cis32-ai — lecture # 23 — mon-1-may-2006

today's topics:

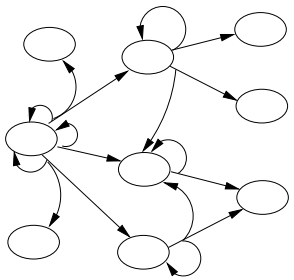
- decision-theoretic planning (finish from last time)
- dealing with uncertainty

Markov decision processes

- So far, there is nothing really new here.
- But it is only a small step to a much better representation.
- In a non-deterministic environment, we don't have a simple transition function.
- Instead an action can lead to one of a number of states.
- When we can tell which state we are in, then we have a Markov decision process (MDP)

- An MDP has the following formal model:
  - a state space S;
  - a set of actions, $A(s) \subseteq A$, applicable in each state $s \in S$;
  - transition probabilities $\Pr_a(s' \mid s)$ for $s, s' \in S$ and $a \in A$;
  - action costs $c(a, s) > 0$; and
  - a set of goal states $G \subseteq S$
- Thus for each state we have a set of actions we can apply, and these take us to other states with some probability.
- We don't know which state we will end up in, but we know which one we are in after the action (we have *full observability*).

- This gives us a problem space that looks like:



- A solution is now choice of action in every possible state that the agent might end up in.

- We can think of this solution as a function π which maps states into applicable actions, $\pi(s_i) = a_i$.
- This function is called a *policy*.
- What a policy allows us to compute is a probability distribution across all the trajectories from a given initial state.
- This is the product of all the transition probabilities, $\Pr_{a_i}(s_{i+1} \mid s_i)$, along the trajectory.
- Goal states are taken to have no cost, no effects, so that if $s \in G$:
  - $c(a, s) = 0$
  - $\Pr(s \mid s) = 1$

- We can then calculate the expected cost of a policy starting in state s_0.
- This is just the probability of the policy multiplied by the cost of traversing it:

$$\sum_{i=0}^{\infty} c(\pi(s_i), s_i)$$

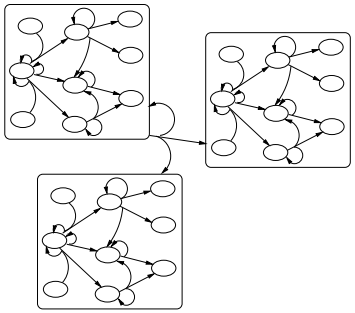
- An optimal policy is then a π^* that has minimum expected cost for all states s.
- As with the search version of the problem, we can solve this by searching, albeit through a much larger space.
- Later we will look at ways to do this search.

Partially observable MDPs

- Full observability is a big assumption (it requires an accessible environment). Much more likely is *partial observability*.
- This means that we don't know what state we are in, but instead we have some set of beliefs about which state we are in.
- We represent these beliefs by a probability distribution over the set of possible states.
- These probabilities are obtained by making observations.
- The effect of observations are modelled as probabilities $\Pr_a(o \mid s)$, where o are observations.

- Formally a POMDP is:
  - a state space S;
  - a set of actions, $A(s) \subseteq A$, applicable in each state $s \in S$;
  - transition probabilities $\Pr_a(s' \mid s)$ for $s, s' \in S$ and $a \in A$;
  - action costs $c(a, s) > 0$;
  - a set of goal states, G;
  - an initial belief state b_0;
  - a set of final belief states b_F;
  - observations o after action a with probabilities $\Pr_a(o \mid s)$

- So we have a situation which looks like:



- This is just an MDP over belief states.

- The goal states of an MDP are just replaced by, for example, states in which we are pretty sure we have reached a goal:

$$\sum_{s \in G} b(s) > 1 - \epsilon$$

- We solve a POMDP by looking for a function which maps belief states into actions, where belief states b are probability distributions over the set of states S.
- Given a belief state b, the effect of carrying out action a is:

$$b_a(s) = \sum_{s' \in S} \Pr_a(s \mid s') b(s')$$

- If we carry out a in b and then observe o, we get to state b_a^o:

$$b_a^o(s) = \frac{\Pr_a(o \mid s) b_a(s)}{\Sigma_{s' \in S} \Pr_a(o \mid s') b_a(s')}$$

- The term on the bottom is the probability of observing o after doing a in b.
- Thus actions map between belief states with probability:

$$b_a(o) = \sum_{s' \in S} \Pr_a(o \mid s') b_a(s')$$

and we want to find a trajectory from b_0 to b_F at minimum cost.

dealing with uncertainty

- We have considered logic-based approaches to reasoning about the world and planning what to do.
- These techniques deal with some of the problems agents face.
- However, they stop some way short of what we need to deal with real environments.
- They have problems when environments are:
  - non-deterministic
  - inaccessible
  - dynamic
- This lecture will discuss some issues when dealing with these more complex environments and start on some solutions

Environment Issues

- When environments are non-deterministic, actions don't have predictable outcomes.
- Actions can fail.
- This can partly be handled by interleaving planning and acting.
- A more robust solution is to try and model the non-determinism.
- This can be done by handling the inherent *uncertainty* in actions.

- When environments are inaccessible, we don't know everything about an environment.
- An agent's knowledge is *incomplete*.
- We can make assumptions to "fill in" gaps in the knowledge-base but these may turn out to be false.
- Again, interleaving planning and acting can help, but...
- ...we need mechanisms for making assumptions, and...
- ...we need mechanisms for retracting false assumptions

- When environments are dynamic, the word can change as a result of things other than the agent's actions
- One way of looking at this is that:
  - It makes the world non-deterministic
  - It makes the world inaccessible
- Thus we have to:
  - model non-determinism/dynamism
  - handle assumptions.
- Doing these things is somewhat out of the scope of first order logic.

Incompleteness of knowledge

- Consider the blocks world with three blocks A, B, and C
- We have a goal

$$On(A, B), On(B, C)$$

- However, we cannot observe C, so the only knowledge about the world that we have is:

$$OnTable(A), OnTable(B)$$

- How do we proceed?
- Well, one thing we can do is to assume that unless we know otherwise, a block is on the table.

- This allows us to complete the knowledge-base to get:

$$OnTable(A), OnTable(B), OnTable(C)$$

  and from this we can build a plan.
- The problem is, what happens when we find out that C is really on top of B.
- Not only does this invalidate the plan, but the knowledge-base becomes:

$$OnTable(A), OnTable(B), OnTable(C), On(C, B)$$

- This will confuse the planner, and so we need a mechanism to identify that:

$$On(C, B) \Leftrightarrow \neg OnTable(C)$$

  and then remove the assumption.
- Doing this in logic is hard—logic doesn't allow you to remove things.
- Handling this kind of problem is the domain of *non-monotonic logic* and *commonsense* reasoning.

Uncertainty of knowledge

- Consider the block's world with three blocks A, B, and C
- We have a goal

$$On(A, B), On(B, C)$$

- We have an initial state:

$$OnTable(A), On(B, C)$$

- How do we proceed?
- Obviously we want to pick-up A and put it on B.
- What happens if picking up a block is not reliable?

- One thing we can do is to distinguish between things that we know will be true and things that may be true.
- Things that will always be true are *necessary truths*:
  - $2 + 2 = 4$
  - Simon is mortal
- Things that may be true are *possible truths*:
  - Tomorrow will be sunny
  - $Holding(A)$ will be true after $Pickup(A)$
- Dealing with "possible" and "necessary" can be done in *modal logic*.

- Another approach is to use probability theory.
- Thus we can say that 90% of the time a $Pickup$ will leave the block in the robot hand, 10% of the time it will leave the block on the table.
- We can then write the STRIPS operator for Pickup as:

$$
\begin{array}{ll}
 & Pickup(x) \\
\text{pre} & Clear(x) \wedge OnTable(x) \wedge ArmEmpty \\
\text{del} & OnTable(x) \wedge ArmEmpty \\
\text{add} & (Holding(x)\ 0.9) \wedge (OnTable(x)\ 0.1) \\
 & \quad \wedge (ArmEmpty\ 0.1)
\end{array}
$$

- When we combine steps in a plan, we have to take these probabilities into account.
- Consider that we add a $Stack(A, B)$ operation after the $Pickup(A)$ operation, and $Stack$ only has a 75% chance of working.
- The chance of having the goal is the probability of picking up the block correctly multiplied by the probability of stacking it:

$$
\begin{aligned}
\Pr(goal) &= 0.9 \times 0.75 \\
&= 0.675
\end{aligned}
$$

- Similarly, the probability that A is still on the table is 0.1, and the probability that it was picked up but then fell on the floor when the $Stack$ failed is 0.225.

- Doing something similar for a more complex plan allows us to calculate the probability that the whole plan succeeds
- Since some operations are more reliable than others, it may be that some plans have a bigger chance of success than others.
- Searching through the whole space of plans we can find the ones most likely to succeed.
- We can also model the interaction between actions this way.
- For instance if a $Stack$ is more reliable when it follows a $Pickup$ than when it follows an $UnStack$.

- While this simple approach works well enough for STRIPS operators, it fails to work well for logic in general.
- This is because if we have A and $A \Rightarrow B$, and both of these are uncertain, modus ponens is not much help.
- From $\Pr(A)$ and $\Pr(A \Rightarrow B)$ we can only calculate that:

$$\Pr(A) + \Pr(A \Rightarrow B) - 1 \leq \Pr(B) \leq \Pr(A \Rightarrow B)$$

- So we get an interval, and these intervals rapidly go towards $[0, 1]$.
- This means we have to find representations other than logic when we have to handle non-determinism.

Probability theory

- We start with a *sample space* Ω.
- For instance, Ω for the action of rolling a die would be $\{1, 2, 3, 4, 5, 6\}$.
- Subsets of Ω then correspond to particular events. The set $\{2, 4, 6\}$ corresponds to the event of rolling an even number.
- We use S to denote the set of all possible events:

$$S = 2^{\Omega}$$

- It is sometimes helpful to think of the sample space in terms of Venn diagrams—indeed all probability calculations can be carried out in this way.

- A probability measure is a function:

$$\Pr : S \mapsto [0, 1]$$

such that:

$$\begin{aligned} \Pr(\emptyset) &= 0 \\ \Pr(\Omega) &= 1 \\ \Pr(E \cup F) &= Pr(E) + \Pr(F), \text{whenever } E \cap F = \emptyset \end{aligned}$$

- Saying $E \cap F = \emptyset$ is the same as saying that E and F cannot occur together.
- They are thus *disjoint* or *exclusive*.
- The meaning of a probability is somewhat fraught; both frequency and subjective belief (Bayesian) interpretations are problematic.

- If the occurrence of an event E has no effect on the occurrence of an event F, then the two are said to be *independent*.
- An example of two independent events are the throwing of a 2 on the first roll of a die, and a 3 on the second.
- If E and F are independent, then:

$$\Pr(E \cap F) = \Pr(E).\Pr(F)$$

- When E and F are not independent, we need to use:

$$\Pr(E \cap F) = \Pr(E).\Pr(F|E)$$

where $\Pr(F|E)$ is the *conditional probability* of F given that E is known to have occurred.

- To see how $\Pr(F)$ and $\Pr(F|E)$ differ, consider F is the event "a 2 is thrown" and E is the event "the number is even".

- We can calculate conditional probabilities from:

$$\Pr(F|E) = \frac{\Pr(E \cap F)}{\Pr(E)}$$
$$\Pr(E|F) = \frac{\Pr(E \cap F)}{\Pr(F)}$$

which, admittedly is rather circular.

- We can combine these two identities to obtain *Bayes' rule*:

$$\Pr(F|E) = \frac{\Pr(E|F)\Pr(F)}{\Pr(E)}$$

- Also of use is *Jeffrey's rule*:

$$\Pr(F) = \Pr(F|E)\Pr(E) + \Pr(F|\neg E)\Pr(\neg E)$$

- More general versions are appropriate when considering events with several different possible outcomes.

- Consider being offered a bet in which you pay \$2 if an odd number is rolled on a die, and win \$3 if an even number appears.
- To analyse this prospect we introduce a *random variable* X, as the function:

$$X : \Omega \mapsto \Re$$

from the sample space to the values of the outcomes. Thus for $\omega \in \Omega$:

$$X(\omega) = \begin{cases} 3, & \text{if } \omega = 2, 4, 6 \\ -2, & \text{if } \omega = 1, 3, 5 \end{cases}$$

- The probability that X takes the value 3 is:

$$\begin{aligned} \Pr(\{2, 4, 6\}) &= \Pr(\{2\}) + \Pr(\{4\}) + \Pr(\{6\}) \\ &= 0.5 \end{aligned}$$

- How do we analyse how much this bet is worth to us?

- To do this, we need to calculate the *expected value* of X.
- This is defined by:

$$E(X) = \sum_k k \Pr(X = k)$$

where the summation is over all values of k for which $\Pr(X = k) \neq 0$.

- Thus the expected value of X is \$1.5, and we take this to be the value of the bet.
- And now we can make a first stab at defining how to decide what to do in an uncertain world.
- We chose the action which has maximum expected value.

Probabilistic network models

- If we have a situation involving a number of random variables, the full probability model requires the joint distribution across those variables.
- The classical view is that we do this by establishing all the relevant joint probabilities.
- Thus, given a set of variables $\{X^1, X^2, \ldots X^n\}$, where X^i has values $x_1^i, \ldots, x_{m_i}^i$ we need to establish the probabilities

$$\begin{aligned} &\Pr(x_1^1, x_1^2, \ldots, x_1^n) \\ &\Pr(x_2^1, x_1^2, \ldots, x_1^n) \\ &\vdots \\ &\Pr(x_{m_1}^1, x_{m_2}^2, \ldots, x_{m_n}^n) \end{aligned}$$

- Even for a relatively small model, involving say 20 binary valued variables, the number of joint values required is large, in this case:

$$2^{20} \approx 1,000,000$$

- Even if all the probabilities can be found, calculating marginal values would require a huge number of calculations—to establish $\Pr(x_1^1)$ would need us to sum over all joint values in which the value x_1^1 appears, and there are:

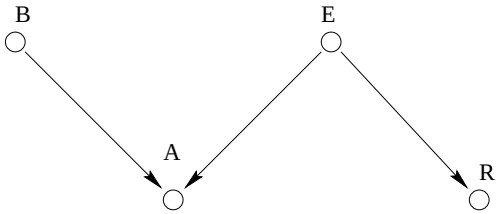
$$2^{19} \approx 500,000$$

of these.

- Furthermore, we would need to do this summation every time we wanted to take account of a new piece of information

- This is because such a change would typically change the marginal value of at least one value of one random variable.
- This in turn would affect every joint probability which includes that value.
- Since many problems in artificial intelligence (AI) are much larger than this, many people argued that probability could not be used in AI.
- As a result, those people turned their attention to different approcahes for dealing with random experiments in the context of AI, and approaches such as certainty factors were developed.

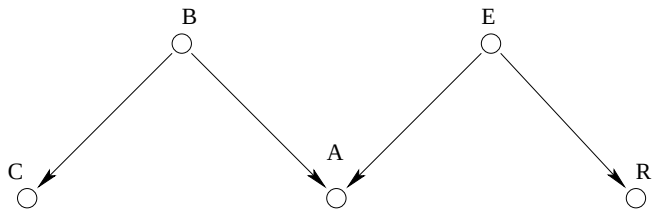
- However, it turns out that while *in theory* the use of probability theory involves a lot of probabilities and as a result is computationally expensive. . .
- . . . *in practice* there are some neat tricks to reduce the number of probabilities and the cost of the computation.
- These tricks revolve around the use of network models, and will be discussed in this lecture.
- The basic idea behind this saving can be best illustated by the following example:



- The nodes in the network represent random variables with the same name.
- For now all we need to know is that the structure of the network tells us that the probability distribution over A depends only on the probability distributions over B and E, and that the probability distribution over R depends on that over E.

- As an aside, the usual view is that the links record some form of causality between random variables.
- Thus E causes R, and B and E are causes of A.
- This particular example encodes the knowledge that Earthquakes are reported on the Radio, and that both Burglaries and Earthquakes cause burglar Alarms to sound.
- We use these causalities to build the network.
- Once we have the network, we use the links to identify which random variables are dependent on which other random variables.
- The exact nature of this dependence will be explained later.

- We can use this kind of knowledge, we can factor the expression for joint probability distributions.
- To see how this works, consider the following network:



- For this we need joint probabilities of the form:

$$\Pr(a, b, c, e, r)$$

and hence require 32 different probability values if all the random variables are binary.

- We can reduce this joint value to:

$$\Pr(a, c, r \mid e, b) \Pr(e, b)$$

by the multiplication law. Now, since r, c and a have no effect on one another (which we know from the graph) we can further reduce this to

$$\Pr(a \mid e, b) \Pr(c \mid e, b) \Pr(r \mid e, b) \Pr(e, b)$$

- Since c is only affected by b and r is only affected by e this becomes

$$\Pr(a \mid e, b) \Pr(c \mid b) \Pr(r \mid e) \Pr(e, b)$$

and since b has no effect on e we end up with:

$$\Pr(a \mid e, b) \Pr(c \mid b) \Pr(r \mid e) \Pr(e) \Pr(b)$$

- If all these variables are binary, the new model requires only 18 different values, and there is a corresponding decrease in the complexity of updating the model.
- The savings are proportionally much greater when dealing with more complex models.
- The next part of the lecture is devoted to establishing more precisely what we mean by saying one variable "has no effect" on another because of the structure of the graph.

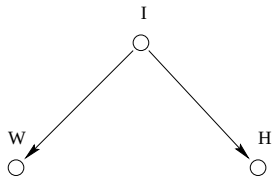
- Before we do this, a little terminology.
- A *directed graph* is a set of variables and a set of directed arcs between them.
- A directed graph is *acyclic* if it is not possible to start at a node, follow the arcs in the direction they point, and end up back at the starting node.
- We will only talk about directed acyclic graphs.
- A is the *parent* of B if there is a directed arc from A to B.
- B is the *child* of A if A is the parent of B.
- Any node with no parents is known as a *root* of the graph.
- Any node with no children is known as a *leaf* of the graph.

- The parents of node A and the parents of those parents, and the parents of those parents, and so on, are the *ancestors* of A.
- If A is an ancestor of B, then B is a *descendent* of A.
- Thus in the network a few slides back, B and E are roots, A and R are leaves.
- B is the parent of A, E is the parent of A and R.
- A is the child of B and E, R is the child of E.
- Neither A nor R has any ancestors other than B and E.
- From this and the prevous example, you might conclude that this notion of "has no effect" has something to do with parent/child relationships.
- However it is more complex than this, since, as we shall see, two parents of a common child can have an effect on each other.

- We say that there is a *link* between two nodes A and B if A is a parent of B or B is a parent of A.
- We say that two nodes A and C in a graph have a *path* between them if it is possible to start at A and follow a series of links through the graph to reach C.
- Note that in defining a path we ignore the direction of the arrows.
- A graph is said to be *singly-connected* if it includes no pairs of nodes with more than one path between them.
- A graph which is not singly-connected is *multiply-connected*.
- A singly-connected graph with one root is called a *tree*.
- A singly-connected graph with several roots is called a *polytree*.

Causal reasoning

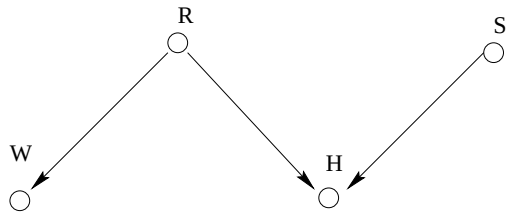
- We start by considering some examples of this notion of "has no effect".
- Consider representing the information that I (icy roads) makes it more likely that Watson will crash W and that Holmes will crash H.
- This information can be captured in the graph:



- Now, if we learn that Watson has crashed, this makes us believe it is more likely that the roads are icy.
- Thus learning something about W allows us to conclude something about I.
- This in turn makes us believe that it is more likely that Holmes has crashed.
- Thus the new information about I can be propagated to learn something about H.
- However, if we learn that the roads are not icy, then the news about Watson has no effect on our beliefs about Holmes.
- Thus knowing the value of I for certain prevents information about W affecting H.

- In other words our notion of "has an effect" depends upon what we know, and changes as we learn new things.
- When nothing is known about I, H is *dependent* on W.
- Once we know the value of I with certainty, then H is *independent* of W.
- This phenomenon is known as *conditional independence*.
- As we shall see this notion of independence is exactly statistical independence, which is why the factorisation given above works.
- Before we can formalise this idea, we need some further examples of exactly what we are formalising.

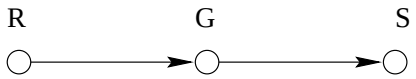
- Holmes knows that there are two causes of his grass being wet, H, either the sprinkler has been on, S, or it has been raining, R.
- If it has been raining, then Watson's grass will be wet as well, W.
- This example may be captured by the graph:



- When Holmes sees that his grass is wet, he is more inclined to believe both that it was raining and that the sprinkler was on.

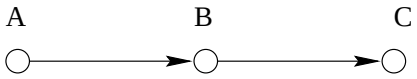
- Thus observing H makes both S and R more likely.
- However, when Holmes sees that Watson's grass is also wet, his belief that it was raining increases.
- Thus observing W provides more evidence for R.
- Because rain seems to be a very likely explanation for the wet grass, Holmes now has little belief that the sprinkler was on.
- Thus the increase in belief in R leads to a decrease in S.
- We say that the increasing probability of rain *explains away* the sprinkler.
- Knowledge about W makes S dependent on R.

- Our third example is a variation on a theme of the second.
- Consider that the grass being wet G is a cause of Holmes' shoes being wet S as he walks to his car.
- A subset of the relevant information is thus:



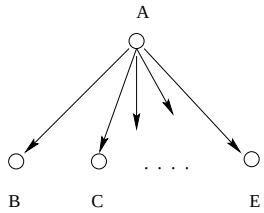
- In a similar way to previous examples, if Holmes finds his shoes are wet, then his belief in the grass being wet increases as does his belief that it was raining.
- Thus knowing S increases belief in G and R.

- However if Holmes is told by his wife that the grass is wet before he leaves the house, then realising his shoes are wet will not change his belief about the likelihood of rain.
- Thus once we know the value of G, R and S are independent.
- This gives us some standard patterns of connection and their associated dependencies.
- In a serial connection such as:



A and C are independent once B is known, and we say that A and C are *d-separated* given B.

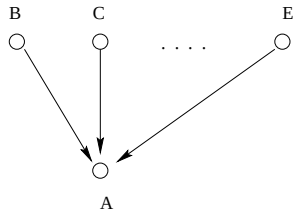
- In the following generalisation of the icy roads example:



B, C, $\ldots E$ are d-separated given A.

- We refer to this connection as *diverging*
- Thus in both the serial and diverging case, learning some thing leads to d-separation.

- The final case is the converging connection which generalises the sprinker example:



- Here the parents of A are independent *unless* something is known about A which does not come from the parents.
- In other words, if there is direct evidence about A, or evidence from the children of A, then the parents of A become dependent upon one another.

- Unlike in the other cases here the change is from independence to dependence.
- We note in passing that there are two types of evidence.
- Evidence which fixes the exact state of a variable (which value it takes) is called *hard* evidence.
- Other evidence is *soft* evidence.
- Hard evidence is required to induce conditional independence is the serial and diverging cases.
- Soft evidence is sufficient to create dependence in the converging case.
- We are now ready to formally define probabilistic networks.
- For that, however, we will need another lecture...