Course Overview

Environment

Deterministic

Stochastic

Problem Type

Constraint Satisfaction

Logic

Static

Sequential

Planning

Variables + Constraints

Search

Arc Consistency

Search

Bayesian Networks

Variable Elimination

Decision Networks

Variable Elimination

Markov Processes

Value Iteration

Course Module

Representation

Reasoning

Technique

Decision theory: acting under uncertainty

Uncertainty

Decision Theory

STRIPS

Search

As CSP (using arc consistency)
Recap: Utility and Expected Utility

- Single-Stage Decision Problems
  - Single-Stage decision networks
  - Variable elimination (VE) for computing the optimal decision

- Sequential Decision Problems
  - General decision networks
  - Time-permitting: Policies
  - Next lecture: variable elimination for finding the optimal policy in general decision networks
Utility

- **Utility**: a measure of desirability of possible worlds to an agent
  - Let $U$ be a real-valued function such that $U(w)$ represents an agent's degree of preference for world $w$

- Simple goals can still be specified: e.g.
  - Worlds that satisfy the goal have utility 100
  - Other worlds have utility 0

- Utilities can be more complicated
  - For example, in the robot delivery domains, they could involve
    - Amount of damage
    - Reached the target room?
    - Energy left
    - Time taken
Delivery Robot Example

- Decision variable 1: the robot can choose to wear pads
  - Yes: protection against accidents, but extra weight
  - No: fast, but no protection
- Decision variable 2: the robot can choose the way
  - Short way: quick, but higher chance of accident
  - Long way: safe, but slow
- Random variable: is there an accident?
  - Agent decides
  - Chance decides

Utility

<table>
<thead>
<tr>
<th></th>
<th>short way</th>
<th>accident</th>
<th>no accident</th>
<th>Utility</th>
</tr>
</thead>
<tbody>
<tr>
<td>wear pads</td>
<td>w0</td>
<td>w1</td>
<td></td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>w2</td>
<td>w3</td>
<td></td>
<td>30</td>
</tr>
<tr>
<td>don't wear pads</td>
<td>w4</td>
<td>w5</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>w6</td>
<td>w7</td>
<td></td>
<td>100</td>
</tr>
</tbody>
</table>

Agent decides: 5
Chance decides: 95
35
95
30
75
3
100
0
80
Possible worlds and decision variables

- A **possible world** specifies a value for each random variable and each decision variable.
- For each assignment of values to all decision variables, the probabilities of the worlds satisfying that assignment sum to 1.

![Diagram showing possible worlds and decision variables with conditional probabilities and utilities.]

<table>
<thead>
<tr>
<th>Conditional probability</th>
<th>Wear pads</th>
<th>Long way</th>
<th>Accident</th>
<th>Utility</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>35</td>
<td>95</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.8</td>
<td>30</td>
<td>75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.01</td>
<td>3</td>
<td>80</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.99</td>
<td>80</td>
<td>80</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.2</td>
<td>35</td>
<td>95</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.8</td>
<td>30</td>
<td>75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.01</td>
<td>3</td>
<td>80</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.99</td>
<td>80</td>
<td>80</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Expected utility of a decision

- The expected utility of a decision is:

\[ E[U|D = d] = \sum_w P(w|D = d)U(w) \]
Lecture Overview

- Recap: Utility and Expected Utility

Single-Stage Decision Problems
  - Single-Stage decision networks
  - Variable elimination (VE) for computing the optimal decision

Sequential Decision Problems
  - General decision networks
  - Time-permitting: Policies
  - Next lecture: variable elimination for finding the optimal policy in general decision networks
Single Action vs. Sequence of Actions

- **Single Action (aka One-Off Decisions)**
  - One or more primitive decisions that can be treated as a single macro decision to be made before acting
  - E.g., “WearPads” and “WhichWay” can be combined into macro decision (WearPads, WhichWay) with domain \{yes, no\} \times \{long, short\}

- **Sequence of Actions (Sequential Decisions)**
  - Repeat:
    - make observations
    - decide on an action
    - carry out the action
  - Agent has to take actions not knowing what the future brings
    - This is fundamentally different from everything we’ve seen so far
    - Planning was sequential, but we still could still think first and then act
Optimal single-stage decision

- Given a single (macro) decision variable $D$
  - the agent can choose $D = d_i$ for any value $d_i \in \text{dom}(D)$

**Definition (optimal single-stage decision)**

An optimal single-stage decision is the decision $D = d_{\text{max}}$ whose expected value is maximal:

$$d_{\text{max}} \in \arg\max_{d_i \in \text{dom}(D)} E[U | D = d_i]$$
What is the optimal decision in the example?

**Definition (optimal single-stage decision)**

An optimal single-stage decision is the decision $D=d_{\text{max}}$ whose expected value is maximal:

$$d_{\text{max}} \in \arg\max_{d_i \in \text{dom}(D)} E[U|D=d_i]$$

(Wear pads, short way)
(Wear pads, long way)
(No pads, short way)
(No pads, long way)

| Decision | Conditional Probability | Utility | $E[U|D]$ |
|----------|-------------------------|---------|----------|
| Wear pads, short way | 0.2 | 35 | 83 |
| Wear pads, long way | 0.8 | 95 | |
| No pads, short way | 0.01 | 30 | 74.55 |
| No pads, long way | 0.99 | 75 | |
| Wear pads, short way | 0.2 | 3 | 80.6 |
| Wear pads, long way | 0.8 | 100 | |
| No pads, short way | 0.01 | 0 | 79.2 |
| No pads, long way | 0.99 | 80 | |
Definition (optimal single-stage decision)

An **optimal single-stage decision** is the decision $D=d_{\text{max}}$ whose expected value is maximal:

$$d_{\text{max}} \in \arg\max_{d_i \in \text{dom}(D)} E[U|D=d_i]$$

Best decision: (wear pads, short way)
Single-Stage decision networks

- Extend belief networks with:
  - **Decision nodes**, that the agent chooses the value for
    - Parents: only other decision nodes allowed
    - Domain is the set of possible actions
    - Drawn as a rectangle
  - Exactly one utility node
    - Parents: all random & decision variables on which the utility depends
    - Does not have a domain
    - Drawn as a diamond

- Explicitly shows dependencies
  - E.g., which variables affect the probability of an accident?
Types of nodes in decision networks

• A random variable is drawn as an ellipse.
  – Arcs into the node represent probabilistic dependence
  – As in Bayesian networks: a random variable is conditionally independent of its non-descendants given its parents

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

• A utility node is drawn as a diamond.
  – Arcs into the node represent variables that the utility depends on.
  – Specifies a utility for each instantiation of its parents
Example Decision Network

- Decision nodes do not have an associated table.
- The utility node does not have a domain.
Computing the optimal decision: we can use VE

• Denote
  – the random variables as $X_1, \ldots, X_n$
  – the decision variables as $D$
  – the parents of node $N$ as $\text{pa}(N)$

\[
E(U) = \sum_{X_1, \ldots, X_n} P(X_1, \ldots, X_n \mid D) U(\text{pa}(U))
\]

\[
= \sum_{X_1, \ldots, X_n} \prod_{i=1}^{n} P(X_i \mid \text{pa}(X_i)) U(\text{pa}(U))
\]

• To find the optimal decision we can use VE:
  1. Create a factor for each conditional probability and for the utility
  2. Sum out all random variables, one at a time
     • This creates a factor on $D$ that gives the expected utility for each $d_i$
  3. Choose the $d_i$ with the maximum value in the factor
VE Example: Step 1, create initial factors

### Abbreviations:
- **W** = Which Way
- **P** = Wear Pads
- **A** = Accident

#### $f_1(A, W)$

| Which Way W | Accident A | $P(A|W)$ |
|-------------|------------|----------|
| long        | true       | 0.01     |
| long        | false      | 0.99     |
| short       | true       | 0.2      |
| short       | false      | 0.8      |

#### $f_2(A, W, P)$

<table>
<thead>
<tr>
<th>Which Way W</th>
<th>Accident A</th>
<th>Pads P</th>
<th>Utility</th>
</tr>
</thead>
<tbody>
<tr>
<td>long</td>
<td>true</td>
<td>true</td>
<td>30</td>
</tr>
<tr>
<td>long</td>
<td>true</td>
<td>false</td>
<td>0</td>
</tr>
<tr>
<td>long</td>
<td>false</td>
<td>true</td>
<td>75</td>
</tr>
<tr>
<td>long</td>
<td>false</td>
<td>false</td>
<td>80</td>
</tr>
<tr>
<td>short</td>
<td>true</td>
<td>true</td>
<td>35</td>
</tr>
<tr>
<td>short</td>
<td>true</td>
<td>false</td>
<td>3</td>
</tr>
<tr>
<td>short</td>
<td>false</td>
<td>true</td>
<td>95</td>
</tr>
<tr>
<td>short</td>
<td>false</td>
<td>false</td>
<td>100</td>
</tr>
</tbody>
</table>

\[
E(U) = \sum_A P(A|W)U(A, W, P) = \sum_A f_1(A, W) f_2(A, W, P)
\]
VE example: step 2, sum out A

Step 2a: compute product $f_1(A,W) \times f_2(A,W,P)$

What is the right form for the product $f_1(A,W) \times f_2(A,W,P)$?

- $f(A,W)$
- $f(A,P)$
- $f(A)$
- $f(A,P,W)$
VE example: step 2, sum out A

Step 2a: compute product $f(A,W,P) = f_1(A,W) \times f_2(A,W,P)$

What is the right form for the product $f_1(A,W) \times f_2(A,W,P)$?

• It is $f(A,P,W)$: the domain of the product is the union of the multiplicands’ domains
  - $f(A,P,W) = f_1(A,W) \times f_2(A,W,P)$
  - I.e., $f(A=a,P=p,W=w) = f_1(A=a,W=w) \times f_2(A=a,W=w,P=p)$
VE example: step 2, sum out A

Step 2a: compute product

\[ f(A,W,P) = f_1(A,W) \times f_2(A,W,P) \]

\[ f(A=a,P=p,W=w) = f_1(A=a,W=w) \times f_2(A=a,W=w,P=p) \]

<table>
<thead>
<tr>
<th>Which way W</th>
<th>Accident A</th>
<th>( f_1(A,W) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>long</td>
<td>true</td>
<td>0.01</td>
</tr>
<tr>
<td>long</td>
<td>false</td>
<td>0.99</td>
</tr>
<tr>
<td>short</td>
<td>true</td>
<td>0.2</td>
</tr>
<tr>
<td>short</td>
<td>false</td>
<td>0.8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Which way W</th>
<th>Accident A</th>
<th>Pads P</th>
<th>( f_2(A,W,P) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>long</td>
<td>true</td>
<td>true</td>
<td>30</td>
</tr>
<tr>
<td>long</td>
<td>true</td>
<td>false</td>
<td>0</td>
</tr>
<tr>
<td>long</td>
<td>false</td>
<td>true</td>
<td>75</td>
</tr>
<tr>
<td>long</td>
<td>false</td>
<td>false</td>
<td>80</td>
</tr>
<tr>
<td>short</td>
<td>true</td>
<td>true</td>
<td>35</td>
</tr>
<tr>
<td>short</td>
<td>true</td>
<td>false</td>
<td>3</td>
</tr>
<tr>
<td>short</td>
<td>false</td>
<td>true</td>
<td>95</td>
</tr>
<tr>
<td>short</td>
<td>false</td>
<td>false</td>
<td>100</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Which way W</th>
<th>Accident A</th>
<th>Pads P</th>
<th>( f(A,W,P) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>long</td>
<td>true</td>
<td>true</td>
<td>0.01 * 30</td>
</tr>
<tr>
<td>long</td>
<td>true</td>
<td>false</td>
<td>0.99 * 30</td>
</tr>
<tr>
<td>long</td>
<td>false</td>
<td>true</td>
<td>0.2 * 30</td>
</tr>
<tr>
<td>long</td>
<td>false</td>
<td>false</td>
<td>0.8 * 30</td>
</tr>
<tr>
<td>short</td>
<td>true</td>
<td>true</td>
<td>0.99 * 80</td>
</tr>
<tr>
<td>short</td>
<td>true</td>
<td>false</td>
<td>0.01 * 80</td>
</tr>
<tr>
<td>short</td>
<td>false</td>
<td>true</td>
<td>0.99 * 80</td>
</tr>
<tr>
<td>short</td>
<td>false</td>
<td>false</td>
<td>0.8 * 30</td>
</tr>
</tbody>
</table>
VE example: step 2, sum out A

Step 2a: compute product
\[ f(A,W,P) = f_1(A,W) \times f_2(A,W,P) \]

\[ f(A=a,P=p,W=w) = f_1(A=a,W=w) \times f_2(A=a,W=w,P=p) \]

<table>
<thead>
<tr>
<th>Which way W</th>
<th>Accident A</th>
<th>( f_1(A,W) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>long</td>
<td>true</td>
<td>0.01</td>
</tr>
<tr>
<td>long</td>
<td>false</td>
<td>0.99</td>
</tr>
<tr>
<td>short</td>
<td>true</td>
<td>0.2</td>
</tr>
<tr>
<td>short</td>
<td>false</td>
<td>0.8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Which way W</th>
<th>Accident A</th>
<th>Pads P</th>
<th>( f_2(A,W,P) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>long</td>
<td>true</td>
<td>true</td>
<td>30</td>
</tr>
<tr>
<td>long</td>
<td>true</td>
<td>false</td>
<td>0</td>
</tr>
<tr>
<td>long</td>
<td>false</td>
<td>true</td>
<td>75</td>
</tr>
<tr>
<td>long</td>
<td>false</td>
<td>false</td>
<td>80</td>
</tr>
<tr>
<td>short</td>
<td>true</td>
<td>true</td>
<td>35</td>
</tr>
<tr>
<td>short</td>
<td>true</td>
<td>false</td>
<td>3</td>
</tr>
<tr>
<td>short</td>
<td>false</td>
<td>true</td>
<td>95</td>
</tr>
<tr>
<td>short</td>
<td>false</td>
<td>false</td>
<td>100</td>
</tr>
</tbody>
</table>
VE example: step 2, sum out A

Step 2b: sum A out of the product $f(A,W,P)$:

$$f_3(W,P) = \sum_A f(A,W,P)$$

<table>
<thead>
<tr>
<th>Which way W</th>
<th>Pads P</th>
<th>$f_3(W,P)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>long</td>
<td>true</td>
<td>$0.01 \times 30 + 0.99 \times 75 = 74.55$</td>
</tr>
<tr>
<td>long</td>
<td>false</td>
<td>??</td>
</tr>
<tr>
<td>short</td>
<td>true</td>
<td>$0.2 \times 35 + 0.8 \times 95$</td>
</tr>
<tr>
<td>short</td>
<td>false</td>
<td>$0.99 \times 80 + 0.8 \times 95$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Which way W</th>
<th>Accident A</th>
<th>Pads P</th>
<th>f(A,W,P)</th>
</tr>
</thead>
<tbody>
<tr>
<td>long</td>
<td>true</td>
<td>true</td>
<td>0.01*30</td>
</tr>
<tr>
<td>long</td>
<td>true</td>
<td>false</td>
<td>0.01*0</td>
</tr>
<tr>
<td>long</td>
<td>false</td>
<td>true</td>
<td>0.99*75</td>
</tr>
<tr>
<td>long</td>
<td>false</td>
<td>false</td>
<td>0.99*80</td>
</tr>
<tr>
<td>short</td>
<td>true</td>
<td>true</td>
<td>0.2*35</td>
</tr>
<tr>
<td>short</td>
<td>true</td>
<td>false</td>
<td>0.2*3</td>
</tr>
<tr>
<td>short</td>
<td>false</td>
<td>true</td>
<td>0.8*95</td>
</tr>
<tr>
<td>short</td>
<td>false</td>
<td>false</td>
<td>0.8*100</td>
</tr>
</tbody>
</table>
VE example: step 2, sum out A

Step 2b: sum A out of the product $f(A, W, P)$:

$$f_3 (W, P) = \sum_A f(A, W, P)$$

<table>
<thead>
<tr>
<th>Which way W</th>
<th>Pads P</th>
<th>$f_3(W, P)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>long</td>
<td>true</td>
<td>0.01<em>30+0.99</em>75=74.55</td>
</tr>
<tr>
<td>long</td>
<td>false</td>
<td>0.01<em>0+0.99</em>80=79.2</td>
</tr>
<tr>
<td>short</td>
<td>true</td>
<td>0.2<em>35+0.8</em>95=83</td>
</tr>
<tr>
<td>short</td>
<td>false</td>
<td>0.2<em>3+0.8</em>100=80.6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Which way W</th>
<th>Accident A</th>
<th>Pads P</th>
<th>f(A,W,P)</th>
</tr>
</thead>
<tbody>
<tr>
<td>long</td>
<td>true</td>
<td>true</td>
<td>0.01 * 30</td>
</tr>
<tr>
<td>long</td>
<td>true</td>
<td>false</td>
<td>0.01*0</td>
</tr>
<tr>
<td>long</td>
<td>false</td>
<td>true</td>
<td>0.99*75</td>
</tr>
<tr>
<td>long</td>
<td>false</td>
<td>false</td>
<td>0.99*80</td>
</tr>
<tr>
<td>short</td>
<td>true</td>
<td>true</td>
<td>0.2*35</td>
</tr>
<tr>
<td>short</td>
<td>true</td>
<td>false</td>
<td>0.2*3</td>
</tr>
<tr>
<td>short</td>
<td>false</td>
<td>true</td>
<td>0.8*95</td>
</tr>
<tr>
<td>short</td>
<td>false</td>
<td>false</td>
<td>0.8*100</td>
</tr>
</tbody>
</table>
VE example: step 3, choose decision with max E(U)

The final factor encodes the expected utility of each decision

- Thus, taking the short way but wearing pads is the best choice, with an expected utility of 83
Variable Elimination for Single-Stage Decision Networks: Summary

1. Create a factor for each conditional probability and for the utility
2. Sum out all random variables, one at a time
   - This creates a factor on $D$ that gives the expected utility for each $d_i$
3. Choose the $d_i$ with the maximum value in the factor
Lecture Overview

• Recap: Utility and Expected Utility

• Single-Stage Decision Problems
  – Single-Stage decision networks
  – Variable elimination (VE) for computing the optimal decision

Sequential Decision Problems
  – General decision networks and Policies
  – Next lecture: variable elimination for finding the optimal policy in general decision networks
Sequential Decision Problems

• An intelligent agent doesn't make a multi-step decision and carry it out blindly
  – It would take new observations it makes into account

• A more typical scenario:
  – The agent observes, acts, observes, acts, …

• Subsequent actions can depend on what is observed
  – What is observed often 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

• General Decision networks:
  – Just like single-stage decision networks, with one exception: the parents of decision nodes can include random variables
Sequential Decision Problems: Example

- Example for sequential decision problem
  - Treatment depends on Test Result (and others)

- Each decision $D_i$ has an information set of variables $pa(D_i)$, whose value will be known at the time decision $D_i$ is made
  - $pa(\text{Test}) = \{\text{Symptoms}\}$
  - $pa(\text{Treatment}) = \{\text{Test, Symptoms, TestResult}\}$
Another example for sequential decision problems
 – Call depends on Report and SeeSmoke (and on CheckSmoke)
Sequential Decision Problems

• What should an agent do?
  – What an agent should do depends on what it will do in the future
    • E.g. agent only needs to check for smoke if that will affect whether it calls
  – What an agent does in the future depends on what it did before
    • E.g. when making the decision it needs to whether it checked for smoke
  – We will get around this problem as follows
    • The agent has a conditional plan of what it will do in the future
    • We will formalize this conditional plan as a policy
Policies for Sequential Decision Problems

Definition (Policy)
A policy is a sequence of $\delta_1, \ldots, \delta_n$ decision functions

$$\delta_i : \text{dom}(\rho a(D_i)) \rightarrow \text{dom}(D_i)$$

This policy means that when the agent has observed $o \in \text{dom}(\rho D_i)$, it will do $\delta_i(o)$

There are $2^2=4$ possible decision functions $\delta_{cs}$ for Check Smoke:

- Decision function needs to specify a value for each instantiation of parents

<table>
<thead>
<tr>
<th>Report</th>
<th>$\delta_{cs}$1</th>
<th>$\delta_{cs}$2</th>
<th>$\delta_{cs}$3</th>
<th>$\delta_{cs}$4</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
</tbody>
</table>

Call
Policies for Sequential Decision Problems

**Definition (Policy)**
A policy is a sequence of $\delta_1, \ldots, \delta_n$ decision functions

$$\delta_i : \text{dom}(\text{pa}(D_i)) \rightarrow \text{dom}(D_i)$$

There are $2^8=256$ possible decision functions $\delta_{cs}$ for Call:

- Decision function needs to specify a value for each instantiation of parents

---

**Diagram:***

- **Nodes:** Tampering, Fire, Alarm, Smoke, Leaving, Check Smoke, See Smoke, Report, Utility, Call

**Table:**

<table>
<thead>
<tr>
<th>Report</th>
<th>CheckS</th>
<th>SeeS</th>
<th>$\delta_{\text{call}1}$</th>
<th>$\delta_{\text{call}n}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>false</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>
How many policies are there?

- If a decision D has k binary parents, how many assignments of values to the parents are there?

\[ 2k \quad 2+k \quad k^2 \quad 2^k \]
How many policies are there?

• If a decision D has k binary parents, how many assignments of values to the parents are there?
  – $2^k$

• If there are b possible value for a decision variable, how many different decision functions are there for it if it has k binary parents?

\[ 2^{kp} \quad b \times 2^k \quad b^{2k} \quad 2^{kb} \]
Learning Goals For Today’s Class

• Compare and contrast stochastic single-stage (one-off) decisions vs. multistage decisions
• Define a Utility Function on possible worlds
• Define and compute optimal one-off decisions
• Represent one-off decisions as single stage decision networks
• Compute optimal decisions by Variable Elimination

Next time:
  – Variable Elimination for finding optimal policies
Announcements

• Assignment 4 is due on Monday
• Final exam is on Monday, April 11
  – The list of short questions is online … please use it!
• Office hours next week
  – Simona: Tuesday, 10-12 (no office hours on Monday!)
  – Mike: Wednesday 1-2pm, Friday 10-12am
  – Vasanth: Thursday, 3-5pm
  – Frank:
    • X530: Tue 5-6pm, Thu 11-12am
    • DMP 110: 1 hour after each lecture

• Optional Rainbow Robot tournament: Friday, April 8
  – Hopefully in normal classroom (DMP 110)
  – Vasanth will run the tournament,
    I’ll do office hours in the same room (this is 3 days before the final)