Decision Theory: Sequential Decisions

CPSC 322 Lecture 32

March 29, 2006
Textbook §12.3

Lecture Overview

Decision Variables

- Decision variables are like random variables that an agent gets to choose the value of.
- A possible world specifies the value for each decision variable and each random variable.
- For each assignment of values to all decision variables, the measures of the worlds satisfying that assignment sum to 1.
- The probability of a proposition is undefined unless you condition on the values of all decision variables.

Single decisions

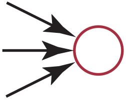
- Given a single decision variable, the agent can choose $D = d_i$ for any $d_i \in dom(D)$.
- The expected utility of decision $D = d_i$ is $\mathcal{E}(U|D = d_i)$.
- An optimal single decision is the decision $D = d_{max}$ whose expected utility is maximal:

$$d_{max} = \arg\max_{d_i \in dom(D)} \mathcal{E}(U|D = d_i).$$

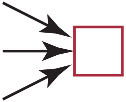
Decision Networks

- A decision network is a graphical representation of a finite sequential decision problem.
- Decision networks extend belief networks to include decision variables and utility.
- A decision network specifies what information is available when the agent has to act.
- A decision network specifies which variables the utility depends on.

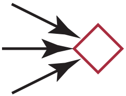
Decision Networks



- A random variable is drawn as an ellipse. Arcs into the node represent probabilistic dependence.



- A decision variable is drawn as an rectangle. Arcs into the node represent information available when the decision is made.



- A value node is drawn as a diamond. Arcs into the node represent values that the value depends on.

Lecture Overview

Sequential Decisions

- An intelligent agent doesn't make a multi-step decision and carry it out without considering revising it based on future information.
- A more typical scenario is where the agent:
  observes, acts, observes, acts, . . .
- Subsequent actions can depend on what is observed.
  - What is observed depends on previous actions.
- Often the sole reason for carrying out an action is to provide information for future actions.
  - For example: diagnostic tests, spying.

Sequential decision problems

- A sequential decision problem consists of a sequence of decision variables $D_1, \ldots, D_n$.
- Each D_i has an information set of variables pD_i, whose value will be known at the time decision D_i is made.

- What should an agent do?
  - What an agent should do at any time depends on what it will do in the future.
  - What an agent does in the future depends on what it did before.

Policies

- A policy specifies what an agent should do under each circumstance.
- A policy is a sequence $\delta_1, \ldots, \delta_n$ of decision functions

$$\delta_i : dom(pD_i) \to dom(D_i).$$

This policy means that when the agent has observed $O \in dom(pD_i)$, it will do $\delta_i(O)$.

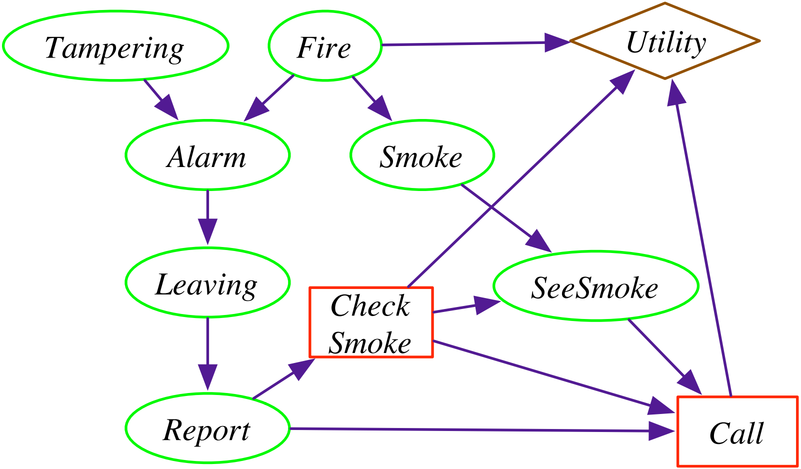
Expected Value of a Policy

- Possible world ω satisfies policy δ, written $\omega \models \delta$ if the world assigns the value to each decision node that the policy specifies.
- The expected utility of policy δ is

$$\mathcal{E}(U|\delta) = \sum_{\omega \models \delta} U(\omega) \times P(\omega),$$

- An optimal policy is one with the highest expected utility.

Decision Network for the Alarm Problem



Lecture Overview

Finding the optimal policy

- Remove all variables that are not ancestors of a value node
- Create a factor for each conditional probability table and a factor for the utility.
- Sum out variables that are not parents of a decision node.
- Select a variable D that is only in a factor f with (some of) its parents.
  - this variable will be one of the decisions that is made latest
- Eliminate D by maximizing. This returns:
  - the optimal decision function for D, $\arg\max_D f$
  - a new factor to use in VE, $\max_D f$
- Repeat till there are no more decision nodes.
- Sum out the remaining random variables. Multiply the factors: this is the expected utility of the optimal policy.

Complexity of finding the optimal policy

- If there are k binary parents, to a decision D, there are 2^k assignments of values to the parents.
- If there are b possible actions, there are b^{2^k} different decision functions.
- If there are d decisions, each with k binary parents and b possible actions, there are $\left(b^{2^k}\right)^d$ policies.
- Doing variable elimination lets us find the optimal policy after considering only $d \cdot b^{2^k}$ policies
  - The dynamic programming algorithm is much more efficient than searching through policy space.

Lecture Overview

Value of Information

- The value of information X for decision D is the utility of the the network with an arc from X to D minus the utility of the network without the arc.
  - The value of information is always non-negative.
  - It is positive only if the agent changes its action depending on X.
- The value of information provides a bound on how much you should be prepared to pay for a sensor. How much is a better weather forecast worth?

Value of Control

- The value of control of a variable X is the value of the network when you make X a decision variable minus the value of the network when X is a random variable.
- You need to be explicit about what information is available when you control X.
  - If you control X without observing, controlling X can be worse than observing X.
  - If you keep the parents the same, the value of control is always non-negative.

Lecture Overview

Agents as Processes

Agents carry out actions:

- forever infinite horizon
- until some stopping criteria is met indefinite horizon
- finite and fixed number of steps finite horizon

Decision-theoretic Planning

What should an agent do under these different planning horizons, when

- it gets rewards (and punishments) and tries to maximize its rewards received
- actions can be noisy; the outcome of an action can't be fully predicted
- there is a model that specifies the probabilistic outcome of actions
- the world is fully observable

Lecture Overview

Recap

Sequential Decisions

Finding Optimal Policies

Value of Information, Control

Decision Processes

MDPs

World State

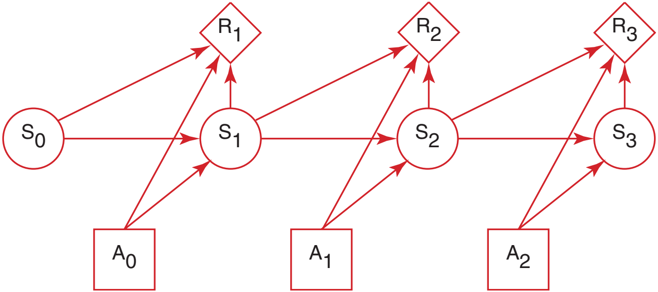
- The world state is the information such that if you knew the world state, no information about the past is relevant to the future. Markovian assumption.
- Let S_i be the state at time i

$$P(S_{t+1}|S_0, A_0, \ldots, S_t, A_t) = P(S_{t+1}|S_t, A_t)$$

  $P(s'|s, a)$ is the probability that the agent will be in state s' immediately after doing action a in state s.
- The dynamics is stationary if the distribution is the same for each time point.

Decision Processes

- A Markov decision process augments a stationary Markov chain with actions and values:

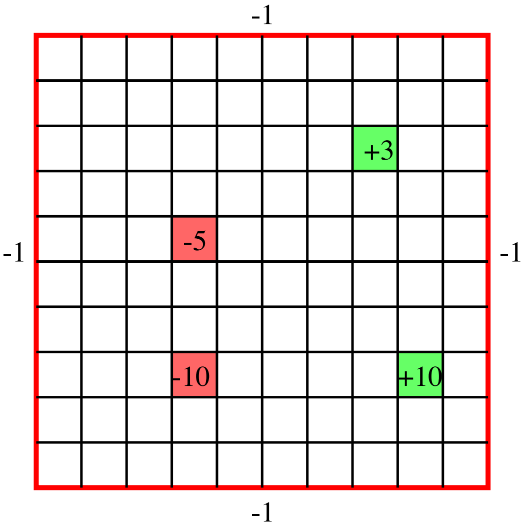


Markov Decision Processes

An MDP is defined by:

- set S of states.
- set A of actions.
- $P(S_{t+1}|S_t, A_t)$ specifies the dynamics.
- $R(S_t, A_t, S_{t+1})$ specifies the reward. The agent gets a reward at each time step (rather than just a final reward).
  - $R(s, a, s')$ is the reward received when the agent is in state s, does action a and ends up in state s'.

Example: Simple Grid World



Grid World Model

- Actions: up, down, left, right.
- 100 states corresponding to the positions of the robot.
- Robot goes in the commanded direction with probability 0.7, and one of the other directions with probability 0.1.
- If it crashes into an outside wall, it remains in its current position and has a reward of -1.
- Four special rewarding states; the agent gets the reward when leaving.

Planning Horizons

The planning horizon is how far ahead the planner looks to make a decision.

- The robot gets flung to one of the corners at random after leaving a positive ($+10$ or $+3$) reward state.
  - the process never halts
  - infinite horizon
- The robot gets $+10$ or $+3$ entering the state, then it stays there getting no reward. These are absorbing states.
  - The robot will eventually reach the absorbing state.
  - indefinite horizon

Information Availability

What information is available when the agent decides what to do?

- fully-observable MDP the agent gets to observe S_t when deciding on action A_t.
- partially-observable MDP (POMDP) the agent has some noisy sensor of the state. It needs to remember its sensing and acting history.

We'll only consider (fully-observable) MDPs.