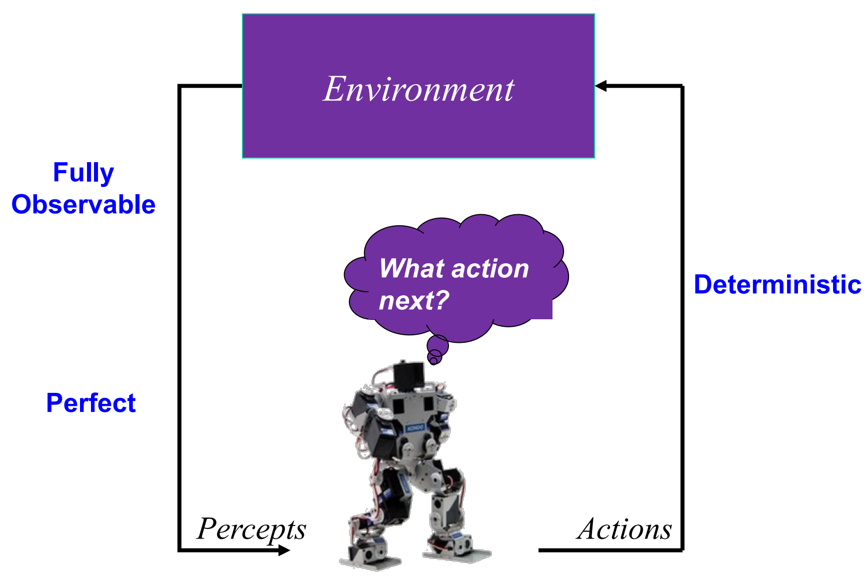
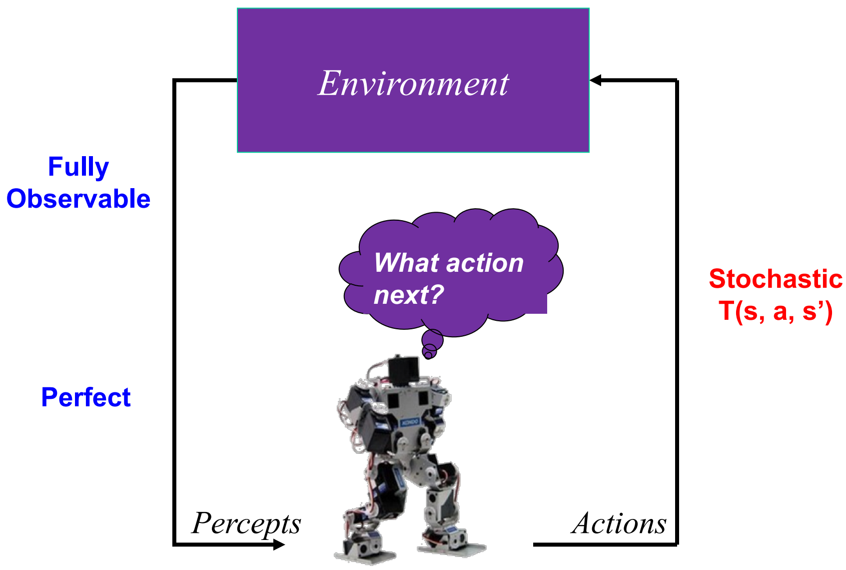
CSE-573 Artificial Intelligence

Partially-Observable MDPS (POMDPs)

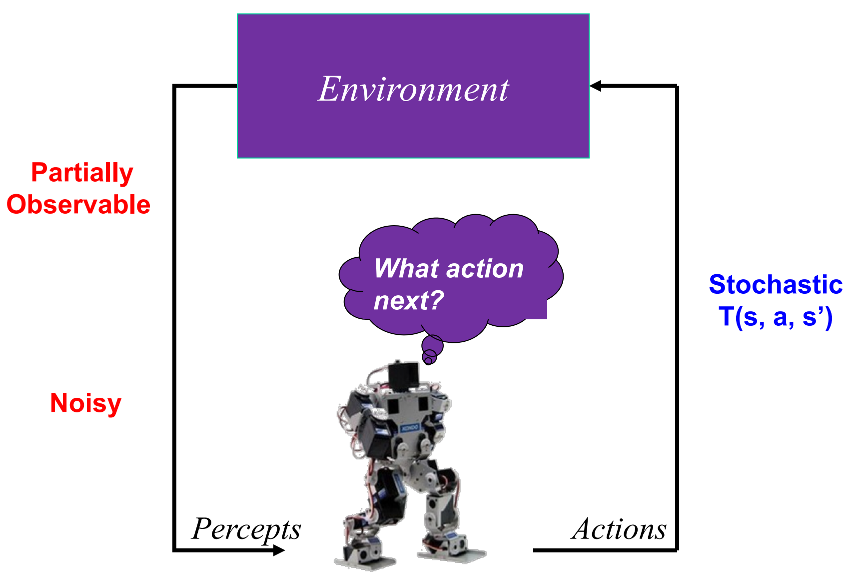
Classical



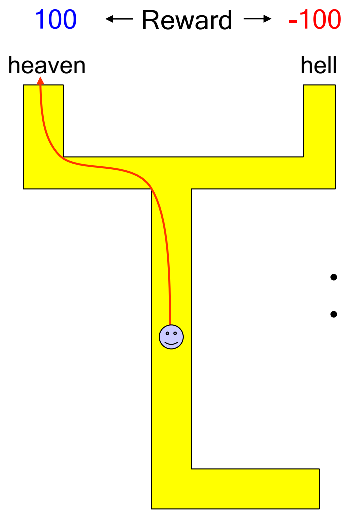
Stochastic (MDP)



Partially-Observable Stochastic (POMDP)



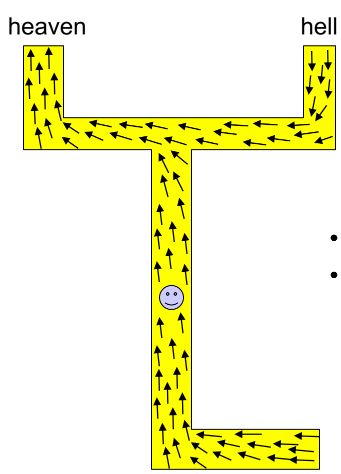
Classical Planning



- Sequential Plan
- World deterministic
- State observable

▪5

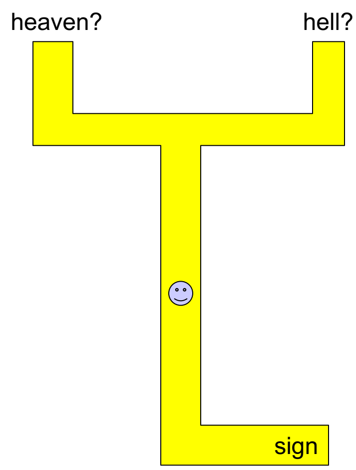
MDP-Style Planning



- Policy
- World stochastic
- State observable

▪6

Stochastic, *Partially* Observable



▪7

Markov Decision Process (MDP)

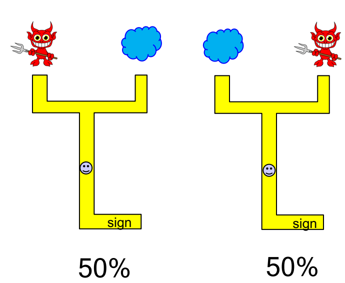
- **S**: set of states
- **A**: set of actions (a also sometimes denoted u
- **P**r(s'|s,a): transition model
- **R**(s,a,s'): reward model
- γ: discount factor
- s_0: start state

Partially-Observable MDP

- **S**: set of states
- **A**: set of actions (a or u)
- **P**r(s’|s,a): transition model
- **R**(s,a,s’): reward model
- γ: discount factor
- s_0: start state
- **Z** set of possible observations
- **P**r(z|s)

Belief State

- State of agent’s mind
- Not just of world



Probs Σ = 1

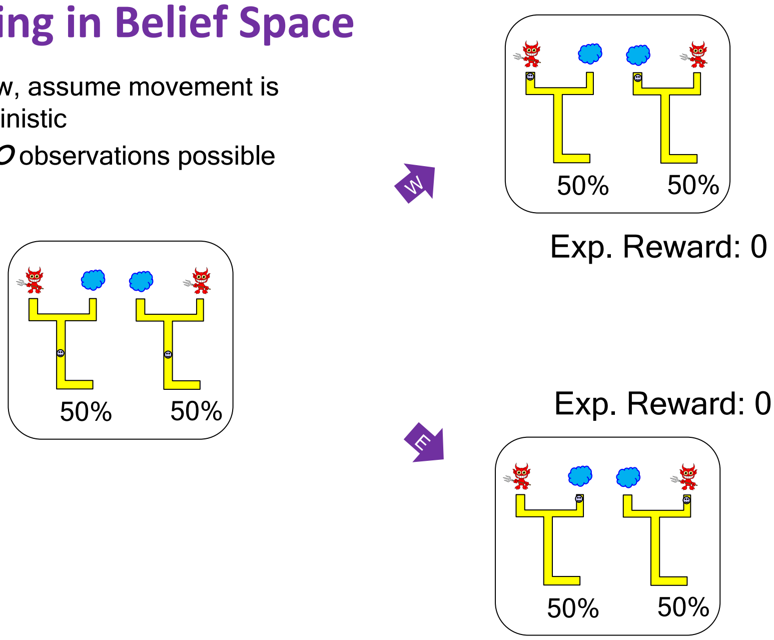
Note: POMDP

▪10

Planning in Belief Space

For now, assume movement is deterministic

And ***NO*** observations possible



Partially-Observable MDP

- **S**: set of states
- **A**: set of actions
- **Pr**($s'|s,a$): transition model
- **R**(s,a,s'): reward model
- γ: discount factor
- s_0: start state
- **Z** set of possible observations (aka evidence, measurements)
- **Pr**($z|s$)

Evidence Model

e/w = location of devil
b/m/ul/ur = location of agent

- **S** = $\{s_{wb}, s_{eb}, s_{wm}, s_{em}\ s_{wul}, s_{eul}\ s_{wur}, s_{eur}\}$
- **Z** = {heat, noheat}
- **Pr(z|s):**

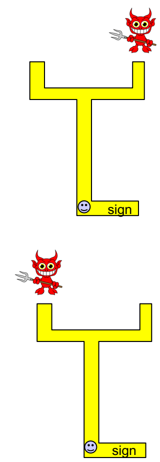
  $Pr(heat \mid s_{eb}) = 1.0$

  $Pr(heat \mid s_{wb}) = 0.2$

  $Pr(heat \mid s_{other}) = 0.0$

s_{eb}

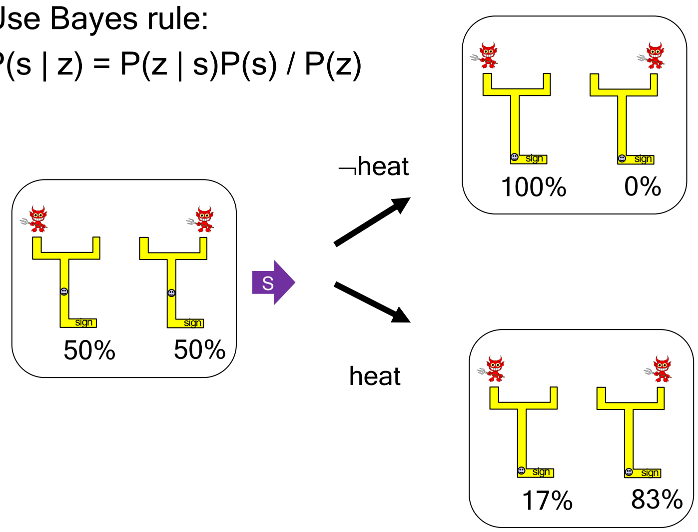
s_{wb}



Updating beliefs given evidence

$Pr(heat \mid s_{eb}) = 1.0$
$Pr(heat \mid s_{wb}) = 0.2$

Use Bayes rule:
$P(s \mid z) = P(z \mid s)P(s) / P(z)$



Objective of a Fully Observable MDP

- Find a policy
  π: **S → A**

- which maximizes expected discounted reward
  - given an infinite horizon
  - assuming full observability

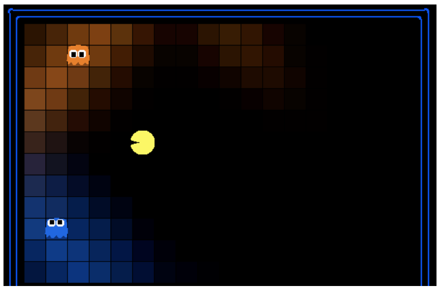
Objective of a POMDP

- Find a policy
  π: BeliefStates(**S**) → **A**
  A belief state is a ***probability distribution*** over states

- which maximizes expected discounted reward
  - given an infinite horizon
  - assuming ***partial*** & ***noisy*** observability

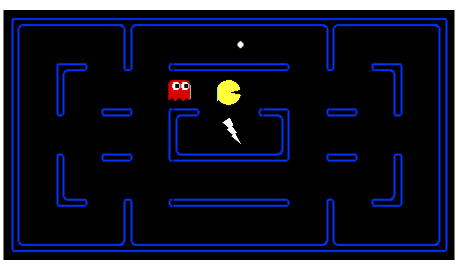
Planning in last HW

- Maximum a posteriori (MAP) Estimate
- Now “know” state
- Solve MDP

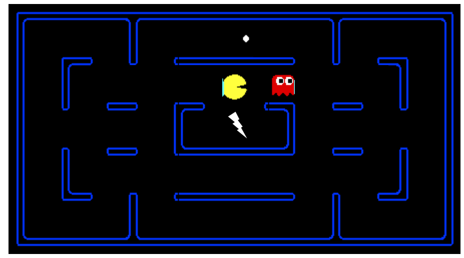


1

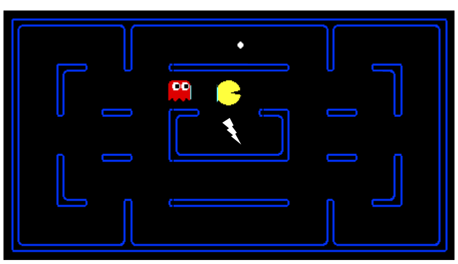
Best plan to eat final food?



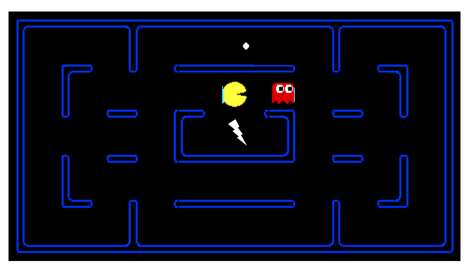
Best plan to eat final food?



Problem with Planning from MAP Estimate



49%



51%

- Best action for belief state over k worlds may not be the best action in ***any one*** of those worlds

POMDPs

- In POMDPs we apply the very same idea as in MDPs.
- Since the state is not observable, the agent has to make its decisions based on the ***belief state*** which is a ***posterior distribution over states***.

  π : beliefs → actions, denoted u

- Let *b* be the belief of the agent about the state under consideration.
- POMDPs compute a value function over belief space:

$$V_T(b) \quad = \quad \gamma \max_u \left[r(b,u) + \int V_{T-1}(b^*) p(b^* \mid u, b) \, db^* \right]$$

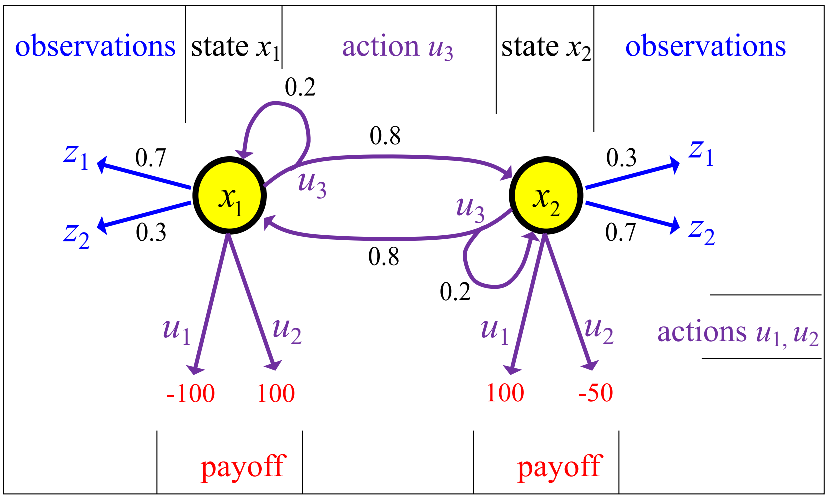
21

Problems

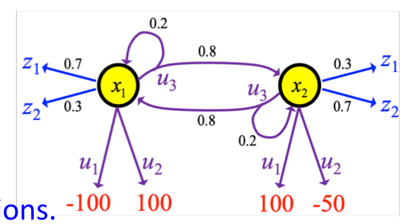
- A belief is a probability distribution over world states, thus, each value in a **POMDP is a function of an entire probability distribution**.
- **This is challenging, since probabilities are real-valued**.
  - Given 2 world states, s_1 & s_2, how many belief states are there?
- For **finite worlds** with finite state, action, and evidence spaces and finite horizons, we can **effectively represent the value functions by piecewise linear functions**.

22

An Illustrative Example



Example Parameters



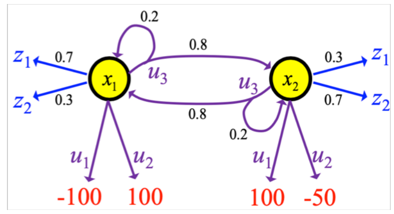
- The actions u_1 and u_2 are terminal actions.
- u_3 is a sensing action, potentially leading to a state transition.
- The horizon is finite and γ=1.

$$
\begin{aligned}
r(x_1,u_1) &= -100 & r(x_2,u_1) &= +100 \\
r(x_1,u_2) &= +100 & r(x_2,u_2) &= -50 \\
r(x_1,u_3) &= -1 & r(x_2,u_3) &= -1
\end{aligned}
$$

$$
\begin{aligned}
p(x_1'|x_1,u_3) &= 0.2 & p(x_2'|x_1,u_3) &= 0.8 \\
p(x_1'|x_2,u_3) &= 0.8 & p(z_2'|x_2,u_3) &= 0.2
\end{aligned}
$$

$$
\begin{aligned}
p(z_1|x_1) &= 0.7 & p(z_2|x_1) &= 0.3 \\
p(z_1|x_2) &= 0.3 & p(z_2|x_2) &= 0.7
\end{aligned}
$$

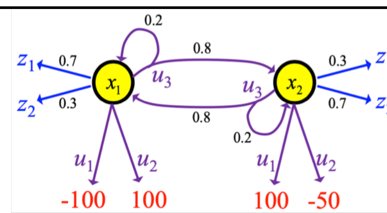
Expected Reward in POMDPs



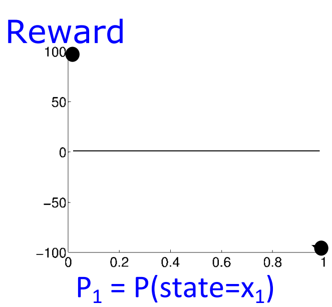
- In MDPs, the reward depends on the state of the system.
- In POMDPs, we don't know what state we are in?
- Therefore, we compute the **expected reward** by **integrating over all states**:

$$
\begin{aligned}
r(b,u) &= E_x[r(x,u)] \\
&= \int r(x,u)p(x)\ dx \\
&= p_1\ r(x_1,u) + p_2\ r(x_2,u)
\end{aligned}
$$

Example

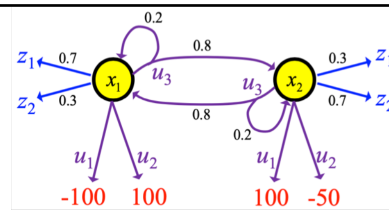


- If we are totally certain that we are in state x_1 and execute action u_1, we receive a reward of -100
- If, on the other hand, we definitely know that we are in x_2 and execute u_1, the reward is +100.
- In between: probability weighted linear combination

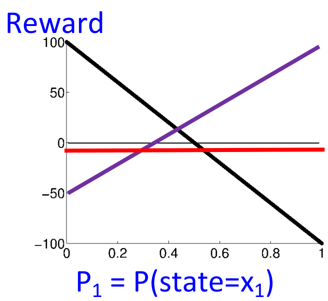


$$
\begin{aligned}
r(b,u_1) &= -100\ p_1 + 100\ p_2 \\
&= -100\ p_1 + 100\ (1-p_1) \\
&= 100 - 200\ p_1
\end{aligned}
$$

Example



- If we are totally certain that we are in state x_1 and execute action u_1, we receive a reward of -100
- If, on the other hand, we definitely know that we are in x_2 and execute u_1, the reward is +100.
- In between: probability weighted linear combination



$$
\begin{aligned}
r(b, u_1) &= -100\ p_1 + 100\ p_2 \\
&= -100\ p_1 + 100\ (1 - p_1) \\
&= 100 - 200\ p_1
\end{aligned}
$$

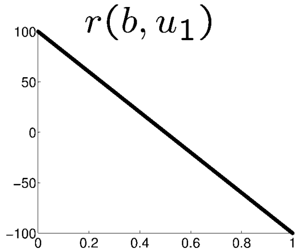
$$
\begin{aligned}
r(b, u_2) &= 100 p_1 - 50(1\text{-}p_1) \\
&= 150\ p_1 - 50
\end{aligned}
$$

$$r(b, u_3) = -1$$

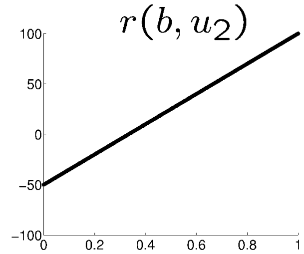
28

One Step Reward in Our Example

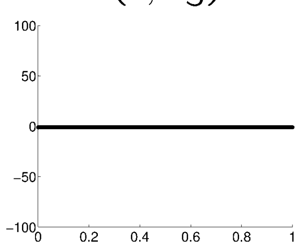
$r(b, u_1)$



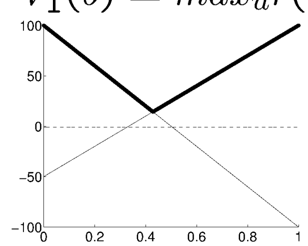
$r(b, u_2)$



$r(b, u_3)$



$V_1(b) = max_u r(b, u)$



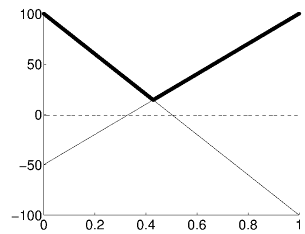
29

The Resulting Policy for T=1

- Given a finite POMDP with time horizon = 1
- Use $V_1(b)$ to determine the optimal policy.

$$\pi_1(b) = \begin{cases} u_1 & \text{if } p_1 \leq \frac{3}{7} \\ u_2 & \text{if } p_1 > \frac{3}{7} \end{cases} \quad = 0.429$$

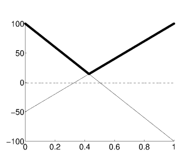
- Corresponding value:



Piecewise Linearity, Convexity

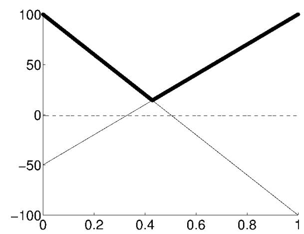
The resulting value function $V_1(b)$ is the maximum of the three functions at each point

$$V_1(b) = \max_u r(b,u)$$
$$= \max \left\{ \begin{array}{ll} -100\, p_1 & +100\,(1-p_1) \\ 100\, p_1 & -50\,(1-p_1) \\ 0 & \end{array} \right\}$$



I.e., it's piecewise linear and convex.

Pruning

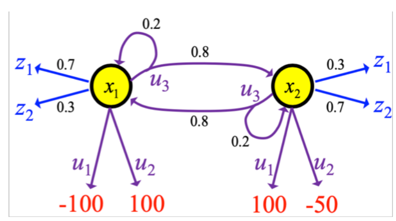


- With $V_1(b)$, note that only the first two components contribute.
- The third component can be safely pruned

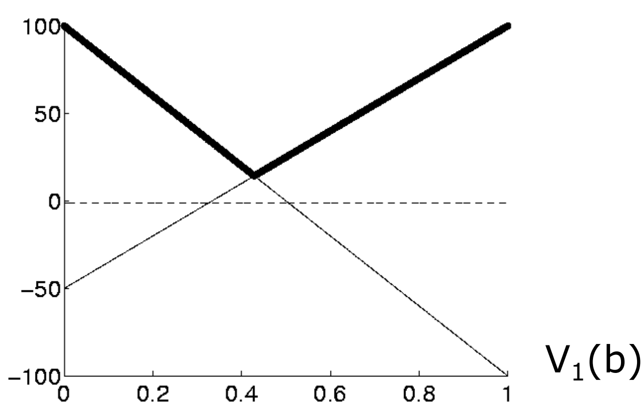
$$V_1(b) \;=\; \max\left\{ \begin{array}{ll} -100\;p_1 & +100\;(1-p_1) \\ 100\;p_1 & -50\;(1-p_1) \end{array} \right\}$$

32

Incorporating Observation



Suppose that the robot can magically receive an observation before deciding on an action.



33

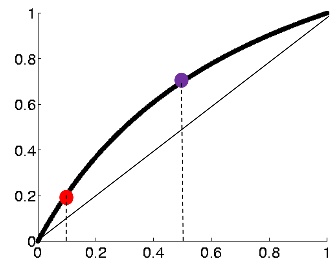
Incorporating Observation

- Suppose agent perceives z_1
- We know $p(z_1 \mid x_1)=0.7$ and $p(z_1 \mid x_2)=0.3$.
- Given the obs z_1 we update the belief using ...?

Bayes rule!!

$$p'_1 = \frac{0.7 p_1}{p(z_1)} \quad \text{where} \quad p(z_1) = 0.7 p_1 + 0.3(1-p_1) = 0.4 p_1 + 0.3$$

- So...
  - If $p_1 = 0.5$
  - then $p_1' = .35 / .50 = 0.7$
  - If $p_1 = 0.1$
  - then $p_1' = .07 / .34 = 0.206$



$b'(b|z_1)$

Incorporating Observation

- Suppose agent perceives z_1
- We know $p(z_1 \mid x_1)=0.7$ and $p(z_1 \mid x_2)=0.3$.
- Given the obs z_1 we update the belief using ...?

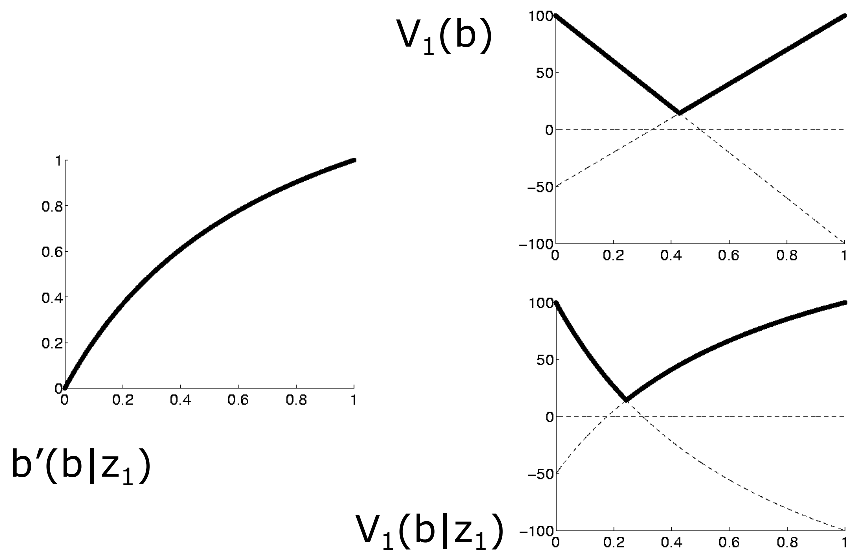
Bayes rule!!

$$p'_1 = \frac{0.7 p_1}{p(z_1)} \quad \text{where} \quad p(z_1) = 0.7 p_1 + 0.3(1-p_1) = 0.4 p_1 + 0.3$$

- Now, $V_1(b \mid z_1)$ is given by

$$\begin{aligned} V_1(b \mid z_1) &= \max \left\{ \begin{array}{ll} -100 \cdot \frac{0.7\, p_1}{p(z_1)} & +100 \cdot \frac{0.3\,(1-p_1)}{p(z_1)} \\ 100 \cdot \frac{0.7\, p_1}{p(z_1)} & -50 \cdot \frac{0.3\,(1-p_1)}{p(z_1)} \end{array} \right\} \\ &= \frac{1}{p(z_1)} \max \left\{ \begin{array}{ll} -70\, p_1 & +30\,(1-p_1) \\ 70\, p_1 & -15\,(1-p_1) \end{array} \right\} \end{aligned}$$

Value Function



36

Expected Value after Measuring

- But, we do not know ***in advance*** what the next measurement will be,
- So we must compute the expected belief

$$\bar{V}_1(b) = E_z[V_1(b \mid z)] = \sum_{i=1}^{2} p(z_i) V_1(b \mid z_i)$$

$$= \sum_{i=1}^{2} p(z_i) V_1\left(\frac{p(z_i \mid x_1) p_1}{p(z_i)}\right)$$

$$= \sum_{i=1}^{2} V_1\left(p(z_i \mid x_1) p_1\right)$$

37

Expected Value after Measuring

- But, we do not know ***in advance*** what the next measurement will be,
- So we must compute the expected belief

$$
\begin{aligned}
\bar{V}_1(b) &= E_z[V_1(b \mid z)] \\
&= \sum_{i=1}^{2} p(z_i)\, V_1(b \mid z_i) \\
&= \max \left\{ \begin{array}{ll} -70\, p_1 & +30\,(1-p_1) \\ 70\, p_1 & -15\,(1-p_1) \end{array} \right\} \\
&\quad + \max \left\{ \begin{array}{ll} -30\, p_1 & +70\,(1-p_1) \\ 30\, p_1 & -35\,(1-p_1) \end{array} \right\}
\end{aligned}
$$

38

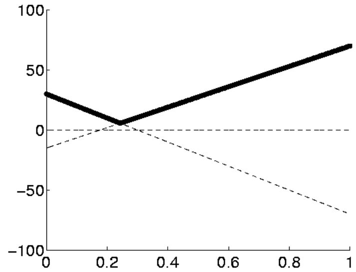
Resulting Value Function

- The four possible combinations yield the following function which then can be simplified and pruned.

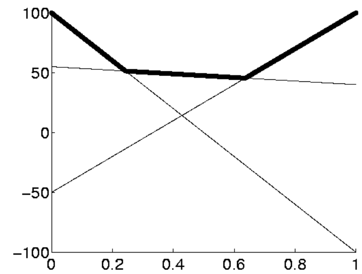
$$
\begin{aligned}
\bar{V}_1(b) &= \max \left\{ \begin{array}{llll} -70\, p_1 & +30\,(1-p_1) & -30\, p_1 & +70\,(1-p_1) \\ -70\, p_1 & +30\,(1-p_1) & +30\, p_1 & -35\,(1-p_1) \\ +70\, p_1 & -15\,(1-p_1) & -30\, p_1 & +70\,(1-p_1) \\ +70\, p_1 & -15\,(1-p_1) & +30\, p_1 & -35\,(1-p_1) \end{array} \right\} \\
&= \max \left\{ \begin{array}{ll} -100\, p_1 & +100\,(1-p_1) \\ +40\, p_1 & +55\,(1-p_1) \\ +100\, p_1 & -50\,(1-p_1) \end{array} \right\}
\end{aligned}
$$

39

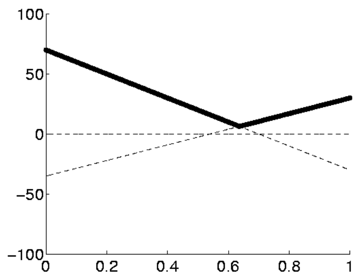
Value Function



$p(z_1)\ V_1(b|z_1)$



$\overline{V}_1(b)$



$p(z_2)\ V_2(b|z_2)$

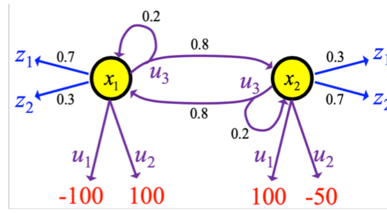
But….

- That was assuming that we were going to get an observation for free…
- What if the only way to get an observation is to execute the sensing action?

Increasing the Time Horizon

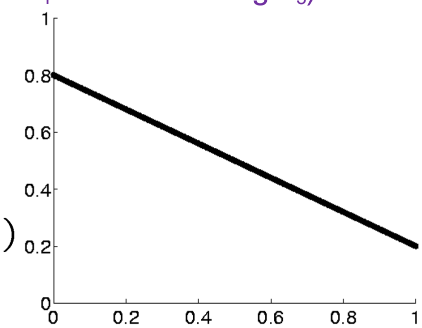
When the agent selects u_3 its state may change.

When computing the value function, we have to take these potential state changes into account.



$$
\begin{aligned}
p'_1 &= E_x[p(x_1 \mid x, u_3)] \\
&= \sum_{i=1}^{2} p(x_1 \mid x_i, u_3) p_i \\
&= 0.2p_1 + 0.8(1-p_1) \\
&= 0.8 - 0.6p_1
\end{aligned}
$$

P(x=x_1 after executing u_3)



P(x=x_1 originally)

42

Resulting Value Function after executing u_3

Taking the state transitions into account, we finally obtain.

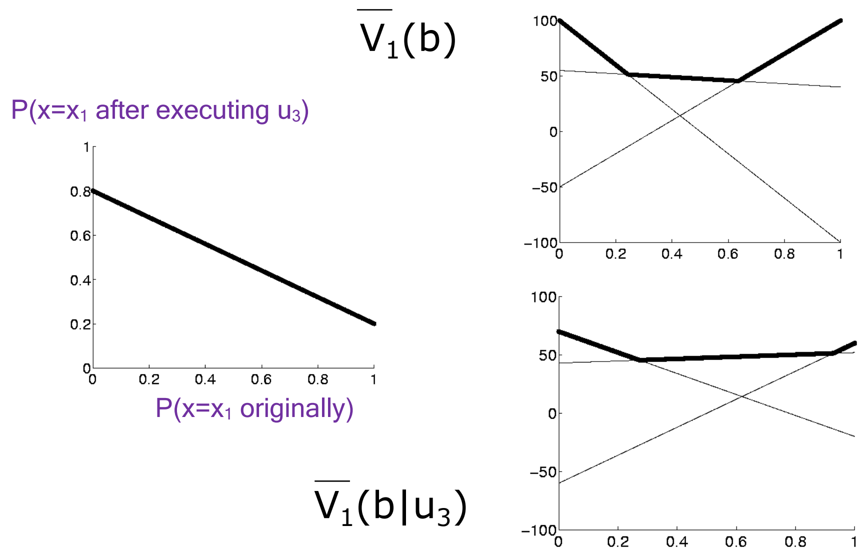
$$
\bar{V}_1(b) = \max \left\{ \begin{array}{llll} -70\,p_1 & +30\,(1-p_1) & -30\,p_1 & +70\,(1-p_1) \\ -70\,p_1 & +30\,(1-p_1) & +30\,p_1 & -35\,(1-p_1) \\ +70\,p_1 & -15\,(1-p_1) & -30\,p_1 & +70\,(1-p_1) \\ +70\,p_1 & -15\,(1-p_1) & +30\,p_1 & -35\,(1-p_1) \end{array} \right\}
$$

$$
= \max \left\{ \begin{array}{ll} -100\,p_1 & +100\,(1-p_1) \\ +40\,p_1 & +55\,(1-p_1) \\ +100\,p_1 & -50\,(1-p_1) \end{array} \right\}
$$

$$
\bar{V}_1(b \mid u_3) = \max \left\{ \begin{array}{ll} 60\,p_1 & -60\,(1-p_1) \\ 52\,p_1 & +43\,(1-p_1) \\ -20\,p_1 & +70\,(1-p_1) \end{array} \right\}
$$

43

Value Function after executing u_3



44

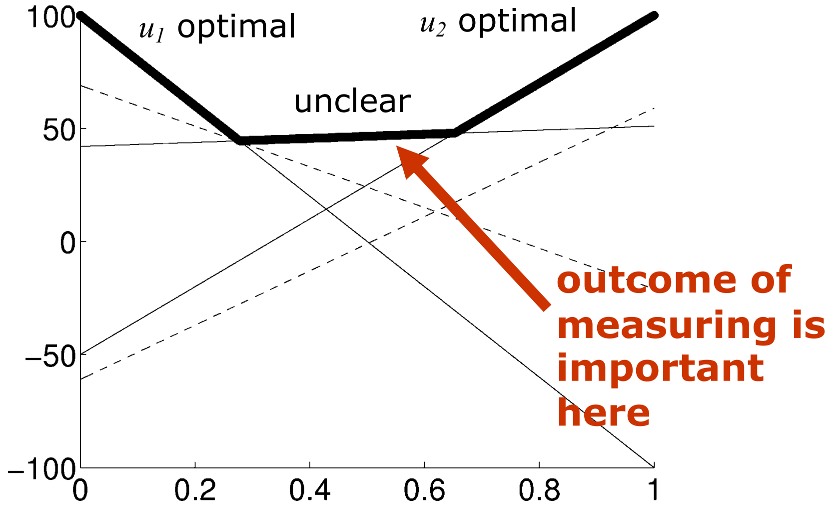
Value Function for T=2

- Taking into account that the agent can either directly perform u_1 or u_2 or first u_3 and then u_1 or u_2, we obtain (after pruning)

$$\bar{V}_2(b) \;=\; \max\left\{\begin{array}{rr} -100\,p_1 & +100\,(1-p_1) \\ 100\,p_1 & -50\,(1-p_1) \\ 51\,p_1 & +42\,(1-p_1) \end{array}\right\}$$

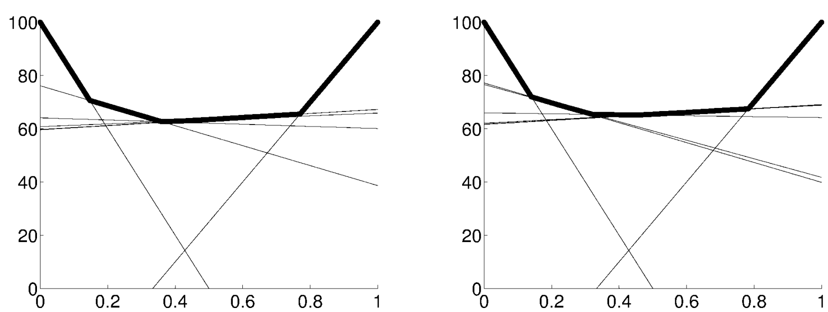
45

Graphical Representation of $V_2(b)$



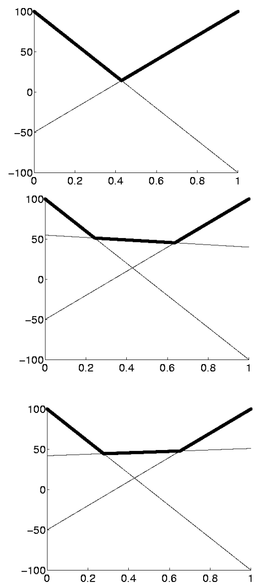
Deep Horizons

- We have now completed a full backup in belief space.
- This process can be applied recursively.
- The value functions for T=10 and T=20 are



Deep Horizons and Pruning

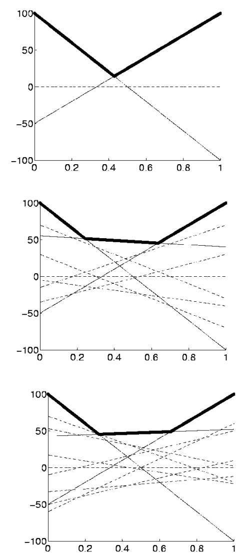
With Pruning **Without**



Horizon=1

Horizon=2

Horizon=3



48

Why Pruning is Essential

- Each **update introduces additional linear components** to *V*.
- Each **measurement squares** **the number of linear components**.
- Thus, an unpruned value function for T=20 includes more than $10^{547,864}$ linear functions.
- .
- The pruned value functions at T=20, in comparison, contains only 12 linear components.
- The combinatorial explosion of linear components in the value function are the major reason why the **exact solution of POMDPs is usually impractical**

49

POMDP Approximations

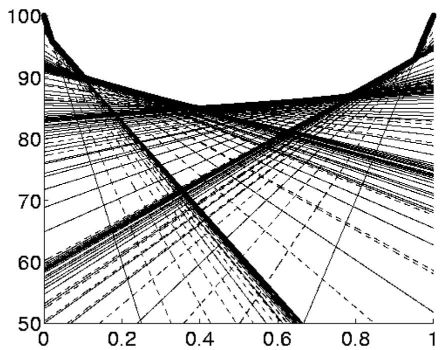
- Point-based value iteration
- QMDPs
- AEMS

Point-based Value Iteration

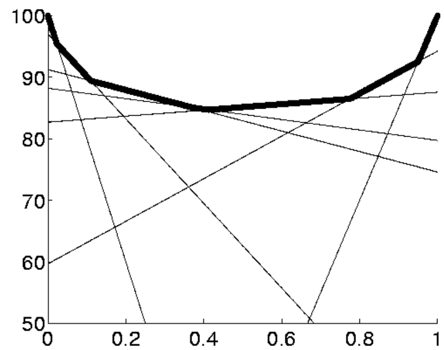
- Kind of like particle filtering…
- Maintains a set of example beliefs
- Only considers constraints that maximize value function for at least one of the examples

Point-based Value Iteration

Value functions for T=30



Exact value function



PBVI

53

POMDP Summary

- POMDPs compute the optimal action in partially observable, stochastic domains.
- For finite horizon problems, the resulting value functions are piecewise linear and convex.
- In each iteration the number of linear constraints grows exponentially.
- Until recently, POMDPs only applied to very small state spaces with small numbers of possible observations and actions.
  - But with PBVI, |S| = millions

57