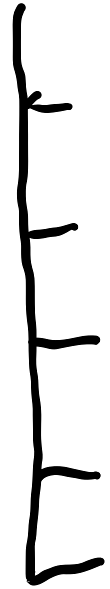
Recap

Recap

DMU



Recap

DMU

- Probabilistic Models

Recap

DMU
- Probabilistic Models
- MDPs
- Reinforcement Learning
-
-

Recap

DMU

- Probabilistic Models
- MDPs
- Reinforcement Learning
- POMDPs
- Games

Probabilistic Models

Probabilistic Models

3 Rules

$P(A)$

$P(A,B)$

$P(A|B)$

Probabilistic Models

$P(A)$

$P(A,B)$

$P(A|B)$

3 Rules

1. $0 \leq P(X \mid Y) \leq 1$

   $\sum_{x \in X} P(x \mid Y) = 1$

Probabilistic Models

$P(A)$

$P(A, B)$

$P(A \mid B)$

3 Rules

1. $0 \le P(X \mid Y) \le 1$

   $\sum_{x \in X} P(x \mid Y) = 1$

2. $P(X) = \sum_{y \in Y} P(X, y)$

Probabilistic Models

$P(A)$
$P(A, B)$
$P(A \mid B)$

3 Rules

1. $0 \leq P(X \mid Y) \leq 1$
   $\sum_{x \in X} P(x \mid Y) = 1$

2. $P(X) = \sum_{y \in Y} P(X, y)$

3. $P(X \mid Y) \ = \frac{P(X,Y)}{P(Y)}$

Probabilistic Models

$P(A)$

$P(A, B)$

$P(A \mid B)$

3 Rules

1. $0 \leq P(X \mid Y) \leq 1$

   $\sum_{x \in X} P(x \mid Y) = 1$

2. $P(X) = \sum_{y \in Y} P(X, y)$

3. $P(X \mid Y) = \frac{P(X,Y)}{P(Y)}$

Bayes Rule

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

Probabilistic Models

P(A)
P(A,B)
P(A|B)

3 Rules

1. $0 \leq P(X \mid Y) \leq 1$
   $\sum_{x \in X} P(x \mid Y) = 1$

2. $P(X) = \sum_{y \in Y} P(X, y)$

3. $P(X \mid Y) = \frac{P(X,Y)}{P(Y)}$

Bayes Rule

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

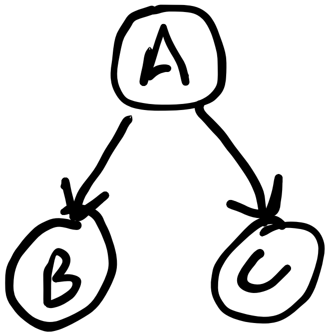
Independence

$$A \perp B \iff P(A, B) = P(A)P(B)$$

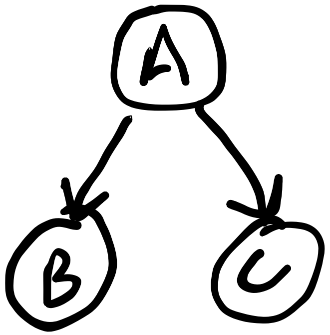
$$A \perp B \mid C \iff P(A, B \mid C) = P(A \mid C)P(B \mid C)$$

Bayesian Networks

Bayesian Networks

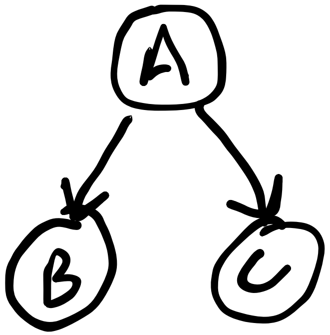


Bayesian Networks



$$P(X_i \mid X_{1:n \setminus i}) \stackrel{?}{=} P(X_i \mid Pa(X_i))$$

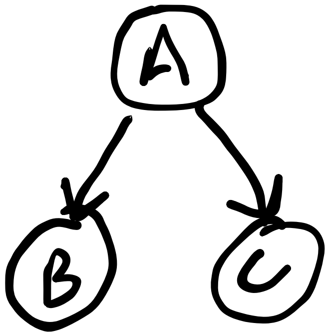
Bayesian Networks



$$\cancel{P(X_i \mid X_{1:n\setminus i}) \stackrel{?}{=} P(X_i \mid Pa(X_i))}$$

No!

Bayesian Networks



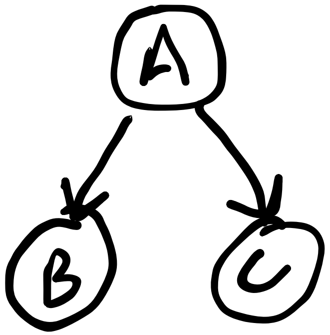
$$\cancel{P(X_i \mid X_{1:n\setminus i}) \stackrel{?}{=} P(X_i \mid Pa(X_i))}$$

No!

Chain Rule

$$P(X_{1:n}) = \prod_i P(X_i \mid Pa(X_i))$$

Bayesian Networks



$$\cancel{P(X_i \mid X_{1:n \setminus i}) \stackrel{?}{=} P(X_i \mid Pa(X_i))}$$

No!

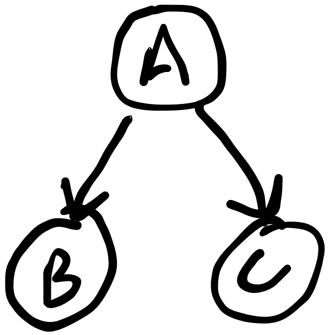
Chain Rule

$$P(X_{1:n}) = \prod_i P(X_i \mid Pa(X_i))$$

Sampling

Topological sort, then sample from each node

Bayesian Networks



$$\cancel{P(X_i \mid X_{1:n\setminus i}) \stackrel{?}{=} P(X_i \mid Pa(X_i))}$$

No!

Sampling

Topological sort, then sample from each node

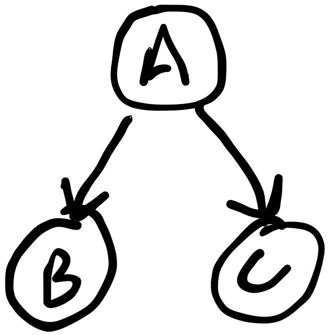
Chain Rule

$$P(X_{1:n}) = \prod_i P(X_i \mid Pa(X_i))$$

Conditional Independence

$X \perp Y \mid \mathcal{C}$ if all paths between X and Y are d-separated by $\mathcal{C}$

Bayesian Networks



$$\cancel{P(X_i \mid X_{1:n\setminus i}) \stackrel{?}{=} P(X_i \mid Pa(X_i))}$$

No!

Sampling

Topological sort, then sample from each node

Chain Rule

$$P(X_{1:n}) = \prod_i P(X_i \mid Pa(X_i))$$

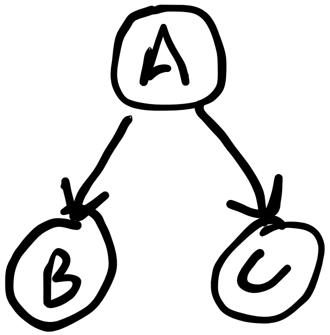
Conditional Independence

$X \perp Y \mid \mathcal{C}$ if all paths between X and Y are d-separated by $\mathcal{C}$

Inference

- Input: BN, evidence values
- Output: Distribution of targets

Bayesian Networks



$$\cancel{P(X_i \mid X_{1:n\setminus i}) \stackrel{?}{=} P(X_i \mid Pa(X_i))}$$

No!

Chain Rule

$$P(X_{1:n}) = \prod_i P(X_i \mid Pa(X_i))$$

Sampling

Topological sort, then sample from each node

Conditional Independence

$X \perp Y \mid \mathcal{C}$ if all paths between X and Y are d-separated by $\mathcal{C}$

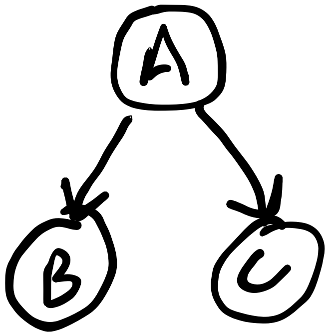
Inference

- Input: BN, evidence values
- Output: Distribution of targets

Exact: NP-Hard

Approximate via sampling: Direct, Likelihood Weighted, Gibbs

Bayesian Networks



$$\cancel{P(X_i \mid X_{1:n\setminus i}) \stackrel{?}{=} P(X_i \mid Pa(X_i))}$$

No!

Chain Rule

$$P(X_{1:n}) = \prod_i P(X_i \mid Pa(X_i))$$

Sampling

Topological sort, then sample from each node

Conditional Independence

$X \perp Y \mid \mathcal{C}$ if all paths between X and Y are d-separated by $\mathcal{C}$

Inference

- Input: BN, evidence values
- Output: Distribution of targets

Exact: NP-Hard

Approximate via sampling: Direct, Likelihood Weighted, Gibbs

Learning

- Input: Data
- Output: BN structure and parameters

Markov Decision Processes

Markov Decision Processes

$$(S, A, R, T, \gamma)$$

Markov Decision Processes

$$(S, A, R, T, \gamma)$$

Examples: $S = \{1, 2, 3\}$ or $S = \mathbb{R}^2$

Markov Decision Processes

$$(S, A, R, T, \gamma)$$

Examples: $S = \{1, 2, 3\}$ or $S = \mathbb{R}^2$

$$s = (x, \dot{x}) \in S = \mathbb{R}^2$$

Markov Decision Processes

$$(S, A, R, T, \gamma)$$

Examples: $S = \{1, 2, 3\}$ or $S = \mathbb{R}^2$

$$s = (x, \dot{x}) \in S = \mathbb{R}^2$$

$$\underset{\pi}{\text{maximize}} \; E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$$

Markov Decision Processes

$$(S, A, R, T, \gamma)$$

Examples: $S = \{1, 2, 3\}$ or $S = \mathbb{R}^2$

$$s = (x, \dot{x}) \in S = \mathbb{R}^2$$

$$\underset{\pi}{\text{maximize}} \; E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$$

$$Q^\pi(s, a) = E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \,\middle|\, s_0 = s, a_0 = a, a_t = \pi(s_t)\right]$$

Markov Decision Processes

$$(S, A, R, T, \gamma)$$

Examples: $S = \{1, 2, 3\}$ or $S = \mathbb{R}^2$

$$s = (x, \dot{x}) \in S = \mathbb{R}^2$$

$$\underset{\pi}{\text{maximize}} \; E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$$

$$Q^\pi(s,a) = E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \,\middle|\, s_0 = s,\, a_0 = a,\, a_t = \pi(s_t)\right]$$

$$V^\pi(s) = \max_a Q^\pi(s,a)$$

Markov Decision Processes

$$(S, A, R, T, \gamma)$$

Examples: $S = \{1, 2, 3\}$ or $S = \mathbb{R}^2$

$$s = (x, \dot{x}) \in S = \mathbb{R}^2$$

$$\underset{\pi}{\text{maximize}} \; E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$$

$$Q^\pi(s,a) = E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \,\middle|\, s_0 = s, a_0 = a, a_t = \pi(s_t)\right]$$

$$V^\pi(s) = \max_a Q^*(s,a)$$

$$V^\pi(s) = R(s,a) + \gamma E[V^\pi(s')]$$

$$V^*(s) = \max_a \{R(s,a) + \gamma E[V^*(s')]\}$$

$$B[V](s) = \max_a \{R(s,a) + \gamma E[V(s')]\}$$

Markov Decision Processes

$$(S, A, R, T, \gamma)$$

Examples: $S = \{1, 2, 3\}$ or $S = \mathbb{R}^2$

$$s = (x, \dot{x}) \in S = \mathbb{R}^2$$

$$\underset{\pi}{\text{maximize}} \; E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$$

$$Q^\pi(s,a) = E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \,\middle|\, s_0 = s, a_0 = a, a_t = \pi(s_t)\right]$$

$$V^\pi(s) = \max_a Q^*(s,a)$$

$$V^\pi(s) = R(s,a) + \gamma E[V^\pi(s')]$$

Policy Evaluation

$$V^*(s) = \max_a \{R(s,a) + \gamma E[V^*(s')]\}$$

$$B[V](s) = \max_a \{R(s,a) + \gamma E[V(s')]\}$$

Markov Decision Processes

$$(S, A, R, T, \gamma)$$

Examples: $S = \{1, 2, 3\}$ or $S = \mathbb{R}^2$

$$s = (x, \dot{x}) \in S = \mathbb{R}^2$$

$$\underset{\pi}{\text{maximize}}\ E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$$

$$Q^\pi(s,a) = E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \,\middle|\, s_0 = s, a_0 = a, a_t = \pi(s_t)\right]$$

$$V^\pi(s) = \max_a Q^\pi(s,a)$$

$$V^\pi(s) = R(s,a) + \gamma E[V^\pi(s')]$$ Policy Evaluation

$$V^*(s) = \max_a \{R(s,a) + \gamma E[V^*(s')]\}$$ Bellman's Equation: Certificate of Optimality

$$B[V](s) = \max_a \{R(s,a) + \gamma E[V(s')]\}$$

Markov Decision Processes

$$(S, A, R, T, \gamma)$$

Examples: $S = \{1, 2, 3\}$ or $S = \mathbb{R}^2$

$$s = (x, \dot{x}) \in S = \mathbb{R}^2$$

$$\underset{\pi}{\text{maximize}}\ E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$$

$$Q^{\pi}(s,a) = E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \,\middle|\, s_0 = s, a_0 = a, a_t = \pi(s_t)\right]$$

$$V^{\pi}(s) = \max_a Q^{*}(s,a)$$

$V^{\pi}(s) = R(s,a) + \gamma E[V^{\pi}(s')]$ Policy Evaluation

$V^{*}(s) = \max_a \{R(s,a) + \gamma E[V^{*}(s')]\}$ Bellman's Equation: Certificate of Optimality

$B[V](s) = \max_a \{R(s,a) + \gamma E[V(s')]\}$ Bellman's Operator

Offline MDP Algorithms

Offline MDP Algorithms

Policy Iteration

loop

    Evaluate Policy

    Improve Policy

Offline MDP Algorithms

Policy Iteration

loop

Evaluate Policy

Improve Policy

Value Iteration

loop

$V \leftarrow B[V]$

Offline MDP Algorithms

Policy Iteration

loop

  Evaluate Policy

  Improve Policy

Converges because policy always improves and there are a finite number of policies

Value Iteration

loop

  $V \leftarrow B[V]$

Offline MDP Algorithms

Policy Iteration

loop

Evaluate Policy

Improve Policy

Converges because policy always improves and there are a finite number of policies

Value Iteration

loop

$V \leftarrow B[V]$

Converges because B is a contraction mapping

Online MDP Planning

Online MDP Planning

Monte Carlo Tree Search

Online MDP Planning

Monte Carlo Tree Search

Search

Expand

Rollout

Backup

Online MDP Planning

Monte Carlo Tree Search

Search

$$Q(s,a) + c\sqrt{\frac{\log N(s)}{N(s,a)}}$$

Expand

Rollout

Backup

Online MDP Planning

Monte Carlo Tree Search

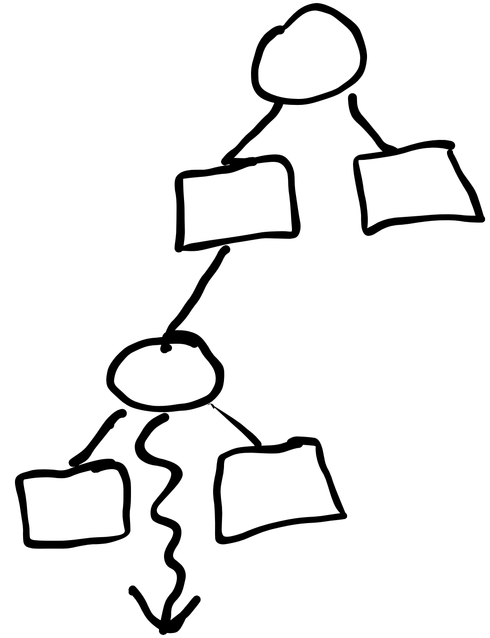
Search

Expand

Rollout

Backup

$$Q(s,a) + c\sqrt{\frac{\log N(s)}{N(s,a)}}$$



Online MDP Planning

Monte Carlo Tree Search

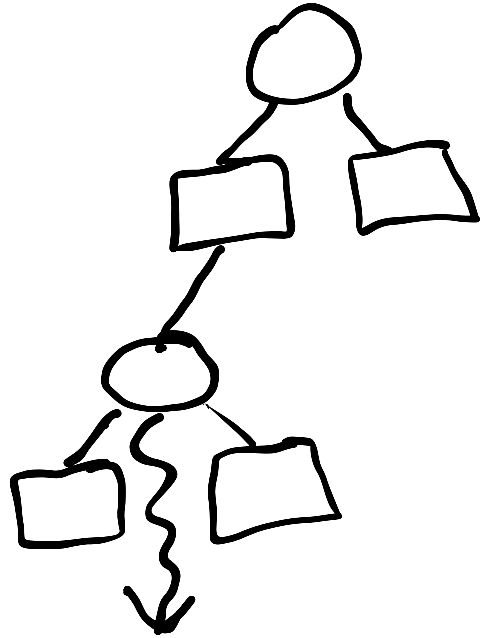
Search

Expand

Rollout

Backup

$$Q(s,a) + c\sqrt{\frac{\log N(s)}{N(s,a)}}$$



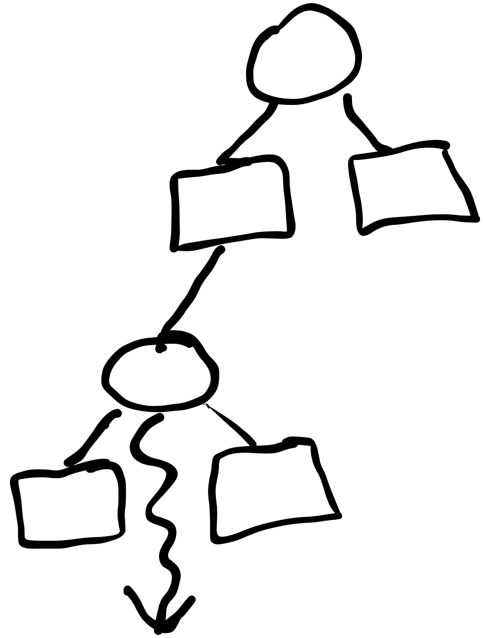
Sparse Sampling

Online MDP Planning

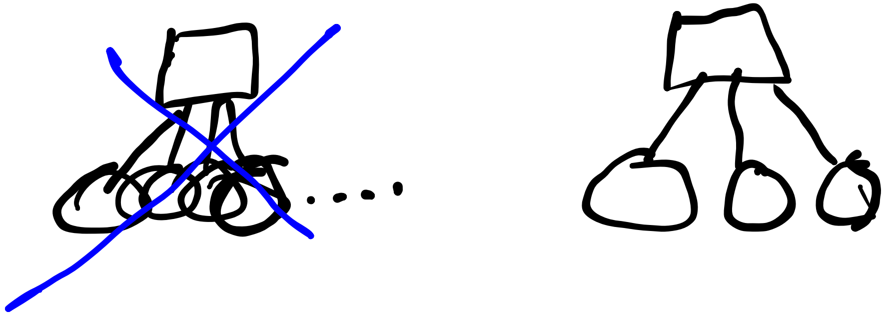
Monte Carlo Tree Search

- Search
- Expand
- Rollout
- Backup

$$Q(s,a) + c\sqrt{\frac{\log N(s)}{N(s,a)}}$$



Sparse Sampling

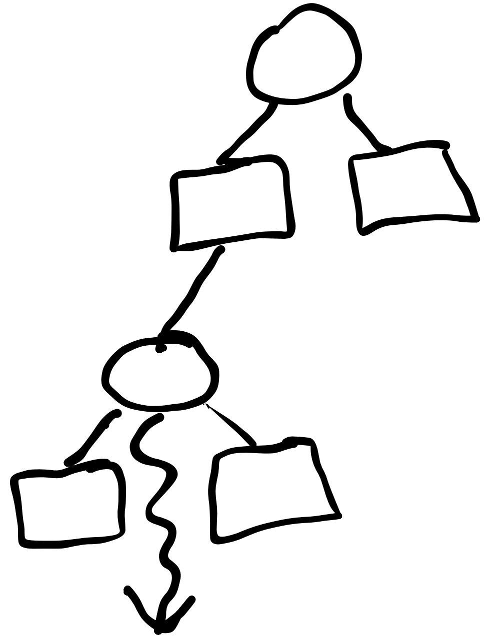


Online MDP Planning

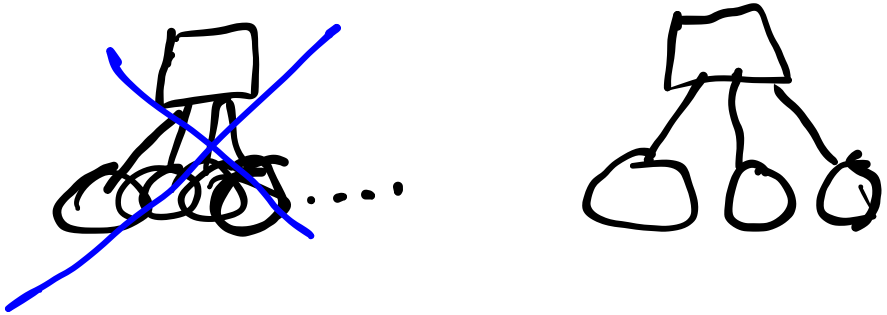
Monte Carlo Tree Search

Search
Expand
Rollout
Backup

$$Q(s,a) + c\sqrt{\frac{\log N(s)}{N(s,a)}}$$

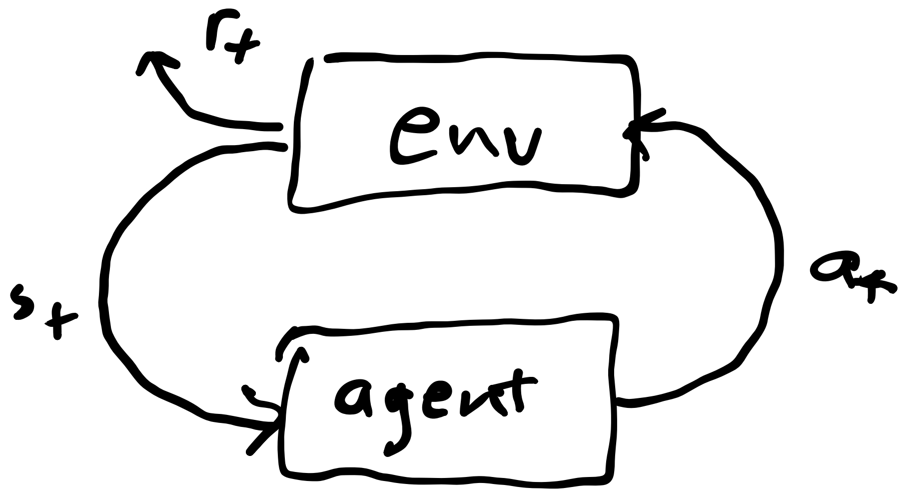


Sparse Sampling



Guarantees *independent* of $|S|$!!

Reinforcement Learning



Challenges:

1. Exploration and Exploitation
2. Credit Assignment
3. Generalization

Exploration

Exploration

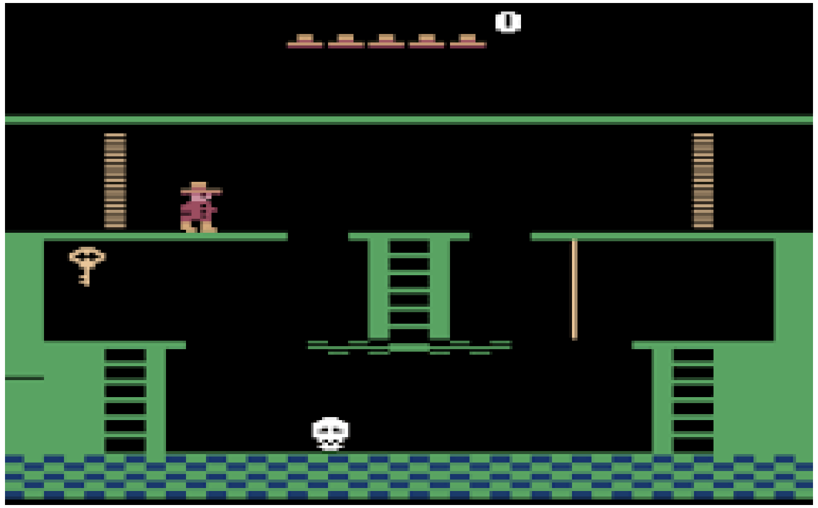
Bandits

- ϵ-greedy
- softmax
- UCB
- Thompson Sampling
- Optimal DP Solution (solving a POMDP!)

Exploration

Bandits

- ϵ-greedy
- softmax
- UCB
- Thompson Sampling
- Optimal DP Solution (solving a POMDP!)

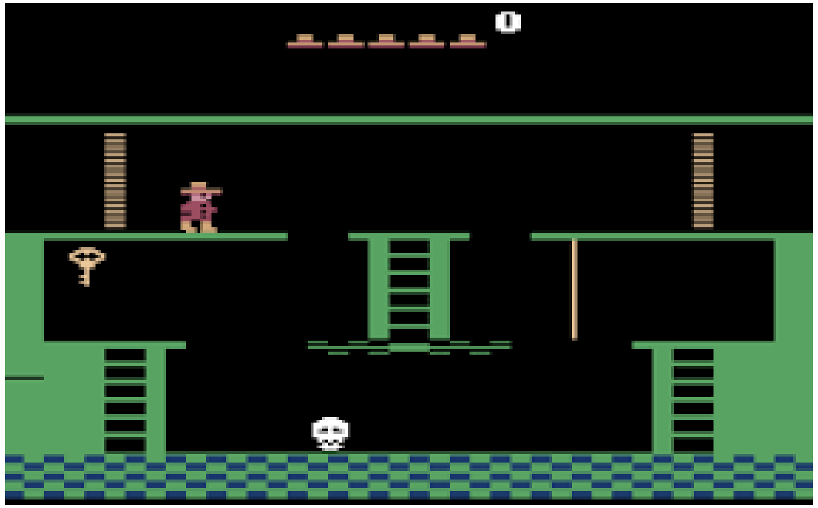


Montezuma's Revenge!

Exploration

Bandits

- ϵ-greedy
- softmax
- UCB
- Thompson Sampling
- Optimal DP Solution (solving a POMDP!)



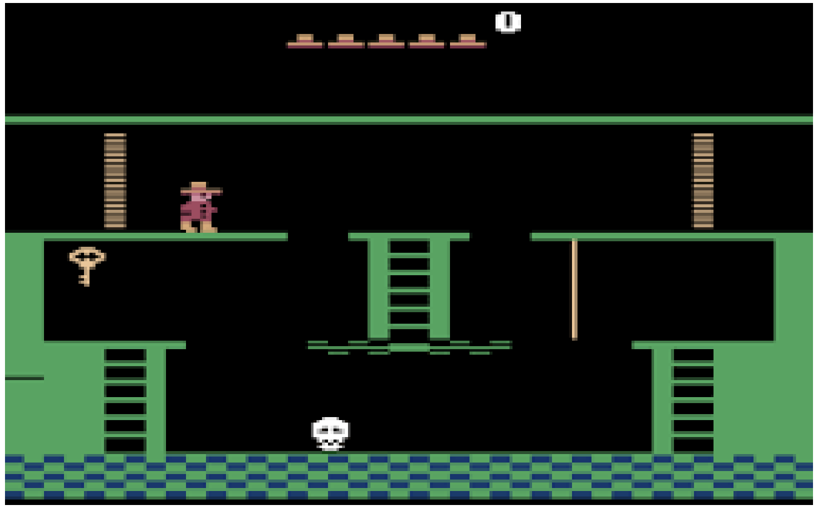
Montezuma's Revenge!

- Pseudocounts

Exploration

Bandits

- ϵ-greedy
- softmax
- UCB
- Thompson Sampling
- Optimal DP Solution (solving a POMDP!)



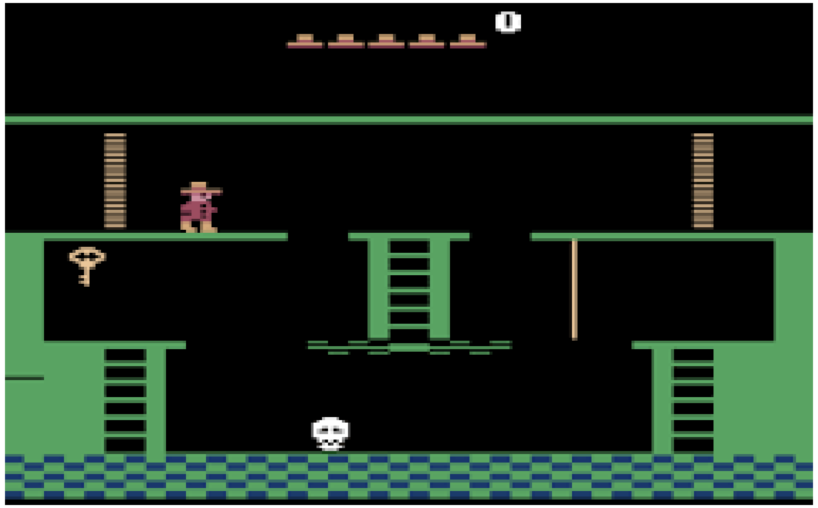
Montezuma's Revenge!

- Pseudocounts
- Curiosity: extra reward for bad prediction

Exploration

Bandits

- ϵ-greedy
- softmax
- UCB
- Thompson Sampling
- Optimal DP Solution (solving a POMDP!)



Montezuma's Revenge!

- Pseudocounts
- Curiosity: extra reward for bad prediction
- Random network distillation

RL Algorithms

RL Algorithms

Model Based

Model Free

RL Algorithms

Model Based

Model Free

On Policy

Off Policy

RL Algorithms

Model Based

Model Free

On Policy

Off Policy

MLMBTRL
(learn T, R)

RL Algorithms

Model Based

Model Free

learn Q

learn π

On Policy

Off Policy

MLMBTRL
(learn T,R)

RL Algorithms

Model Based $\longleftrightarrow$ Model Free

learn Q

learn π

Policy Gradient

On Policy $\updownarrow$ Off Policy

MLMBTRL
(learn T,R)

RL Algorithms

Model Based

Model Free

learn Q

SARSA

learn π

Policy Gradient

On Policy

Off Policy

MLMBTRL
(learn T,R)

RL Algorithms

Model Based

Model Free

learn Q

learn π

On Policy

SARSA

Policy Gradient

MLMBTRL
(learn T,R)

Q-learning

Off Policy

RL Algorithms

Model Based

Model Free

learn Q

SARSA

Q-learning

learn π

Policy Gradient

On Policy

Off Policy

MLMBTRL
(learn T,R)

BAMDP
(POMDP $\tilde{s}=(s,T,R)$)

Policy Gradient

Policy Gradient

- Likelihood ratio trick

Policy Gradient

- Likelihood ratio trick

$$\nabla_\theta p_\theta(\tau) = p_\theta(\tau) \log p_\theta(\tau)$$

Policy Gradient

- Likelihood ratio trick
- Causality

$$\nabla_\theta p_\theta(\tau) = p_\theta(\tau) \log p_\theta(\tau)$$

Policy Gradient

- Likelihood ratio trick
- Causality
- Baseline Subtraction

$$\nabla_\theta p_\theta(\tau) = p_\theta(\tau) \log p_\theta(\tau)$$

Policy Gradient

- Likelihood ratio trick
- Causality
- Baseline Subtraction

$\nabla_\theta p_\theta(\tau) = p_\theta(\tau) \log p_\theta(\tau)$

$$\nabla U(\theta) = \mathbb{E}_\tau\left[\sum_{k=1}^{d} \nabla_\theta \log \pi_\theta(a^{(k)} \mid s^{(k)}) \gamma^{k-1}\left(r_{\text{to-go}}^{(k)} - r_{\text{base}}(s^{(k)})\right)\right]$$

Policy Gradient

- Likelihood ratio trick
- Causality
- Baseline Subtraction

$\nabla_\theta p_\theta(\tau) = p_\theta(\tau) \log p_\theta(\tau)$

$$\nabla U(\theta) = \mathbb{E}_\tau\left[\sum_{k=1}^{d} \nabla_\theta \log \pi_\theta(a^{(k)} \mid s^{(k)}) \gamma^{k-1} \left(r_{\text{to-go}}^{(k)} - r_{\text{base}}(s^{(k)})\right)\right]$$

- Natural Gradient

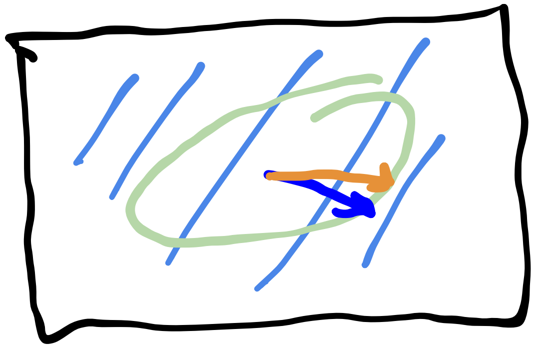
Policy Gradient

- Likelihood ratio trick
- Causality
- Baseline Subtraction

$$\nabla_\theta p_\theta(\tau) = p_\theta(\tau) \log p_\theta(\tau)$$

$$\nabla U(\theta) = \mathbb{E}_\tau \left[\sum_{k=1}^{d} \nabla_\theta \log \pi_\theta(a^{(k)} \mid s^{(k)}) \gamma^{k-1} \left(r_{\text{to-go}}^{(k)} - r_{\text{base}}(s^{(k)}) \right) \right]$$

- Natural Gradient



KL div.
Bound

Q-Learning

Q-Learning

SARSA

$$Q(s,a) \leftarrow Q(s,a) + \alpha(r_t + \gamma Q(s',a') - Q(s,a))$$

Q-Learning

SARSA

$$Q(s,a) \leftarrow Q(s,a) + \alpha(r_t + \gamma Q(s',a') - Q(s,a))$$

Eligibility Traces

Q-Learning

SARSA

$$Q(s,a) \leftarrow Q(s,a) + \alpha(r_t + \gamma Q(s',a') - Q(s,a))$$

Eligibility Traces

Q-learning

$$Q(s,a) \leftarrow Q(s,a) + \alpha(r_t + \gamma \max_{a'} Q(s',a') - Q(s,a))$$

Q-Learning

SARSA

$$Q(s,a) \leftarrow Q(s,a) + \alpha(r_t + \gamma Q(s',a') - Q(s,a))$$

Eligibility Traces

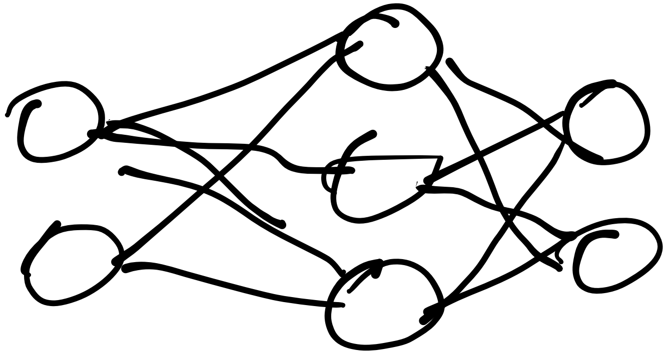
Q-learning

$$Q(s,a) \leftarrow Q(s,a) + \alpha(r_t + \gamma \max_{a'} Q(s',a') - Q(s,a))$$

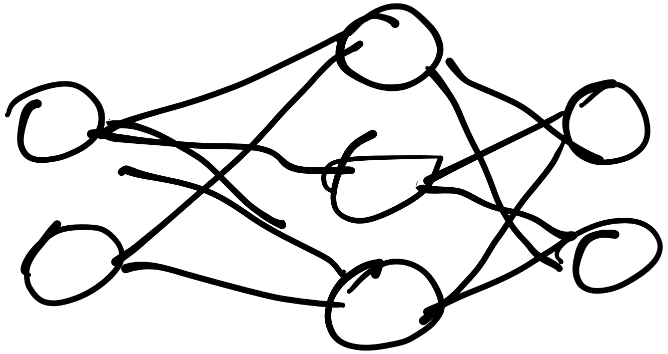
Double Q Learning

Neural Networks and DQN

Neural Networks and DQN

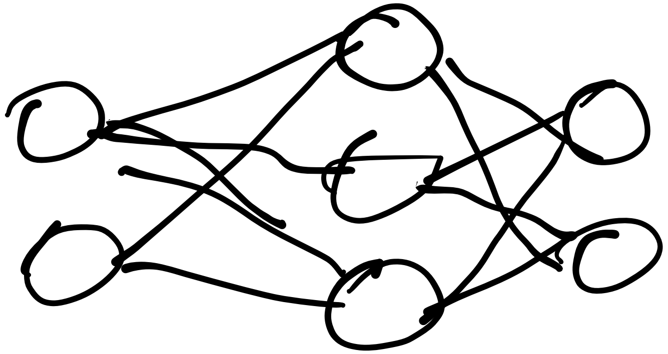


Neural Networks and DQN



$$f_\theta(x) = \sigma(W_2\sigma(W_1 x + b_1) + b_2)$$

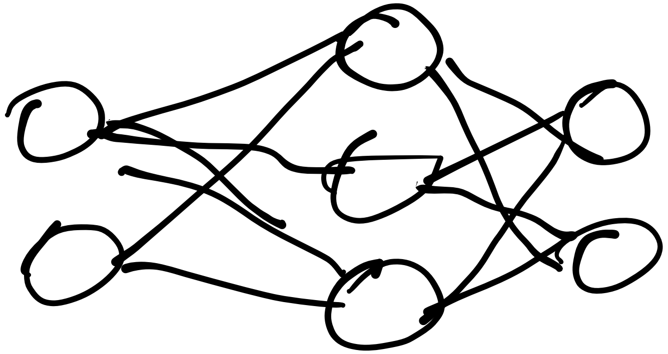
Neural Networks and DQN



$$f_\theta(x) = \sigma(W_2\sigma(W_1 x + b_1) + b_2)$$

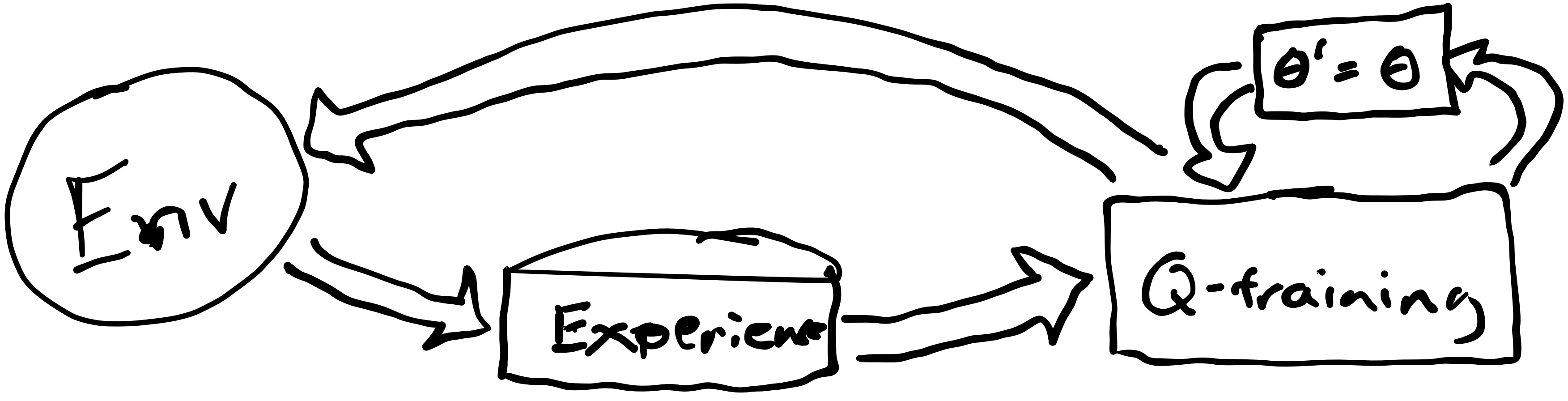
Backprop

Neural Networks and DQN



$$f_\theta(x) = \sigma(W_2\sigma(W_1x + b_1) + b_2)$$

Backprop



Actor-Critic

- Actor: π_θ
- Critic: Q_ϕ

Soft Actor Critic

Actor-Critic

- Actor: π_θ
- Critic: Q_ϕ

Soft Actor Critic

$$J(\pi) = E\left[\sum_{t=0}^{\infty} \gamma^t \left(r_t + \alpha \mathcal{H}(\pi(\cdot \mid s_t))\right)\right]$$

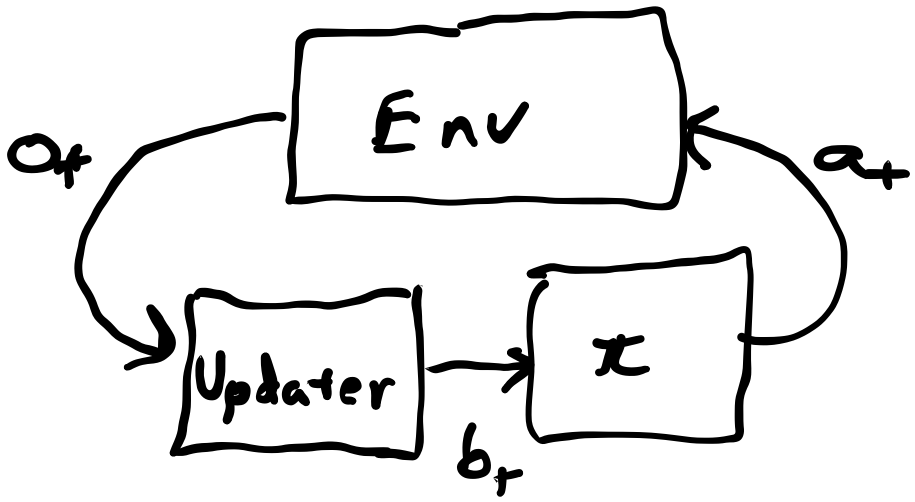
POMDPs

POMDPs

$$(S, A, T, R, O, Z, \gamma)$$

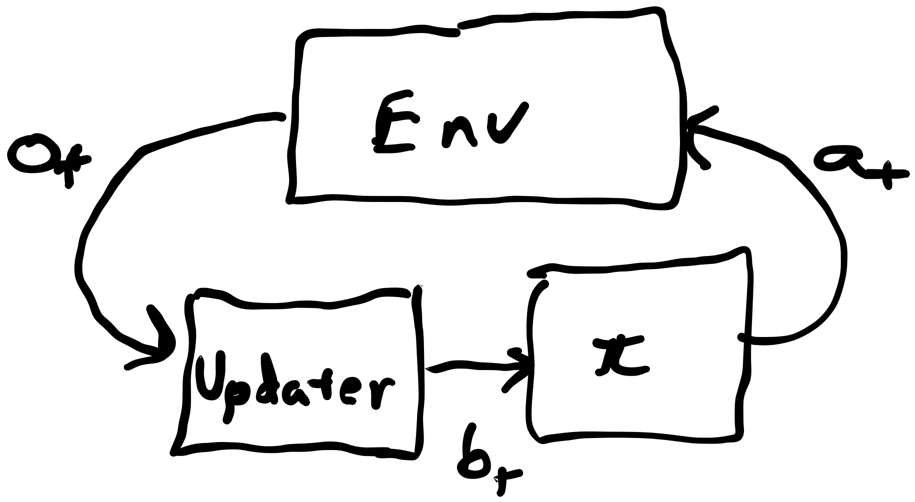
POMDPs

$$(S, A, T, R, O, Z, \gamma)$$



POMDPs

$$(S, A, T, R, O, Z, \gamma)$$

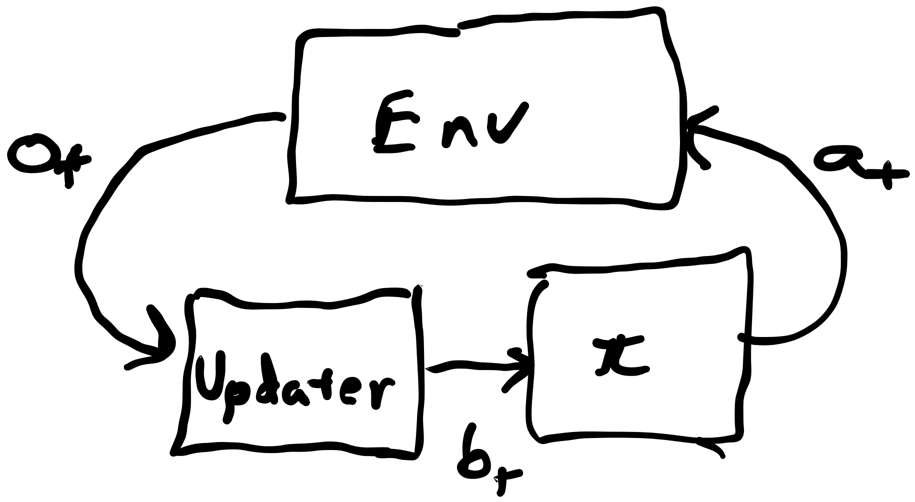


Belief Updates

- Discrete Bayesian Filter
- Particle Filter

POMDPs

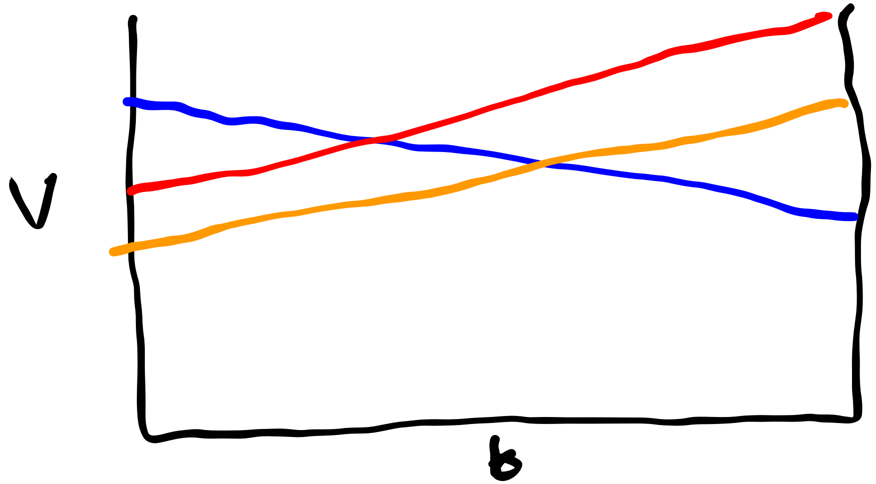
$$(S, A, T, R, O, Z, \gamma)$$



Belief Updates

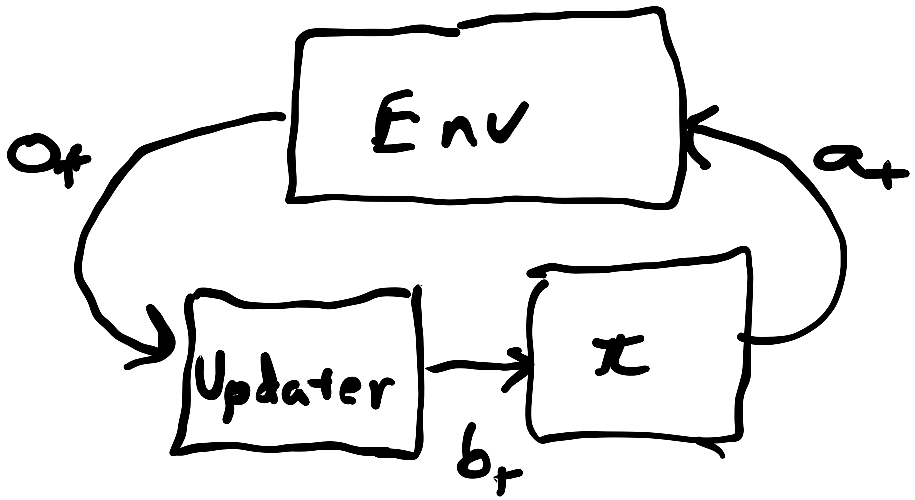
- Discrete Bayesian Filter
- Particle Filter

Alpha Vectors



POMDPs

$$(S, A, T, R, O, Z, \gamma)$$

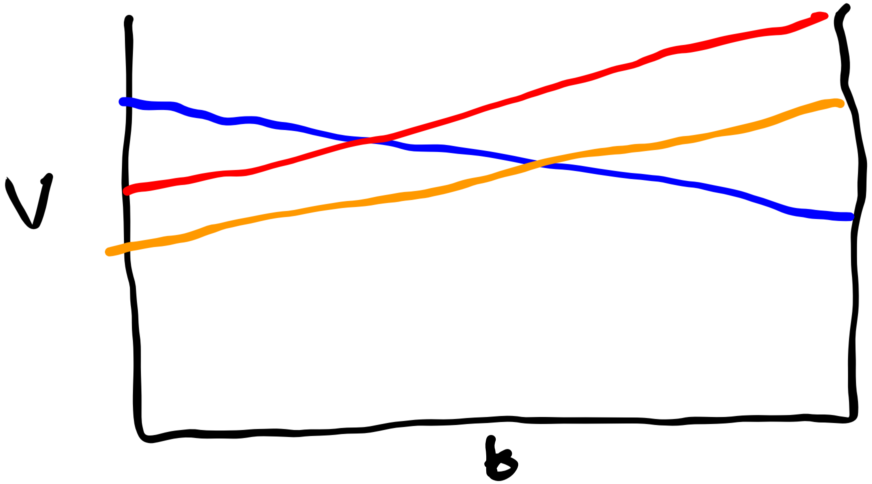


- Each alpha vector corresponds to a conditional plan

Belief Updates

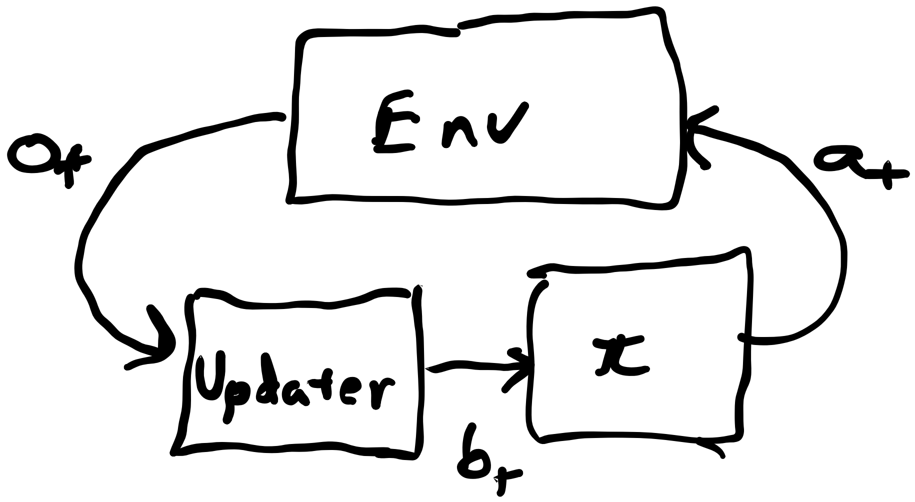
- Discrete Bayesian Filter
- Particle Filter

Alpha Vectors



POMDPs

$$(S, A, T, R, O, Z, \gamma)$$

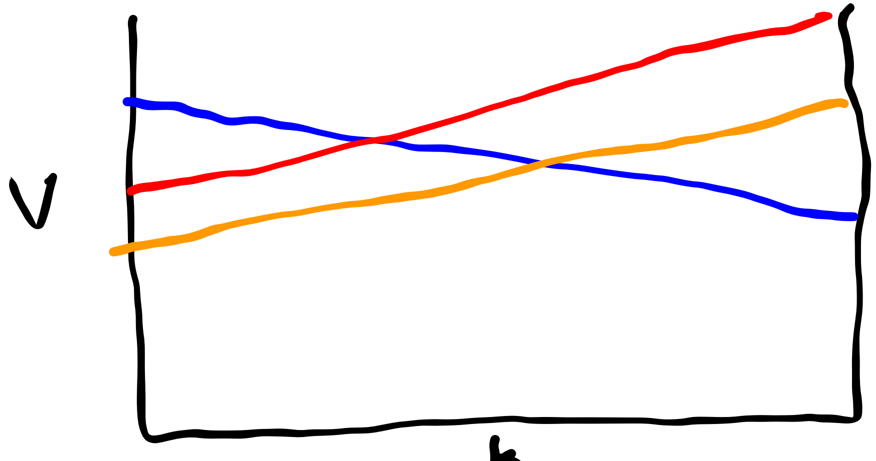


- Each alpha vector corresponds to a conditional plan
- You can prune alpha vectors by solving an LP

Belief Updates

- Discrete Bayesian Filter
- Particle Filter

Alpha Vectors



POMDP Approximations

POMDP Approximations

Formulation

- Certainty Equivalence
- QMDP

POMDP Approximations

Formulation

- Certainty Equivalence
- QMDP

Numerical

POMDP Approximations

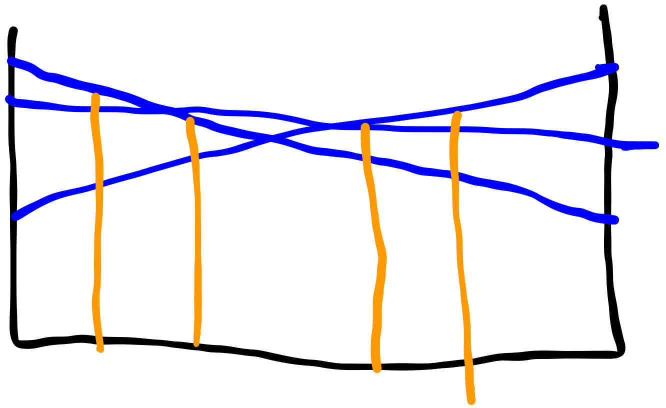
Formulation

- Certainty Equivalence
- QMDP

Numerical

Offline

- Point-Based Value Iteration
- SARSOP



POMDP Approximations

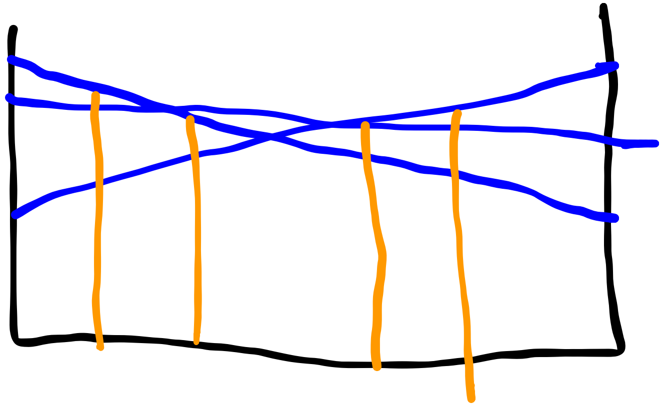
Formulation

- Certainty Equivalence
- QMDP

Numerical

Offline

- Point-Based Value Iteration
- SARSOP



Online

- POMCP
- DESPOT

POMDP Approximations

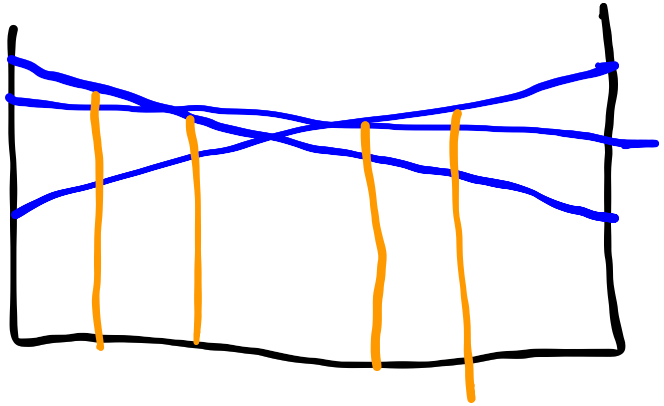
Formulation

- Certainty Equivalence
- QMDP

Numerical

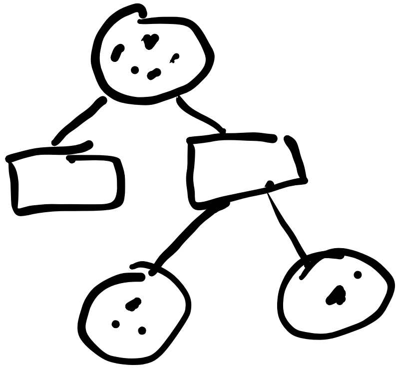
Offline

- Point-Based Value Iteration
- SARSOP



Online

- POMCP
- DESPOT



Simple Games

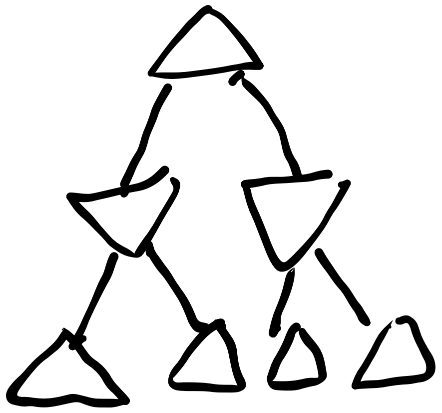
- ~~Optimal Solutions~~ No!
- Equilibria (e.g. Nash Equilibria)

-1,-1	-3,0
0,-3	-2,-2

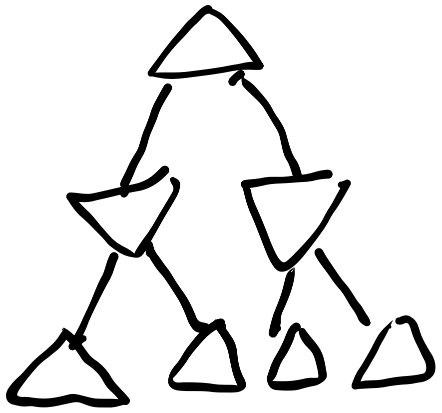
- Every Game has at least 1 Nash Equilibrium
- Might be Pure or Mixed

Turn Taking Games

Turn Taking Games

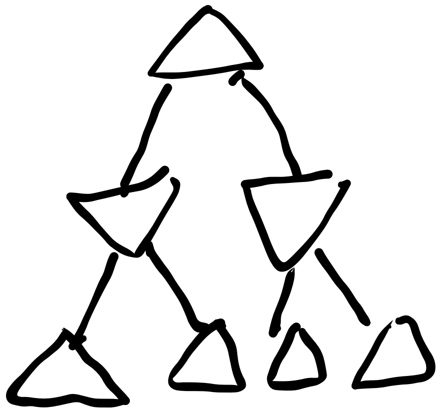


Turn Taking Games



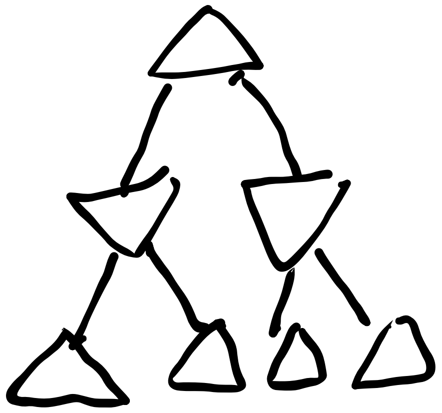
- Value Function Backup

Turn Taking Games



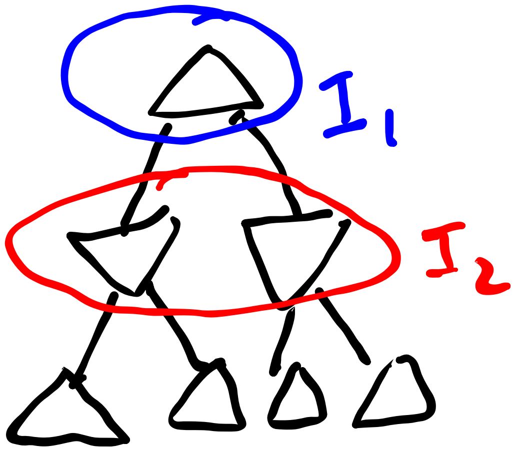
- Value Function Backup
- $\alpha\beta$ Pruning

Turn Taking Games



- Value Function Backup
- $\alpha\beta$ Pruning
- Incomplete Information Extensive Form

Turn Taking Games



- Value Function Backup
- $\alpha\beta$ Pruning
- Incomplete Information Extensive Form

Recap

Recap

Big Problems

1. Immediate and Future Rewards
2. Unknown Models
3. Partial Observability
4. Other Agents

$Q(s,a)$
RL
POMDP
Games