected utility. Thus, if none of the utility nodes contain D in the domain, then with evidence e we calculate

$$\mathrm{EU}(D \mid e) = \sum_{\mathcal{X}_1} U_1(\mathcal{X}_1)P(\mathcal{X}_1 \mid D, e) + \cdots + \sum_{\mathcal{X}_n} U_n(\mathcal{X}_n)P(\mathcal{X}_n \mid D, e),$$

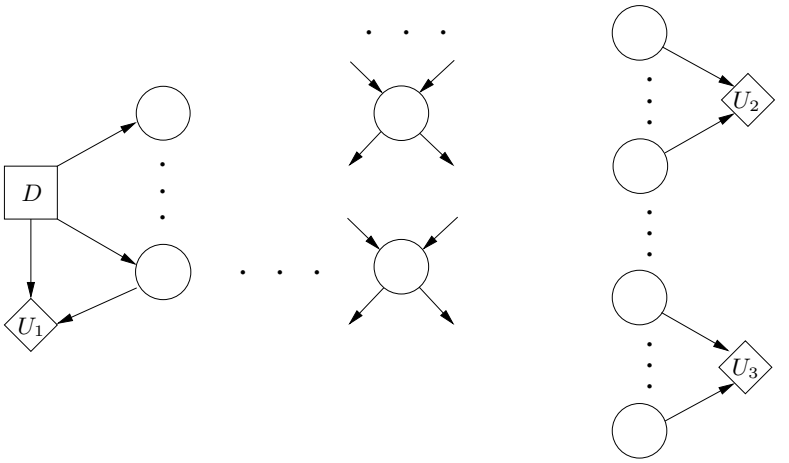


Fig. 9.6. A graphical representation of a one-action decision scenario.

and a state d maximizing $\mathrm{EU}(D = d \mid e)$ is chosen as an optimal decision. When D is contained in the domain of a utility node, such as U_1 in Figure 9.6, then we should perform the summation only over $\mathcal{X}_1 \setminus \{D\}$, and accordingly, we should use the probability distribution $P(\mathcal{X}_1 \setminus \{D\} \mid D, e)$.

A requirement of the method described above is that the decision problem contains only a single decision. When one is working with decision problems involving several decisions, things become a bit more complicated (we shall return to this issue in Sections 9.3 and 9.4).

9.2 Utilities

We treat decision problems in the framework of *theory*. Decisions are made because they may be of use in some way. Therefore, the various decisions should be evaluated on the basis of the usefulness of their consequences. We assume that "usefulness" is measured on a numerical scale called a *utility scale*, and if several kinds of utilities are involved in the same decision problem, then the scales have a common unit.

Management of Effort

In your computer science studies you attend two courses, *Graph Algorithms* and *Machine Intelligence*. In the middle of the term, you realize that you cannot keep pace. You can either reduce your effort in both courses slightly or you can decide to attend one of the courses superficially. What is the best decision?

You have three possible actions:

Gm: Keep pace in Graph Algorithms and follow Machine Intelligence superficially.
SB: Slow down in both courses.
Mg: Keep pace in Machine Intelligence and follow Graph Algorithms superficially.

The results of the actions are your final marks for the courses. The marks are integers between 0 and 5, where 0 and 1 are failing marks. You have certain expectations for the marks given your effort in the rest of the term. They are shown in Table 9.1.

	kp	*sd*	*fs*
0	0	0	0.1
1	0.1	0.2	0.1
2	0.1	0.1	0.4
3	0.2	0.4	0.2
4	0.4	0.2	0.2
5	0.2	0.1	0

$P(GA \mid effort)$

	kp	*sd*	*fs*
0	0	0	0.1
1	0	0.1	0.2
2	0.1	0.2	0.2
3	0.2	0.2	0.3
4	0.4	0.4	0.2
5	0.3	0.1	0

$P(MI \mid effort)$

Table 9.1. The conditional probabilities of the final marks in Graph Algorithms (GA) and Machine Intelligence (MI) given the efforts *keep pace* (*kp*), *slow down* (*sd*), and *follow superficially* (*fs*).

A way of solving your decision problem would be to say that the numeric value of the mark is a utility, and you want to maximize the sum of the expected marks. The calculations would then be

$$\mathrm{EU}(Gm) = \sum_{m \in GA} P(m \mid kp)m + \sum_{m \in MI} P(m \mid fs)m = 3.5 + 2.3 = 5.8,$$

$$\mathrm{EU}(SB) = \sum_{m \in GA} P(m \mid sd)m + \sum_{m \in MI} P(m \mid sd)m = 2.9 + 3.2 = 6.1,$$

$$\mathrm{EU}(Mg) = \sum_{m \in GA} P(m \mid fs)m + \sum_{m \in MI} P(m \mid kp)m = 2.3 + 3.9 = 6.2.$$

From this, you would conclude that you should follow Graph Algorithms superficially but keep pace in Machine Intelligence.

However, do the marks really reflect your utilities? If, for example, you had the same number of marks but the numeric values were $0, 5, 6, 8, 9, 10$, you would have come to another conclusion. The problem is that you cannot expect that a difference of 1 in mark number always represents the same difference in utility. Actually, in this case your subjective utility is probably not increasing in the numeric value of the mark: the rule at your university

is that if you fail, you are given another chance, but if you pass, you are not allowed to try again to get a better mark. Therefore, you find that the worst mark to get is a 2 rather than a 0!

To overcome this problem, the mark scale is mapped into a utility scale going from 0 to 1. The best possible mark (5) is given the utility 1, and the worst possible mark (2) gets the utility 0.

The intermediate marks are given utilities between 0 and 1 by imagining that you have a choice between two games:

Game 1: You get for certain the mark x;
Game 2: You get mark 5 with probability p, and you get mark 2 with probability $1 - p$.

Which game would you prefer?

If $p = 0$, you would prefer Game 1, and for $p = 1$, Game 2 would be best. For some p between 0 and 1, you would be indifferent, and *this p is the utility for the mark x.* Specifically, if you should find a value for p that would make you indifferent between games 1 and 2, then it should hold that

$$\mathrm{EU}(\textit{Game 1}) = \mathrm{EU}(\textit{Game 2}).$$

This can be rewritten as $1 \cdot U(x) = (1-p) \cdot U(2) + p \cdot U(5)$, and by exploiting that $U(2) = 0$ and $U(5) = 1$ we get $U(x) = p$.

In Table 9.2, we have performed the utility assessment for you. The utilities assessed are for only one course. We will now assume that the utility of marks for several courses is the sum of the individual utilities. Note that this is not evident (it might, for example, be that you prefer two 2's to failing both courses, which would delay your studies considerably), and an alternative could be to construct a single utility function for both courses.

Mark	0	1	2	3	4	5
Utility	0.05	0.1	0	0.6	0.8	1

Table 9.2. Utilities for the various marks (the same for both courses).

In Figure 9.7, the decision model is illustrated. To find an optimal decision, the calculations are

$$\mathrm{EU}(\textit{action}) = \sum_{m \in GA} P(m \mid \textit{action}) U_{GA}(m) + \sum_{m \in MI} P(m \mid \textit{action}) U_{MI}(m).$$

We get $\mathrm{EU}(Gm) = 1.015, \mathrm{EU}(SB) = 1.07, \mathrm{EU}(Mg) = 1.045$, and the optimal decision is therefore SB.

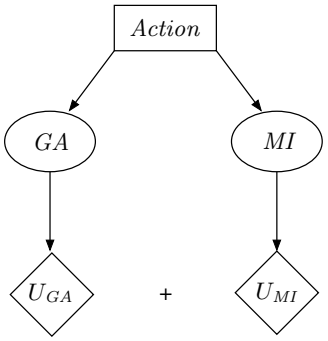


Fig. 9.7. A decision model for effort.

9.2.1 Instrumental Rationality

Beneath the principle of maximal expected utility there is a normative claim that rational decision making shall be represented as a task of calculating expected utilities and to choose an option of maximal expected utility. The question is whether this claim includes all kinds of human choice (private decisions, company decisions, political decisions, etc.). Does it cover choosing the dinner for tomorrow as well as whether to kill your husband or leave him? Does it include setting of tax rates and building dams for flood protection?

It is not claimed that humans/companies/politicians act in accordance with the principle of maximal expected utility (which can easily be disproved). The claim is that if the decision maker takes his time to analyze the situation to find out which choice seems the best, then it is irrational not to choose one of maximal expected utility.

In order not to enter into a circular argument, you need to be precise about the term *rational* without referring to utilities, and a way of doing so is to put up a set of rules that characterize rational choice. The rules need not be exhaustive or independent, but they should have the character that everybody agrees that it is irrational not to obey them.

Below we present the first such set of rules, presented by von Neuman and Morgenstern in 1947. The rules have been called *axioms of instrumentally rational choice*, and they are formulated in terms of preferences over lotteries. Formally, a *lottery* is a probability distribution over a set of outcomes/prices, denoted by X, where an outcome $X = x$ can be a bundle of commodities, services, resources, etc. The lottery with a certain outcome of the price x is denoted by $[x]$. The decision maker is supposed to rank the lotteries by preference. The notation $A \succeq B$ denotes that B is not preferred to lottery A, $A \succ B$ denotes that A is (strictly) preferred to B; and $A \sim B$ denotes that the decision maker is indifferent between A and B (shorthand for $A \succeq B$ and $B \succeq A$).

Construction of mixed lotteries. From two lotteries A and B we can construct compound lotteries. Let $\alpha \in [0,1]$. Then $\alpha A + (1-\alpha)B$ is a new lottery: with probability α, A is drawn, else B.

Axioms of instrumentally rational choice:

1. *Reflexivity.* For any lottery A, $A \succeq A$.
2. *Completeness.* For any pair (A, B) of lotteries, $A \succeq B$ or $B \succeq A$.
3. *Transitivity.* If $A \succeq B$ and $B \succeq C$, then $A \succeq C$.
4. *Preference increasing with probability.* If $A \succeq B$ then $\alpha A + (1-\alpha)B \succeq \beta A + (1-\beta)B$ if and only if $\alpha \geq \beta$.
5. *Continuity.* If $A \succeq B \succeq C$ then there exists $\alpha \in [0,1]$ such that $B \sim \alpha A + (1-\alpha)C$.
6. *Independence.* If $C = \alpha A + (1-\alpha)B$ and $A \sim D$, then $C \sim (\alpha D + (1-\alpha)B)$.

Theorem 9.1. *For an individual who acts according to a preference ordering satisfying rules 1–6 above, there exists a utility function over the outcomes so that the expected utility is maximized.*

Proof. Since the set of prices X is finite, there is a best price, x_B, and a worst price, x_W. Without loss of generality we set $U(x_B) = 1$ and $U(x_W) = 0$. The continuity axiom [5] then yields that for any price x there is an $\alpha \in [0,1]$ such that $[x] \sim \alpha[x_B] + (1-\alpha)[x_W]$. We set $U(x) = \alpha$.

Now let x_i denote prices and let t_i be probabilities. From standard probability calculus we have that if $A = \alpha B + (1-\alpha)C$, $B = \sum_i t_i^B [x_i]$, and $C = \sum_i t_i^C [x_i]$, then $A = \sum_i (\alpha t_i^B + (1-\alpha) t_i^C)[x_i]$. That is, any lottery A can be written in the form

$$A = \sum_i t_i [x_i],$$

and

$$\mathrm{EU}(A) = \sum_i t_i U(x_i).$$

Since $[x_i] \sim U(x_i)[x_B] + (1 - U(x_i))[x_W]$, we get (axiom [6])

$$A \sim \sum_i t_i (U(x_i)[x_B] + (1 - U(x_i))[x_W]).$$

Since $U(x_i)$ is independent of t_i, we have

$$A \sim \left(\sum_i t_i U(x_i) \right) [x_B] + \left(\sum_i t_i (1 - U(x_i)) \right) [x_W].$$

Hence, for all lotteries A we have (axiom [3])

$$A \sim \alpha[x_B] + (1-\alpha)[x_W],$$

where $\alpha = \mathrm{EU}(A)$. Now let $A \sim \alpha[x_B] + (1-\alpha)[x_W]$ and $B \sim \beta[x_B] + (1-\beta)[x_W]$. By axiom [4] we have that $A \succeq B$ if and only if $\alpha \geq \beta$ if and only if $\mathrm{EU}(A) \geq \mathrm{EU}(B)$.

□

The theorem says that if you agree that rules 1–6 apply for your decision problem, then you have to choose a decision that maximizes your expected utility. If you do not wish to follow the recommendation of a perfect max-EU analysis of your problem, your only way out is to attack the rules.

To illustrate this point, consider the following example (Allais' paradox). You have a choice between two lotteries:

- Lottery $A = [\$1\text{mill.}]$,
- Lottery $B = 0.1[\$5\text{mill.}] + 0.89[\$1\text{mill.}] + 0.01[\$0]$.

Most probably you would strictly prefer A to B because your life would be completely changed if you got \$1 million, and in B there is a risk of this not happening. This reasoning is perfectly rational. It reflects only that your subjective utility of \$1 million is very close to your utility of \$5 million. This must also be the case in other situations. Assume that you are faced with a new choice between two lotteries:

- Lottery $C = 0.11[\$1\text{mill.}] + 0.89[\$0]$,
- Lottery $D = 0.1[\$5\text{mill.}] + 0.9[\$0]$.

In turns out that if you chose D (as many people would do) you would not maximize expected utility. In other words, if you seriously mean that the difference in utility between \$1 million and \$5 million is very small, you must take the extra 1% chance of winning \$1 million.

The following calculations show that choosing D does not maximize your expected utility. Let $U(\$5\text{mill.}) = 1, \mathrm{U}(0) = 0, \mathrm{U}(\$1\text{mill.}) = \mathrm{u}$. If you prefer A to B, you have

$$u > 0.1 + 0.89u.$$

Hence

$$u > \frac{10}{11}$$

and now

$$\mathrm{EU}(C) = 0.11u > 0.11\frac{10}{11} = 0.1 = \mathrm{EU}(D).$$

The rules presented here cover a simple type of decision problem. There is an extensive scientific debate about how wide the scope is for the principle of maximizing expected utilities in a world assigned with subjective probabilities. Axioms similar to the axioms presented here have been devised, and theorems similar to Theorem 9.1 have been proved.

9.3 Decision Trees

A classical way of representing decision problems with several decisions is with *decision trees*. A decision tree is a model that encodes the structure of the decision problem by representing all possible sequences of decisions and observations explicitly in the model.

The nonleaf nodes in a decision tree are decision nodes (rectangular boxes) or chance nodes (circles or ellipses), and the leaves are utility nodes (diamond shaped). The links in the tree have labels. A link from a decision node is labeled with the action chosen, and a link from a chance node is labeled by a state.

Example 9.1 (The Two-Test Milk Problem). Consider the infected milk scenario from Figure 3.1 and Section 3.2.1 (to keep things simple, we assume that the infections and tests are independent between the days). The farmer has 50 cows, and the milk from each cow is poured into a common container and transported to the dairy. The value of the milk is \$2 per cow. The dairy checks the milk carefully, and if it is infected it is thrown away. After having milked a cow, the farmer may perform two different tests of the milk, T_A and T_B, before pouring it into the container. The price of the first test is 6 cents and it has a false positive/negative rate of 0.01, and the price of the second test is 20 cents and it has a false positive/negative rate of 0.001.

To establish the utilities, let us assume that the farmer has clean milk from the 49 other cows. If the farmer pours the milk into the container, he will gain \$100 if it is not infected, and he will gain nothing if it is infected. If he throws the milk away, he will gain \$98 regardless of the state of the milk.

The question is whether he should perform the tests and in which order. Figures 9.8 and 9.9 show the graphical part of a decision tree for the milk example with two tests.

A decision tree is read from the root downward. When you pass a decision node, the label tells you what the decision is, and when you pass a chance node, the label tells you the state of the node. If a decision node follows a chance node, then the chance node is observed before the decision is made. Hence the sequence in which we visit the nodes corresponds to the sequence of observations and decisions. We assume *no-forgetting*: when a decision is to be taken, the decision maker knows all the labels on the path from the root down to the current position in the decision tree. We adopt the shorthand *past* for the set of labels from the root to a position in the tree.

Each path from the root to a leaf specifies a complete sequence of observations and decisions, and we call such a sequence a *decision scenario*. Furthermore, we require decision trees to be complete: from a chance node there must be a link for each possible state, and from a decision node there must be a link for each possible decision option. This also means that a decision tree specifies all the possible scenarios in the decision problem.

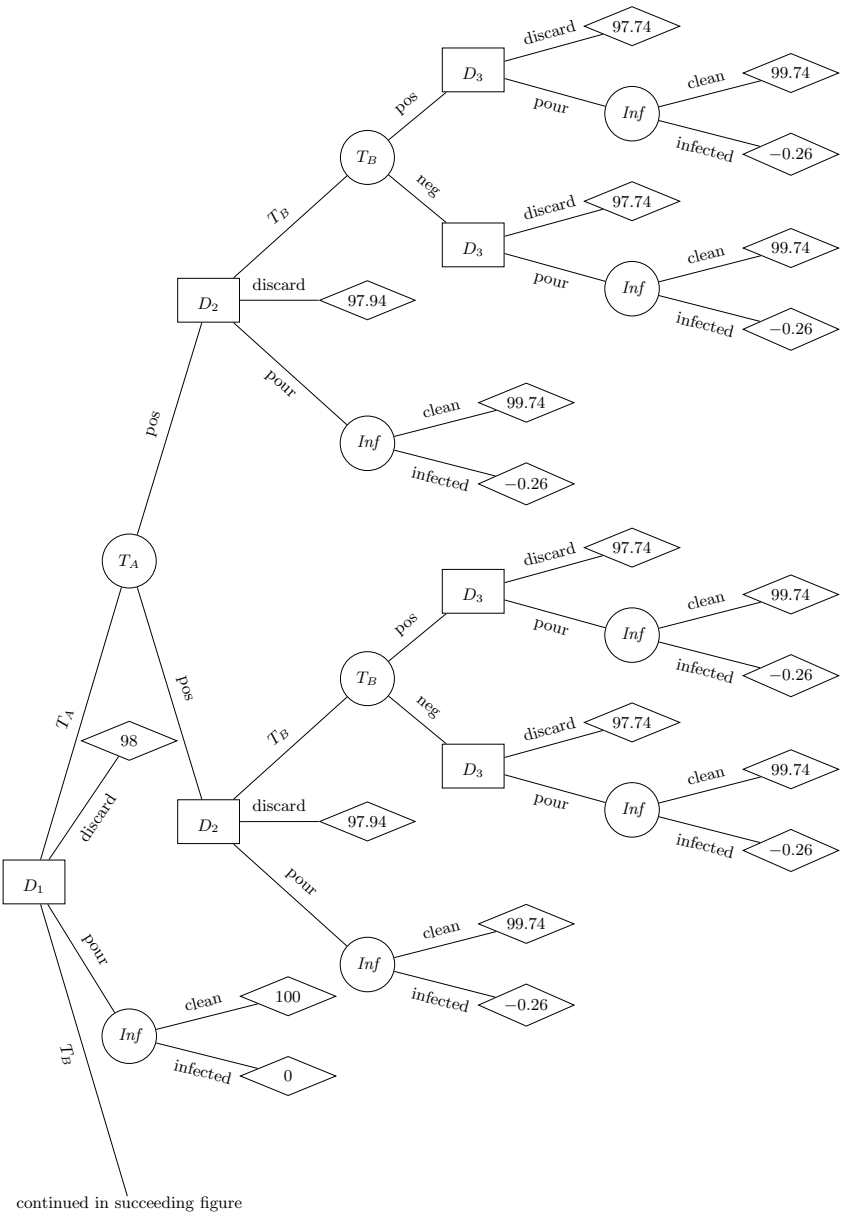


Fig. 9.8. The graphical part of a decision tree for the milk problem from Example 9.1. The tree reflects that no test is performed when the milk has been poured or discarded. Note that nodes in a decision tree may share names.

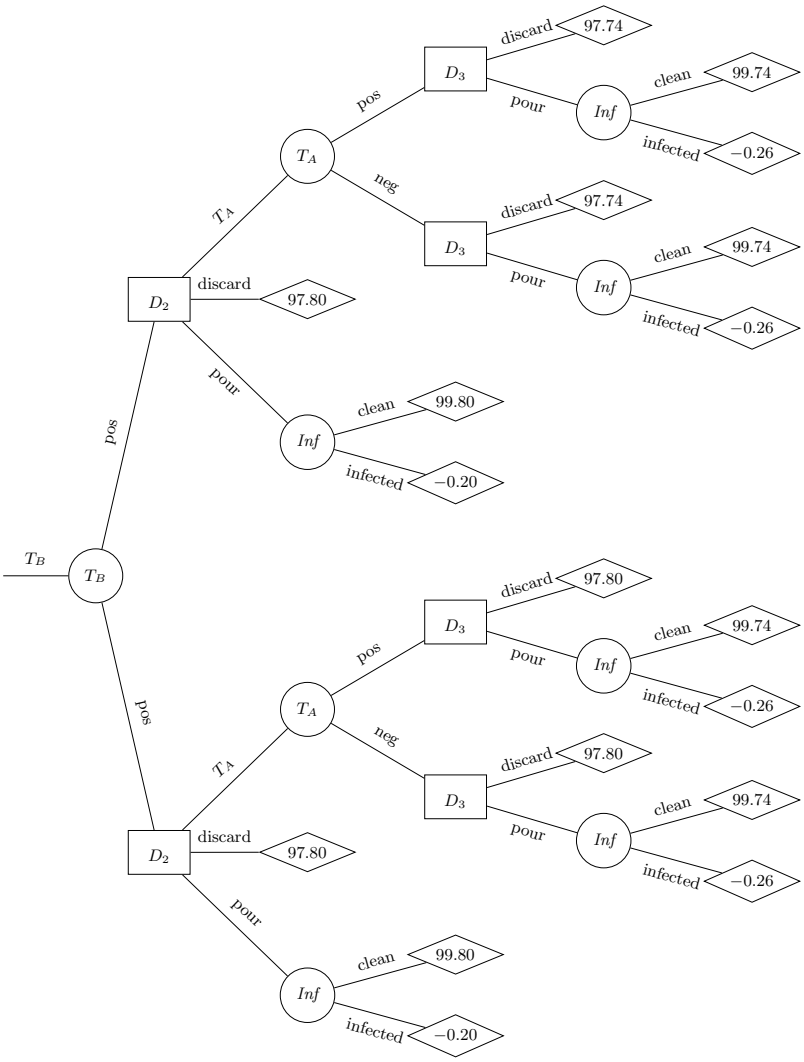


Fig. 9.9. Continuation of diagram in Figure 9.8.

The quantitative part of a decision tree consists of utilities and probabilities. Each leaf has a utility value attached to it. This utility reflects the utility of the decision scenario identified by the path from the root to the leaf in question. For the chance nodes, we associate a probability with each of the links emanating from them. See Figure 9.11 for an example. Let A be a chance node at a particular position in the tree with past o, and let l be an outgoing link labeled with a. We then associate $P(A = a \mid o)$ with this link. Either you can have the probabilities explicitly attached to the links (which can be rather impractical to work with), or you can use your Bayesian network model as a

reference. You can, for example, complement the graphical part in Figures 9.8 and 9.9 with the Bayesian network in Figure 9.10 and then use the Bayesian network to calculate the required probabilities.

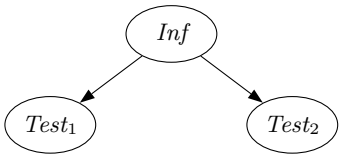


Fig. 9.10. A Bayesian network for calculating the probabilities for the decision tree in Figures 9.8 and 9.9.

9.3.1 A Couple of Examples

We now give two other examples of decision problems involving a sequence of decisions.

Example 9.2 (The Car Start Problem). In the morning, my car will not start. There are three possible faults: the *spark plugs* may be dirty, with probability 0.3; the *ignition system* may be malfunctioning, with probability 0.2; or there is some *other* cause, with probability 0.5. I can perform two repair actions myself: SP, which at the cost of 4 minutes always fixes spark plugs; and IS, which takes 2 minutes and fixes the ignition system with probability 0.5. I can also perform a test T, namely to check the charge on the spark plugs when starting. It takes half a minute, and it says *ok* if and only if the ignition system is okay. Finally, I can call road service RS, which at the cost of 15 minutes fixes everything. The car was okay yesterday evening, so I assume that there is at most one fault.

To work with utilities rather than costs, let us say that I have 30 minutes to fix the car and arrive at work, and I want to find a test−repair sequence that expectedly gives me as much time as possible for getting to work. Therefore, the utility of a test−repair sequence is the remaining time for getting to work.

A decision tree for this Car Start Problem is shown in Figure 9.11. The probabilities for the decision tree are calculated from the model in Figure 9.12, where the technique from Section 3.3.9 is used.

Example 9.3 (The Reactor Problem).

An electric utility firm must decide whether to build (B) a reactor of advanced design (a), a reactor of conventional design (c), or no reactor (n) at all. If the reactor is successful, an advanced reactor is more profitable, but it is also more risky.

If the firm builds a conventional reactor, the profits are \$8B if it is a success (cs), and −\$4B if there is a failure (cf). If the firm builds an advanced

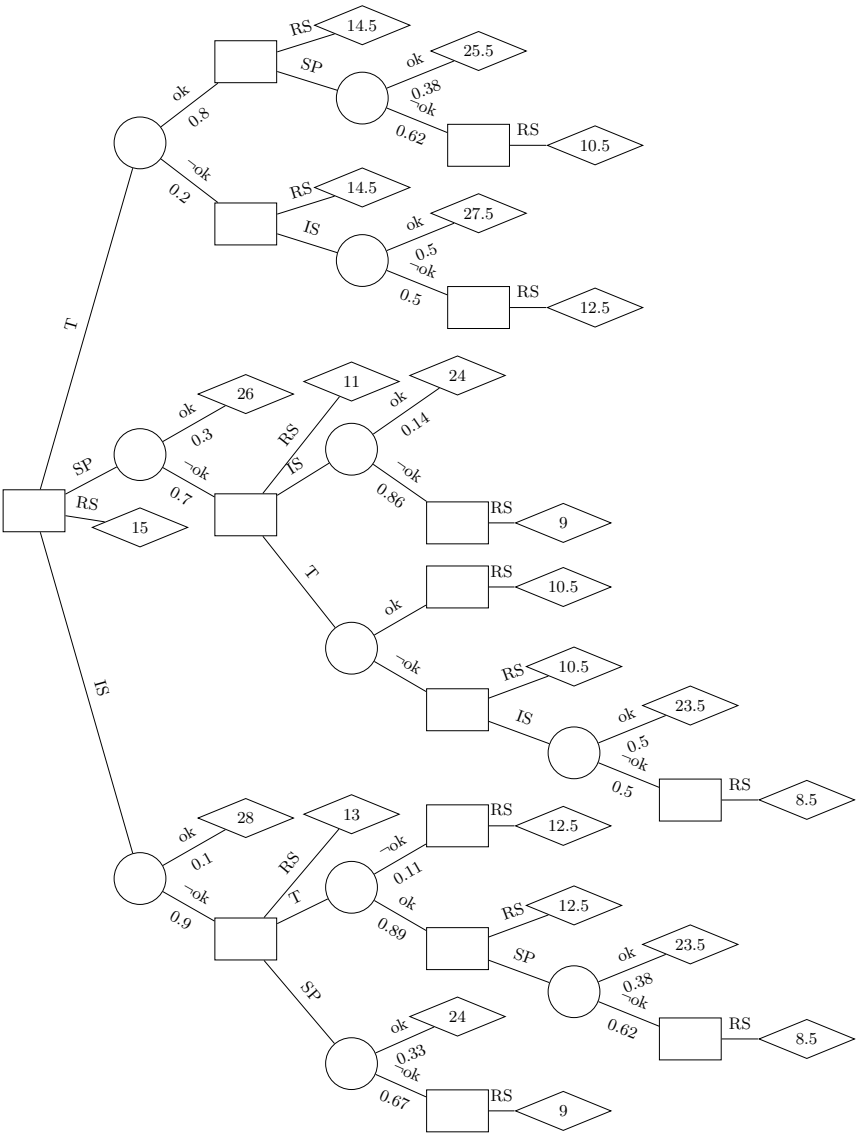


Fig. 9.11. A decision tree for the Car Start Problem in Example 9.2.

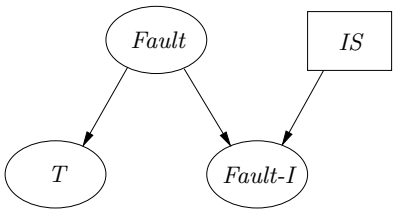


Fig. 9.12. A model for calculating the probabilities for a decision tree for the Car Start Problem in Example 9.2. Due to the assumption of exactly one fault, the faults are collected in the node *Fault* with states *is, sp*, and *other*.

reactor, the profits are \$12B if it is a success (as), −\$6B if there is a limited accident (al), and −\$10B if there is a major accident (am). The firm's utility is assumed to be linear in dollars. Before making the decision to build, the firm has the option to conduct a test ($T = t$) or not (nt) of the components (Cp) of the advanced reactor. The test results (R) can be classified as either bad (b), good (g), or excellent (e). The cost of the test is \$1B. If the test results are bad, then the Nuclear Regulatory Commission (NRC) will not permit the construction of an advanced reactor.

Figure 9.17 shows a decision tree representation of the problem, where the probabilities can be found from the Bayesian network in Figure 9.14.

The specification of the quantitative part (Figure 9.14) can be extended with decision nodes and utility nodes as shown in Figure 9.15, which can also be considered a model of the relevant world.

9.3.2 Coalesced Decision Trees

The main drawback of decision trees is that they grow exponentially with the number of decision and chance variables, and – as illustrated in the two examples – even very small decision problems require a relatively large decision tree. There are, however, methods for reducing the complexity by exploiting symmetries in the decision problem.

The idea is that when a decision tree contains identical subtrees, they can be collapsed. In the milk problem, if both tests are negative, the situations will be the same regardless of the order in which the tests are performed. The succeeding parts of the decision tree must therefore be the same, both in terms of structure and numerical information (probabilities and utilities); hence we can have the links to these parts meet in a common decision node. Figure 9.16 shows the structure of a coalesced decision tree for the milk problem, and Figure 9.17 shows the coalesced decision tree for the reactor problem.

The procedure for solving a coalesced decision tree is the same as the procedure for normal decision trees (see the next section).

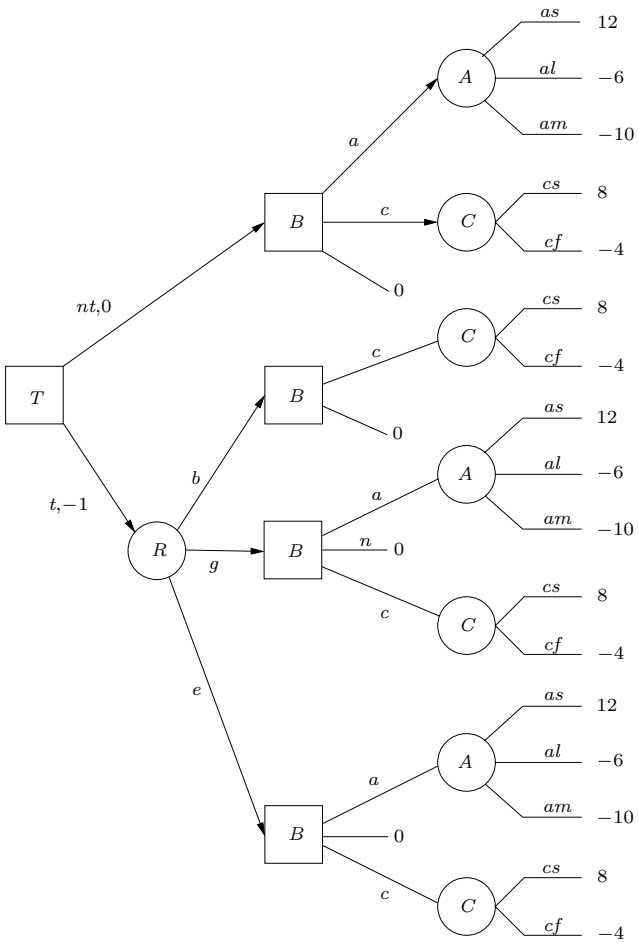


Fig. 9.13. A decision tree for the Reactor Problem. Note that the cost of the test is attached to the link $T = t$, indicating that the cost will be the same for all ensuing scenarios.

9.3.3 Solving Decision Trees

A solution to a decision tree is a *strategy* that specifies how we should act at the various decision nodes. An example of a strategy is illustrated in Figure 9.18 by the boldfaced links. Strategies are compared based on their expected utilities, and finding an *optimal strategy* amounts to finding a strategy with highest expected utility; such a strategy is not necessarily unique.

By assigning to each node in the decision tree a value corresponding to the maximum expected utility achievable at that node, an optimal strategy will pick an action leading to a child of maximum value. Looking at the end

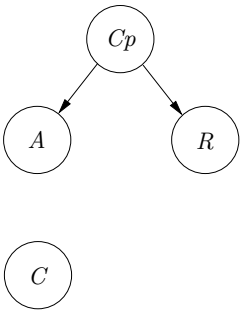


Fig. 9.14. A Bayesian network providing probabilities for the decision tree representation of the Reactor Problem shown in Figure 9.13.

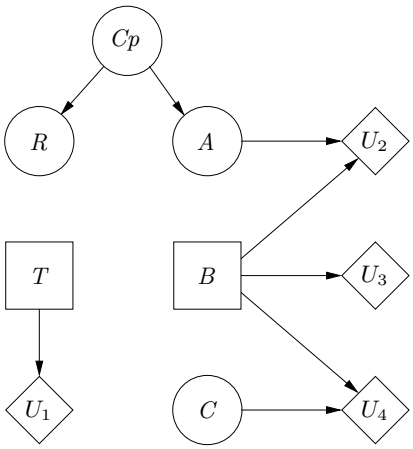


Fig. 9.15. A model of the world relevant for the reactor problem.

of the decision tree, one sees that the value of a leaf node is simply the utility assigned to that node. If we go one step further up the tree, then the value of a decision node D is the maximum value associated with its children/leaves, since D is under our full control. For a chance node, its value corresponds to the utility you can expect to achieve from that point in the decision tree: the value is the sum of the utilities of the leaves weighted with the probabilities of their outcomes. When all children of a node N have been assigned a value, we can calculate the value to assign to N. If N is a decision node, we assign it the maximum of the children's values, and if N is a chance node, we assign the weighted sum.

These observations form the basis for a procedure known as "average-out and fold-back" for calculating an optimal strategy and the maximum expected utility: start with nodes that have only leaves as children. If the node is a

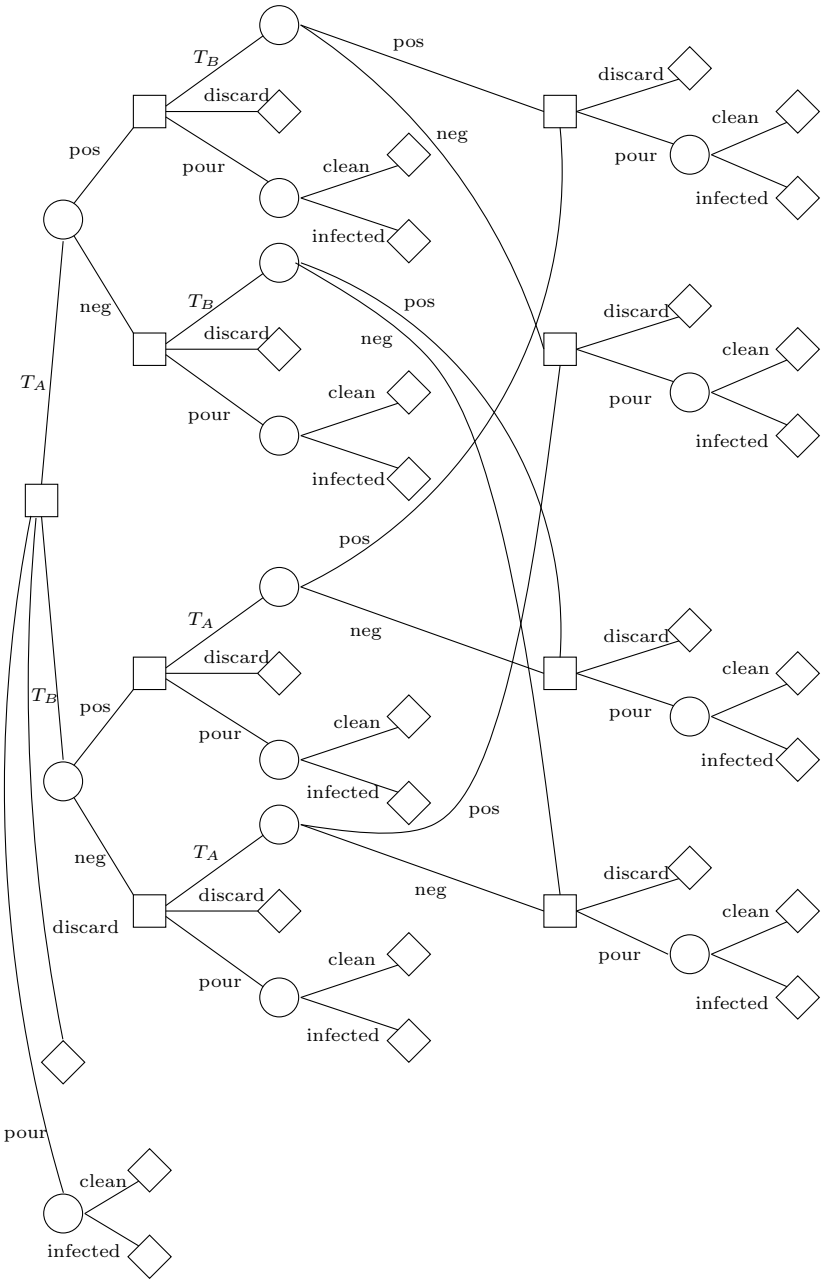


Fig. 9.16. The structure of a coalesced decision tree for the milk problem.

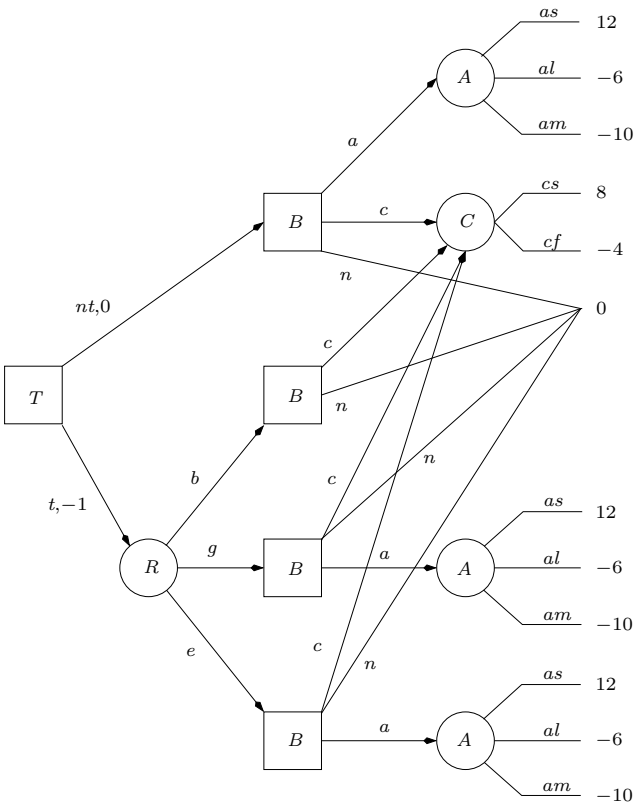


Fig. 9.17. A coalesced decision tree for the reactor problem. If we decide to build a conventional reactor the resulting subtrees will be the same regardless of our previous decisions and observations.

chance node A, the expected utility for A is calculated. Each child of A is an outcome o and has a utility $U(o)$ attached, and the link has a probability $P(A = a)$. We calculate the product $U(o) \cdot P(A = a)$ from each child, and their sum is attached to A. If the node is a decision node D, each child of D has an (expected) utility attached. Choose a child with maximal expected utility, highlight the link, and attach the value to D.

This is done repeatedly until the root is reached. The resulting value for the root is the expected utility if you adhere to the strategy of always maximizing the expected utility, and the paths from root to leaves following highlighted links when possible represent an optimal strategy for the decision problem.

Example 9.4 (The Car Start Problem, continued).

Figure 9.18 illustrates the calculations for solving the troubleshooting problem.

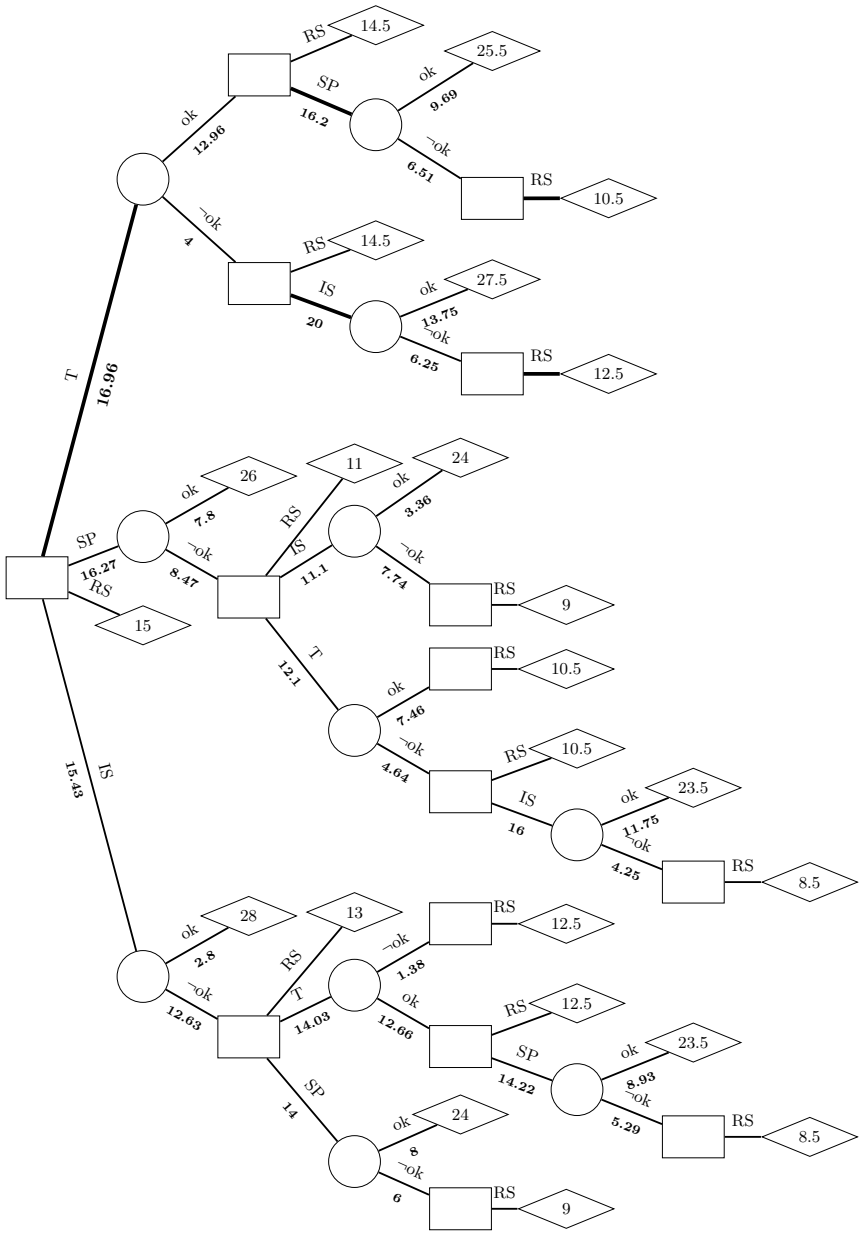


Fig. 9.18. Results when solving the decision tree from Figure 9.11. The boldfaced links indicate the optimal strategy.

As can be seen from Figure 9.18, the maximum expected utility is 16.96. A strategy close to the optimal one (in terms of expected utility) is to start performing SP and if unsuccessful to follow with T.

More formally, if we use $N(X = x)$ to denote the node following X by the link labeled x, then the "average-out and fold-back" algorithm can be specified recursively as follows.

Algorithm 9.1 [**Expected-Utility** (EU)] *Let X be a node in a decision tree T. To calculate an optimal strategy and the maximum expected utility for the subtree rooted at X, do:*

1. *If X is a utility node, then return $U(X)$.*
2. *If X is a chance node, then return*

$$\mathrm{EU}(X) = \sum_{x \in \mathrm{sp}(X)} P(X = x \mid \mathrm{past}(X))\, \mathrm{EU}(N(X = x)).$$

3. *If X is a decision node, then return*

$$\mathrm{EU}(X) = \max_{x \in \mathrm{sp}(X)} \mathrm{EU}(N(X = x)),$$

and mark the arc labeled:

$$x' = \arg \max_{x \in \mathrm{sp}(X)} \mathrm{EU}(N(X = x)).$$

□

By unfolding the calculations in the algorithm, we see that the expected utility of an optimal strategy Δ is the sum of the utilities of the possible outcomes o (the leaves in the decision tree) weighted by the probability of the path down to o under the strategy Δ:

$$\mathrm{EU}(\Delta) = \sum_{o} U(o) P(o \mid \Delta).$$

The probability $P(o \mid \Delta)$ is the product of the probabilities attached to the arcs on the path from the root to o, where arcs emanating from decision nodes contribute 1 if they are part of Δ and 0 otherwise. For example, the strategy in Figure 9.18 is first to perform the test T, and if it says ok then follow with SP and possibly RS. If T says $\neg ok$, then follow with IS and possibly RS. The strategy has four possible outcomes, and the expected utility is

$$\begin{aligned}
\mathrm{EU}(\Delta) &= 25.5 \cdot P(T = ok, SP = ok \mid \Delta) + 10.5 \cdot P(T = ok, SP = \neg ok \mid \Delta) + \\
&\quad 12.5 \cdot P(T = \neg ok, IS = \neg ok \mid \Delta) + 27.5 \cdot P(T = \neg ok, IS = ok \mid \Delta) \\
&= 25.5 \cdot 0.8 \cdot 0.38 + 10.5 \cdot 0.8 \cdot 0.62 + 12.5 \cdot 0.2 \cdot 0.5 + 27.5 \cdot 0.2 \cdot 0.5 \\
&= 16.96.
\end{aligned}$$

In general, this procedure can be used for calculating the expected utility of any strategy; hence the identification of an optimal strategy could also be formulated as

$$\Delta = \arg\max_{\Delta'} \mathrm{EU}(\Delta').$$

This approach, however, clearly has a complexity problem, since we should explore all possible strategies. The reason that this problem is not as apparent in the algorithm above is that it exploits a general principle known as *dynamic programming*. The idea is that the contribution from, say, the subtree rooted at $T = \neg ok$ is independent of the subtree rooted at $T = ok$; hence a strategy that is optimal for the subtree at $T = \neg ok$ will be part of an optimal strategy for the full decision tree.

9.4 Influence Diagrams

Decision trees are very easy to use, but they have a serious drawback: the number of decisions and observations need not be large before it becomes an inhuman task to specify the problem. We therefore look for other modeling frameworks that in a much more compact way can be used to represent decision problems with several decisions and observations.

In this section we present the *influence diagram* framework. It is particularly well suited for so-called *symmetric* decision problems.

In the decision tree framework, we used two models for describing a decision problem: a Bayesian network for calculating probabilities and a decision tree for representing the sequence of decisions and observations. In the influence diagram framework the approach is different: the Bayesian network is extended with syntactic features that will allow it to encode the probability model as well as the structure of the decision problem.

9.4.1 Extended Poker Model

In the poker problem described in Section 3.2.3, the final decision is whether to call or fold. When taking this decision I have information about my own hand (*MH*) as well as the number of cards my opponent has discarded in the first and in the second round of changing cards. However, before I come that far I would also have had to decide on my first change of cards (*MFC*) and my second change of cards (*MSC*). In order to make these two decisions explicit in the representation, you can extend the model in Figure 9.3 with *MFC* and *MSC* as well as two variables representing my initial hand (*MH0*) and my hand after the first change of cards (*MH1*). The resulting model is shown in Figure 9.19.

Looking at Figure 9.19 we see that even though all relevant variables are included in the model, it does not convey the order in which the decisions

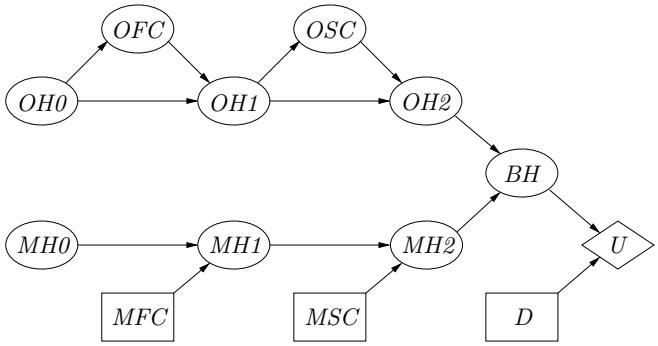


Fig. 9.19. The poker model in Figure 9.3 extended with variables for my initial hand (*MH0*), my first change of cards (*MFC*), my second hand (*MH1*), and my second change of cards (*MSC*).

are taken; nor does it specify the variables that are observed before a particular decision: before deciding on the first decision *MFC* I observe *MH0*; then I observe my opponent's first change of cards *OFC* as well as my second hand *MH1* before I decide on *MSC*; and finally, I observe both *MH2* and my opponent's second change of cards *OSC* prior to deciding on *D*.

An immediate way to encode this information directly in the model is to extend the model with so-called *information arcs*. An information arc is a directed arc $X \rightarrow D$ going into a decision node D from either a chance node or another decision X. Semantically it specifies that X is either observed (if it is a chance node) or decided on (if it is a decision node) before we decide on D. By extending the model in Figure 9.19 with information arcs we get the model in Figure 9.20, where we can see, for example, that when deciding on *MSC* we know the state of *OFC*, *MH0*, *MFC*, and *MH1*.

Now assume that we adopt the no-forgetting assumption from the decision tree framework, i.e., the decision maker remembers all previous observations and decisions. Given this assumption, we see that the model in Figure 9.20 contains redundant information arcs. For example, the arc *MFC* → *MSC* indicates that we decide on *MFC* before deciding on *MSC*, and the two arcs from *MH0* into *MFC* and *MSC* specify that the state of *MH0* is known when we decide on both *MFC* and *MSC*. However, under the no-forgetting assumption the link from *MH0* to *MSC* is redundant and it can therefore be removed. Similarly, *MFC* has an impact on *MH1*, which is observed before *MSC*. Therefore, *MFC* must precede *MSC*, and the link from *MFC* to *MSC* can be removed. By iteratively removing all redundant information arcs we obtain the model in Figure 9.21.

A model such as the one shown in Figure 9.21 is also called an *influence diagram*, and it encodes information about the probability model as well as the relevant information about the structure of the decision problem: the directed

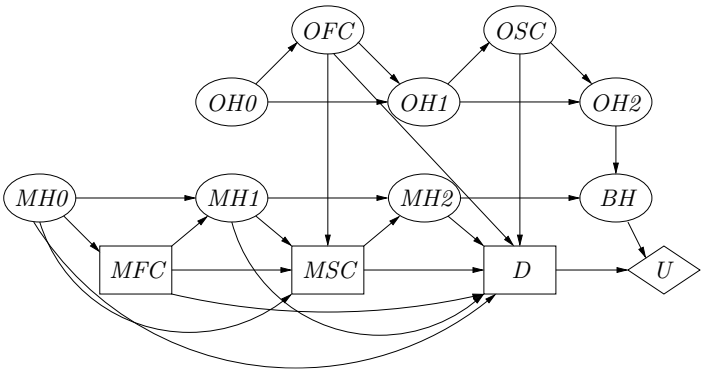


Fig. 9.20. The poker model in Figure 9.19 extended with information arcs into the decision variables.

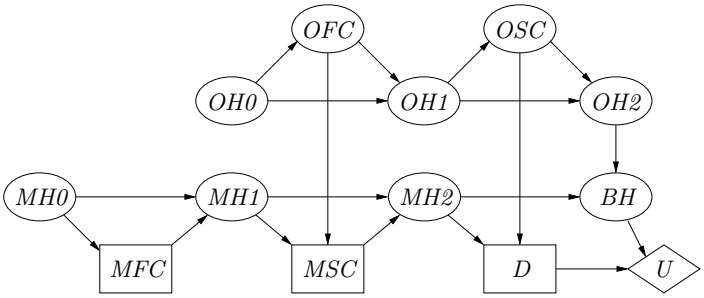


Fig. 9.21. The poker model in Figure 9.19, where the redundant information arcs have been removed.

path going through all the decision variables specifies the sequence in which the decisions are made, and the chance variables appearing as parents of a decision variable are the set of chance variables observed immediately before that decision. For example, since *MH2* and *OSC* are parents of *D*, they are observed immediately before *D* but after the decisions *MFC* and *MSC*. Note that we do not specify the sequence in which *MH2* and *OSC* are revealed, but their ordering will not affect the solution of the influence diagram (see also Section 9.3.3 and Section 10.1). In summary, the sequence of observations and decisions can be described as follows:

$$\{MH0\} \prec MFC \prec \{MH1, OFC\} \prec MSC \prec \{MH2, OSC\} \prec D \\ \prec \{OH0, OH1, OH2, BH\}.$$

For the last set of variables it should be noted that whether a variable will eventually be observed depends on the semantics of that variable and cannot be deduced from the syntax of the influence diagram. Finally, we also see that due to the no-forgetting assumption we can read that at the time of deciding

on D, I will know the states of the parents *MH2* and *OSC*, and by assuming that I do not forget my past, I will also know the states of *MH0*, *MFC*, *MH1*, *OFC*, and *MSC*.

9.4.2 Definition of Influence Diagrams

In the previous section we exemplified the influence diagram framework as an alternative to the decision tree framework. Historically, influence diagrams were invented as a compact representation of decision trees for symmetric decision problems (see Section 9.5). Now they are seen more as a decision tool extending Bayesian networks, and below we formally introduce the influence diagram framework in this way.

Syntax

An *influence diagram* consists of a directed acyclic graph over chance nodes, decision nodes, and utility nodes with the following structural properties:

- there is a directed path comprising all decision nodes;
- the utility nodes have no children;
- the decision nodes and the chance nodes have a finite set of states;
- the utility nodes have no states.

An influence diagram is *realized* when the following quantities have been specified:

- a conditional probability table $P(A \mid \mathrm{pa}(A))$ is attached to each chance node A;
- a real-valued function over $\mathrm{pa}(U)$ is attached to each utility node U.

Unless the context requires a distinction we let the term "influence diagram" include a specification of probabilities and utilities.

Figure 9.22 shows an example of an influence diagram (the states of the variables are not specified).

Semantics

Links into a decision node yield no quantitative requirements. They are called *information links*, and they indicate that the states of the parents are known prior to taking the decision. On the other hand, links into chance nodes or utility nodes represent functional relations.

The structural requirement that there be a path comprising all decision nodes ensures that the influence diagram defines a temporal sequence of decisions. This yields a partitioning of the chance variables into disjoint subsets according to the time of observation. The set $\mathcal{I}_0$ is the set of variables observed before any decision is taken. The set $\mathcal{I}_1$ is the set of variables observed after

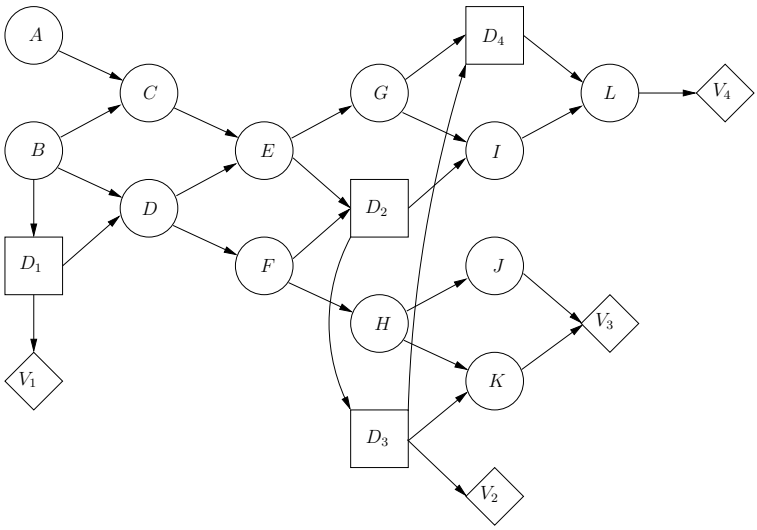


Fig. 9.22. An example of an influence diagram.

the first decision D_1 is taken but before the second decision D_2, and the set $\mathcal{I}_i$ is the set of chance variables observed after decision D_i but before decision D_{i+1}. If there are n decisions, $\mathcal{I}_n$ is the set of variables that are observed after D_n or not at all:

$$\mathcal{I}_0 \prec D_1 \prec \mathcal{I}_1 \prec \ldots \prec \mathcal{I}_{n-1} \prec D_n \prec \mathcal{I}_n.$$

For example, in Figure 9.22 we have $\mathcal{I}_0 = \{B\}$, $\mathcal{I}_1 = \{E, F\}$, $\mathcal{I}_2$ is empty, $\mathcal{I}_3 = \{G\}$, and $\mathcal{I}_4 = \{A, C, D, H, I, J, K, L\}$. The ordering $\prec$ therefore specifies a *partial temporal ordering* over the variables in the influence diagram; the ordering is partial since we do not have an ordering over the variables in each of the sets $\mathcal{I}_i$.

There is a hidden assumption behind the semantics of influence diagrams, namely *no-forgetting*: the decision maker remembers the past observations and decisions. Thus, at D_i we know the state of the variables appearing before D_i under $\prec$.

In some decision problems, two decisions may be independent in the sense that they can be taken in any order without changing the expected utilities. In Figure 9.22, the two decisions D_2 and D_3 are independent. Therefore, the link from D_2 to D_3 puts an unnecessary restriction on the decision maker. It could be removed and the representation would still be meaningful, although the first structural requirement would be violated. Unfortunately, it is not always easy to characterize situations in which decisions are independent, and we will keep the first structural requirement, which ensures a well-specified

decision problem. We shall, however, return to this issue in Section 9.5.2 and Section 11.2.

If there is more than one utility node, then the entire utility can be either the sum or the product of the individual utilities. Due to the intuitive appeal, local utilities are usually treated as components in a sum. For instance, in the mildew example (Section 9.1.2) we have two local utility functions: C, which represents the cost of the various treatments, and U, which represents the utility of the harvest for each state of the crops. The total utility is the sum of C and U, and if we assume that both C and U are the actual costs and payoffs, then the sum simply encodes the overall monetary value of the different scenarios as described by the parents of C and U. Should it happen that the total utility is the product rather than the sum of the local utilities, then taking the logarithm of the utilities will transform the problem into an influence diagram in which the total utility is the sum of the transformed utilities.

Solving an Influence Diagram

An influence diagram provides a description of a decision problem and should subsequently be used to aid the decision maker in the decision process. This amounts to prescribing an action for each decision variable conditioned on the previous observations and decisions. A way of doing the prescription is to transform the influence diagram into a decision tree and then apply the "average-out and fold-back" algorithm. The influence diagram's decision tree representation has the property that each node representing a decision D has the same variables in the past. Let past(D) denote the variables in D's past. Thus, if in the decision tree we have an action for each such decision node, these actions will collectively specify an action for each possible configuration past(D). Such a specification is called a *policy* (denoted by δ) for D:

$$\delta_D : \mathrm{sp}(\mathrm{past}(D)) \to \mathrm{sp}(D)\,.$$

If we have a policy for each decision variable in an influence diagram, we call it a *strategy*. For example, a strategy for the influence diagram in Figure 9.21 will consist of three policies:

$$\begin{aligned}
\delta_{MFC} &: \mathrm{sp}(\mathit{MH0}) \to \mathrm{sp}(\mathit{MFC});\\
\delta_{MSC} &: \mathrm{sp}(\mathit{MH0}, \mathit{MFC}, \mathit{MH1}, \mathit{OFC}) \to \mathrm{sp}(\mathit{MSC});\\
\delta_D &: \mathrm{sp}(\mathit{MH0}, \mathit{MFC}, \mathit{MH1}, \mathit{OFC}, \mathit{MH2}, \mathit{OSC}) \to \mathrm{sp}(D)\,.
\end{aligned}$$

If the strategy encodes the solution of the "average-out and fold-back" algorithm (i.e., the strategy maximizes the expected utility), then the strategy is called an *optimal strategy* and each of its policies is called an *optimal policy*.

Definition 9.1. *A* policy *for decision D_i is a mapping δ_i that for any configuration of the past of D_i yields a decision for D_i. That is,*

$$\delta_i(\mathcal{I}_0, D_1, \ldots, D_{i-1}, \mathcal{I}_{i-1}) \in \mathrm{sp}(D_i)\,.$$

A strategy *for an influence diagram is a set of policies, one for each decision. A* solution *to an influence diagram is a strategy maximizing the expected utility.*

By transforming the influence diagram into a decision tree in order to solve it, the complexity problem inherent in the decision tree framework is still present in the solution phase. However, solution methods working directly on the influence diagram have also been developed (see Chapter 10).

9.4.3 Repetitive Decision Problems

Fishing in the North Sea

Every year, the European Union undertakes very delicate political and biological negotiations to determine a volume of fishing for most kinds of fish in the North Sea. Simplified, you can say that each year the EU has a test for the volume of fish, and based on this test the volume of allowable catch is decided. This decision has an impact on the volume for next year (note that the decision on volume does not mean that only this volume is actually caught – quotas have a status similar to speed limits on highways). Figure 9.23 gives an influence diagram for a five-year strategy, where each variable is given ten states.[1]

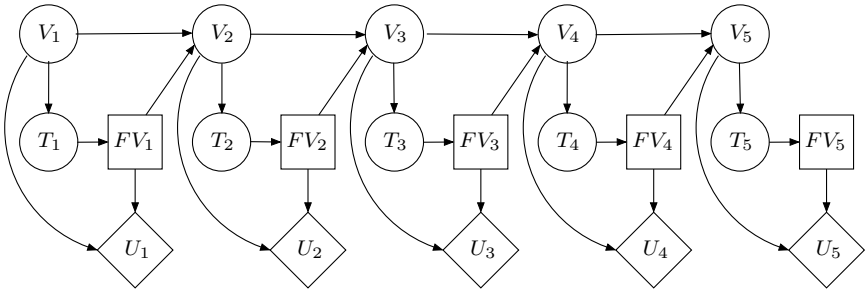


Fig. 9.23. An influence diagram for a five-year strategy for fishing volumes of herring in the North Sea.

The fishing model above has a complexity problem. For the fifth decision, all the past is relevant. Because there are nine ten-state variables in the past, the domain of the policy function for FV_5 has 10^9 elements.

[1] The model in Figure 9.23 is an example of a partially observable Markov decision process (POMDP), which we shall consider further in Section 9.6.2.

This does not mean that whenever the past is intractably large, the computer must give up. It fortunately often happens that not all information from the past is relevant (see Section 11.2).

Sometimes solving even fairly small influence diagrams represents an intractable task, and then you must use various approximation methods. One method is *blocking*. The principle in information blocking is to introduce variables that when observed, d-separate most of the past from the present decision.

Fishing Again

The problem with the model in Figure 9.23 is that all information from the past has an impact on how we will estimate the current volume of fish. We can make an approximation by allowing only this year's test and fishing volume to be used for estimating next year's volume of fish. In the model, we delete the arrows $V_i \rightarrow V_{i+1}$ and instead introduce the arrows $T_i \rightarrow V_{i+1}$ (see Figure 9.24).

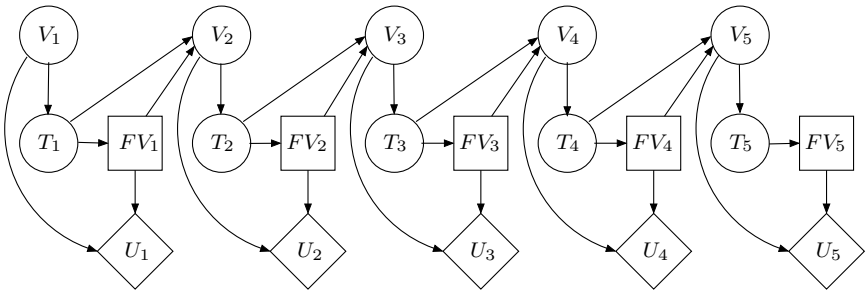


Fig. 9.24. The influence diagram from Figure 9.23 approximated through information blocking.

To establish the potential $P(V_{i+1} \mid T_i, FV_i)$, we can use the model in Figure 9.23.

$$P(V_2, T_1 \mid FV_1) = \sum_{V_1} P(V_1)P(T_1 \mid V_1)P(V_2 \mid V_1, FV_1),$$

$$P(T_1 \mid FV_1) = \sum_{V_2} P(V_2, T_1 \mid FV_1),$$

$$P(V_2 \mid T_1, FV_1) = \frac{P(V_2, T_1 \mid FV_1)}{P(T_1 \mid FV_1)}.$$

This last potential is used for all time slices.

The trick just shown is an example of a general information-blocking technique whereby you abstract the past into a *history variable* and allow only

temporal links from observed variables and from the history variable (see Figure 9.25 for another example).

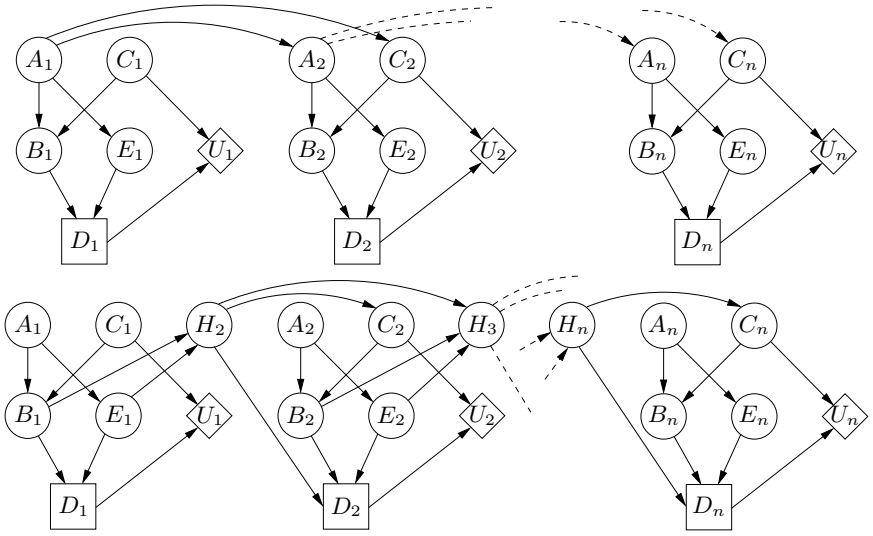


Fig. 9.25. In the top figure we have to take the entire past into account when deciding on D_n. In the lower figure, history variables have been introduced to perform information blocking.

9.5 Asymmetric Decision Problems

From the specification of the syntax for the influence diagram we see that the sequence in which the nodes are observed and decided on is the same in all possible scenarios (up to a permutation of the chance nodes in the sets $\mathcal{I}_i$). For instance, in the poker example we always start by observing *MH0*, and regardless of the outcome we then decide on *MFC*, etc. These types of decision problems are also called *symmetric* decision problems, because they can be represented by a decision tree that is completely symmetric (see Figure 9.26 for an example). If a decision problem is not symmetric we call it *asymmetric*.

Definition 9.2. *A decision problem is said to be* symmetric *if:*

- *in all of its decision tree representations, the number of scenarios is the same as the cardinality of the Cartesian product of the state spaces of all chance and decision variables, and*
- *in at least one decision tree representation, the sequence of chance and decision variables is the same in all scenarios.*

In particular, the first requirement ensures that the possible outcomes and decision options for a variable do not depend on previous observations and decisions. Moreover, the reason why the definition deals with several decision tree representations for a decision problem is that two consecutive observations (without intermediate decisions) or two consecutive decisions can be swapped without affecting the solution to the decision problem. For example, in Figure 9.26 the cardinality of the product of the state spaces of all variables is $2 \cdot 2 \cdot 2 \cdot 2 = 16$. This is also the number of scenarios in the decision tree, and since the decision tree also adheres to the second condition in the definition above, the underlying decision problem is symmetric.

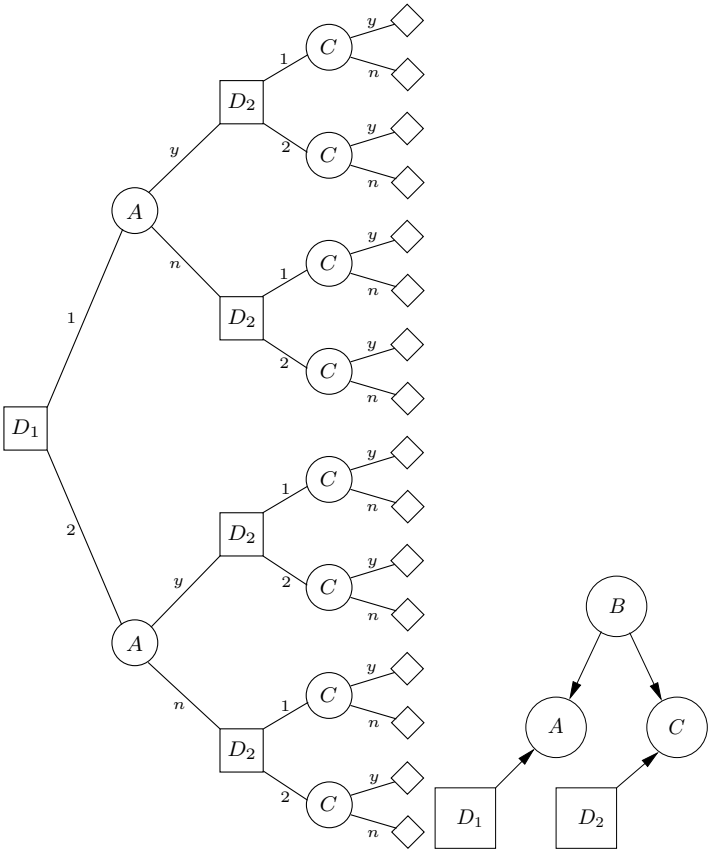


Fig. 9.26. A symmetric decision tree and the associated probability model.

The influence diagram corresponding to the decision problem shown in Figure 9.26 is given in Figure 9.27. From this example we see that the influence diagram provides a much more compact representation of the decision

problem than does the decision tree. However, this holds only for symmetric problems: in the (asymmetric) decision tree shown in Figures 9.8 and 9.9 we observe only the result of the first test $Test_1$ if we decide to actually perform the test ($D_1 = T_1$ or $D_2 = T_1$). That is, the sequence in which we make observations and decisions may vary in the different scenarios, but the influence diagram does not provide an immediate mechanism for representing such types of conditional orderings.

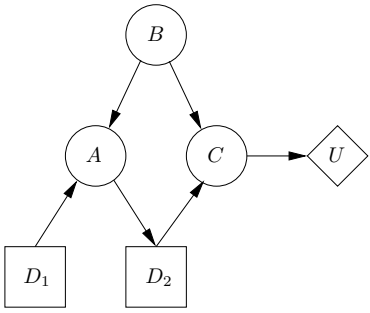


Fig. 9.27. An influence diagram representation corresponding to the decision tree from Figure 9.26.

The use of test decisions (like the ones in Figure 9.8 and 9.9 and in Example 9.1) is a frequent causes of asymmetry in decision problems: if you decide to perform a test, you will eventually observe the test result, but if you decide not to perform the test then a result will never be observed. Influence diagrams do not contain a special representation of test decisions. However, there is a general way of representing test decisions as ordinary decision variables. Assume, in the crop example in Figure 9.1, that I am in the situation that I can test the severity of the mildew attack before I decide on whether to spray. The node *Attack* represents the severity of the attack before spraying, so to model the impact of spraying we introduce a new chance node, *A-Attack*, representing the attack after the spray decision P. The decision is connected to the model by inserting a link from P to *A-Attack*. To model the test decision we insert a decision node T. This decision is basically a decision on whether the state of the chance node *Attack* should be revealed before deciding on P (we assume the test to be accurate). One way to model this situation is to introduce an additional node *Attack′* with the same states as *Attack* and with the additional state, *unobserved*, for handling the situation in which we decide not to perform the test. Next we add an arc from T and *Attack* to *Attack′* and an informational arc from *Attack′* to P. The final model is shown in Figure 9.28.

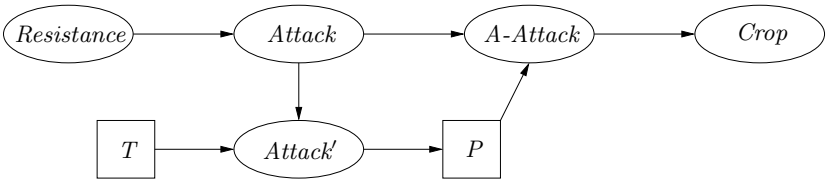


Fig. 9.28. An influence diagram representation (without utility nodes) of the crop problem: should you investigate the severity of the mildew attack before deciding on spraying against mildew?

The table for *Attack′* given T and *Attack* is specified so that the state is *unobserved* if T is *no*, and if T is *yes*, then *Attack′* is in the same state as *Attack* (see Table 9.3 and 9.4).

		Attack	
		y	n
T	y	$(1,0,0)$	$(0,1,0)$
	n	$(0,0,1)$	$(0,0,1)$

Table 9.3. The probability table $P(Attack' = (y, n, unobserved) \mid Attack, T)$ associated with *Attack′* in Figure 9.28.

This construction is general, and it is illustrated in Figure 9.29 and Table 9.4. In this way, methods developed for computing decision strategies can also be used for decision scenarios containing test decisions.

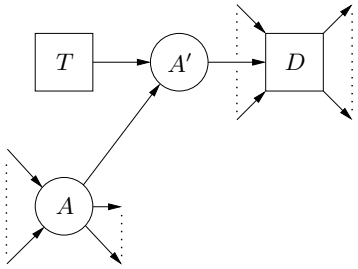


Fig. 9.29. A general way to model a decision on whether to observe A before deciding on D.

The construction can be made a bit simpler by extending the node A with the extra state *unobserved* and thereby avoiding the extra node A'. However, usually it is preferable not to change the nodes of the initial (causal) model.

		A		
		a_1	...	a_n
T	y	$(1,\ldots,0,0)$	...	$(0,\ldots,1,0)$
	n	$(0,\ldots,0,1)$	...	$(0,\ldots,0,1)$

Table 9.4. The probability table $P(A' = (a_1, \ldots, a_n, \textit{unobserved}) \mid A, T)$ associated with A' in Figure 9.29.

As the modeling technique illustrates, influence diagrams can be used to model decision problems even when the decision problem is not completely symmetric. However, this comes at a cost since we need to introduce artificial states (e.g. the state *unobserved*) and in some situations it may also be necessary to introduce artificial nodes. In the extreme case in which the decision problem does not contain any symmetric substructures, the decision tree will provide a more compact representation than the influence diagram.

9.5.1 Different Sources of Asymmetry

As we have discussed above, influence diagrams are not really suitable for modeling asymmetric decision problems. However, decision trees are not really an alternative either when there are many observations and decisions. Therefore, much research has been directed at finding specification languages that much more compactly can represent the information needed for describing the decision problem. The following two examples shed additional light on some of the problems we face when constructing such languages.

Example 9.5 (The Diagnosis Problem). Consider a two-test problem like the one in Example 9.1, Page 290; after an initial observation I you have two tests, T_A and T_B, and a decision *Pour?*. The decision on pouring is the last decision, but the two tests can be performed in any order.

To represent this problem by an influence diagram we have to represent the unspecified ordering of the tests as a linear ordering of decisions. Introduce two decision nodes, $\textit{Test}_1$ and $\textit{Test}_2$, with options, t_A, t_B, and *no-test*; introduce two chance nodes, O_1 and O_2, as children of *Inf?* with states $\textit{pos}_A$, $\textit{neg}_B$, $\textit{pos}_A$, $\textit{neg}_B$, and *no-test*. To specify that two consecutive tests of the same type will give the same results, you introduce a link from O_1 to O_2 (See Figure 9.30).

Example 9.6 (The Dating Problem). Joe needs to decide whether he should ask (*Ask*) Emily for a date for Friday evening. He is not sure whether Emily likes him (*LikesMe*). If he decides not to ask Emily or if he decides to ask and she turns him down, he will then decide whether to go to a nightclub or watch a movie on TV at home (*NClub?*). Before making this decision, he will consult the TV guide to see whether there are any movies he would like to see (*TV*). If he decides to go to a nightclub, he will have to pay a cover charge and pay for drinks. His overall nightclub experience (*NCExp*) will depend on whether he

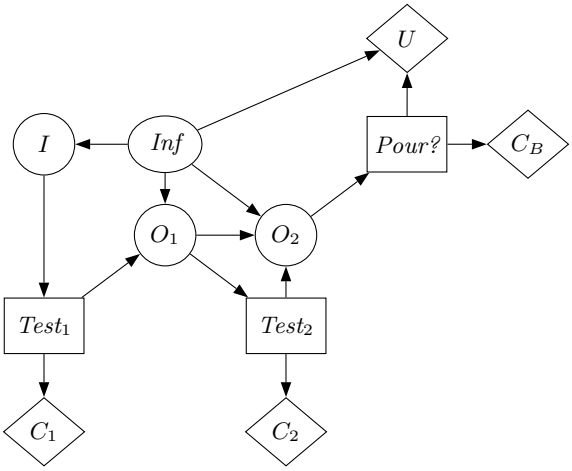


Fig. 9.30. An influence diagram representation of two tests and a decision on pouring. The *Test* nodes have three options, t_A, t_B, and *no-test*. The O nodes have five states, pos_A, pos_B, neg_A, neg_B, *no-test*. The arc $O_1 \rightarrow O_2$ indicates that repeating a test will give identical results.

meets his friends (*MeetFr*), the quality of the live music, etc (*Club*). If Emily accepts (*Accept*), then he will ask her whether she wishes to go to a restaurant or to a movie (*ToDo*); Joe cannot afford to do both. If Emily decides on a movie, Joe will have to decide (*Movie*) whether to see an action movie he likes or a romantic movie that he does not really care for, but which may put Emily in the right mood (*mMood*) to enhance his post movie experience with Emily (*mExp*). If Emily decides on a restaurant, he will have to decide (*Rest*) whether to select a cheap restaurant or an expensive restaurant. He knows that his choice will have an impact on his wallet and on Emily's mood (*rMood*), which in turn will affect his post restaurant experience with Emily (*rExp*).

From the examples above we can identify three types of asymmetry:

Functional asymmetry: The possible outcomes or decision options of a variable may vary depending on the past. We saw this in the reactor problem, where the options of the build decision are dependent on the result of a test.

Structural asymmetry: The very occurrence of an observation or a decision depends on the past. In the Dating Problem, for example, the restaurant options exist only if Emily accepts the invitation.

Order asymmetry: The ordering of the decisions and observations is not settled at the time the model is specified. For instance, in the Diagnosis Problem the ordering of the two tests is unspecified.

9.5.2 Unconstrained Influence Diagrams

In this section we shall look at a particular class of decision problems in which only order asymmetry is present.

Example

Consider again the two-test problem from Example 9.1 (Page 290) and its influence diagram representation shown in Figure 9.30. A much more direct specification would be to use decision nodes representing each test explicitly. If we knew, for example, that $Test_A$ comes before $Test_B$, it can done with an influence diagram (see Figure 9.31(a)). However, in practice this is rarely the case.

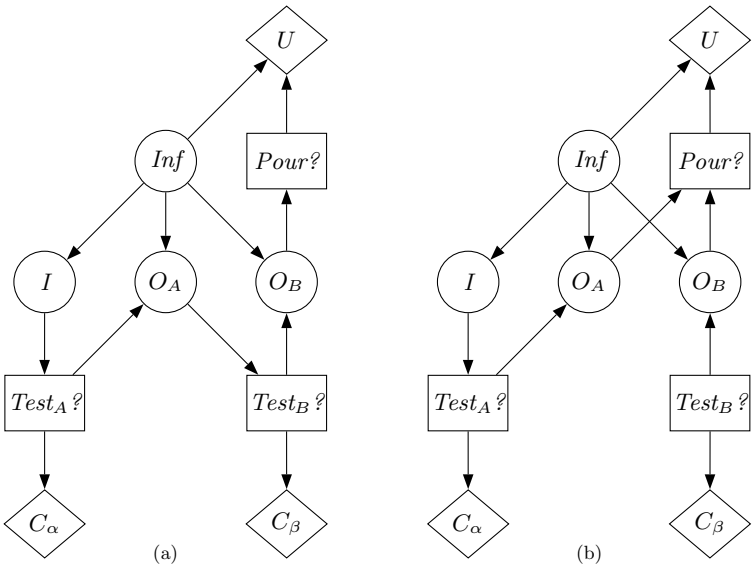


Fig. 9.31. (a) An ID representing the scenario in which you first decide on $Test_A$? and next on $Test_B$?. (b) An attempt to remove the temporal constraint on the test decisions.

To relax the temporal constraint on the test decisions, you may remove the link from O_A to $Test_B$? (Figure 9.31(b)). However, now there is no specification that the result of the first test is known when deciding on the next test. To specify this we introduce a new type of chance variables, *observables*. They are drawn as double circles, and they are observed when all preceding decision nodes have been decided (Figure 9.32). In that case we say that the observable is *free* and that the last preceding decision *released* the observable.

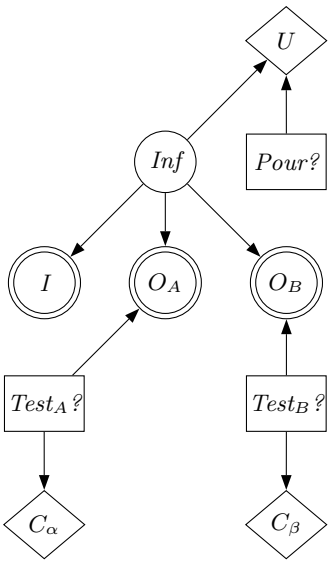


Fig. 9.32. A graphical representation of two tests and a decision of pouring. Here I is observed prior to any decision, O_A is observed when $Test_A$? has been decided, and O_B is observed when $Test_B$? has been decided.

Looking at Figure 9.32 it may seem that we have not specified that O_A is actually observed immediately after deciding on $Test_A$?. However, since the expected utility cannot increase by delaying an observation free of cost, we can safely introduce the rule that an observable chance node is observed immediately after it has been released. This means that the decision problem has been uniquely specified, and the rest can be left to a computer. The specification in Figure 9.32 yields that solving the decision problem boils down to solving two influence diagrams (one for each order of the test decisions) and choosing the order and strategy from the one giving the highest expected utility. This also means that while the influence diagram encoded the possible sequences of observations and decisions at the graphical level, this new framework has postponed it to the solution phase.

Next, consider a more complex situation. A patient may suffer from two different diseases. After an initial observation O_I, there are two possible tests, T_A and T_B, and each disease has a specific treatment, Tr_1 and Tr_2. After each treatment, the new state of the disease is observed (cost free). In Figure 9.33 the problem is specified graphically.

Even for a simple problem like the one above it is extremely cumbersome to draw a decision tree, and it is rather tricky to squeeze the scenario into the ID straightjacket; the problem is that all possible sequences must be represented explicitly.

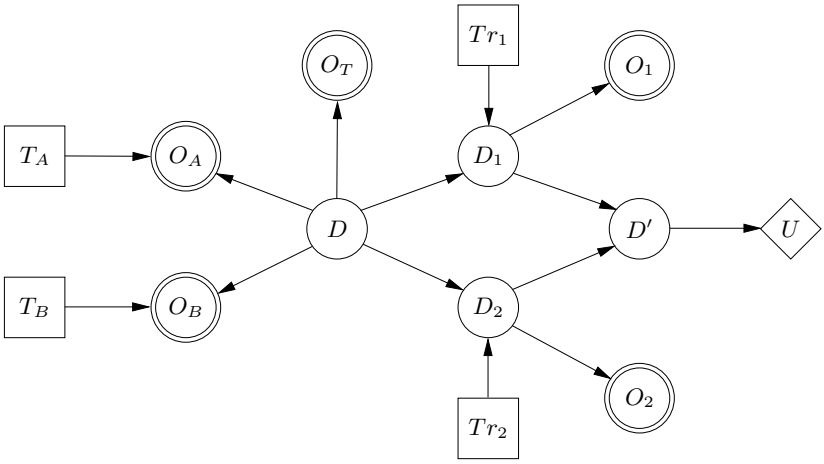


Fig. 9.33. A graphical representation of a situation with two tests and two treatments.

Definition of UIDs

As the examples above illustrate, we can meaningfully relax the linear temporal order constraint for influence diagrams without getting an ambiguous representation.

Definition 9.3. *An* unconstrained influence diagram *(UID) is an acyclic directed graph over decision variables (rectangular shaped), chance variables (circular shaped), and utility variables (diamond shaped). Utility variables have no children. There are two types of chance variables,* observables *(doubly circled) and* nonobservables *(singly circled). A nonobservable cannot have a decision as a child.*

Let U be a UID. The set of decision variables is denoted by $\mathcal{D}_U$, and the set of observables is denoted by $\mathcal{O}_U$. The partial temporal order induced by U is denoted by $\prec_U$. When obvious from the context we avoid the subscript.

The quantitative specification required is similar to the specification for influence diagrams: conditional probabilities and utility functions. We add the convention that each decision variable D has a cost. If this cost depends only on D, it is not represented graphically. We say that a UID is *realized* when the structure has been extended with the required quantitative specifications.

The semantics of a UID are similar to the semantics of an ID. A link into a decision variable represents temporal precedence; a link into a chance variable represents causal influence; a link into a utility variable represents functional dependence. We assume *no-forgetting*: at each point of the decision process the decision maker knows all previous decisions and observations.

An observable can be observed when all its antecedent decision variables have been decided on. In that case we say that the observable is *free*, and we *release* an observable when the last decision in its ancestral set is taken.

The structural specification of a UID yields a partial temporal ordering of the decisions and observations. An extension to a linear ordering is called an *admissible order*. Any admissible order yields an influence diagram.

S-DAGs and Strategies

As for decision trees and influence diagrams, the graphical language and its suitability as a language supporting human modeling are the most important properties. Having constructed an adequate model, you can hand it over to a computer, which may then unfold the model to a decision tree and compute an optimal strategy.

In dealing with UIDs, the concept of *strategy* is more complex than in the case of IDs (see Section 9.4.2). In principle we look for a set of rules telling us what to do given the current information, where "what to do" is to choose the next action as well as to choose a decision option if the next action is a decision. That is, a strategy consists of a function prescribing the next step and a set of functions for choosing decisions. The structure of the step function can be represented in a graphical structure, called an *S-DAG* (strategy DAG).

Definition 9.4. *Let U be a UID. An* S-DAG *is a directed acyclic graph G. The nodes are labeled with variables from $\mathcal{D}_U \cup \mathcal{O}_U$ such that each maximal directed path in G represents an admissible ordering of $\mathcal{D}_U \cup \mathcal{O}_U$. For notational convenience we add two unary nodes* Source*, and* Sink*.* Source *is the only node with no parents and* Sink *is the only node with no children.*

Note that an S-DAG need not contain all admissible orderings. Figure 9.34 gives an example of an S-DAG for the two-tests-two-treatments problem.

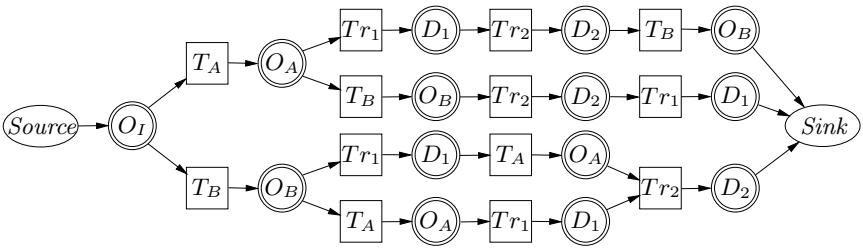


Fig. 9.34. An example of an S-DAG for the UID in Figure 9.33.

For a node N in an S-DAG G, the *history* of N is defined as the union of the labels of N and its ancestors, denoted by $\mathrm{hst}_G(N)$. When the S-DAG is

obvious from the context we drop the subscript. For example, the O_B-node at the bottom path in Figure 9.34 has the history $\{O_I, T_B, O_B\}$ and the children $\{T_A, Tr_1\}$; the set of labels of N's children is denoted by $\mathrm{ch}(N)$. A *step policy* for node N is now defined as a function

$$\sigma : \mathrm{sp}(\mathrm{hst}(N)) \rightarrow \mathrm{ch}(N).$$

Recall that $\mathrm{sp}(\mathrm{hst}(N))$ denotes all possible configurations of the variables in $\mathrm{hst}(N)$.

A *step strategy* for a UID U is a pair $(G, \mathcal{S})$, where G is an S-DAG for U and $\mathcal{S}$ is a set of step policies, one for each node in G (except for *Sink*); when a node has only one child, the step policy is trivial. For a decision node N a *decision policy* is a function

$$\delta : \mathrm{sp}(\mathrm{past}(N)) \rightarrow \mathrm{sp}(N)\,.$$

A *strategy* for U is a step strategy together with a decision policy for each decision node.

Example

Consider the UID in Figure 9.35. A strategy may have the structure illustrated by the S-DAG and the simple policy rules in Figure 9.36. Note that the policies combine step policies and decision policies.

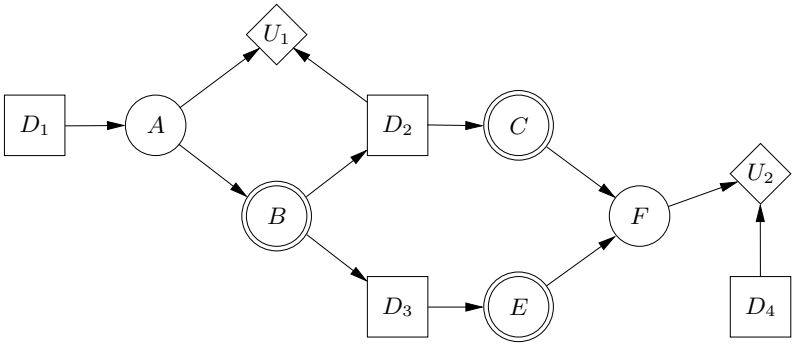
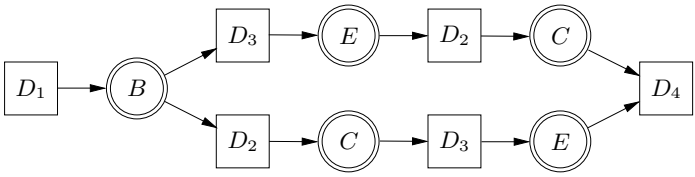


Fig. 9.35. An example UID.

The strategy represented in Figure 9.36 can be unfolded to the *strategy tree* in Figure 9.37. The expected utility from following the strategy can be calculated in the same way as for decision trees, where the UID is used for calculating the probabilities.



$$\emptyset: \text{ choose option } d_1^1$$

$$D_1, B: \text{ choose } \begin{cases} d_2^3 & \text{if } B = b_1, \\ d_1^2 & \text{if } B = b_2. \end{cases}$$

$$D_1, B, D_3, E: \text{ choose } \begin{cases} d_2^2 & \text{if } E = e_1, \\ d_1^2 & \text{if } E = e_2. \end{cases}$$

$$D_1, B, D_2, C: \text{ choose } \begin{cases} d_1^3 & \text{if } C = c_1, \\ d_2^3 & \text{if } C = c_2. \end{cases}$$

Fig. 9.36. The structure of a strategy for the UID in Figure 9.35.

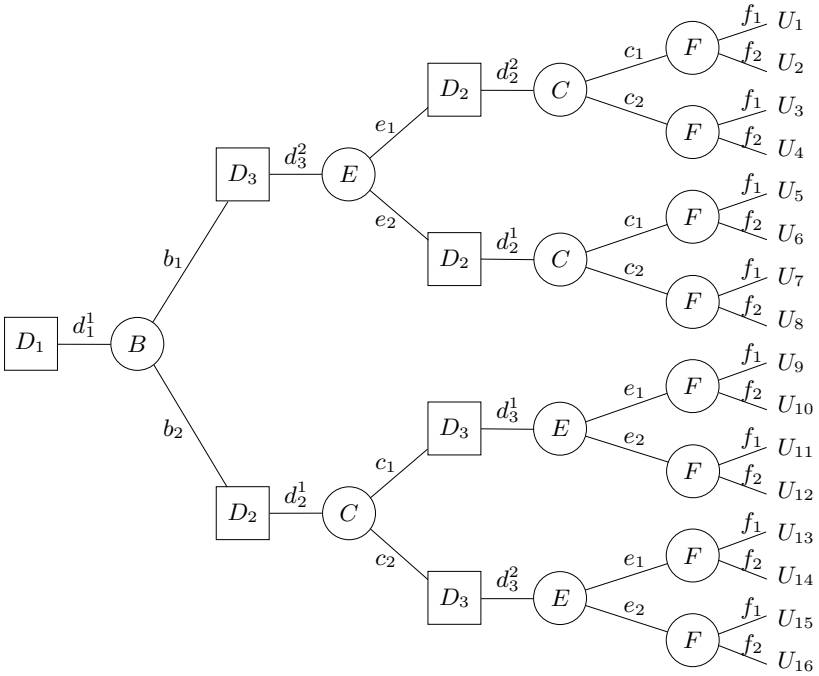


Fig. 9.37. The strategy from Figure 9.36 unfolded to a strategy tree.

Definition 9.5. *Let Δ be a strategy for the UID U. The* expected utility *of Δ is the expected utility of the corresponding unfolded strategy tree for Δ with respect to U. A* solution *to U is a strategy of maximal expected utility. Such a strategy is called an* optimal strategy. *The S-DAG for an optimal strategy is called optimal, and the step policies as well as the decision policies are also called optimal.*

Rather than trying out all possible strategy trees in looking for an optimal strategy, there are efficient solution algorithms that exploit dynamic programming and work on a (single) S-DAG representation of the UID (see Section 10.4).

9.5.3 Sequential Influence Diagrams

There is no widely recognized graphical language that compactly can cope with all types of asymmetry. Here we shall indicate only one attempt, called *sequential influence diagrams* (SIDs). The SID framework has its source in a (causal) world model like the one in Figure 9.15. To extend this world model to also represent the structure of the decision problem we need to specify the order of the decisions and observations as well as any asymmetry constraints. There are various ways of doing so. In the case of influence diagrams, the order is specified in the same graph through information links, but you may also have a separate specification (as in decision trees).

The SID framework takes the former approach by extending the world model with features specifying order and asymmetry constraints. This is done in Figure 9.38. The world model is extended with dashed arrows (*structural links*) indicating informational precedence. A label on a link is a *guard* reflecting asymmetry constraints. A guard consists of two parts. The first part takes care of structural asymmetry, and the second part describes functional asymmetry. That is, the first part describes the condition for following the link. If the condition is satisfied we say that the link is *open*. For example, if we decide to perform the test $T = t$ in Figure 9.38, then the next node will be R. If there are constraints on the choices at a decision node, then this is specified in the second part of the guard (this part is empty when there are no constraints). In Figure 9.38 the choice a in B can be taken only if a test is not performed or a test is performed and the result is either good or excellent (i.e., the scenario satisfies $(T = nt) \vee (T = t \wedge (R = e \vee R = g))$).

The specification can be unfolded to a decision tree by iteratively following the open arcs from a source node (a node with no incoming structural arcs) until a node is reached with no open outgoing arcs.

An SID specification of the Dating Problem is shown in Figure 9.39. The framework partly adopts the UID method of representing order asymmetry by introducing *clusters* of nodes (encapsulated in a dashed ellipse). In terms of information precedence, we can think of a cluster $\mathbf{C}$ of nodes as a single node in the sense that a structural arc going into $\mathbf{C}$ from a node X indicates

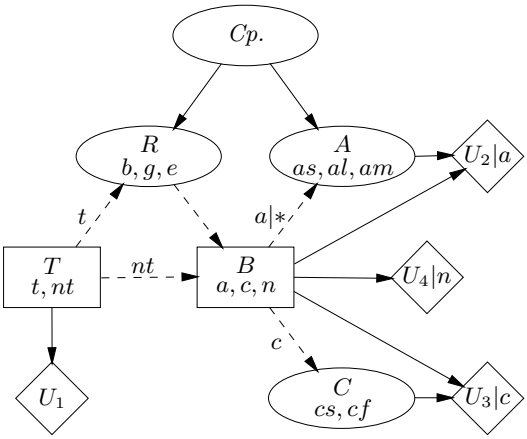


Fig. 9.38. A graphical representation of the Reactor Problem; the $*$ denotes that the choice $B = a$ is allowed only in scenarios that satisfy $(T = nt) \vee (T = t \wedge (R = e \vee R = g))$.

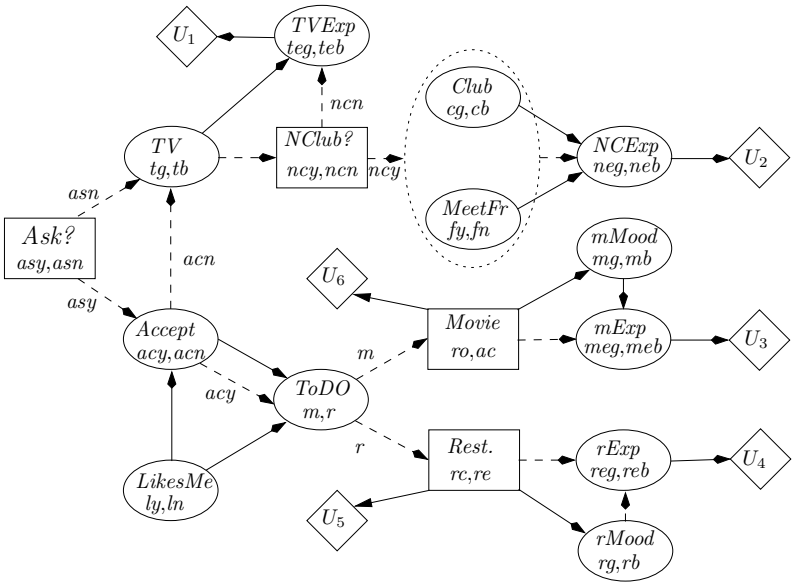


Fig. 9.39. An SID representation of the Dating Problem.

that when X has been observed or decided on, the next node is a node in $\mathbf{C}$. A structural arc from $\mathbf{C}$ to a node Y indicates that Y will be the next node in the ordering on leaving $\mathbf{C}$. Figure 9.39 illustrates the use of clusters for representing the partial temporal ordering over the chance nodes *Club* and *MeetFr*. From the model we see that these two nodes will be observed only after a decision on *NClub?* but before *NCExp* is observed.

A sequential influence diagram can be solved by unfolding it into a decision tree. There are, however, more efficient ways, which identify symmetric subtrees and solve them as influence diagrams, but that is outside the scope of this book.

9.6 Decision Problems with Unbounded Time Horizons

Consider a problem of robot navigation in which a robot is placed in some environment and its task is to find a path from its current position to a certain goal position. Each time the robot moves from one position to another it incurs a loss (fuel expenditure), but when it reaches the goal state it receives a reward and the navigation task ends. The aim is now to find a sequence of moves that will maximize the robot's expected reward (and minimize its expected loss):

The problem above is an example of a general type of problem called *planning under uncertainty*:

- at each step we are faced with the same type of decision,
- at each step we are given a certain reward (possibly negative) determined by the chosen decision and the state of the world,
- the outcome of a decision may be uncertain,
- the time horizon of the decision problem is unbounded.

Examples of other problems of this type include factory process control and transportation logistics.

In Section 9.4.3 we discussed a related type of decision problem, namely repetitive decision problems with a bounded time horizon. In what follows we extend this discussion to unbounded time horizons.

9.6.1 Markov Decision Processes

In the robot navigation problem above, the robot's process can roughly be described as an unbounded loop over the following events:

1. observe the state of the world (for example the robot's position in the world),
2. decide on the next action and collect the reward (possibly negative),
3. perform the action.

Using the influence diagram modeling language, we can represent the qualitative part of this problem by the structure in Figure 9.40. The node S_i represents the state of the world at step i; D_i is the ith decision of the robot; and R_i is the reward received when action D_i is performed in state S_i. The dashed arcs indicate that the future time horizon may be unbounded.

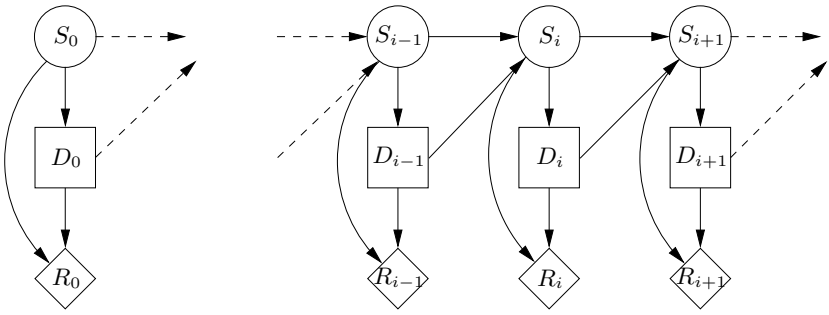


Fig. 9.40. A snapshot of a model of a Markov decision process. S_i represents the state at step i; D_i represents the ith decision; and R_i represents the reward of taking decision D_i in state S_i.

In order to specify the quantitative part of the model we need some additional information about the problem domain. Specifically, we shall assume that the robot is placed in the 3×3 grid environment shown in Figure 9.41. The robot can move *north*, *east*, *south*, and *west*, and for each move it incurs a loss of 0.1. If the robot decides to move, say, *north*, then this move will succeed with probability 0.7, and with probability 0.3 the robot will "slip" and move in one of the other three directions with equal probability; if the robot moves into a wall it will remain at its current position. At any point in time the robot can observe its exact position, and the aim is now to find a sequence of moves that will take it to the goal state at position $(3, 1)$ in the upper right corner. At the goal state it will receive a reward of 10, and from this state it cannot exit. Such a state is called a *terminal state*. At positions $(2, 2)$ and $(3, 2)$ two obstacles are placed that will incur a loss of 5 and 1, respectively. Although the environment is bounded, the decision problem is in principle unbounded. The robot may, for example, cycle between two positions an indefinite number of times before entering the goal state.

Returning to the model in Figure 9.40, we see that the variable S_i has a state for each possible position of the robot (a total of nine), and based on the description above, the associated transition function can (for $D_i = \mathit{north}$) be specified as in Table 9.5; the structure of the transition function is similar for the other actions. For this particular example, the reward function is independent of the chosen decision, and $R(S_i, D)$ $(= R(S_i))$ specifies a value for each position.

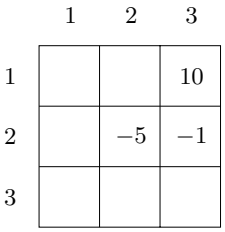


Fig. 9.41. A 3×3 grid world.

		S_i								
		$(1,1)$	$(1,2)$	$(1,3)$	$(2,1)$	$(2,2)$	$(2,3)$	$(3,1)$	$(3,2)$	$(3,3)$
S_{i+1}	$(1,1)$	0,8	0,7	0	0,1	0	0	0	0	0
	$(1,2)$	0,1	0,1	0,7	0	0,1	0	0	0	0
	$(1,3)$	0	0.1	0.2	0	0	0.1	0	0	0
	$(2,1)$	0.1	0	0	0.7	0.7	0	0	0	0
	$(2,2)$	0	0.1	0	0.1	0	0.7	0	0.1	0
	$(2,3)$	0	0	0.1	0	0.1	0.1	0	0	0.1
	$(3,1)$	0	0	0	0.1	0	0	1	0.7	0
	$(3,2)$	0	0	0	0	0.1	0	0	0.1	0.7
	$(3,3)$	0	0	0	0	0	0.1	0	0.1	0.2

Table 9.5. The transition function $P(S_{i+1} \mid \mathit{north}, S_i)$ for the robot navigation problem.

The robot navigation problem is an example of a *Markov decision process* (MDP). In general, in a Markov decision process:

- the world is *fully observable*, i.e., the agent can observe the true state of the world at any point in time,
- the uncertainty in the system is a result of the consequences of the actions being nondeterministic (when performing an action we make a state transition with a certain probability), and
- for each decision we get a reward (which may be negative) that may depend on the current world state.

More formally:

Definition 9.6 (Markov decision Processes). *An MDP consists of an unbounded set of identical time steps. Each time step i consists of:*

1. *A finite set of states of the world represented by the chance variable S_i.*
2. *A finite set of actions represented by the decision variable D_i.*
3. *A* transition function $P(S_{i+1} = s' \mid S_i = s, D_i = a)$ *specifying the probability that the next state is s' when action a is taken in state s.*
4. *A* reward function $R(S_i = s, D_i = a)$ *specifying the reward of taking action a in state s, for each $a \in \mathrm{sp}(D_i)$ and $s \in \mathrm{sp}(S_i)$.*

5. *An* initial state $s_0 \in S_0$.

The transition function and the reward function are the same for all time steps.

In the definition above, we require that at any given point in time the world state be represented by a single variable. This means that when specifying the transition function we need to elicit $|\text{sp}(S) \times \text{sp}(S)|$ probabilities for each decision. In order to make this elicitation task easier, you may exploit the internal structure of the world and represent S as a Bayesian network.

Types of Strategies

A policy for a decision variable is in general a function that returns a decision option for each possible configuration of the variables previously observed and decided on. In dealing with MDPs, however, the past is irrelevant in determining the optimal decision. More precisely, from the d-separation properties of the MDP model in Figure 9.40 we see that the future is independent of the past given the current state of the world S_i (this is also called the *Markov property*). Hence, instead of considering the past for decision D_i, it is sufficient to include only S_i:

$$\delta_{D_i} : \text{sp}(S_i) \rightarrow \text{sp}(D_i)\,.$$

In decision problems with a bounded time horizon we have previously defined an optimal strategy as a collection of optimal polices, one for each decision. However, in dealing with unbounded time horizons the situation is a bit different. To illustrate the difference, consider again the model in Figure 9.24 approximating the fishing in the North Sea decision problem (described in Section 9.4.3). Strictly speaking, according to Definition 9.6, this model is not an MDP, but by marginalizing out the unobserved variables we obtain the equivalent MDP structure in Figure 9.42.

In looking for an optimal strategy for this model it is obvious that the optimal policy for FV_1, say, is not necessarily the same as the optimal policy for FV_5 ($\delta_{FV_1}(T_1) \neq \delta_{FV_5}(T_5)$); even though the tests conducted at year 1 and year 5 produce the same results, the decisions at these two points in time will in general be different. For example, at year 1 the optimal policy may set the allowable catch to a conservative number to ensure that there will be enough fish in the forthcoming years. On the other hand, at year 5 these concerns are irrelevant, since the time horizon stops at that year, and the optimal policy may set the allowable catch to a higher volume. To take another example, in the robot navigation problem we look for a strategy for arriving at the goal state from some starting position, say $(2, 3)$. Suppose now that we have a finite time horizon and require that the robot should reach the goal state within 4 steps. With this constraint we do not have time to follow the route left around the center state $(2, 2)$ corresponding to the relative sequence of positions (*west*, *north*, *north*, *east*, *east*). Instead we would have to pass either $(2, 2)$ or $(3, 2)$, both of which incur a loss.

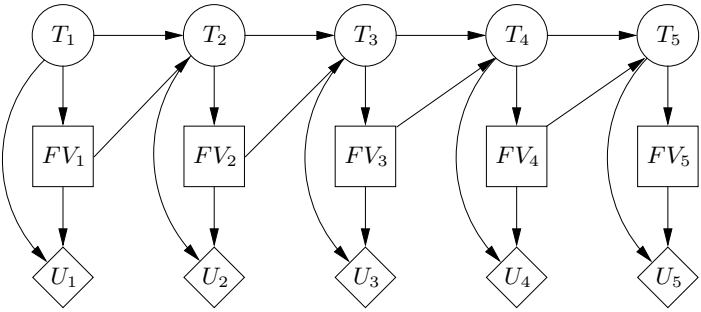


Fig. 9.42. The approximated model for fishing in the North Sea obtained from the original model in Figure 9.24 by marginalizing out the unobserved state variables V_i.

In general, we can say that optimal decisions at the end will be different from the ones at the beginning. For these situations we say that the optimal strategy is a *nonstationary strategy*.

Consider now the case in which we have an unbounded time horizon. At any time step, the optimal decision can depend only on the current state and what may happen in the future. If two time steps are in the same state, then they also have the same possibilities in the future, and therefore the optimal decision must be the same.

In the fishing in the North Sea example with unbounded time horizon, the optimal policy for deciding on the allowable catch at year 1 will not be any different from the policy at year 5. Similarly, in the robot example the optimal policy at state $(2, 3)$ does not depend on the point in time at which the robot entered that state. That is, when there is no fixed time horizon there is no reason to change the optimal policy for a given state at different points in time. For the robot example, this allows us to represent the optimal policy as in Figure 9.43.

More formally, an optimal strategy Δ consists of a set of identical policies, which are functions of only the current state. Such a strategy is called *stationary*, since it can be completely described by a single policy. We will not distinguish between a stationary strategy and a policy, and these terms will also be used interchangeably.

Optimality in Markov Decision Process

When evaluating a strategy for a decision problem with an unbounded time horizon, you might be tempted to simply consider the expected utilities/rewards for each time step and sum them up over time. However, if the process never stops, the sum may not be bounded, and you cannot compare two strategies with an expected reward of $+\infty$. This is not a problem for the

	1	2	3
1	→	→	
2	↑	↑	↑
3	↑	←	←

Fig. 9.43. A strategy for the robot in the 3×3 grid world.

robot example, since it has a terminal state in which the robot will eventually end up. However, in the fishing example, any catching policy that at each time step gives a positive reward will have an infinite sum. An immediate approach for handling this problem could be to specify some fixed horizon k so that the utility of a state sequence $s_0, s_1, s_2, \ldots$ is simply the sum of the rewards obtained at the first k states. For notational convenience we shall in this section assume that the reward is independent of the chosen action:

$$U(s_0, s_1, s_2, \ldots) = R(s_0) + R(s_1) + \cdots + R(s_k).$$

However, this raises the question of how to choose k, and, more importantly, it has the effect of postponing unpleasant decisions to after the horizon; in the extreme case in which $k = 0$ we care only about the immediate reward. The bounded fishing model in Figure 9.42 illustrates this point. With a fixed time horizon, you will be very greedy, in the end not caring about the volume of fish in later years.

Another approach is to weigh rewards in the immediate future higher than rewards in the distant future. This can be done by introducing a *discounting factor* γ, $0 \leq \gamma \leq 1$, so that the utility of a state sequence $s_0, s_1, s_2, \ldots$ is the accumulated discounted reward of each of the states:

$$U(s_0, s_1, s_2, \ldots) = R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \cdots .$$

In the extreme case that $\gamma = 0$, the agent considers all future rewards as being insignificant (corresponding to $k = 0$ above), and if $\gamma = 1$ then the discounted utility corresponds to having additive rewards as in the robot navigation problem. When $\gamma < 1$ the utility of an infinite sequence is always finite:

$$U(s_0, s_1, s_2, \ldots) = \sum_{i=0}^{\infty} \gamma^i R(s_i) \leq \sum_{i=0}^{\infty} \gamma^i \max R = \frac{\max R}{1 - \gamma}, \tag{9.1}$$

where $\max R$ is the maximum reward we can achieve in any state. A problem domain in which the discounted reward model has been applied is economics; here the discounting factor has been used, for example, to represent inflation or an interest rate. Discounted rewards have also been used to model unbounded

decision problems, in which the decision process may terminate at any point in time with probability $(1-\gamma)$. This could, for example, be used to model that there is a risk of $(1-\gamma)$ that the robot will break down after it has performed a move.

Some decision problems cannot naturally be modeled using discounted rewards. The robot navigation problem with no terminal state is an example of such a decision problem: the navigation task is not only to reach the goal state but also to avoid the obstacles, and if we disregard the potential problem of the robot breaking down, then there is no real justification for using discounted rewards (why should it be worse to hit an obstacle now than in the future?). In this situation, the *average reward* may be a more appropriate model:

$$U(s_0, s_1, s_2, \ldots) = \lim_{N\to\infty} \frac{1}{N} \sum_{i=0}^{N-1} R(s_i) \leq \max R.$$

No matter whether we use discounted reward or average reward, we should take into account that each strategy Δ corresponds to a set of different state sequences due to the actions being nondeterministic. For example, if the robot starts at $(1,3)$, then a performed action sequence (*north*, *north*, *east*, *east*) will result in the state sequence $[(1,2),(1,1),(2,1),(3,1)]$ with probability $0.7^4 = 0.2401$. Thus, we evaluate strategies based on their *expected reward*. Let $P(S_i \mid \Delta, s_0)$ be the probability distribution for S_i given that we start in s_0 and follow the strategy Δ. Then

$$\sum_{S_i} R(S_i) P(S_i \mid \Delta, s_0)$$

is the expected reward at step i, and $\gamma^i \sum_{S_i} R(S_i) P(S_i \mid \Delta, s_0)$ is the discounted expected reward. The expected reward of Δ is defined as

$$U^*(s, \Delta) = \lim_{N\to\infty} \sum_{i=0}^{N} \gamma^i \left(\sum_{S_i} R(S_i) P(S_i \mid \Delta, s_0) \right).$$

A standard notation for $U^*(s_0, \Delta)$ is also

$$\mathbb{E}\left[\sum_{i=0}^{\infty} \gamma^i R(s_i) \,\middle|\, \Delta, s \right].$$

In Section 10.6 we shall return to the actual calculation of these expectations.

9.6.2 Partially Observable Markov Decision Processes

In many decision problems, the assumption that the environment is fully observable is not realistic. For example, the sensors used by a robot for positioning may be inaccurate, and they will therefore provide only a blurred picture

of the state of the world. We call such an environment *partially observable*, and in the Bayesian framework we can encode this uncertainty with a probability distribution over the possible world states. For bounded horizons, we have actually encountered such a decision problem before, namely in the form of the more exact model for the fishing in the North Sea decision problem specified in Figure 9.23.

In general, we can model that type of decision problem as a so-called *partially observable Markov decision process* (POMDP) illustrated in Figure 9.44. In the POMDP model the node O_i represents the observation at step i, and the conditional probability distribution attached to this node encodes the uncertainty associated with the observation; the information arc from O_i to D_i specifies that only O_i is observed immediately before decision D_i. More formally, a POMDP consists of:

1. A set of states and actions as in the MDP framework.
2. A transition function and a reward function as specified for the MDP.
3. A set of possible observations represented by the chance variable O_i at time step i.
4. An *observation function* $P(O_i \mid S_i, D_{i-1})$ that specifies the probability of the possible observations conditioned on the current state of the world and the last decision.

Observe that as for the MDP we use a single variable to represent the observation and the state at the ith time step. However, as for the MDP, we can consider these variables as being the products of several variables, so that both the transition function and the observation function can be specified more compactly using a Bayesian network. To simplify the model, we will stick to the single-variable representations.

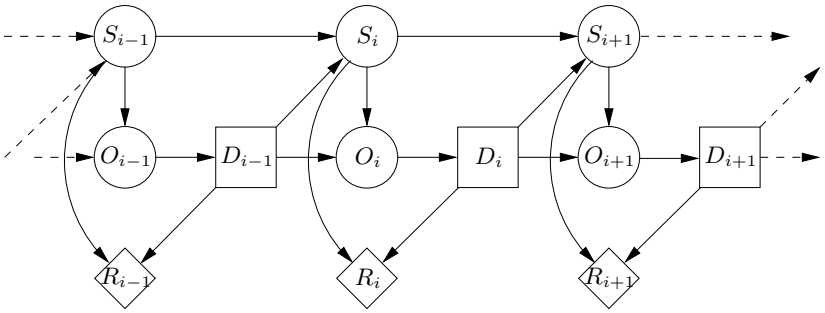


Fig. 9.44. A snapshot of a model of a partially observable Markov decision process. The state of the world S_i is observed only indirectly through the observation node O_i.

When the world is only partially observable we can no longer execute an action based on the current state of the world. In fact, based on the d-separation properties of the model in Figure 9.44, we see that when decision D_i is taken, all previous observations and decisions are d-connected to the current and future state variables, hence the entire past is relevant when the decision is taken. Another way of interpreting this situation is that all our previous observations and decisions have an impact on our current beliefs about the state of the world, and our ensuing action is based on these beliefs. This also means that while for MDPs we specified a policy conditionally on the observed state of the world, we should now specify a policy conditionally on our belief of the state of the world. Since the actual state of the world is not observed, our belief will in general not point to any specific state but will rather be a probability distribution over the possible states. That is, our belief can be expressed as a probability distribution $P(S_i \mid D_1, O_1, \ldots, D_{i-1}, O_i)$, and an optimal policy for step i will therefore specify an action for each possible probability distribution over S_i. This implies that if $P(S_i|past_i) = P(S_j|past_j)$, then the optimal decisions for D_i and D_j are the same.

9.7 Summary

One Action

Decision D, utility functions $U_1, \ldots, U_n$ over domains $X_1, \ldots, X_n$, evidence e. The expected utility is

$$\mathrm{EU}(D \mid e) = \sum_{X_1} U_1(X_1) P(X_1 \mid D, e) + \cdots + \sum_{X_n} U_n(X_n) P(X_n \mid D, e),$$

and a state d maximizing $\mathrm{EU}(D \mid e)$ is chosen as an optimal action.

Instrumental Rationality

For an individual who acts according to a preference ordering satisfying the rules below, there exists a utility function so that the individual maximizes the expected utility.

1. *Reflexivity.* For any lottery A, $A \succeq A$.
2. *Completeness.* For any pair (A, B) of lotteries, $A \succeq B$ or $B \succeq A$.
3. *Transitivity.* If $A \succeq B$ and $B \succeq C$, then $A \succeq C$.
4. *Preference increasing with probability.* If $A \succeq B$ then $\alpha A + (1-\alpha)B \succeq \beta A + (1-\beta)B$ if and only if $\alpha \geq \beta$.
5. *Continuity.* If $A \succeq B \succeq C$ then there exists $\alpha \in [0, 1]$ such that $B \sim \alpha A + (1-\alpha)C$.
6. *Independence.* If $C = \alpha A + (1-\alpha)B$ and $A \sim D$, then $C \sim (\alpha D + (1-\alpha)B)$.

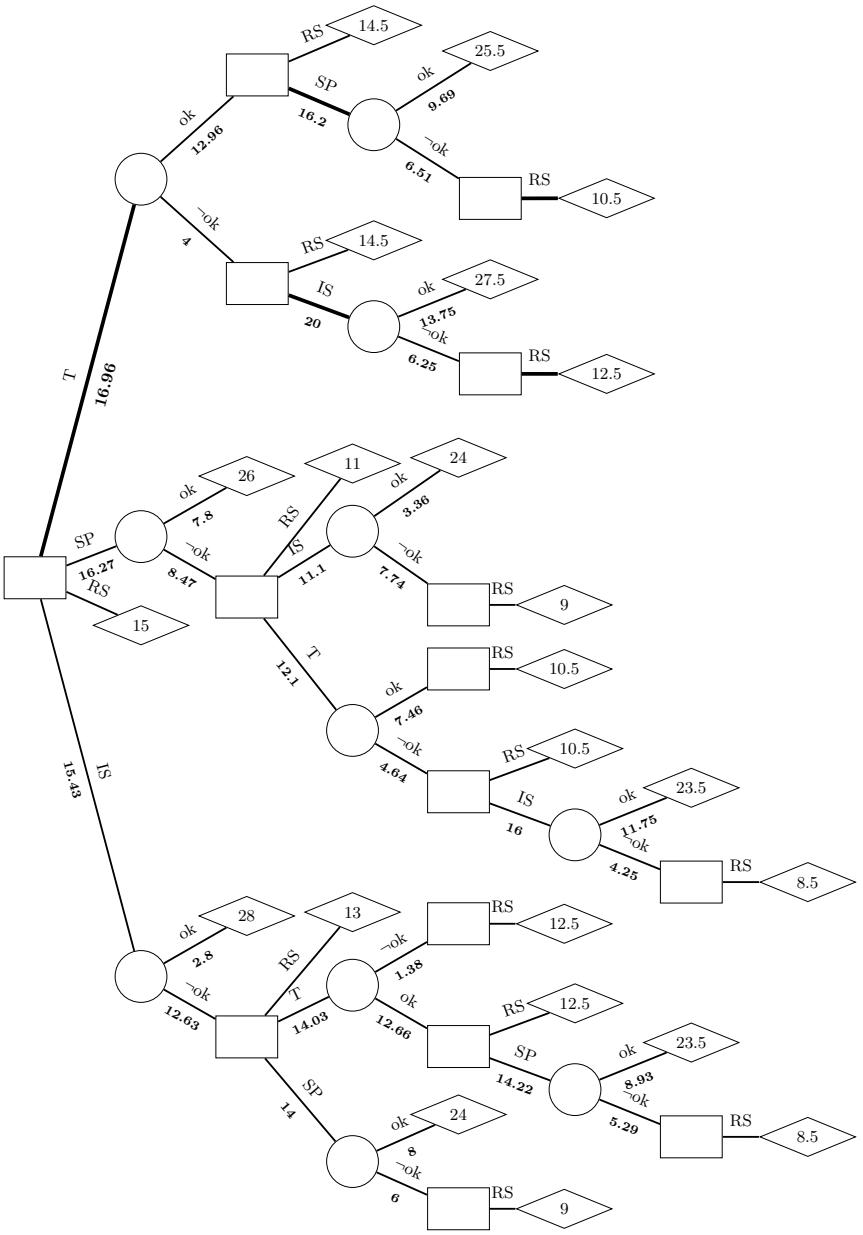


Fig. 9.45. An example of a decision tree. The probabilities may be taken from a Bayesian network. The bold links indicate an optimal strategy.

Decision Trees

An example is shown in Figure 9.45.
To calculate an optimal strategy and the maximum expected utility for the subtree rooted at node X, do:

1. If X is a utility node, then return $U(X)$.
2. If X is a chance node, then return

$$\mathrm{EU}(X) = \sum_{x \in \mathrm{sp}(X)} P(X = x \mid \mathrm{past}(X)) \, \mathrm{EU}(N(X = x)).$$

3. If X is a decision node, then return

$$\mathrm{EU}(X) = \max_{x \in \mathrm{sp}(X)} \mathrm{EU}(N(X = x)),$$

and mark the arc labeled

$$x' = \arg \max_{x \in \mathrm{sp}(X)} \mathrm{EU}(N(X = x)).$$

Influence Diagrams

An *influence diagram* consists of a directed acyclic graph over chance nodes, decision nodes, and utility nodes with the following structural properties:

- there is a directed path comprising all decision nodes;
- the utility nodes have no children.

For the quantitative specification, we require that:

- the decision nodes and the chance nodes have a finite set of mutually exclusive states;
- the utility nodes have no states;
- to each chance node A there be attached a conditional probability table $P(A \mid pa(A))$;
- to each utility node V there be attached a real-valued function over $pa(V)$.

Figure 9.46 gives an example of the structural part of an influence diagram.

A *policy* for decision D_i is a mapping δ_i that for any configuration of the past of D_i yields a decision for D_i. That is

$$\delta_i(\mathcal{I}_0, D_1, \ldots, D_{i-1}, \mathcal{I}_{i-1}) \in \mathrm{sp}(D_i)\,.$$

A *strategy* for an influence diagram is a set of policies, one for each decision. A *solution* to an influence diagram is a strategy maximizing the expected utility.

Methods for determining optimal strategies from influence diagrams are given in Chapter 10.

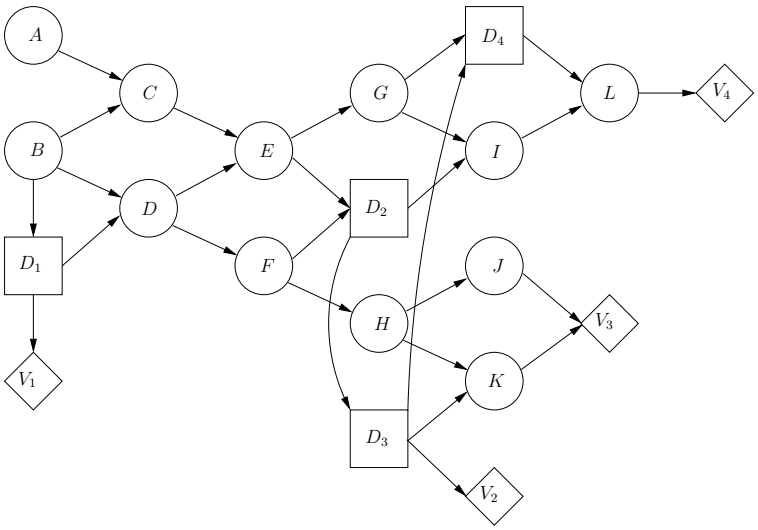


Fig. 9.46. An example of the structure of an influence diagram. We have $\mathcal{I}_0 = \{B\}, \mathcal{I}_1 = \{E, F\}, \mathcal{I}_2$ is empty, $\mathcal{I}_3 = \{G\}, \mathcal{I}_4 = \{A, C, D, H, I, J, K, L\}$.

Asymmetric Decision Problems

A decision problem is said to be *symmetric* if:

- in all of its decision tree representations, the number of scenarios is the same as the cardinality of the Cartesian product of the state spaces of all chance and decision variables, and
- in at least one decision tree representation, the sequence of chance and decision variables is the same in all scenarios.

There are three types of asymmetry:

Functional asymmetry: The possible outcomes or decision options of a variable may vary depending on the past.

Structural asymmetry: The very occurrence of an observation or a decision depends on the past.

Order asymmetry: The ordering of the decisions and observations is not settled at the time the model is specified.

Unconstrained Influence Diagrams

Unconstrained influence diagrams are used to model order asymmetry. Compared to influence diagrams there need not be a total ordering of the decisions, and the chance variables are partitioned into two sets: observable chance variables and nonobservable chance variables. An observable chance variable is

released (for observation) when all its antecedent decision variables have been decided on.

Solving an unconstrained influence diagram involves finding the next action as well as finding an optimal policy if the next action is a decision (that is, finding the conditional sequence of action and observations maximizing the expected utility). The solution is specified in terms of an S-DAG:

An S-DAG is a directed acyclic graph G. The nodes are labeled with variables from $\mathcal{D}_U \cup \mathcal{O}_U$ such that each maximal directed path in G represents an admissible ordering of $\mathcal{D}_U \cup \mathcal{O}_U$.

A *step policy* for a node N in an S-DAG G is a function

$$\sigma : \mathrm{sp}(\mathrm{hst}(N)) \rightarrow \mathrm{ch}(N).$$

A *step strategy* for U is a pair $(G, \mathcal{S})$, where G is an S-DAG for U and $\mathcal{S}$ is a set of step policies, one for each node in G (except for *Sink*). A *policy* for N is a function

$$\delta : \mathrm{sp}(\mathit{past}(N)) \rightarrow \mathrm{ch}(N).$$

A *strategy* for U is a step strategy together with a policy for each node.

Decision Problems with an Unbounded Time Horizon

An MDP consists of an unbounded set of identical time steps. Each time step i consists of:

1. A finite set of states of the world (represented by the chance variable S_i).
2. A finite set of actions (represented by the decision variable D_i).
3. A *transition function* $P(S_{i+1} = s' \mid S_i = s, D_i = a)$ specifying the probability that the next state is s' when taking action a in state s.
4. A *reward function* $R(S_i = s, D_i = a)$ specifying the reward of taking action a in state s, for each $a \in \mathrm{sp}(D_i)$ and $s \in \mathrm{sp}(S_i)$.
5. An *initial state* $s_0 \in S_0$.

The transition function and the reward function are the same for all time steps.

There are three standard ways to ensure that the utility of an unbounded state sequence $s_0, s_1, s_2, \ldots$ is bounded:

Fixed time horizon: The sum of the rewards obtained at the first k states:

$$U(s_0, s_1, s_2, \ldots) = R(s_0) + R(s_1) + \cdots + R(s_k).$$

Discounted reward: The accumulated discounted reward of each of the states:

$$U(s_0, s_1, s_2, \ldots) = R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \cdots ,$$

where $0 \geq \gamma < 1$.

Average expected reward: The accumulated average reward at each of the states:

$$U(s_0, s_1, s_2, \ldots) = \lim_{N \to \infty} \frac{1}{N} \sum_{i=0}^{N-1} R(s_i).$$

A POMDP consists of:

1. A set of states and actions as in the MDP framework.
2. A transition function and a reward function as specified for the MDP.
3. A set of possible observations (represented by the chance variable O_i at time step i).
4. An *observation function* $P(O_i \mid S_i, D_{i-1})$ that specifies the probability of the possible observations conditioned on the current state of the world and the last decision.

9.8 Bibliographical Notes

Decision theory has a long history but achieved a breakthrough in the work of von Neumann and Morgenstern (1944), who laid down the axioms for instrumental rationality. Decision trees were introduced by Raiffa and Schlaifer (1961). Influence diagrams were proposed by Howard and Matheson (1981), and were adapted to allow for additive decompositions of utility functions in (Tatman and Shachter, 1990). Unconstrained influence diagrams were introduced in (Jensen and Vomlelova, 2002), and sequential influence diagrams in (Jensen *et al.*, 2006). The latter is a fusion of the valuation networks of Shenoy (1996) and the asymmetric influence diagrams of Nielsen and Jensen (2003a). The study of Markov decision processes can be traced back at least to Howard (1960). A good starting point for further reading is (Puterman, 1994). Partially observed Markov decision processes originate with Drake (1962) and Åström (1965). The reactor problem, as presented here, is due to Covaliu and Oliver (1995).

9.9 Exercises

Exercise 9.1. Consider the management of effort example in Section 9.2.

(i) Let the marks be $0, 5, 6, 8, 9, 10$. What is the optimal decision if the numerical values are used as utilities?

(ii) Consider the approach in which the marks are given subjective utilities. Show that action Gd can be optimal only if the mark 0 is given higher utility than mark 3.

Exercise 9.2. Prove that if U is a utility function for a decision maker and if a ($a > 0$) and b are real numbers, then $aU + b$ is an equivalent utility function.

Exercise 9.3. ^{E} Extend the model from Exercise 3.14 to a model for folding or calling.

Exercise 9.4. ^{E} Extend Exercise 3.18 with the following:

> In golf, the task is to use as few strokes as possible at each hole. I am driving at a hole 260 m long. If the drive is 265 m, I will on average use 1.8 strokes to finish the hole. If the drive is 240 m, on average 2 extra strokes are needed; 220 m requires 2.5 extra strokes; 200 m requires 2.7; 180 m 2.9 extra strokes; 160 m 3.1; 145 m 3.3; a drive of 290 m will carry the ball into a sand trap, requiring 3.5 extra strokes; if the drive is misshit, the ball will drop into a lake, and it will require 4.5 extra strokes to finish the hole.

Construct a system that helps me decide whether to use the 3-wood or the driver in the drive.

Exercise 9.5. ^{E} Consider the stud farm example from Section 3.2.2. Extend the model to be an aid for deciding for each horse whether it should be taken out of breeding. Table 9.6 gives the utilities.

	Carrier	*Pure*
Out	−10	−10
In	−40	100

Stallions

	Carrier	*Pure*
Out	−3	−3
In	−10	40

Mares

Table 9.6. Tables for Exercise 9.5.

Exercise 9.6. Let the hypothesis variable H have n states. Introduce an action variable A with the same states as H; let the utility table be as follows:

$$U(h,a) = \begin{cases} 1 \text{ if } h \text{ and } a \text{ are the same,} \\ 0 \text{ otherwise.} \end{cases}$$

Show that a value function based on U corresponds to selecting a hypothesis state of highest probability.

Exercise 9.7. Construct a decision tree for the mildew decision problem in Section 9.1.2. How many numbers would you need to specify to render it complete?

Exercise 9.8. Solve the decision tree in Figure 9.47.

Exercise 9.9. Consider an altered version of the poker decision problem in which each player is now allowed three rounds of changing hands. What would an influence diagram look like for this altered problem? What is the past for each decision variable in the diagram?

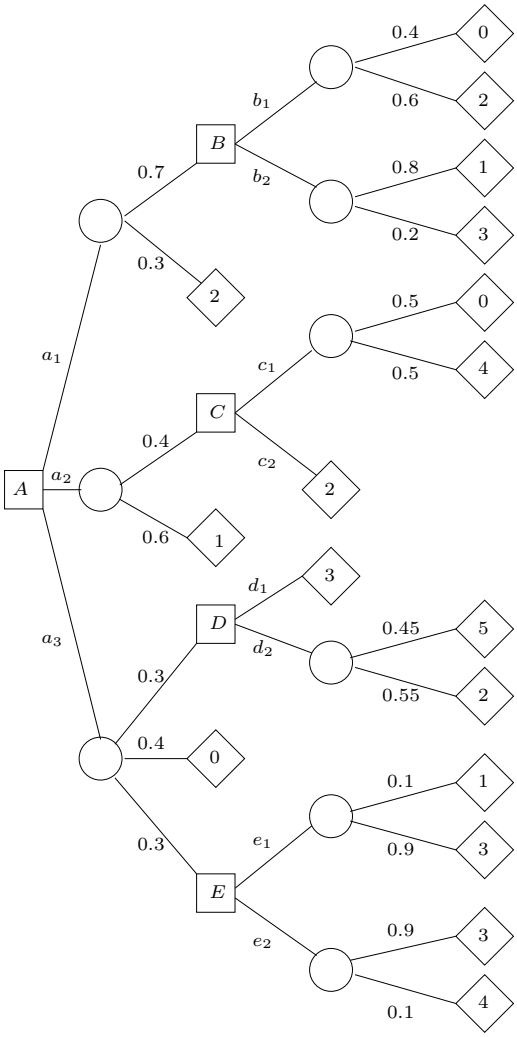


Fig. 9.47. Figure for Exercise 9.8.

Exercise 9.10. What is the partial temporal ordering of observations and decisions in the influence diagrams in Figures 9.23 and 9.24?

Exercise 9.11. ^{E} (The oil wildcatter's problem)

An oil wildcatter must decide whether to drill or not to drill. The cost of drilling is \$70,000. If he decides to drill, the hole may be soaking (with a return of \$270,000), wet (with a return of \$120,000), or dry (with a return of \$0). The prior probabilities for soaking, wet, and dry are (0.2, 0.3, 0.5). At the cost of \$10,000, the oil wildcatter could

decide to take seismic soundings of the geological structure at the site. The specifics of the test are given in Table 9.7.

$T \setminus S$	dr	wt	so
n	0.6	0.3	0.1
o	0.3	0.4	0.4
c	0.1	0.3	0.5

$P(Test \mid Structure)$

Table 9.7. Table for Exercise 9.11. The states n, o, and c are the outcomes of the test.

(i) Solve the problem with a decision tree.
(ii) Solve the problem with an influence diagram.

Exercise 9.12. (The used car buyer's problem)

Joe is considering buying a used car from a dealer for $1,000. The market price of similar cars with no defects is $1,100. Joe is uncertain whether the particular car he is considering is a "peach" or a "lemon." Of the ten major subsystems in the car, a peach has a serious defect in only one subsystem, whereas a lemon has a serious defect in six subsystems. The probability that the used car under consideration is a lemon is 0.2. The cost of repairing one defect is $40, and the cost of repairing six defects is $200.

For an additional $60, Joe can buy the car from the dealer with an "antilemon guarantee." The antilemon guarantee will normally pay for 50% of the repair cost, but if the car is a lemon, then the guarantee will pay 100% of the repair cost.

Before buying the car, Joe has the option of having the car examined by a mechanic for an hour. In this period, the mechanic offers three alternatives t_1, t_2, t_3 as follows:

t_1: test the steering subsystem alone at a cost of $9,

t_2: test the fuel and electrical subsystems for a total cost of $13,

t_3: do a two-test sequence in which Joe can authorize a second test after the result of the first test is known. In this alternative, the mechanic will first test the transmission subsystem at a cost of $10 and report the results to Joe. If Joe approves, the mechanic will then proceed to test the differential subsystem at an additional cost of $4.

All tests are guaranteed to find a defect in the subsystem if a defect exists. We assume that Joe's utility for profit is linear in dollars.

(i) Solve the problem with a decision tree.
(ii) Consider how to represent the problem as an influence diagram (you may add dummy states and variables as you wish).

Exercise 9.13. Draw an influence diagram for the decision problem in Section 9.1.2.

Exercise 9.14. Solve the decision tree in Figures 9.8 and 9.9 (the probabilities can be taken from the model in Figure 9.10).

Exercise 9.15. Complete the reduced decision tree from Figure 9.16 and solve it.

Exercise 9.16. ^{E} Solve Exercise 3.16 as a decision problem.

Exercise 9.17. ^{E} Solve the example in Section 11.1.1 as an influence diagram.

Exercise 9.18. ^{E} Extend the poker model from Exercise 9.3 to the influence diagram in Figure 9.21.

Exercise 9.19. ^{E} Represent the Car Start Problem in Section 9.3.1 as an influence diagram. (What are the decision options at each step?)

Exercise 9.20. Unfold the sequential influence diagram in Figure 9.38 with the following probabilities: A conventional reactor (C) has probability 0.980 of being successful (cs), and a probability 0.020 of a failure (cf). An advanced reactor (A) has probability 0.660 of being successful (as), probability 0.244 of a limited accident (al), and probability 0.096 of a major accident (am).

Exercise 9.21. Consider the Dating Problem in Example 9.6. What are the asymmetries in the decision problem? Which of them are functional asymmetries/structural asymmetries/order asymmetries?

Exercise 9.22. Construct an S-DAG for the UID in Figure 9.48.

Exercise 9.23. Consider the two-player turn-taking game of tic-tac-toe in which each player has three game pieces, and the objective is to place all your pieces in a straight line on a 3×3 board. The players take turns placing a piece in one of the free slots on the board, and when a player has no more pieces off the board, he must take one of his pieces already on the board and place it somewhere else. Formalize the game as a Markov decision process, seen from the point of view of one of the players.

Exercise 9.24. Consider the example of the possibly infected milk from a single cow introduced in Sections 3.1.1 and 3.2.1. Add to that the daily decision of throwing the milk out or pouring it into the tank, and associate the utility of

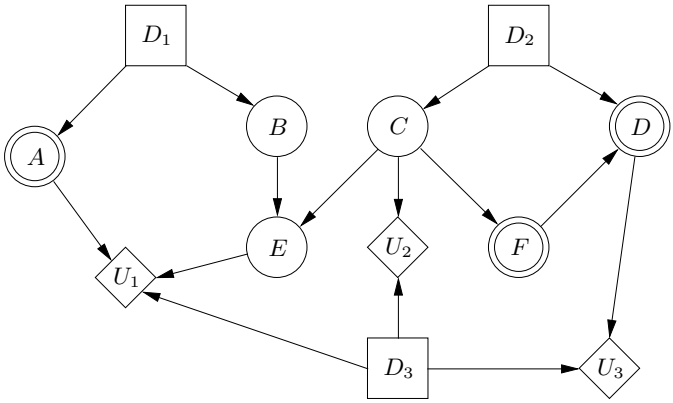


Fig. 9.48. A UID.

- 0 with pouring infected milk into the tank,
- 98 with throwing the milk out, and
- 100 with pouring noninfected milk into the tank.

Formalize the setting as a POMDP.