PATTERN RECOGNITION AND MACHINE LEARNING

CHAPTER 15: MARKOV DECISION PROCESS

Learning Objectives

1、 What is the Markov decision process (MDP)?

2、 What is the partial observable MDP?

3、 What is the Bellman equation?

4、 What are value iteration and policy iteration?

5、 What are policy improvement and policy evaluation?

6、 How to use observation and prediction to update belief?

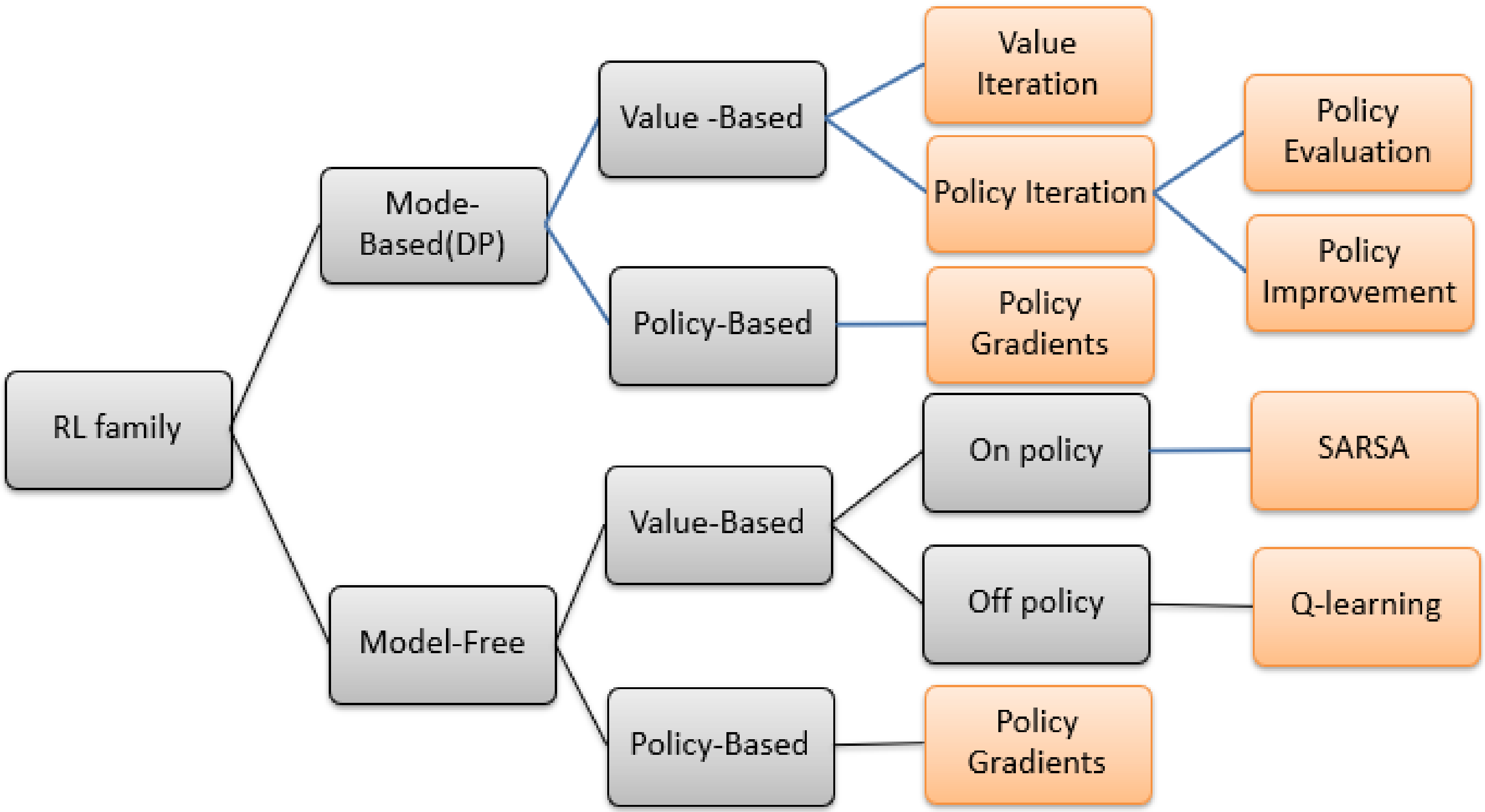
7、 What is the max-sum algorithm?

8、 How to reduce the computational complexity of POMDP?

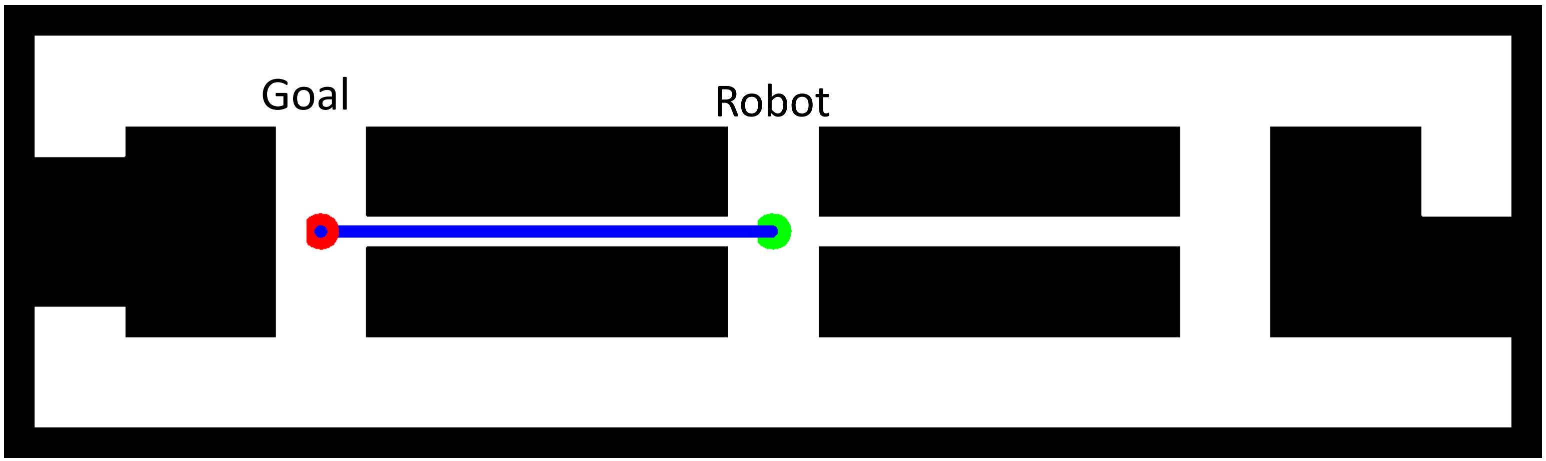
Outlines

- Markov Decision Process (MDP)
- Value Iteration and Policy Iteration
- Partially Observable MDP (POMDP)
- POMDP Observation and Prediction
- POMDP Approximation

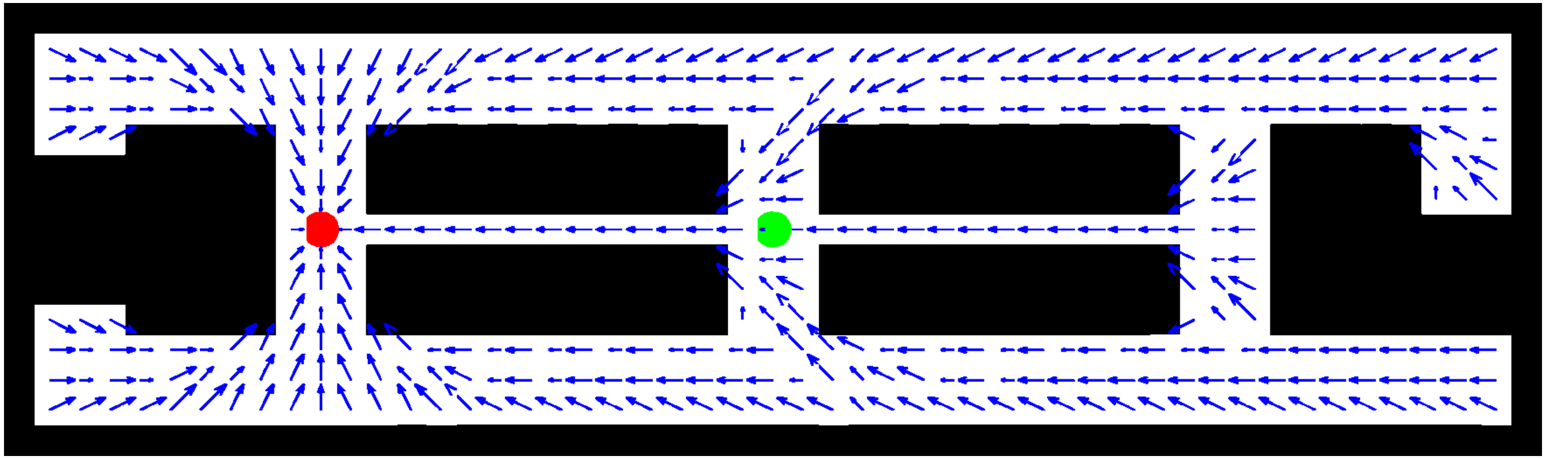
Reinforcement Learning

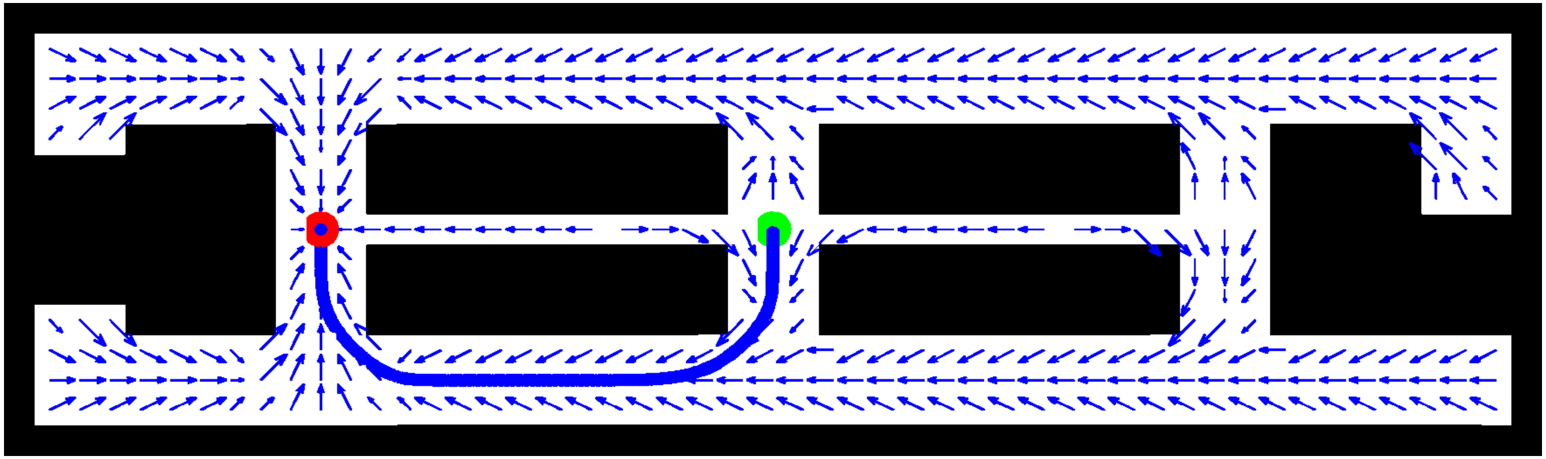


Robot Navigation Problem



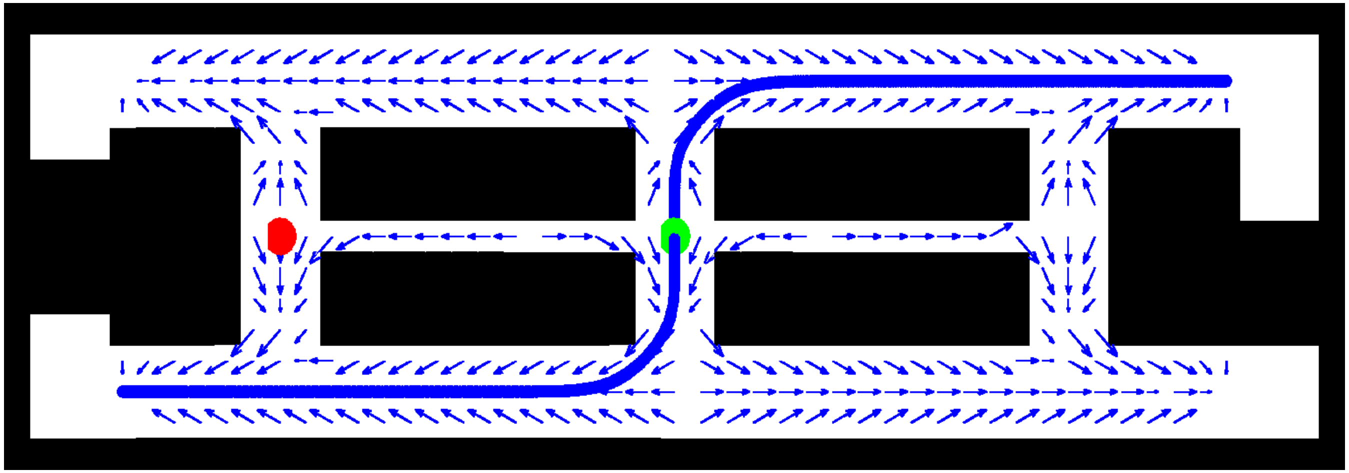
Uncertainty in Motion



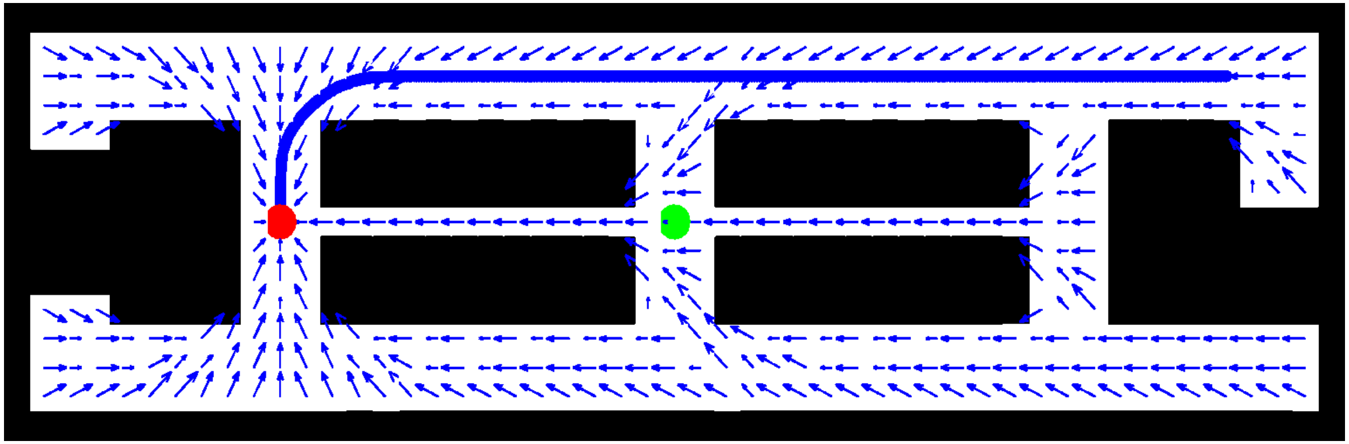


Uncertainty in Motion and Observation

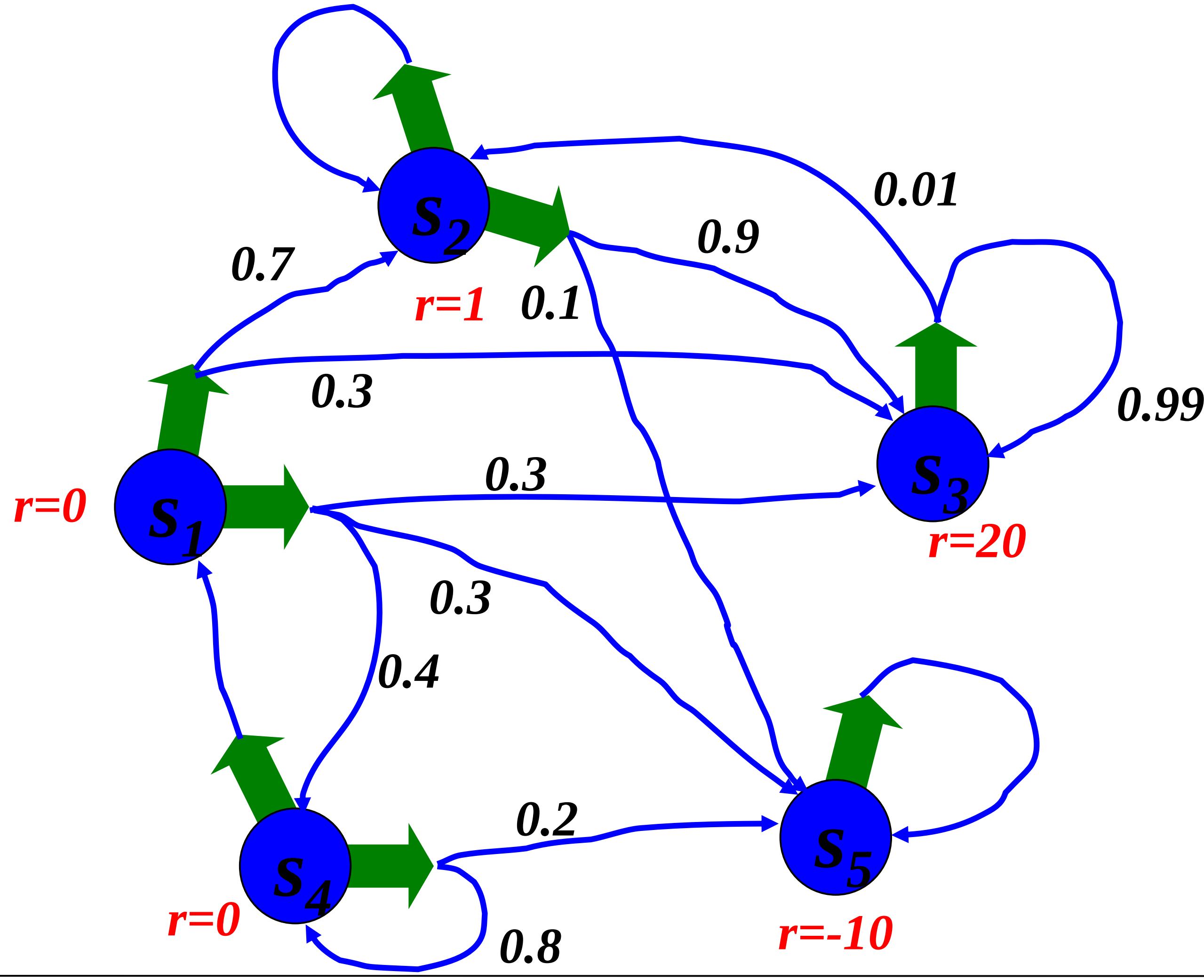
1st Step



2nd Step



Markov Decision Process



Markov Decision Process

			RIGHT GOAL
	OBSTACLE		WRONG GOAL
START POSITION			

Markov Decision Process Setup

- **Given:**

  States x, Actions u

  Transition probabilities $p(x'|u, x)$

  Reward function $r(x, u)$

- **Wanted:**

  Policy $\pi(x)$ that maximizes the future expected reward

Policy and Cumulative Reward

- Policy (fully observable case): $\pi : \quad x_t \to \quad u_t$

- Expected cumulative reward: $R_T \quad = \quad E\left[\sum_{\tau=0}^{T} \gamma^\tau r_{t+\tau}\right]$

$$\boxed{R_\infty \quad \leq \quad r_{\max} + \gamma r_{\max} + \gamma^2 r_{\max} + \gamma^3 r_{\max} + \ldots \quad = \quad \frac{r_{\max}}{1-\gamma}}$$

T=1 : greedy policy
T>1 : finite horizon case, typically no discount
T=infinity: infinite-horizon case, finite reward if discount < 1

Optimal Policy

- Expected cumulative reward of policy:

$$R_T^\pi(x_t) \quad = \quad E\left[\sum_{\tau=0}^{T} \gamma^\tau r_{t+\tau} \mid u_{t+\tau} = \pi\left(x_{t+\tau}\right) \right]$$

- Optimal policy:

$$\pi^* = \underset{\pi}{\operatorname{argmax}} \quad R_T^\pi(x_t)$$

Return from Rewards

- **episodic** (vs. continuing) tasks
  "game over" after N steps
  optimal policy depends on N; harder to analyze

- **additive rewards**
  $R(x_t, x_{t+1}, \ldots) = r(x_t) + r(x_{t+1}) + r(x_{t+2}) + \ldots$
  infinite value for continuing tasks

- **discounted rewards**
  $R(x_t, x_{t+1}, \ldots) = r(x_t) + \gamma * r(x_{t+1}) + \gamma^2 * r(x_{t+2}) + \ldots$
  value bounded if rewards bounded

State Value Function

- Expected return when starting from x_t and following policy π:

$$V^{\pi}(x_t) = R^{\pi}_{\infty}(x_t)$$

- Bellman equation for policy π:

$$V^{\pi}(x) = \sum_{u} \pi(x,u) \left[r(x,u) + \gamma \int V^{\pi}(x') p(x' | x,u) dx' \right]$$

Optimal Value Function

- Optimal return for all possible policies:

$$V(x) = \max_{\pi} V^{\pi}(x)$$

- Bellman equation for optimal value function:

$$V(x) = \sum_{u} \pi(x,u) \left[r(x,u) + \gamma \int V(x')p(x'|x,u)dx' \right]$$

1-Step Optimal Policy and Value Function

- 1-step optimal policy:

$$\pi_1(x) = \underset{u}{\operatorname{argmax}} \quad r(x,u)$$

- Optimal value function of 1-step optimal policy:

$$V_1(x) = \max_u \; r(x,u)$$

2-Step Optimal Policy and Value Function

- 2-step optimal policy:

$$\pi_2(x) \quad = \quad \underset{u}{\operatorname{argmax}} \left[r(x,u) + \gamma \underbrace{\int V_1(x')\, p(x' \mid u, x)\, dx'}_{\text{Predicted Value}} \right]$$

Current Reward

- 2-step optimal value function:

$$V_2(x) \quad = \quad \max_{u} \left[r(x,u) + \gamma \underbrace{\int V_1(x')\, p(x' \mid u, x)\, dx'}_{\text{Predicted Value}} \right]$$

Current Reward

T-Step Optimal Policy and Value Function

- T-step optimal policy:

$$\pi_T(x) \quad = \quad \underset{u}{\text{gmax}} \left[\underset{\uparrow \atop \boxed{\text{Current Reward}}}{r(x,u)} + \gamma \int \underbrace{V_{T-1}(x')\, p(x' \mid u, x)\, dx'}_{\text{Predicted Value}} \right]$$

- T-step optimal value function:

$$V_T(x) \quad = \quad \max_{u} \left[\underset{\uparrow \atop \boxed{\text{Current Reward}}}{r(x,u)} + \gamma \int \underbrace{V_{T-1}(x')\, p(x' \mid u, x)\, dx'}_{\text{Predicted Value}} \right]$$

Infinite Horizon

- Optimal value function:

$$V_\infty(x) = \max_u \Big[\underbrace{r(x,u)}_{\text{Current Reward}} + \gamma \underbrace{\int V_\infty(x')\, p(x' \mid u, x)\, dx'}_{\text{Predicted Value}} \Big]$$

- Bellman equation
  - ✓ Fix point is optimal policy
  - ✓ Necessary and sufficient condition

Outlines

- Markov Decision Process (MDP)
- Value Iteration and Policy Iteration
- Partially Observable MDP (POMDP)
- POMDP Observation and Prediction
- POMDP Approximation

Value Iteration

for all *x* do

$$\hat{V} \longleftarrow r_{\min}$$

endfor

repeat until convergence

for all *x* do

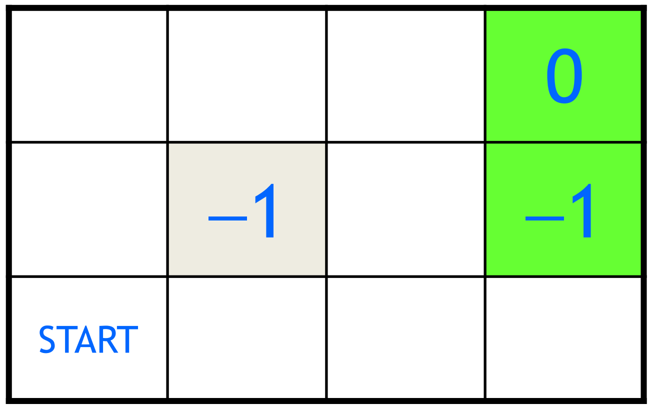
$$\hat{V}(x) \longleftarrow \max_u \left[r(x,u) + \gamma \int \hat{V}(x')\, p(x' \mid u, x)\, dx' \right]$$

endfor

endrepeat

$$\pi(x) = \underset{u}{\operatorname{argmax}} \left[r(x,u) + \gamma \int \hat{V}(x')\, p(x' \mid u, x)\, dx' \right]$$

MDP Model



Environment and reward:

a) Green rectangle: destination, reward = 0 for any action
b) Black rectangle : wall, reward = -1
c) reward = - 0.1 for each step in other states
d) action = {up, down, left, right}

MDP Model

0	1	2	3
4	5	6	7
8	9	10	11

a) Position 3: reward = 0 for any action
b) Positions 5 and 7: wall, reward = -1
c) reward = - 0.1 for each step in other states
d) action = {up/0, down/1, left/2, right/3}

transition probabilities:

$$\{x\text{: }\{u_1\text{: }(x', p(x'|x, u_1), r), u_2\text{: }(x', p(x'|x, u_2), r),$$
$$u_3\text{: }(x', p(x'|x, u_3), r), u_4\text{: }(x', p(x'|x, u_4), r)\ \}\}$$


```
{0: {0: (0, 1.0, -0.1), 1: (4, 1.0, -0.1), 3: (1, 1.0, -0.1), 2: (0, 1.0, -0.1)}, 1: {0: (1,
1.0, -0.1), 1: (1, 1.0, -1), 3: (2, 1.0, -0.1), 2: (0, 1.0, -0.1)}, 2: {0: (2, 1.0, -0.1), 1:
(6, 1.0, -0.1), 3: (3, 1.0, -0.1), 2: (1, 1.0, -0.1)}, 3: {0: (3, 1.0, 0), 1: (3, 1.0, 0), 3:
(3, 1.0, 0), 2: (3, 1.0, 0)}, 4: {0: (0, 1.0, -0.1), 1: (8, 1.0, -0.1), 3: (4, 1.0, -1), 2: (4,
1.0, -0.1)}, 5: {0: (1, 1.0, -0.1), 1: (9, 1.0, -0.1), 3: (6, 1.0, -0.1), 2: (4, 1.0, -0.1)}, 6:
{0: (2, 1.0, -0.1), 1: (10, 1.0, -0.1), 3: (6, 1.0, -1), 2: (6, 1.0, -1)}, 7: {0: (3, 1.0,
-0.1), 1: (11, 1.0, -0.1), 3: (7, 1.0, -1), 2: (6, 1.0, -0.1)}, 8: {0: (4, 1.0, -0.1), 1: (8,
1.0, -0.1), 3: (9, 1.0, -0.1), 2: (8, 1.0, -0.1)}, 9: {0: (9, 1.0, -1), 1: (9, 1.0, -0.1), 3:
(10, 1.0, -0.1), 2: (8, 1.0, -0.1)}, 10: {0: (6, 1.0, -0.1), 1: (10, 1.0, -0.1), 3: (11, 1.0,
-0.1), 2: (9, 1.0, -0.1)}, 11: {0: (11, 1.0, -1), 1: (11, 1.0, -0.1), 3: (11, 1.0, -0.1), 2:
(10, 1.0, -0.1)}}
```

Value Iteration (I)

Value Function V^0

0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0

0	1	2	3
4	5	6	7
8	9	10	11

$\boxed{V^0(0) = 0.0}$ $\boxed{V^1(0) = -0.1}$

$r(0, up) + V^0(0)*p(0|0,up) = -0.1 + (-0.0)*1 = -0.1$

$r(0, do) + V^0(4)*p(4|0,do) = -0.1 + (-0.0)*1 = -0.1$

$\boxed{r(0, rig) + V^0(1)*p(1|0,rig) = -0.1 + (-0.0)*1 = -0.1}$

$r(0, lef) + V^0(0)*p(0|0,lef) = -0.1 + (-0.0)*1 = -0.1$

$\boxed{V^0(1) = 0.0}$ $\boxed{V^1(1) = -0.1}$

$r(1, up) + V^0(1)*p(1|1,up) = -0.1 + (-0.0)*1 = -0.1$

$r(1, do) + V^0(1)*p(1|1,do) = -1.0 + (-0.0)*1 = -1.0$

$\boxed{r(1, rig) + V^0(2)*p(2|1,rig) = -0.1 + (-0.0)*1 = -0.1}$

$r(1, lef) + V^0(0)*p(0|1,lef) = -0.1 + (-0.0)*1 = -0.1$

Value Iteration (II)

Value Function V^1

– 0.1	– 0.1	– 0.1	0.0
– 0.1	0.0	– 0.1	0.0
– 0.1	– 0.1	– 0.1	– 0.1

0	1	2	3
4	5	6	7
8	9	10	11

$\boxed{V^1(0) = -0.1}$ $\boxed{V^2(0) = -0.2}$

$r(0, up) + V^1(0)*p(0|0,up) = -0.1+ (-0.1)*1= -0.2$

$r(0, do) + V^1(4)*p(4|0,do) = -0.1+ (-0.1)*1= -0.2$

$\boxed{r(0, rig) + V^1(1)*p(1|0,rig) = -0.1+ (-0.1)*1= -0.2}$

$r(0, lef) + V^1(0)*p(0|0,lef) = -0.1+ (-0.1)*1= -0.2$

$\boxed{V^1(1) = -0.1}$ $\boxed{V^2(1) = -0.2}$

$r(1, up) + V^1(1)*p(1|1,up) = -0.1+ (-0.1)*1= -0.2$

$r(1, do) + V^1(1)*p(1|1,do) = -1.0+ (-0.1)*1= -1.1$

$\boxed{r(1, rig) + V^1(2)*p(2|1,rig) = -0.1+ (-0.1)*1= -0.2}$

$r(1, lef) + V^1(0)*p(0|1,lef) = -0.1+ (-0.1)*1= -0.2$

Value Iteration (III)

Value Function V^2

– 0.2	– 0.2	– 0.1	0.0
– 0.2	0.0	– 0.2	0.0
– 0.2	– 0.2	– 0.2	– 0.2

$\boxed{V^2(0) = -0.2}$ $\boxed{V^3(0) = -0.3}$

$r(0, up) + V^2(0)*p(0|0,up) = -0.1 + (-0.2)*1 = -0.3$

$r(0, do) + V^2(4)*p(4|0,do) = -0.1 + (-0.2)*1 = -0.3$

$\boxed{r(0, rig) + V^2(1)*p(1|0,rig) = -0.1 + (-0.2)*1 = -0.3}$

$r(0, lef) + V^2(0)*p(0|0,lef) = -0.1 + (-0.2)*1 = -0.3$

0	1	2	3
4	5	6	7
8	9	10	11

$\boxed{V^2(1) = -0.2}$ $\boxed{V^3(1) = -0.2}$

$r(1, up) + V^2(1)*p(1|1,up) = -0.1 + (-0.2)*1 = -0.3$

$r(1, do) + V^2(1)*p(1|1,do) = -1.0 + (-0.2)*1 = -1.2$

$\boxed{r(1, rig) + V^2(2)*p(2|1,rig) = -0.1 + (-0.1)*1 = -0.2}$

$r(1, lef) + V^2(0)*p(0|1,lef) = -0.1 + (-0.2)*1 = -0.3$

Value Iteration (IV)

Value Function V^3

– 0.3	– 0.2	– 0.1	0.0
– 0.3	0.0	– 0.2	0.0
– 0.3	– 0.3	– 0.3	– 0.3

0	1	2	3
4	5	6	7
8	9	10	11

$\boxed{V^3(0) = -0.2}$ $\boxed{V^4(0) = -0.3}$

$r(0, up) + V^3(0)*p(0|0,up) = -0.1 + (-0.3)*1 = -0.4$

$r(0, do) + V^3(4)*p(4|0,do) = -0.1 + (-0.3)*1 = -0.4$

$\boxed{r(0, rig) + V^3(1)*p(1|0,rig) = -0.1 + (-0.2)*1 = -0.3}$

$r(0, lef) + V^3(0)*p(0|0,lef) = -0.1 + (-0.3)*1 = -0.4$

$\boxed{V^3(1) = -0.2}$ $\boxed{V^4(1) = -0.2}$

$r(1, up) + V^3(1)*p(1|1,up) = -0.1 + (-0.2)*1 = -0.3$

$r(1, do) + V^3(1)*p(1|1,do) = -1.0 + (-0.2)*1 = -1.2$

$\boxed{r(1, rig) + V^3(2)*p(2|1,rig) = -0.1 + (-0.1)*1 = -0.2}$

$r(1, lef) + V^3(0)*p(0|1,lef) = -0.1 + (-0.3)*1 = -0.4$

Value Iteration (V)

Value Function V^4

– 0.3	– 0.2	– 0.1	0.0
– 0.4	0.0	– 0.2	0.0
– 0.4	– 0.4	– 0.3	– 0.4

0	1	2	3
4	5	6	7
8	9	10	11

$\boxed{V^4(0) = -0.2}$ $\boxed{V^5(0) = -0.3}$

$r(0, up) + V^1(0)*p(0|0,up) = -0.1 + (-0.3)*1 = -0.4$

$r(0, do) + V^1(4)*p(4|0,do) = -0.1 + (-0.4)*1 = -0.5$

$\boxed{r(0, rig) + V^1(1)*p(1|0,rig) = -0.1 + (-0.2)*1 = -0.3}$

$r(0, lef) + V^1(0)*p(0|0,lef) = -0.1 + (-0.3)*1 = -0.4$

$\boxed{V^4(1) = -0.2}$ $\boxed{V^5(1) = -0.2}$

$r(1, up) + V^1(1)*p(1|1,up) = -0.1 + (-0.2)*1 = -0.3$

$r(1, do) + V^1(1)*p(1|1,do) = -1.0 + (-0.2)*1 = -1.2$

$\boxed{r(1, rig) + V^1(2)*p(2|1,rig) = -0.1 + (-0.1)*1 = -0.2}$

$r(1, lef) + V^1(0)*p(0|1,lef) = -0.1 + (-0.3)*1 = -0.4$

Stationary Value Function

Stationary Value Function

– 0.3	– 0.2	– 0.1	0.0
– 0.4	0.0	– 0.2	0.0
– 0.5	– 0.4	– 0.3	– 0.4

0	1	2	3
4	5	6	7
8	9	10	11

$\boxed{V(0) = -0.3}$

r(0, up) + V(0)*p(0|0,up) = – 0.1+ (– 0.3)*1= – 0.4

r(0, do) + V(4)*p(4|0,do) = – 0.1+ (– 0.4)*1= – 0.5

$\boxed{r(0, rig) + V(1)*p(1|0,rig) = -0.1+ (-0.2)*1= -0.3}$

r(0, lef) + V(0)*p(0|0,lef) = – 0.1+ (– 0.3)*1= – 0.4

$\boxed{V(1) = -0.2}$

r(1, up) + V(1)*p(1|1,up) = – 0.1+ (– 0.2)*1= – 0.3

r(1, do) + V(1)*p(1|1,do) = – 1.0+ (– 0.2)*1= – 1.0

$\boxed{r(1, rig) + V(2)*p(2|1,rig) = -0.1+ (-0.1)*1= -0.2}$

r(1, lef) + V(0)*p(0|1,lef) = – 0.1+ (– 0.3)*1= – 0.4

Optimal Policy for Value Iteration

Stationary Value Function

–0.3	–0.2	–0.1	0.0
–0.4	–0.0	–0.2	–0.0
–0.5	–0.4	–0.3	–0.4

V(0) = – 0.3

Optimal Action: right →

r(0, up) + V(0)*p(0|0,up) = – 0.1+ (– 0.3)*1= – 0.4

r(0, do) + V(4)*p(4|0,do) = – 0.1+ (– 0.4)*1= – 0.5

r(0, rig) + V(1)*p(1|0,rig) = – 0.1+ (– 0.2)*1= – 0.3

r(0, lef) + V(0)*p(1|0,lef) = – 0.1+ (– 0.3)*1= – 0.4

Optimal Policy

→	→	→	●
↑	□	↑	□
↑ →	→	↑	←

V(1) = – 0.2

Optimal Action: right →

r(1, up) + V(1)*p(1|1,up) = – 0.1+ (– 0.2)*1= – 0.3

r(1, do) + V(1)*p(1|1,do) = – 1.0+ (– 0.0)*1= – 1.0

r(1, rig) + V(2)*p(2|1,rig) = – 0.1+ (– 0.1)*1= – 0.2

r(1, lef) + V(0)*p(0|1,lef) = – 0.1+ (– 0.3)*1= – 0.4

Policy Iteration

- Often the optimal policy has been reached long before the value function has converged.
- Policy iteration (1) calculates a new policy based on the current value function and (2) then calculates a new value function based on this policy.

(1) Policy improvement $$\pi^* = \underset{\pi}{\operatorname{argmax}} \quad R_T^\pi(x_t)$$

(2) Policy evaluation

$$R_T^\pi(x_t) = E\left[\sum_{\tau=0}^{T} \gamma^\tau r_{t+\tau} \mid u_{t+\tau} = \pi(x_{t+\tau})\right]$$

- Often converges faster to the optimal policy.

Policy Iteration

- Policy evaluation

$$V_{k+1}^{\pi}(x) = \sum_{u} \pi(x,u) \left[r(x,u) + \gamma \int V_k^{\pi}(x') p(x'|x,u) dx' \right]$$

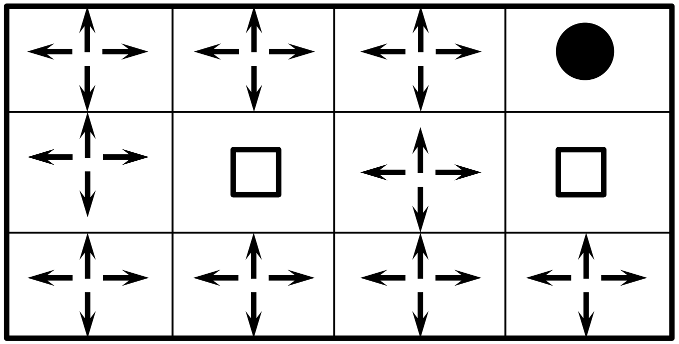
Until converged

- Policy improvement

$$\pi^*(x) = \arg\max_{u} \left[r(x,u) + \gamma \int V^{\pi}(x') p(x'|x,u) dx' \right]$$

Policy Iteration (I)

Policy π^0



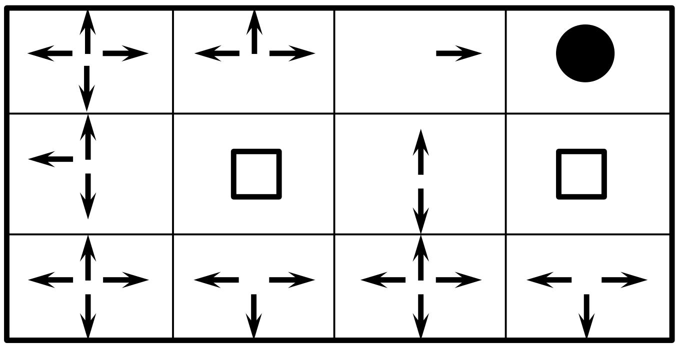
Value Function V^0

–0.1	–0.1	–0.1	0.0
–0.1	–0.0	–0.1	–0.0
–0.1	–0.1	–0.1	–0.1

0	1	2	3
4	5	6	7
8	9	10	11

Policy Iteration (I)

Policy π^1



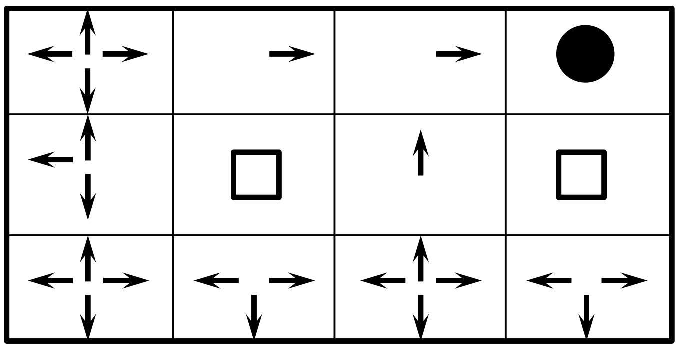
Value Function V^1

–0.2	–0.2	–0.1	0.0
–0.2	–0.0	–0.2	–0.0
–0.2	–0.2	–0.2	–0.2

0	1	2	3
4	5	6	7
8	9	10	11

Policy Iteration (I)

Policy π^2



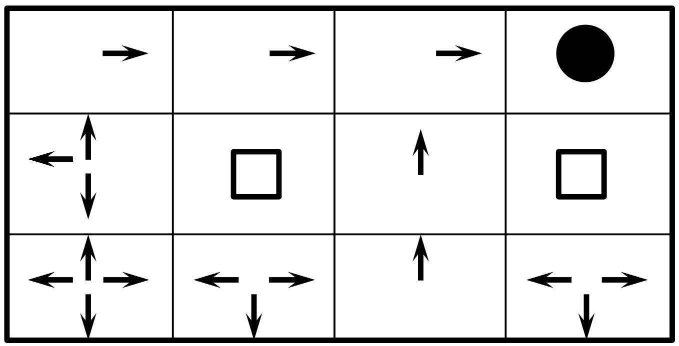
Value Function V^2

–0.3	–0.2	–0.1	0.0
–0.3	–0.0	–0.2	–0.0
–0.3	–0.3	–0.3	–0.3

0	1	2	3
4	5	6	7
8	9	10	11

Policy Iteration (I)

Policy π^3



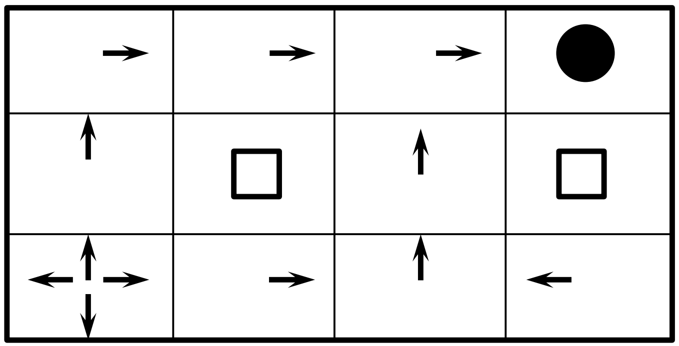
0	1	2	3
4	5	6	7
8	9	10	11

Value Function V^3

–0.3	–0.2	–0.1	0.0
–0.4	–0.0	–0.2	–0.0
–0.4	–0.4	–0.3	–0.4

Policy Iteration (II)

Policy π^4



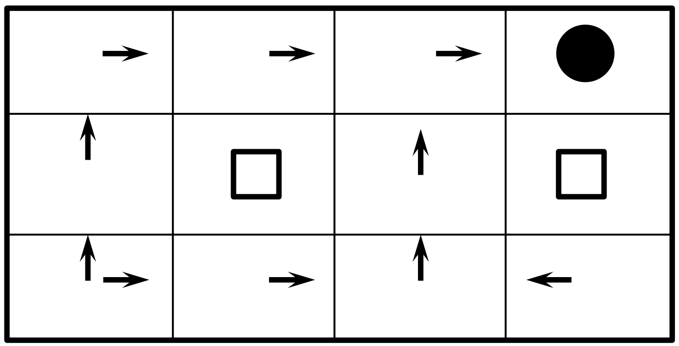
0	1	2	3
4	5	6	7
8	9	10	11

Value Function V^4

–0.3	–0.2	–0.1	0.0
–0.4	–0.0	–0.2	–0.0
–0.5	–0.4	–0.3	–0.4

Policy Iteration (II)

Policy π^5



Value Function V^5

–0.3	–0.2	–0.1	0.0
–0.4	–0.0	–0.2	–0.0
–0.5	–0.4	–0.3	–0.4

0	1	2	3
4	5	6	7
8	9	10	11

Outlines

- Markov Decision Process (MDP)
- Value Iteration and Policy Iteration
- Partially Observable MDP (POMDP)
- POMDP Observation and Prediction
- POMDP Approximation

POMDPs

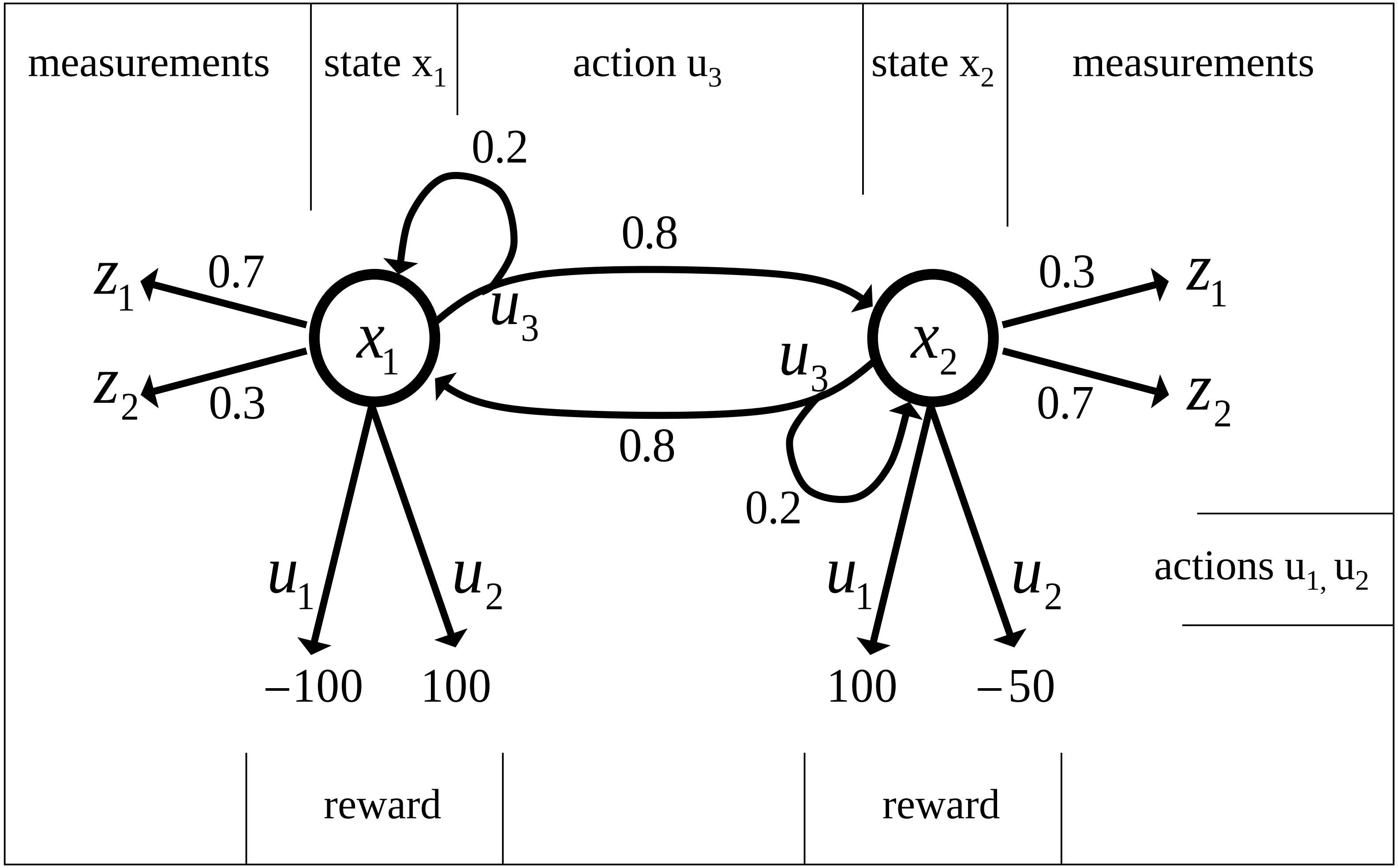
- In POMDPs we apply the very same idea as in MDPs.
- **Since the state is not observable**, the agent has to **make its decisions based on** the belief state which is a **posterior distribution over states**.
- Let b be the belief of the agent about the state under consideration.
- POMDPs compute a **value function over belief space**:

$$V_T(b) \quad = \quad \gamma \max_u \left[r(b,u) + \int V_{T-1}(b') p(b' \mid u, b) \, db' \right]$$

Problem

- Each belief is a probability distribution, and thus each value in a **POMDP is a function of an entire probability distribution**.
- **This is problematic, since probability distributions are continuous**.
- Additionally, we have to deal with the **huge complexity of belief spaces**.
- For **finite worlds** with finite state, action, and measurement spaces and finite horizons, however, we can **effectively represent the value functions by piecewise linear functions**.

An Illustrative Example



Parameters

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

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

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

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

Reward in POMDPs

- In MDPs, the reward depended on the state of the system.
- In POMDPs, however, the true state is not exactly known.
- Therefore, we compute the **expected reward** by **integrating over all states**:

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

Reward of Example (I)

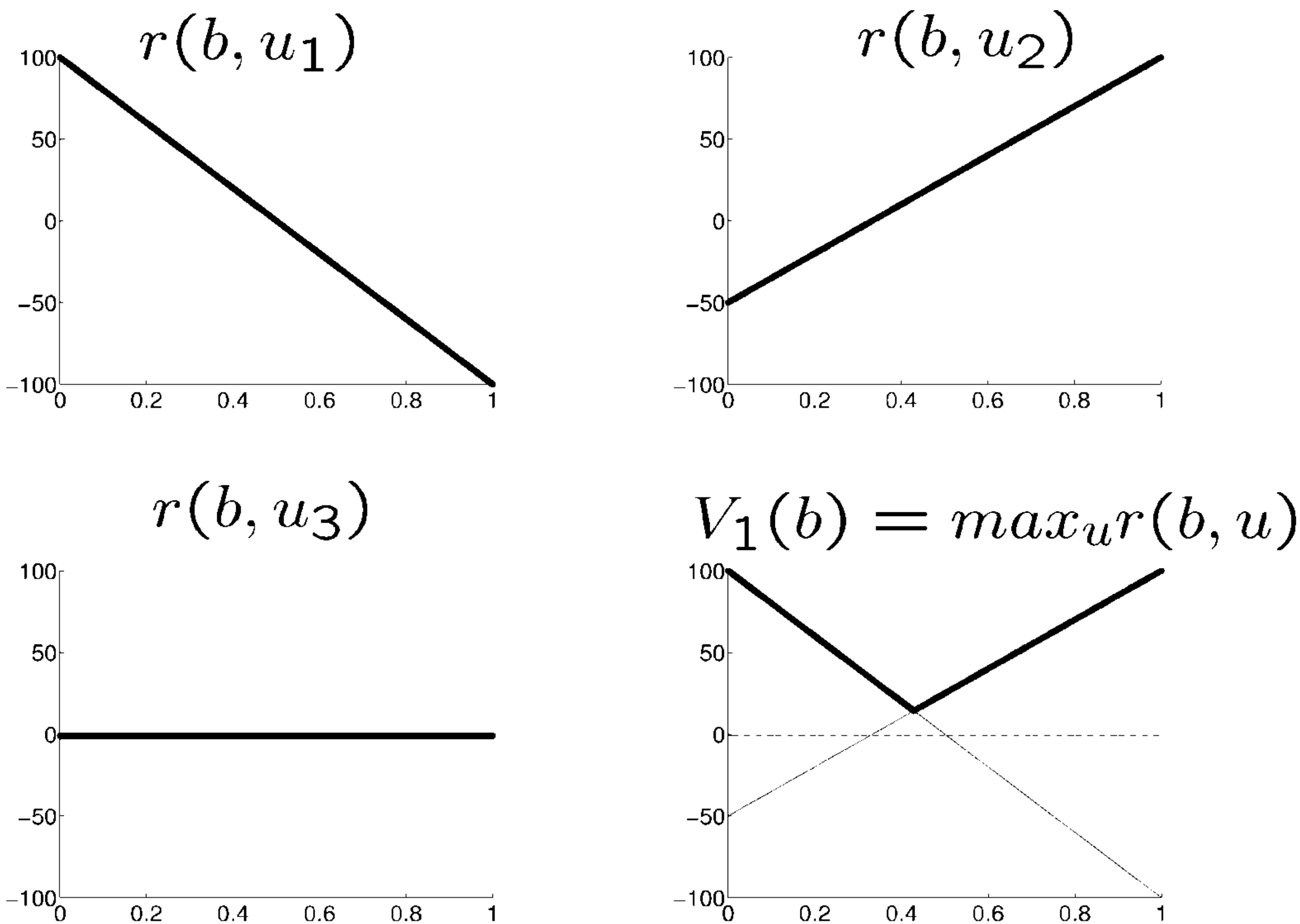
- If we are totally certain that we are in state x_1 and execute action u_1, we receive a reward of -100
- If, on the other hand, we definitely know that we are in x_2 and execute u_1, the reward is +100.
- In between it is the linear combination of the extreme values weighted by the probabilities

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

$$r(b, u_2) = 100\, p_1 - 50\, (1 - p_1)$$

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

Reward of Example (II)



The Resulting Policy for T=1

- Given we have a finite POMDP with T=1, we would use $V_1(b)$ to determine the optimal policy.
- In our example, the optimal policy for T=1 is

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

- This is the upper thick graph in the diagram.

Piecewise Linearity and Convexity

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

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

- It is piecewise linear and convex.

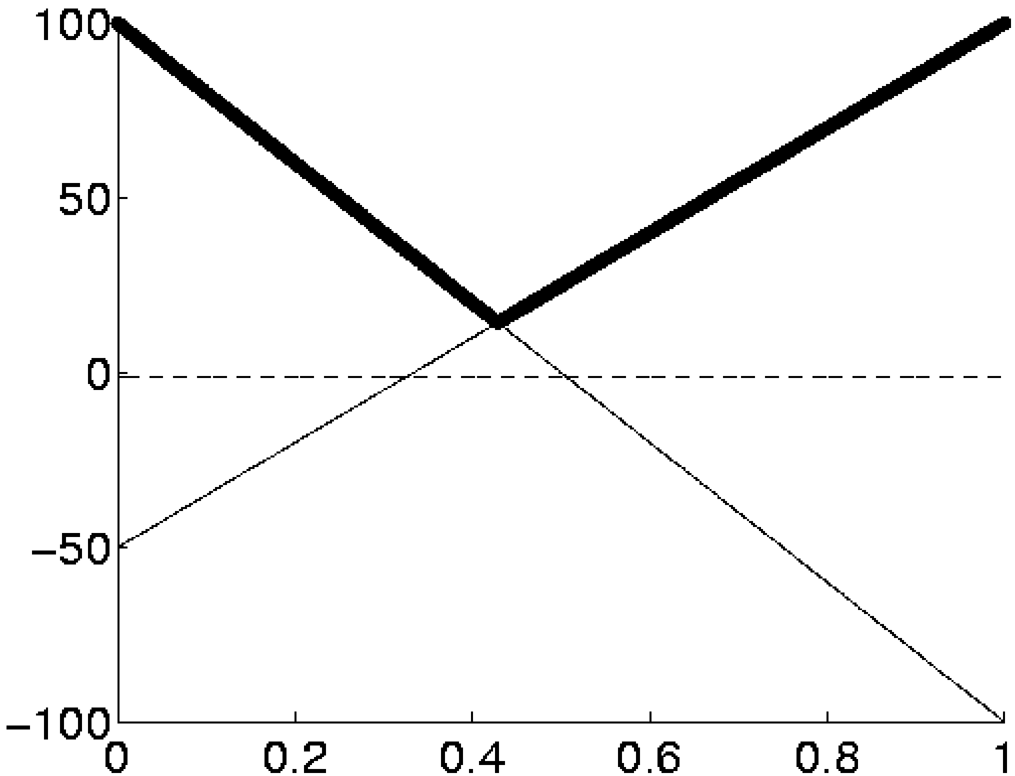
Pruning

- If we carefully consider $V_1(b)$, we see that only the first two components contribute.
- The third component can therefore safely be pruned away from $V_1(b)$.

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

Increasing the Time Horizon

- Assume the robot can make an observation before deciding on an action.



$V_1(b)$

Outlines

- Markov Decision Process (MDP)
- Value Iteration and Policy Iteration
- Partially Observable MDP (POMDP)
- POMDP Observation and Prediction
- POMDP Approximation

Increasing the Time Horizon

- Assume the robot can make an observation before deciding on an action.
- Suppose the robot perceives z_1 for which $p(z_1 \mid x_1)=0.7$ and $p(z_1 \mid x_2)=0.3$.
- Given the observation z_1 we update the belief using Bayes rule.

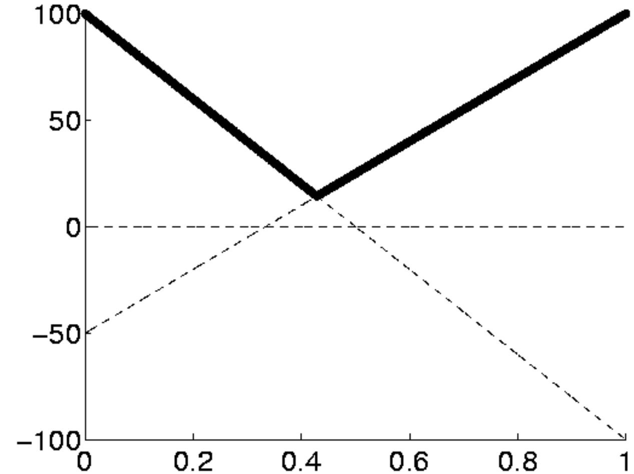
$$p'_1 = \frac{0.7\,p_1}{p(z_1)}$$

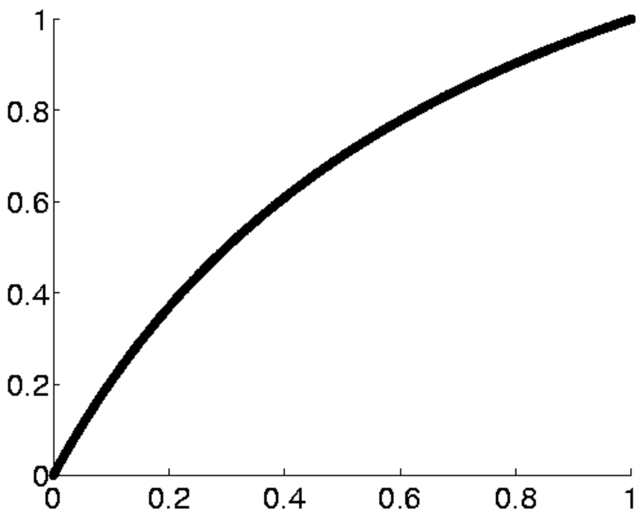
$$p'_2 = \frac{0.3(1-p_1)}{p(z_1)}$$

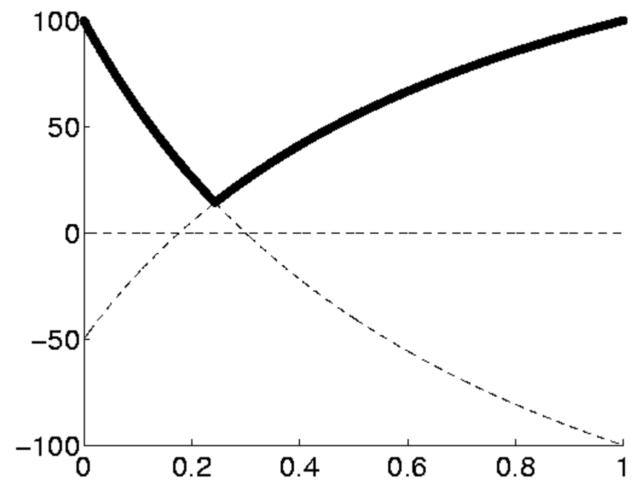
$$p(z_1) = 0.7\,p_1 + 0.3(1-p_1) = 0.4\,p_1 + 0.3$$

Value Function

$V_1(b)$







$b'(b|z_1)$

$V_1(b|z_1)$

Increasing the Time Horizon

- Assume the robot can make an observation before deciding on an action.
- Suppose the robot perceives z_1 for which $p(z_1 \mid x_1)=0.7$ and $p(z_1 \mid x_2)=0.3$.
- Given the observation z_1 we update the belief using Bayes rule. Thus $V_1(b \mid z_1)$ is given by

$$
\begin{aligned}
V_1(b \mid z_1) &= \max \left\{ \begin{array}{ll} -100 \cdot \frac{0.7\, p_1}{p(z_1)} & +100 \cdot \frac{0.3\,(1-p_1)}{p(z_1)} \\[2ex] 100 \cdot \frac{0.7\, p_1}{p(z_1)} & -50 \cdot \frac{0.3\,(1-p_1)}{p(z_1)} \end{array} \right\} \\
&= \frac{1}{p(z_1)} \max \left\{ \begin{array}{ll} -70\, p_1 & +30\,(1-p_1) \\ 70\, p_1 & -15\,(1-p_1) \end{array} \right\}
\end{aligned}
$$

Expected Value after Measuring

- Since we do not know in advance what the next measurement will be, we have to compute the expected belief

$$
\begin{aligned}
\bar{V}_1(b) = E_z[V_1(b \mid z)] &= \sum_{i=1}^{2} p(z_i) V_1(b \mid z_i) \\
&= \sum_{i=1}^{2} p(z_i) V_1\left(\frac{p(z_i \mid x_1)\, p_1}{p(z_i)} \right) \\
&= \sum_{i=1}^{2} V_1\left(p(z_i \mid x_1)\, p_1 \right)
\end{aligned}
$$

Expected Value after Measuring

- Since we do not know in advance what the next measurement will be, we have to compute the expected belief

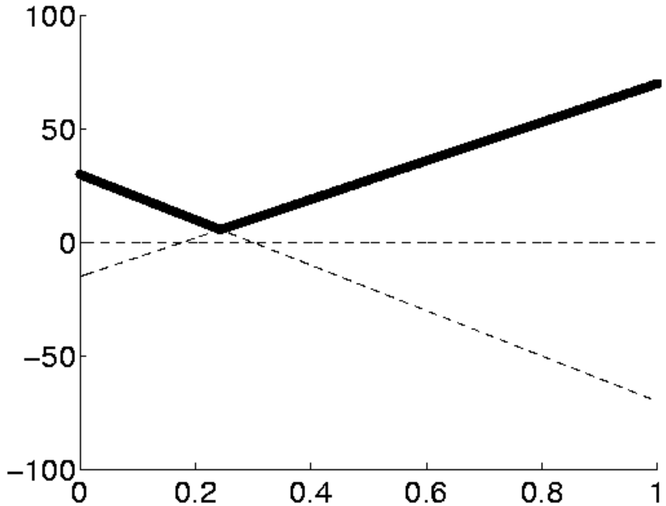
$$
\begin{aligned}
\bar{V}_1(b) &= E_z[V_1(b \mid z)] \\
&= \sum_{i=1}^{2} p(z_i)\, V_1(b \mid z_i) \\
&= \max\left\{ \begin{array}{ll} -70\, p_1 & +30\,(1-p_1) \\ 70\, p_1 & -15\,(1-p_1) \end{array} \right\} \\
&\quad + \max\left\{ \begin{array}{ll} -30\, p_1 & +70\,(1-p_1) \\ 30\, p_1 & -35\,(1-p_1)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

Resulting Value Function

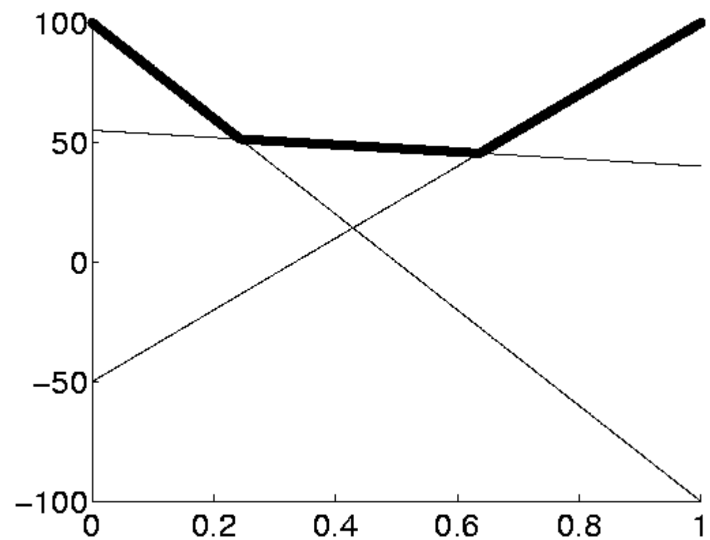
- The four possible combinations yield the following function which then can be simplified and pruned.

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

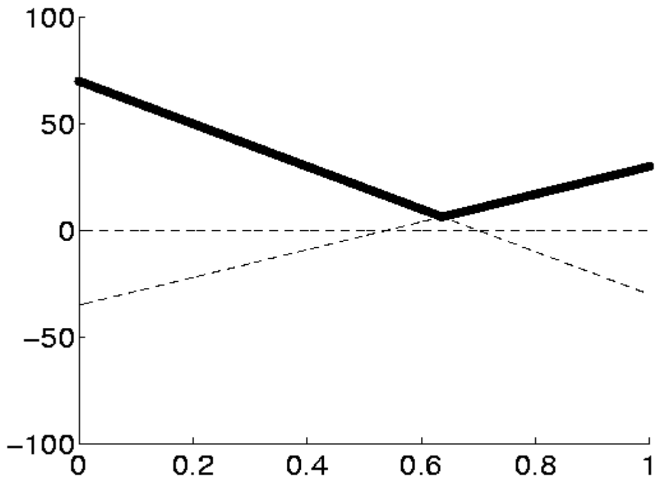
Value Function



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



$\bar{V}_1(b)$



$p(z_2)\ V_1(b|z_2)$

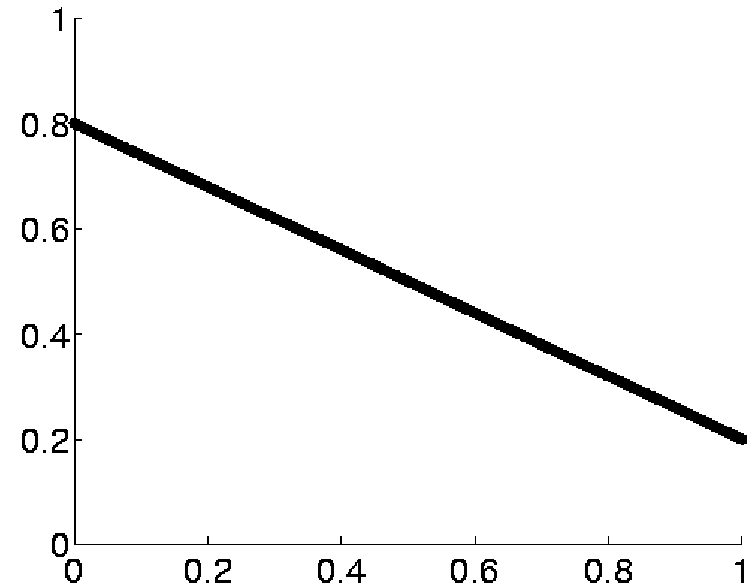
State Transitions (Prediction)

- When the agent selects u_3 its state potentially changes.
- When computing the value function, we have to take these potential state changes into account.

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

State Transitions (Prediction)

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



Value Function after Executing u_3

- Take the state transition into account, we finally get

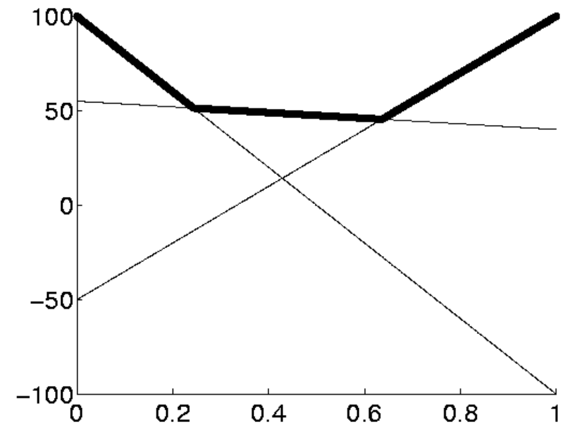
$$\bar{V}_1(b) = \max \left\{ \begin{array}{llll} -70\,p_1 & +30\,(1-p_1) & -30\,p_1 & +70\,(1-p_1) \\ -70\,p_1 & +30\,(1-p_1) & +30\,p_1 & -35\,(1-p_1) \\ +70\,p_1 & -15\,(1-p_1) & -30\,p_1 & +70\,(1-p_1) \\ +70\,p_1 & -15\,(1-p_1) & +30\,p_1 & -35\,(1-p_1) \end{array} \right\}$$

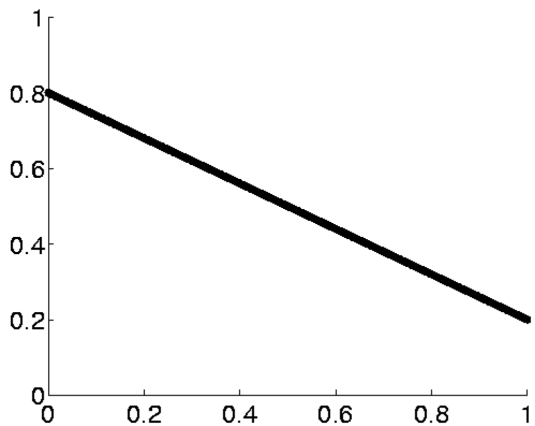
$$= \max \left\{ \begin{array}{ll} -100\,p_1 & +100\,(1-p_1) \\ +40\,p_1 & +55\,(1-p_1) \\ +100\,p_1 & -50\,(1-p_1) \end{array} \right\}$$

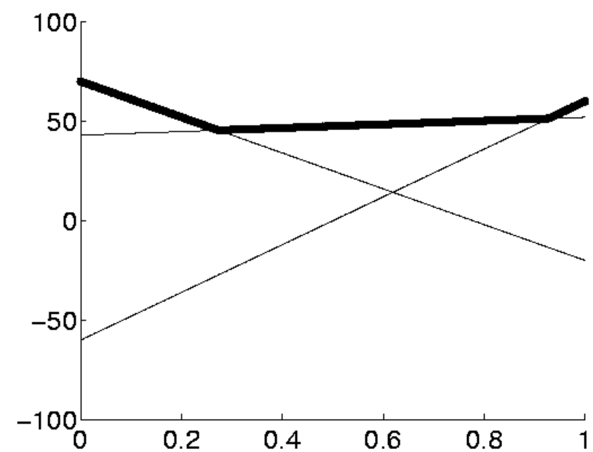
$$\bar{V}_1(b \mid u_3) = \max \left\{ \begin{array}{ll} 60\,p_1 & -60\,(1-p_1) \\ 52\,p_1 & +43\,(1-p_1) \\ -20\,p_1 & +70\,(1-p_1) \end{array} \right\}$$

Value Function after Executing u_3

$$\bar{V}_1(b)$$







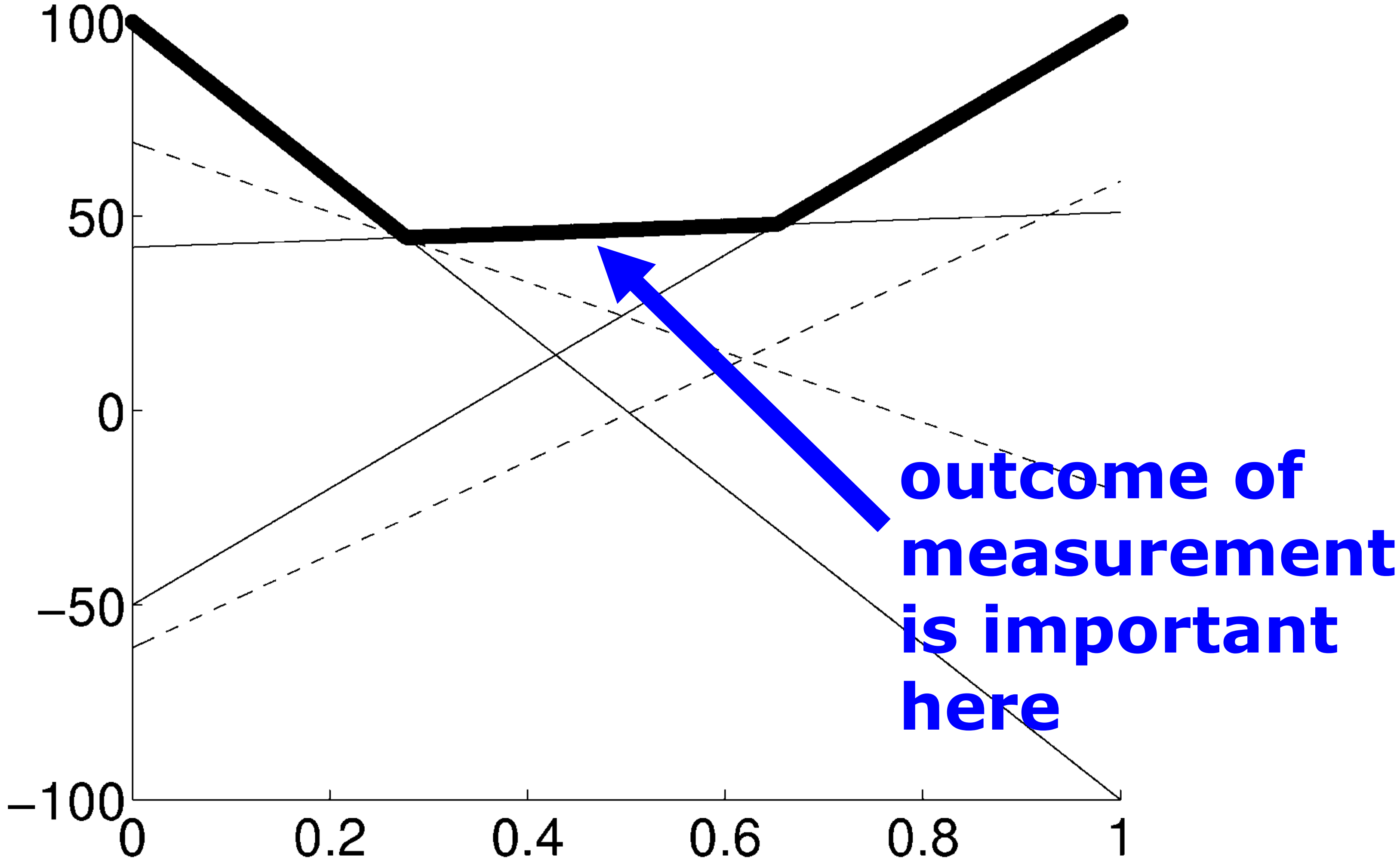
$$\bar{V}_1(b \mid u_3)$$

Value Function for T=2

- Taking into account that the agent can either directly perform u1 or u2
- or first u3 and then u1 or u2, we obtain (after pruning)

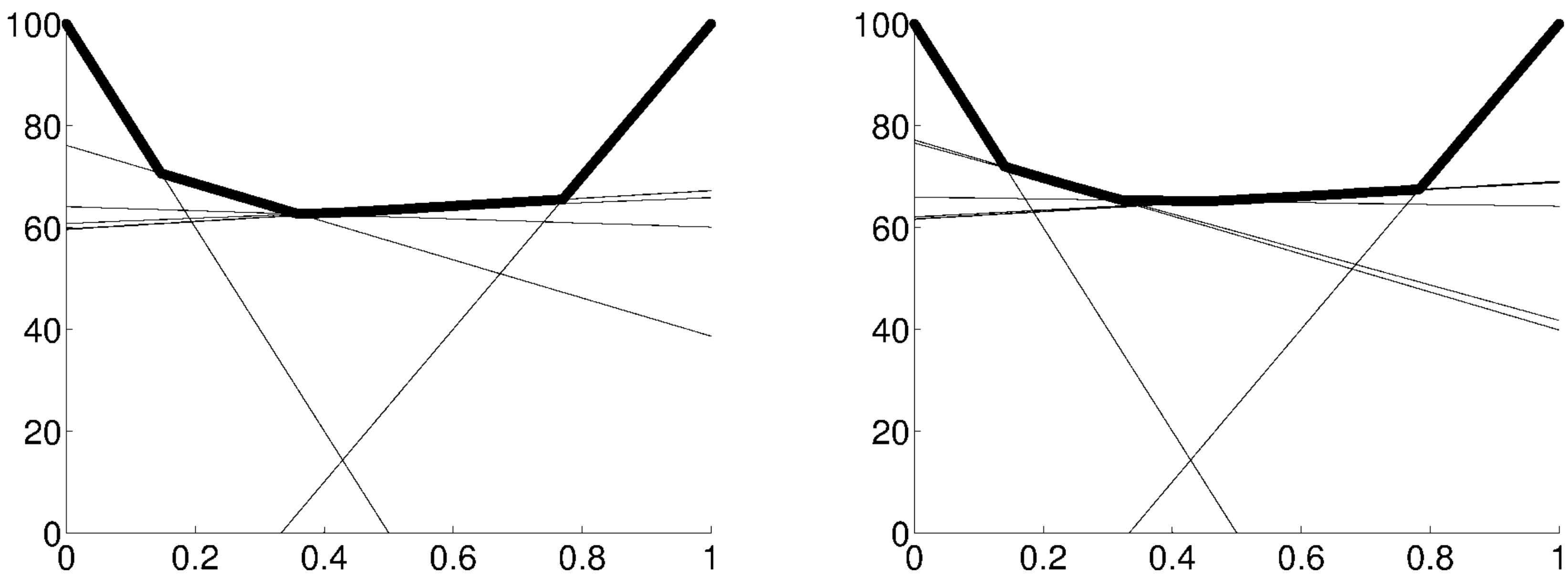
$$\bar{V}_2(b) = \max \left\{ \begin{array}{rr} -100\, p_1 & +100\,(1-p_1) \\ 100\, p_1 & -50\,(1-p_1) \\ 51\, p_1 & +42\,(1-p_1) \end{array} \right\}$$

Value Function for T=2

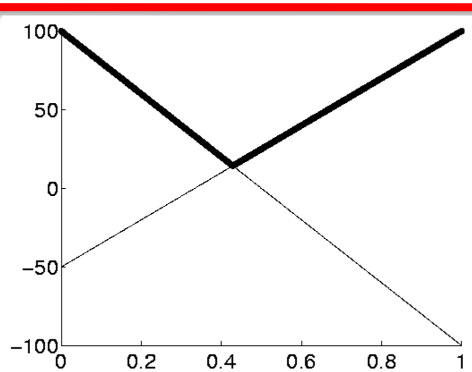


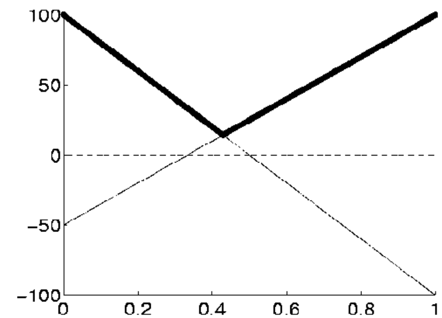
Deep Horizons and Pruning

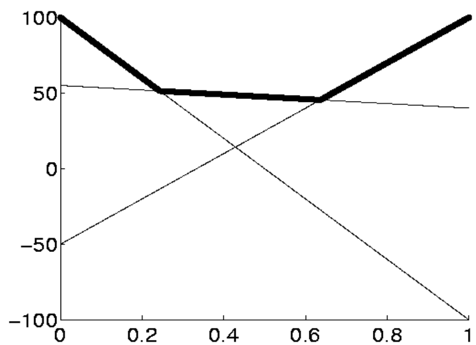
- We have now completed a full backup in belief space
- This process can be applied recursively
  - ✓ The value functions for T=10 and T=20 are

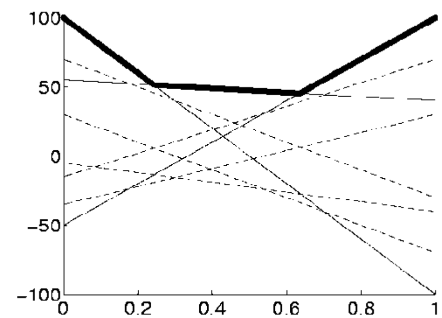


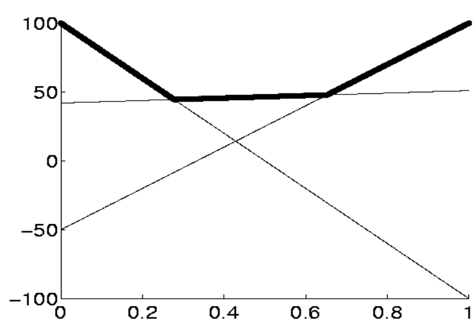
Deep Horizons and Pruning

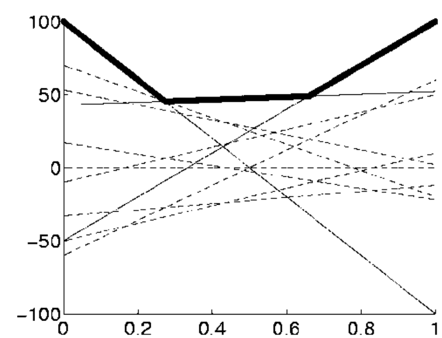












Algorithm POMDP(T):

$\Upsilon = (0; 0, \ldots, 0)$
for $\tau = 1$ *to* T *do*
  $\Upsilon' = \emptyset$
  for all $(u'; v_1^k, \ldots, v_N^k)$ *in* Υ *do*
    for all control actions u *do*
      for all measurements z *do*
        for $j = 1$ *to* N *do*
          $$v_{u,z,j}^k = \sum_{i=1}^{N} v_i^k \, p(z \mid x_i) \, p(x_i \mid u, x_j)$$
        endfor
      endfor
    endfor
  endfor
  for all control actions u *do*
    for all $k(1), \ldots, k(M) = (1, \ldots, 1)$ *to* $(|\Upsilon|, \ldots, |\Upsilon|)$ *do*
      for $i = 1$ *to* N *do*
        $$v_i' = \gamma \left[r(x_i, u) + \sum_z v_{u,z,i}^{k(z)} \right]$$
      endfor
      add $(u; v_1', \ldots, v_N')$ *to* Υ'
    endfor
  endfor
  optional: prune Υ'
  $\Upsilon = \Upsilon'$
endfor
return Υ

Algorithm policy_POMDP(Υ, $b = (p_1, \ldots, p_N)$):

$$\hat{u} = \underset{(u;v_1^k,\ldots,v_N^k)\in\Upsilon}{\operatorname{argmax}} \sum_{i=1}^{N} v_i^k \, p_i$$

return $\hat{u}$

Value Function Representation

$$V(b) \quad = \quad \sum_{i=1}^{N} v_i \, p_i$$

Piecewise linear and convex:

$$V(b) \quad = \quad \max_k \sum_{i=1}^{N} v_i^k \, p_i$$

Value Iteration Backup

- Backup in belief space:

$$V_T(b,u) \quad = \quad \gamma \left[r(b,u) + \sum_z V_{T-1}(B(b,u,z))\; p(z \mid u,b) \right]$$

$$V_T(b) \quad = \quad \max_u V_T(b,u)$$

- Belief update is a function:

$$B(b,u,z)(x') \quad = \quad \frac{1}{p(z \mid u,b)}\; p(z \mid x') \sum_x p(x' \mid u,x)\; b(x)$$

$$p'_j \quad = \quad \frac{1}{p(z \mid u,b)}\; p(z \mid x_j) \sum_{i=1}^{N} p(x_j \mid u,x_i)\; p_i$$

Starting at Previous Belief

$$
\begin{aligned}
V_{T-1}(B(b,u,z)) &= \max_k \sum_{j=1}^{N} v_j^k \; p'_j \\
&= \max_k \sum_{j=1}^{N} v_j^k \; \frac{1}{p(z \mid u,b)} \; p(z \mid x_j) \sum_{i=1}^{N} p(x_j \mid u, x_i) \; p_i \\
&= \frac{1}{p(z \mid u,b)} \; \max_k \sum_{j=1}^{N} v_j^k \; p(z \mid x_j) \sum_{i=1}^{N} p(x_j \mid u, x_i) \; p_i \\
&= \frac{1}{p(z \mid u,b)} \; \max_k \overbrace{\sum_{i=1}^{N} p_i \underbrace{\sum_{j=1}^{N} v_j^k \; p(z \mid x_j) \; p(x_j \mid u, x_i)}_{(*)}}^{(**)}
\end{aligned}
$$

*: constant

**: linear function in parameters of belief space

Putting it Back in

$$V_T(b,u) \;=\; \gamma \left[r(b,u) + \sum_z \max_k \sum_{i=1}^N p_i \sum_{j=1}^N v_j^k \, p(z \mid x_j) \, p(x_j \mid u, x_i) \right]$$

$$r(b,u) \;=\; E_x[r(x,u)] \;=\; \sum_{i=1}^N p_i \, r(x_i,u)$$

Maximization over Actions

$$
\begin{aligned}
V_{T-1}(B(b,u,z)) &= \max_k \sum_{j=1}^{N} v_j^k \; p'_j \\
&= \max_k \sum_{j=1}^{N} v_j^k \; \frac{1}{p(z \mid u,b)} \; p(z \mid x_j) \sum_{i=1}^{N} p(x_j \mid u, x_i) \; p_i \\
&= \frac{1}{p(z \mid u,b)} \; \max_k \sum_{j=1}^{N} v_j^k \; p(z \mid x_j) \sum_{i=1}^{N} p(x_j \mid u, x_i) \; p_i \\
&= \frac{1}{p(z \mid u,b)} \; \max_k \overbrace{\sum_{i=1}^{N} p_i \underbrace{\sum_{j=1}^{N} v_j^k \; p(z \mid x_j) \; p(x_j \mid u, x_i)}_{(*)}}^{(**)}
\end{aligned}
$$

$$
V_T(b,u) = \gamma \left[r(b,u) + \sum_z \max_k \sum_{i=1}^{N} p_i \sum_{j=1}^{N} v_j^k \; p(z \mid x_j) \; p(x_j \mid u, x_i) \right]
$$

Maximization over Actions

$$
\begin{aligned}
V_T(b) &= \max_u V_T(b,u) \\
&= \gamma \max_u \left(\left[\sum_{i=1}^{N} p_i \, r(x_i,u) \right] + \sum_z \max_k \right. \\
&\qquad \left. \sum_{i=1}^{N} p_i \underbrace{\sum_{j=1}^{N} v_j^k \, p(z \mid x_j) \, p(x_j \mid u, x_i)}_{=: \; v_{u,z,i}^k} \right) \\
&= \gamma \max_u \left(\left[\sum_{i=1}^{N} p_i \, r(x_i,u) \right] + \underbrace{\sum_z \max_k \sum_{i=1}^{N} p_i \, v_{u,z,i}^k}_{(*)} \right)
\end{aligned}
$$

$$
v_{u,z,i}^k = \sum_{j=1}^{N} v_j^k \, p(z \mid x_j) \, p(x_j \mid u, x_i)
$$

Max-Sum

$$\max_i \max_j \left[a_i(x) + b_j(x)\right]$$

$$\sum_{j=1}^{m} \max_{i=1}^{N} a_{i,j}(x) = \max_{i(1)=1}^{N} \max_{i(2)=1}^{N} \cdots \max_{i(m)=1}^{N} \sum_{j=1}^{m} a_{i(j),j}$$

$$\sum_{z} \max_{k} \sum_{i=1}^{N} p_i \, v_{u,z,i}^{k} = \max_{k(1)} \max_{k(2)} \cdots \max_{k(M)} \sum_{z} \sum_{i=1}^{N} p_i \, v_{u,z,i}^{k(z)}$$

$$= \max_{k(1)} \max_{k(2)} \cdots \max_{k(M)} \sum_{i=1}^{N} p_i \sum_{z} v_{u,z,i}^{k(z)}$$

Final Result

$$
\begin{aligned}
V_T(b) &= \gamma \max_u \left[\sum_{i=1}^{N} p_i \; r(x_i, u) \right] + \max_{k(1)} \max_{k(2)} \cdots \max_{k(M)} \sum_{i=1}^{N} p_i \sum_z v_{u,z,i}^{k(z)} \\
&= \gamma \max_u \max_{k(1)} \max_{k(2)} \cdots \max_{k(M)} \sum_{i=1}^{N} p_i \left[r(x_i, u) + \sum_z v_{u,z,i}^{k(z)} \right]
\end{aligned}
$$

Individual constraints:

$$
\left(\left[r(x_1, u) + \sum_z v_{u,z,1}^{k(z)} \right] \left[r(x_2, u) + \sum_z v_{u,z,2}^{k(z)} \right] \cdots \left[r(x_N, u) + \sum_z v_{u,z,N}^{k(z)} \right] \right)
$$

Why Pruning is Essential

- Each **update introduces additional linear components** to V.
- Each **measurement squares the number of linear components**.
- Thus, an un-pruned value function
  - ✓ at T=20 includes more than $10^{547,864}$ linear functions.
  - ✓ at T=30 includes $10^{561,012,337}$ 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 **POMDPs are impractical for most applications**.

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.
- POMDPs so far have only been applied successfully to very small state spaces with small numbers of possible observations and actions.

Outlines

- Markov Decision Process (MDP)
- Value Iteration and Policy Iteration
- Partially Observable MDP (POMDP)
- POMDP Observation and Prediction
- POMDP Approximation

POMDP Approximations

- Point-based value iteration
- QMDPs
- AMDPs
- MCMDPs

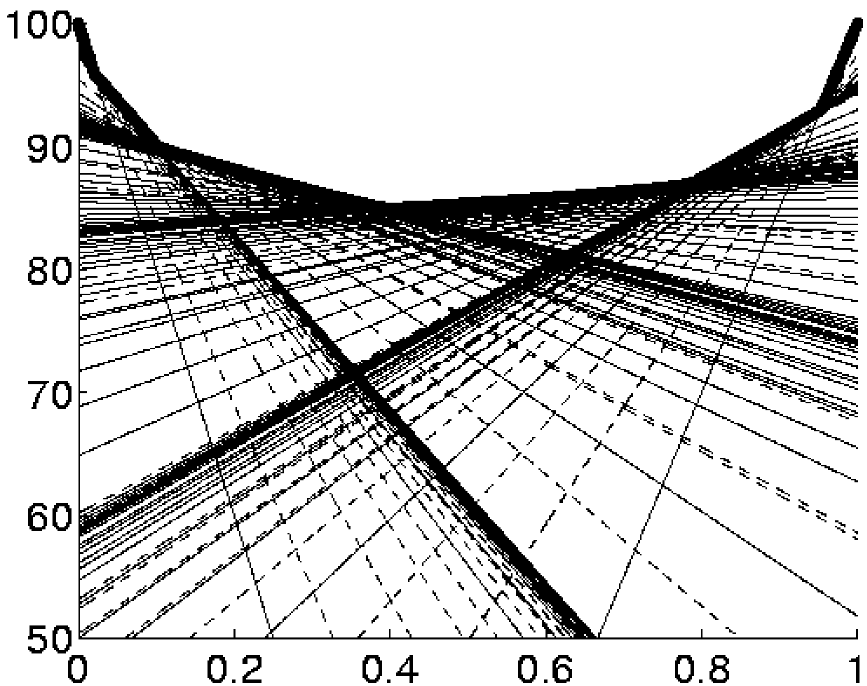
Point-based Value Iteration

- Maintains a set of example beliefs

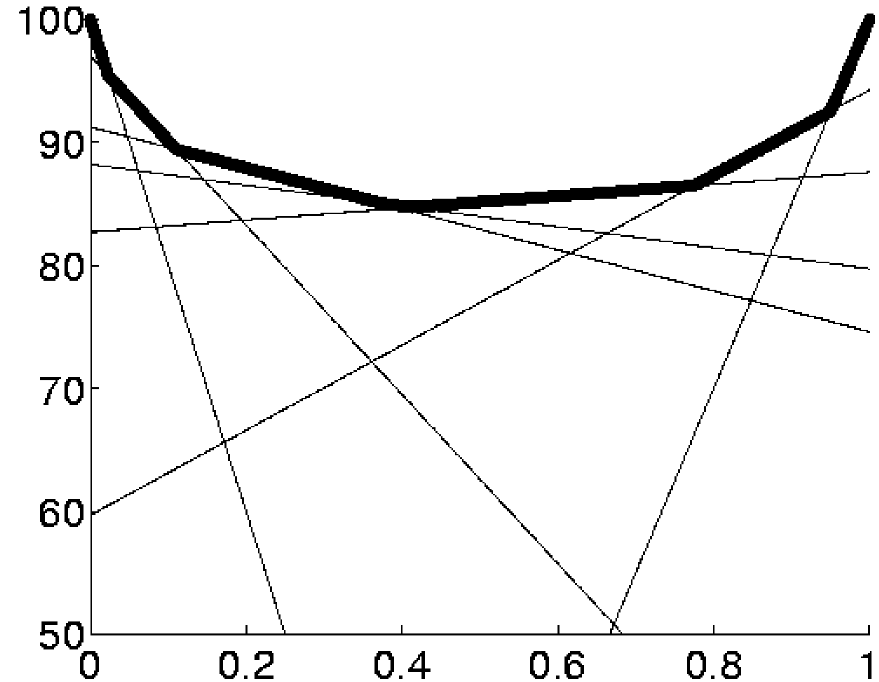
- Only considers constraints that maximize value function for at least one of the examples

Point-based Value Iteration

- Value functions for T=30

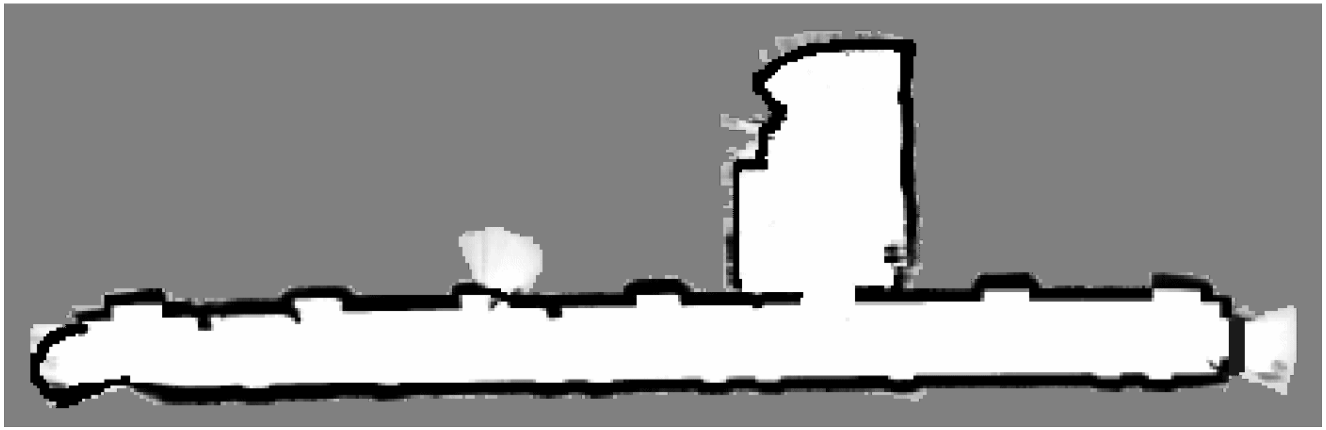


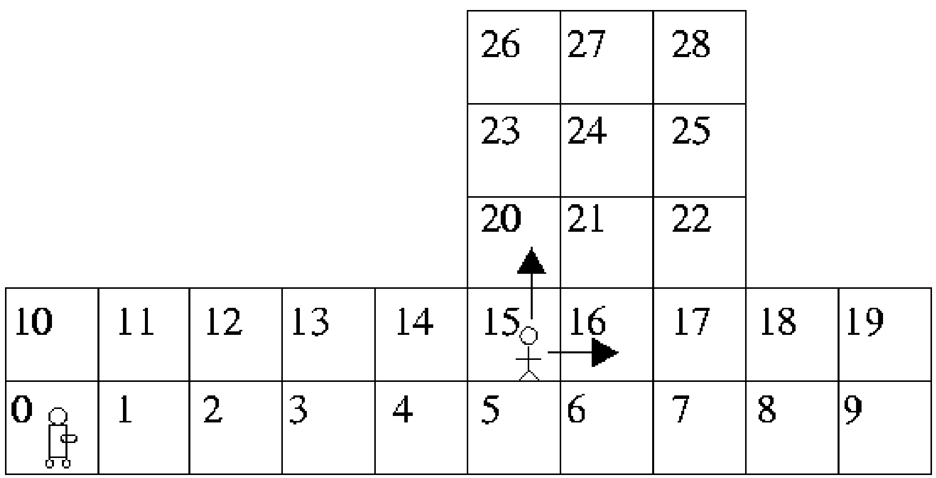
Exact value function



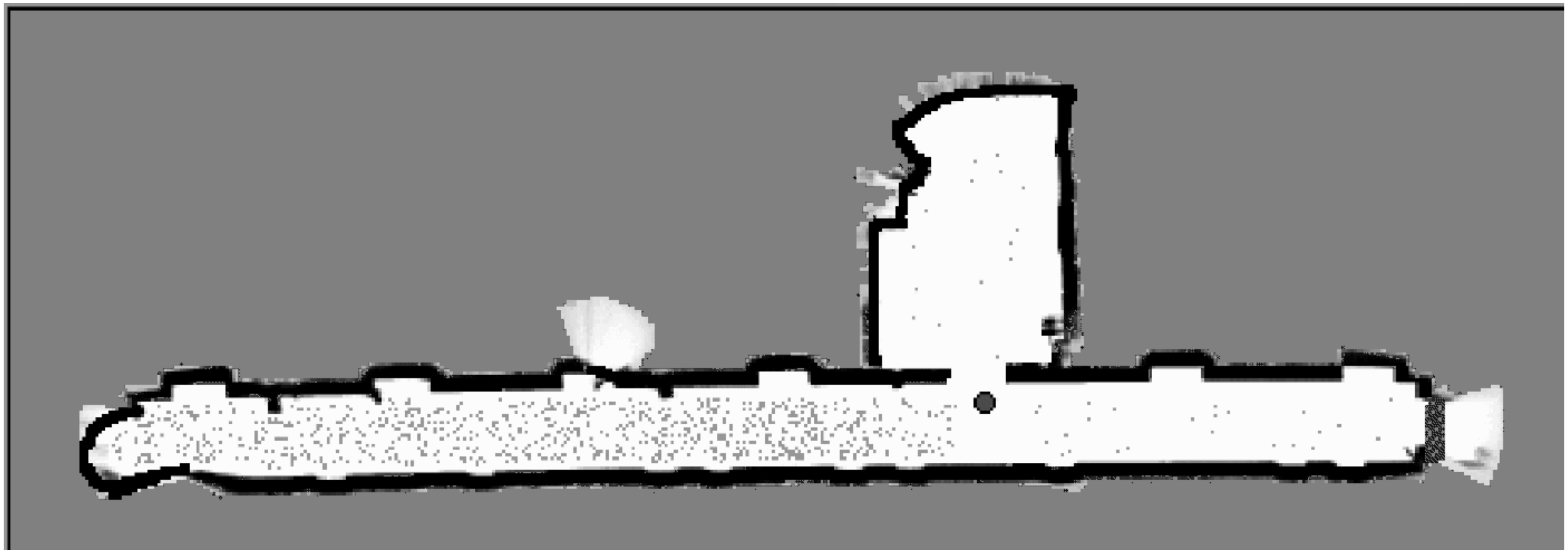
PBVI

Example Application





Example Application



QMDPs

- QMDPs only consider state uncertainty in the first step

- After that, the world becomes fully observable.

QMDP Implementation

Algorithm QMDP($b = (p_1, \ldots, p_N)$)**:**

$\hat{V}$ = MDP_discrete_value_iteration() // *see page 504*

for all control actions u *do*

$$Q(x_i, u) = r(x_i, u) + \sum_{j=1}^{N} \hat{V}(x_j)\, p(x_j \mid u, x_i)$$

endfor

return $\displaystyle \operatorname*{arg\,max}_u \sum_{i=1}^{N} p_i\, Q(x_i, u)$

Augmented POMDP

- Augmentation adds uncertainty component to state space, e.g.,

$$\bar{b} = \begin{pmatrix} \arg\max_x b(x) \\ H_b(x) \end{pmatrix}, \qquad H_b(x) = -\int b(x) \log b(x) dx$$

- Planning is performed by MDP in augmented state space
- Transition, observation and reward models have to be learned

Algorithm AMDP_value_iteration( ):

for all $\bar{b}$ *do* — *// learn model*

for all u *do*

for all $\bar{b}$ *do* — *// initialize model*

$\hat{\mathcal{P}}(\bar{b}, u, \bar{b}') = 0$

endfor

$\hat{\mathcal{R}}(\bar{b}, u) = 0$

repeat n *times* — *// learn model*

generate b *with* $f(b) = \bar{b}$

sample $x \sim b(x)$ — *// belief sampling*

sample $x' \sim p(x' \mid u, x)$ — *// motion model*

sample $z \sim p(z \mid x')$ — *// measurement model*

calculate $b' = B(b, u, z)$ — *// Bayes filter*

calculate $\bar{b}' = f(b')$ — *// belief state statistic*

$\hat{\mathcal{P}}(\bar{b}, u, \bar{b}') = \hat{\mathcal{P}}(\bar{b}, u, \bar{b}') + \frac{1}{n}$ — *// learn transitions prob's*

$\hat{\mathcal{R}}(\bar{b}, u) = \hat{\mathcal{R}}(\bar{b}, u) + \frac{r(u,s)}{n}$ — *// learn payoff model*

endrepeat

endfor

endfor

for all $\bar{b}$ — *// initialize value function*

$\hat{V}(\bar{b}) = r_{\min}$

endfor

repeat until convergence — *// value iteration*

for all $\bar{b}$ *do*

$$\hat{V}(\bar{b}) = \gamma \max_u \left[\hat{\mathcal{R}}(u, \bar{b}) + \sum_{\bar{b}'} \hat{V}(\bar{b}') \, \hat{\mathcal{P}}(\bar{b}, u, \bar{b}') \right]$$

endfor

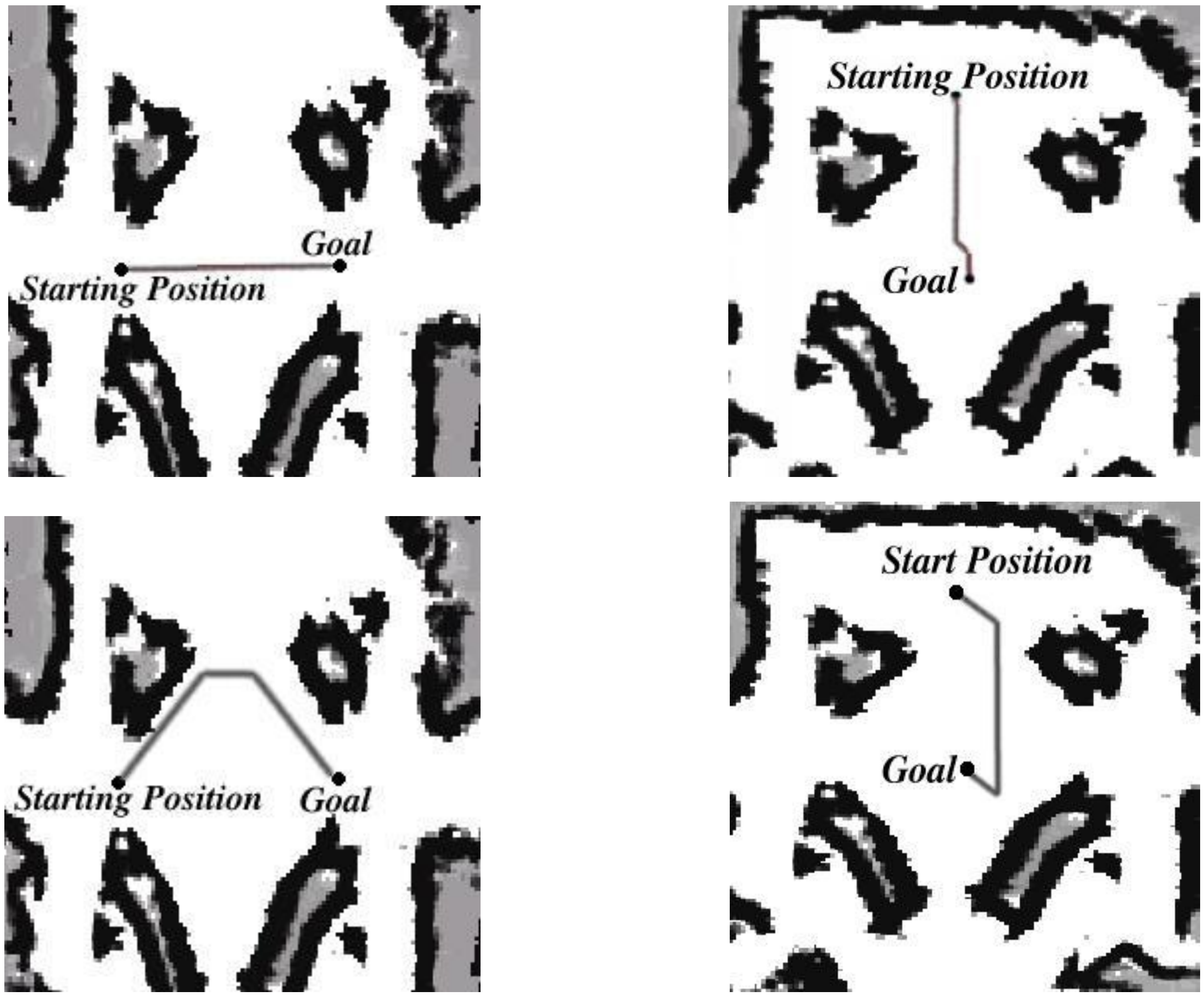
return $\hat{V}, \hat{\mathcal{P}}, \hat{\mathcal{R}}$ — *// return value fct & model*

Algorithm policy_AMDP($\hat{V}$, $\hat{\mathcal{P}}$, $\hat{\mathcal{R}}$, b):

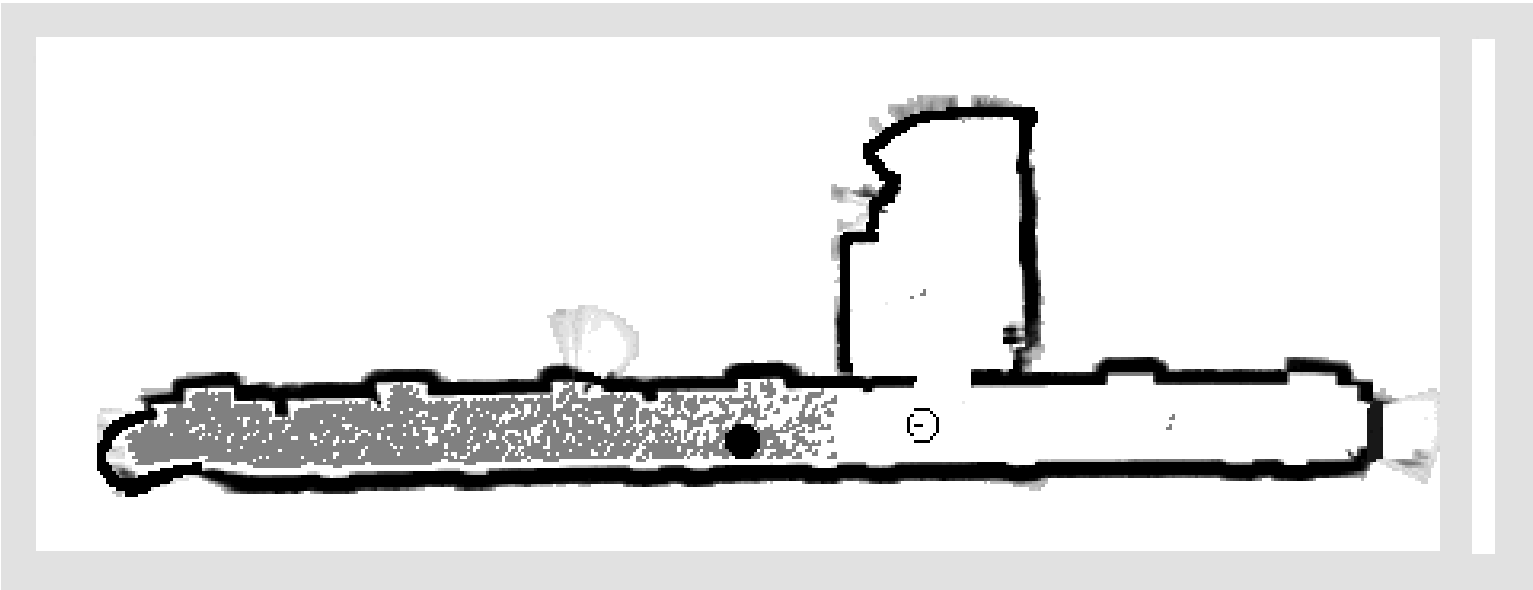
$$\bar{b} = f(b)$$

$$\textit{return} \arg\max_u \left[\hat{\mathcal{R}}(u, \bar{b}) + \sum_{\bar{b}'} \hat{V}(\bar{b}')\ \hat{\mathcal{P}}(\bar{b}, u, \bar{b}') \right]$$

Navigation Example



Dimensionality Reduction on Beliefs



Monte Carlo Method

- Represent beliefs by samples
- Estimate value function on sample sets
- Simulate control and observation transitions between beliefs

Summary

- Markov Decision Process (MDP)
- Value Iteration and Policy Iteration
- Partially Observable MDP (POMDP)
- POMDP Observation and Prediction
- POMDP Approximation