CS 188: Artificial Intelligence

Decision Networks and Value of Information

Instructors: Dan Klein and Pieter Abbeel

University of California, Berkeley
Decision Networks
Decision Networks
**Decision Networks**

- **MEU**: choose the action which maximizes the expected utility given the evidence

- Can directly operationalize this with decision networks
  - Bayes nets with nodes for utility and actions
  - Lets us calculate the expected utility for each action

- New node types:
  - Chance nodes (just like BNs)
  - Actions (rectangles, cannot have parents, act as observed evidence)
  - Utility node (diamond, depends on action and chance nodes)
Decision Networks

- Action selection
  - Instantiate all evidence
  - Set action node(s) each possible way
  - Calculate posterior for all parents of utility node, given the evidence
  - Calculate expected utility for each action
  - Choose maximizing action
Decision Networks

Umbrella = leave

\[ EU(\text{leave}) = \sum_w P(w)U(\text{leave}, w) \]
\[ = 0.7 \cdot 100 + 0.3 \cdot 0 = 70 \]

Umbrella = take

\[ EU(\text{take}) = \sum_w P(w)U(\text{take}, w) \]
\[ = 0.7 \cdot 20 + 0.3 \cdot 70 = 35 \]

Optimal decision = leave

\[ \text{MEU}(\phi) = \max_a EU(a) = 70 \]
Decisions as Outcome Trees

- Almost exactly like expectimax / MDPs
- What’s changed?

- Weather
- Umbrella

- U(t,s)
- U(t,r)
- U(l,s)
- U(l,r)
Example: Decision Networks

Umbrella = leave

\[ EU(\text{leave}|\text{bad}) = \sum_w P(w|\text{bad})U(\text{leave}, w) \]
\[ = 0.34 \cdot 100 + 0.66 \cdot 0 = 34 \]

Umbrella = take

\[ EU(\text{take}|\text{bad}) = \sum_w P(w|\text{bad})U(\text{take}, w) \]
\[ = 0.34 \cdot 20 + 0.66 \cdot 70 = 53 \]

Optimal decision = take

\[ \text{MEU}(F = \text{bad}) = \max_a EU(a|\text{bad}) = 53 \]
Decisions as Outcome Trees

Weather Forecast = bad

Umbrella

Decisions as Outcome Trees

U(t,s) W | {b} U(t,r) W | {b}

W | {b}
take

U(t,s) sun

U(t,r) rain

W | {b}
leave

U(l,s) sun

U(l,r) rain

Forecast = bad

Weather

Umbrella
Ghostbusters Decision Network

Demo: Ghostbusters with probability
Video of Demo Ghostbusters with Probability
Idea: compute value of acquiring evidence
- Can be done directly from decision network

Example: buying oil drilling rights
- Two blocks A and B, exactly one has oil, worth k
- You can drill in one location
- Prior probabilities 0.5 each, & mutually exclusive
- Drilling in either A or B has EU = k/2, MEU = k/2

Question: what’s the value of information of O?
- Value of knowing which of A or B has oil
- Value is expected gain in MEU from new info
- Survey may say “oil in a” or “oil in b”, prob 0.5 each
- If we know OilLoc, MEU is k (either way)
- Gain in MEU from knowing OilLoc?
- VPI(OilLoc) = k/2
- Fair price of information: k/2
VPI Example: Weather

MEU with no evidence

\[
\text{MEU}(\varnothing) = \max_a \text{EU}(a) = 70
\]

MEU if forecast is bad

\[
\text{MEU}(F = \text{bad}) = \max_a \text{EU}(a|\text{bad}) = 53
\]

MEU if forecast is good

\[
\text{MEU}(F = \text{good}) = \max_a \text{EU}(a|\text{good}) = 95
\]

Forecast distribution

<table>
<thead>
<tr>
<th>F</th>
<th>P(F)</th>
<th>0.59</th>
<th>0.41</th>
</tr>
</thead>
<tbody>
<tr>
<td>good</td>
<td>0.59</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>0.41</td>
<td>95</td>
<td>53</td>
</tr>
</tbody>
</table>

\[
VPI(E'|e) = \sum_{e'} P(e'|e) \text{MEU}(e,e') - \text{MEU}(e)
\]
Assume we have evidence \( E=e \). Value if we act now:

\[
\text{MEU}(e) = \max_a \sum_s P(s|e) U(s, a)
\]

Assume we see that \( E' = e' \). Value if we act then:

\[
\text{MEU}(e, e') = \max_a \sum_s P(s|e, e') U(s, a)
\]

BUT \( E' \) is a random variable whose value is unknown, so we don’t know what \( e' \) will be.

Expected value if \( E' \) is revealed and then we act:

\[
\text{MEU}(e, E') = \sum_{e'} P(e'|e) \text{MEU}(e, e')
\]

Value of information: how much MEU goes up by revealing \( E' \) first then acting, over acting now:

\[
VPI(E'|e) = \text{MEU}(e, E') - \text{MEU}(e)
\]
VPI Properties

- **Nonnegative**
  \[ \forall E', e : \text{VPI}(E'|e) \geq 0 \]

- **Nonadditive**
  (think of observing \( E_j \) twice)
  \[ \text{VPI}(E_j, E_k|e) \neq \text{VPI}(E_j|e) + \text{VPI}(E_k|e) \]

- **Order-independent**
  \[ \text{VPI}(E_j, E_k|e) = \text{VPI}(E_j|e) + \text{VPI}(E_k|e, E_j) \]
  \[ = \text{VPI}(E_k|e) + \text{VPI}(E_j|e, E_k) \]
Quick VPI Questions

- The soup of the day is either clam chowder or split pea, but you wouldn’t order either one. What’s the value of knowing which it is?

- There are two kinds of plastic forks at a picnic. One kind is slightly sturdier. What’s the value of knowing which?

- You’re playing the lottery. The prize will be $0 or $100. You can play any number between 1 and 100 (chance of winning is 1%). What is the value of knowing the winning number?
Value of Imperfect Information?

- No such thing (as we formulate it)
- Information corresponds to the observation of a node in the decision network
- If data is “noisy” that just means we don’t observe the original variable, but another variable which is a noisy version of the original one
VPI Question

- VPI(OilLoc) ?
- VPI(ScoutingReport) ?
- VPI(Scout) ?
- VPI(Scout | ScoutingReport) ?

Generally:
If \( \text{Parents}(U) \perp \!\!\!\!\!\!\perp Z \mid \text{CurrentEvidence} \) 
Then \( \text{VPI}(Z \mid \text{CurrentEvidence}) = 0 \)
POMDPs

- MDPs have:
  - States S
  - Actions A
  - Transition function $P(s'|s,a)$ (or $T(s,a,s')$)
  - Rewards $R(s,a,s')$

- POMDPs add:
  - Observations O
  - Observation function $P(o|s)$ (or $O(s,o)$)

- POMDPs are MDPs over belief states $b$ (distributions over $S$)

- We’ll be able to say more in a few lectures
Example: Ghostbusters

- In (static) Ghostbusters:
  - Belief state determined by evidence to date \{e\}
  - Tree really over evidence sets
  - Probabilistic reasoning needed to predict new evidence given past evidence

- Solving POMDPs
  - One way: use truncated expectimax to compute approximate value of actions
  - What if you only considered busting or one sense followed by a bust?
  - You get a VPI-based agent!
Video of Demo Ghostbusters with VPI
General solutions map belief functions to actions
- Can divide regions of belief space (set of belief functions) into policy regions (gets complex quickly)
- Can build approximate policies using discretization methods
- Can factor belief functions in various ways

Overall, POMDPs are very (actually PSPACE-) hard

Most real problems are POMDPs, and we can rarely solve them in their full generality
Next Time: Dynamic Models