Partial Observability in Reinforcement Learning

DRL Summer School

July 30, 2019

Pascal Poupart, CIFAR AI Chair

BOREALIS AI

RBC Institute for Research

Pascal: Borealis AI Principal Researcher

- Research institute funded by RBC
- 5 research centers:
  - Montreal, Toronto, Waterloo, Edmonton and Vancouver

BOREALIS AI

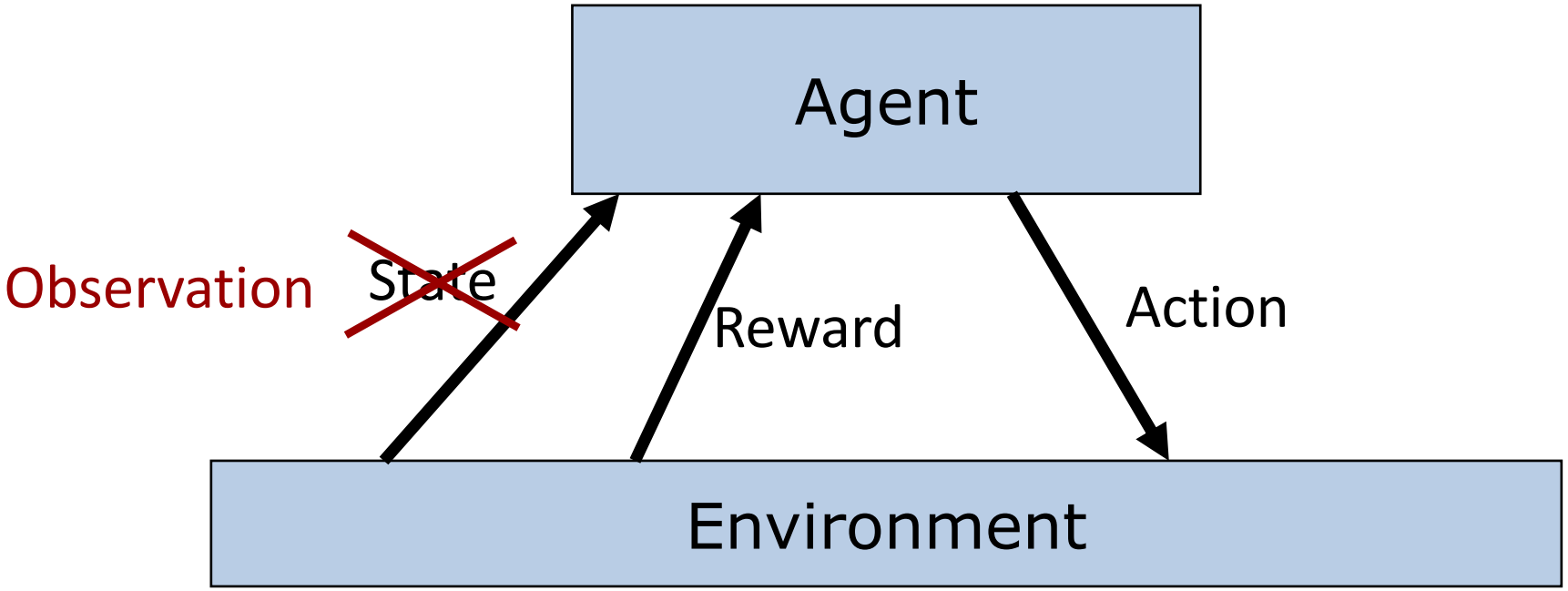
RBC Institute for Research

- 80 researchers:
  - Integrated (applied & fundamental) research model
  - ML, RL, NLP, computer vision, private AI, fintech
- We are hiring!

Pascal: ML Professor at U of Waterloo

- Deep Learning
  - Automated structure learning, sum-product networks, transfer learning
- Reinforcement learning
  - Bayesian RL, Self-supervised RL, Constrained RL, motion-oriented RL, mean-field RL, sport analytics
- NLP
  - Conversational agents, machine translation, automated proofreading
- Theory
  - Convex relaxations of sum-product networks, characterization of local optima in mixture models

Reinforcement Learning Problem



Goal: Learn to choose actions that maximize rewards

Outline

- **Inference** in partially observable domains
  - Hidden Markov models, recurrent neural networks
- **Decision making** in partially observable domains
  - Planning and learning in partially observable MDPs
- **Bayes-optimal exploration**
  - Bayesian RL

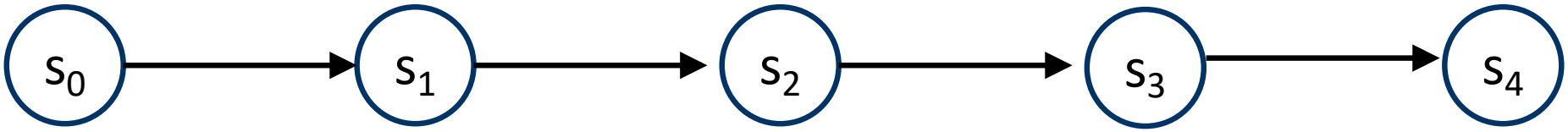
Markov Process

- Assumptions:
  - (first-order) Markovian:

$$\Pr(s_t|s_{t-1}, \dots, s_0) = \Pr(s_t|s_{t-1})$$

  - Stationary:

$$\Pr(s_t|s_{t-1}) = \Pr(s_{t+1}|s_t)\ \forall t$$



Hidden Markov Model

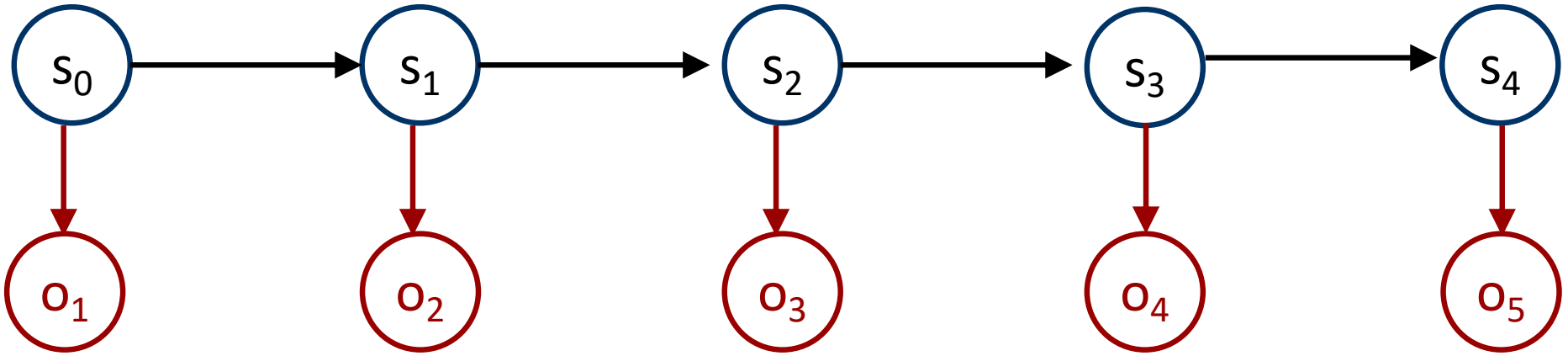
- Assumptions:
  - (first-order) Markovian:

$$\Pr(s_t|s_{t-1}, \dots, s_0) = \Pr(s_t|s_{t-1})$$

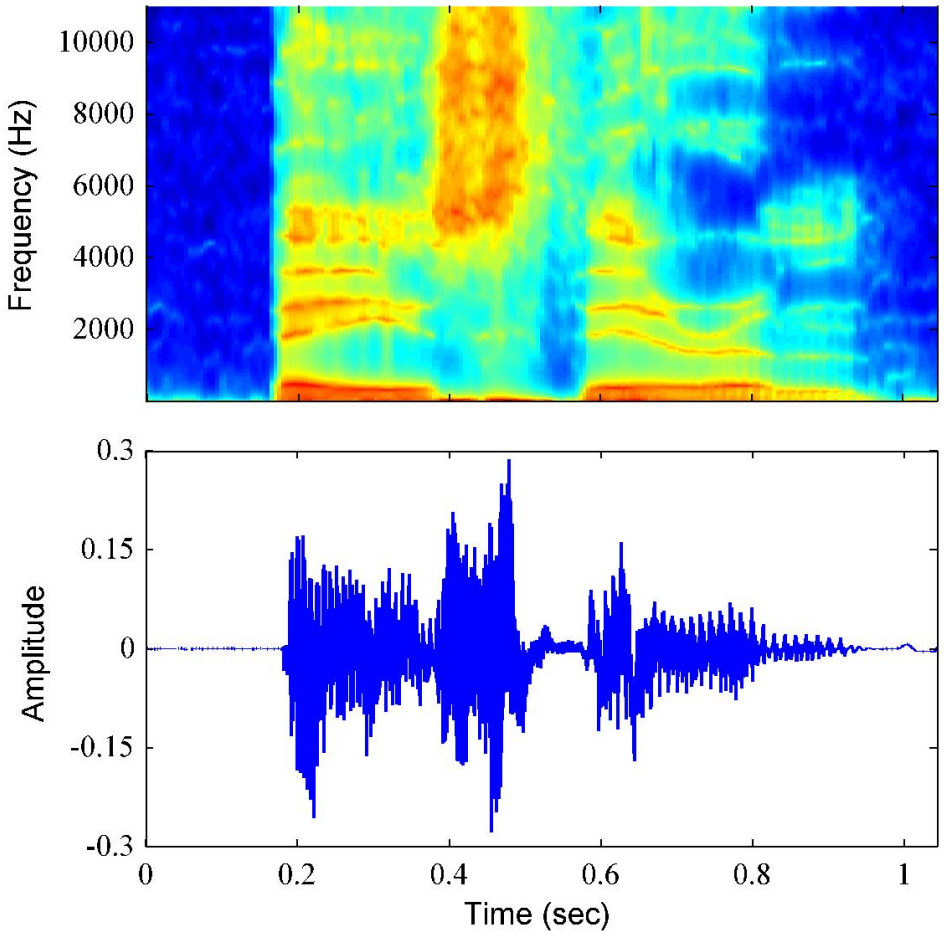
  - Stationary:

$$\Pr(s_t|s_{t-1}) = \Pr(s_{t+1}|s_t)\ \forall t$$

$$\Pr(o_t|s_t) = \Pr(o_{t+1}|s_{t+1})\ \forall t$$

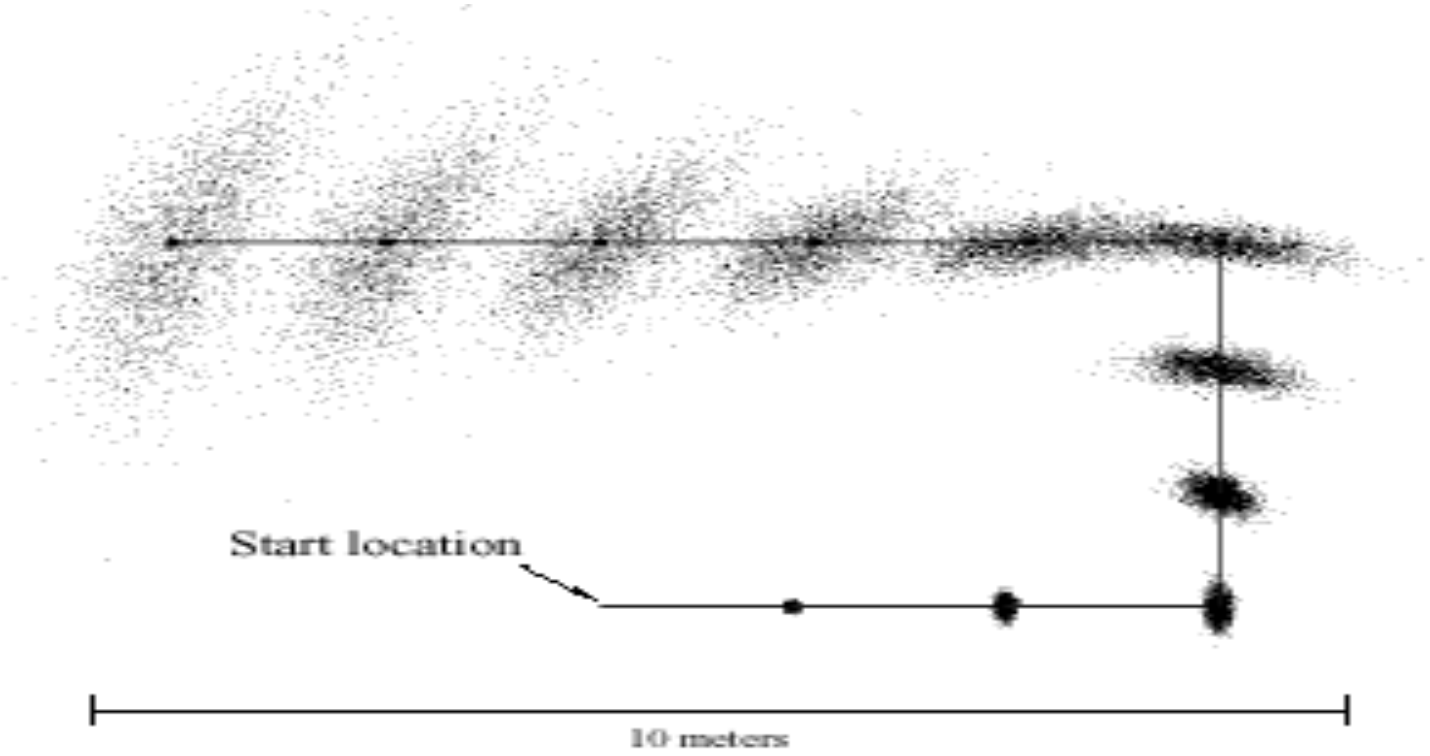


Speech Recognition



Mobile Robot Localisation

- Unobservable location



- Problem: uncertainty grows over time…

Mobile Robot Localisation

- Hidden Markov Model:

  $\boldsymbol{s}$: coordinates of the robot on a map

  $\boldsymbol{o}$: distances to surrounding obstacles (measured by laser range finders or sonars)

  $\mathbf{Pr}(\boldsymbol{s_t}|\boldsymbol{s_{t-1}})$: movement of the robot with uncertainty

  $\mathbf{Pr}(\boldsymbol{o_t}|\boldsymbol{s_t})$: uncertainty in the measurements provided by laser range finders and sonars

- **Localisation:** $\Pr(s_t|o_t, \ldots, o_1)$?

Belief Monitoring

- $\Pr(s_t|o_{1..t})$: distribution over current state given observations
- Examples: robot localisation, activity recognition
- Recursive computation:

$\Pr(s_t|o_{1..t}) \propto \Pr(o_t|s_t, o_{1..t-1})\Pr(s_t|o_{1..t-1})$ by Bayes' thm

$= \Pr(o_t|s_t)\Pr(s_t|o_{1..t-1})$ by conditional independence

$= \Pr(o_t|s_t)\sum_{s_{t-1}}\Pr(s_t, s_{t-1}|o_{1..t-1})$ by marginalization

$= \Pr(o_t|s_t)\sum_{s_{t-1}}\Pr(s_t|s_{t-1}, o_{1..t-1})\Pr(s_{t-1}|o_{1..t-1})$

by chain rule

$= \Pr(o_t|s_t)\sum_{s_{t-1}}\Pr(s_t|s_{t-1})\Pr(s_{t-1}|o_{1..t-1})$ by cond. ind.

Forward Algorithm

- Compute $\Pr(s_t|o_{1..t})$ by forward computation

$\Pr(s_1|o_1) \propto \Pr(o_1|s_1)\Pr(s_1)$

For $i = 2$ to t do

$$\Pr(s_i|o_{1..i}) \propto \Pr(o_i|s_i)\sum_{s_{i-1}} \Pr(s_i|s_{i-1})\Pr(s_{i-1}|o_{1..i-1})$$

End

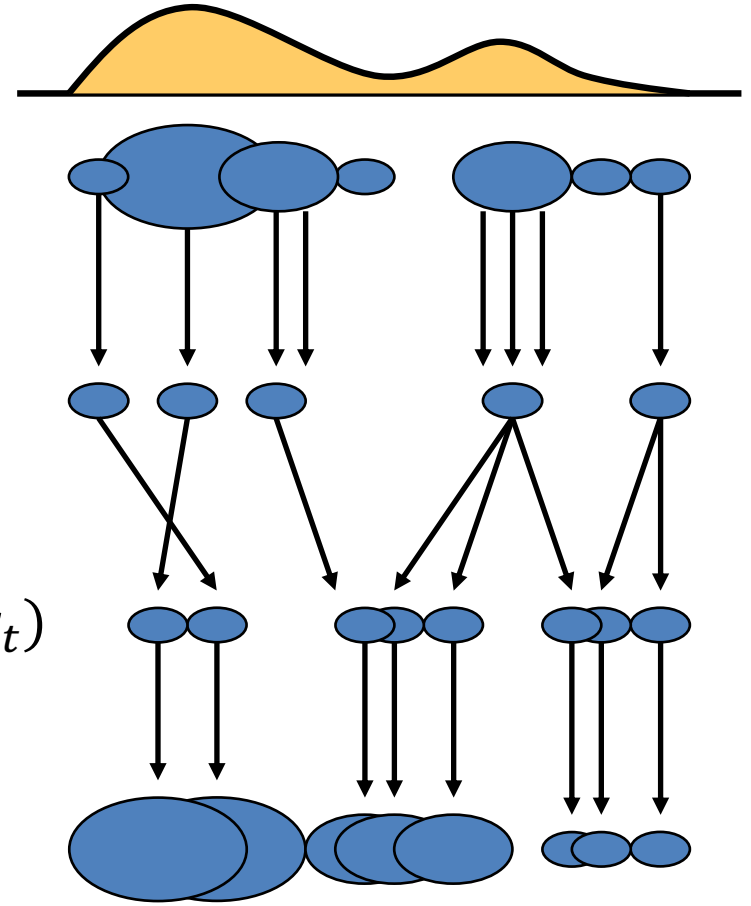
- Linear complexity in t
- How do we deal with continuous states?

Particle Filtering

1) Resample particles

2) Propagate according to transition $\Pr(s_{t+1}|s_t)$

3) Reweight according to observation distribution $\Pr(o_{t+1}|s_{t+1})$



$\Pr(s_t|o_{1..t})$

$\Pr(s_t|o_{1..t})$

$\Pr(s_{t+1}|o_{1..t})$

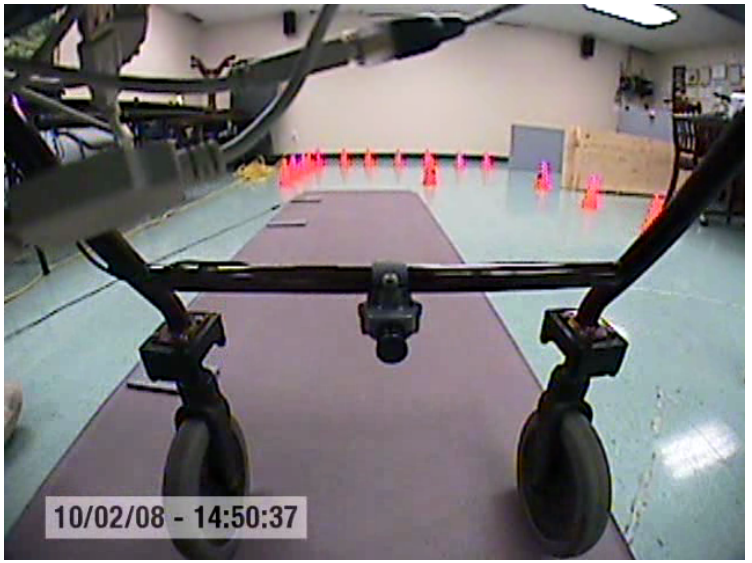
$\Pr(s_t|o_{1..t+1})$

Case Study: Activity Recognition

- [Omar, Sinn et al. UAI 2010]
- Task: infer activities performed by a user of a smart walker
  - Inputs: sensor measurements
  - Output: activity

Backward view

Forward view



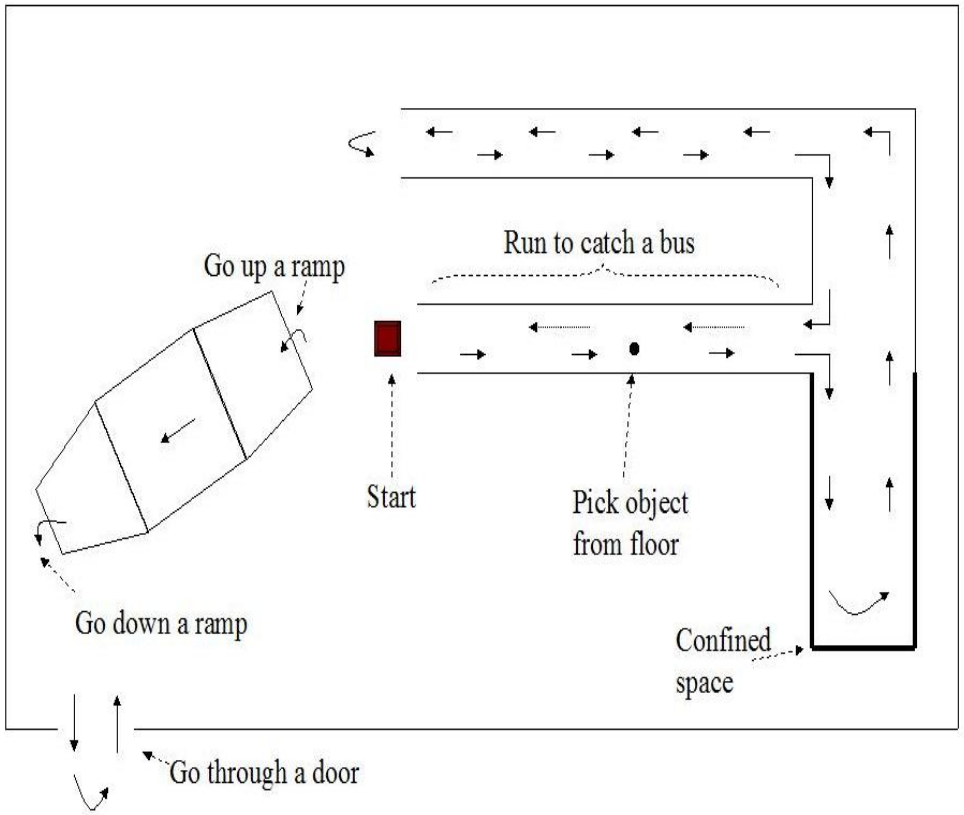
Inputs: Raw Sensor Data

- 8 channels:
  - Forward acceleration
  - Lateral acceleration
  - Vertical acceleration
  - Load on left rear wheel
  - Load on right rear wheel
  - Load on left front wheel
  - Load on right front wheel
  - Wheel rotation counts (speed)

- Data recorded at 50 Hz and digitized (16 bits)

Data Collection

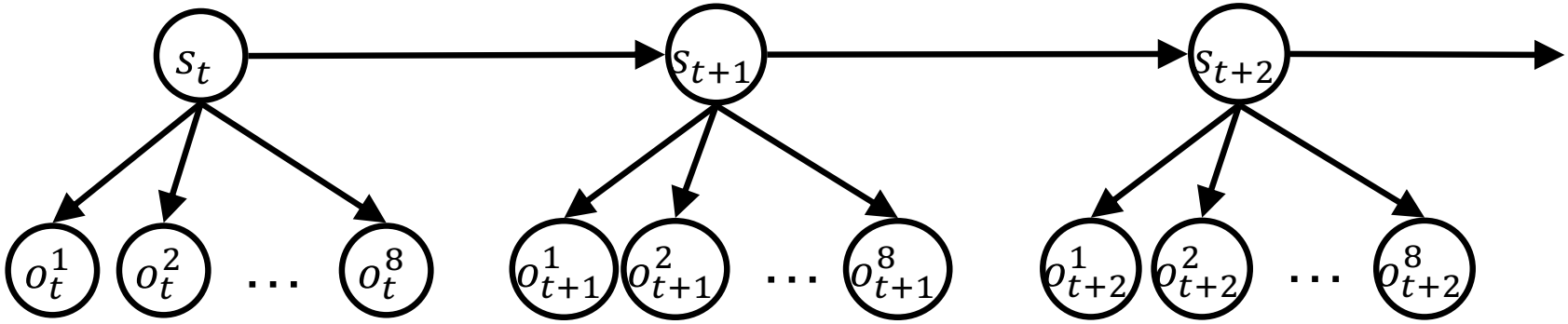
- 8 walker users at Winston Park (84-97 years old)
- 12 older adults (80-89 years old) in the Kitchener-Waterloo area who do not use walkers



Output: Activities

- Not Touching Walker (NTW)
- Standing (ST)
- Walking Forward (WF)
- Turning Left (TL)
- Turning Right (TR)
- Walking Backwards (WB)
- Sitting on the Walker (SW)
- Reaching Tasks (RT)
- Up Ramp/Curb (UR/UC)
- Down Ramp/Curb (DR/DC)

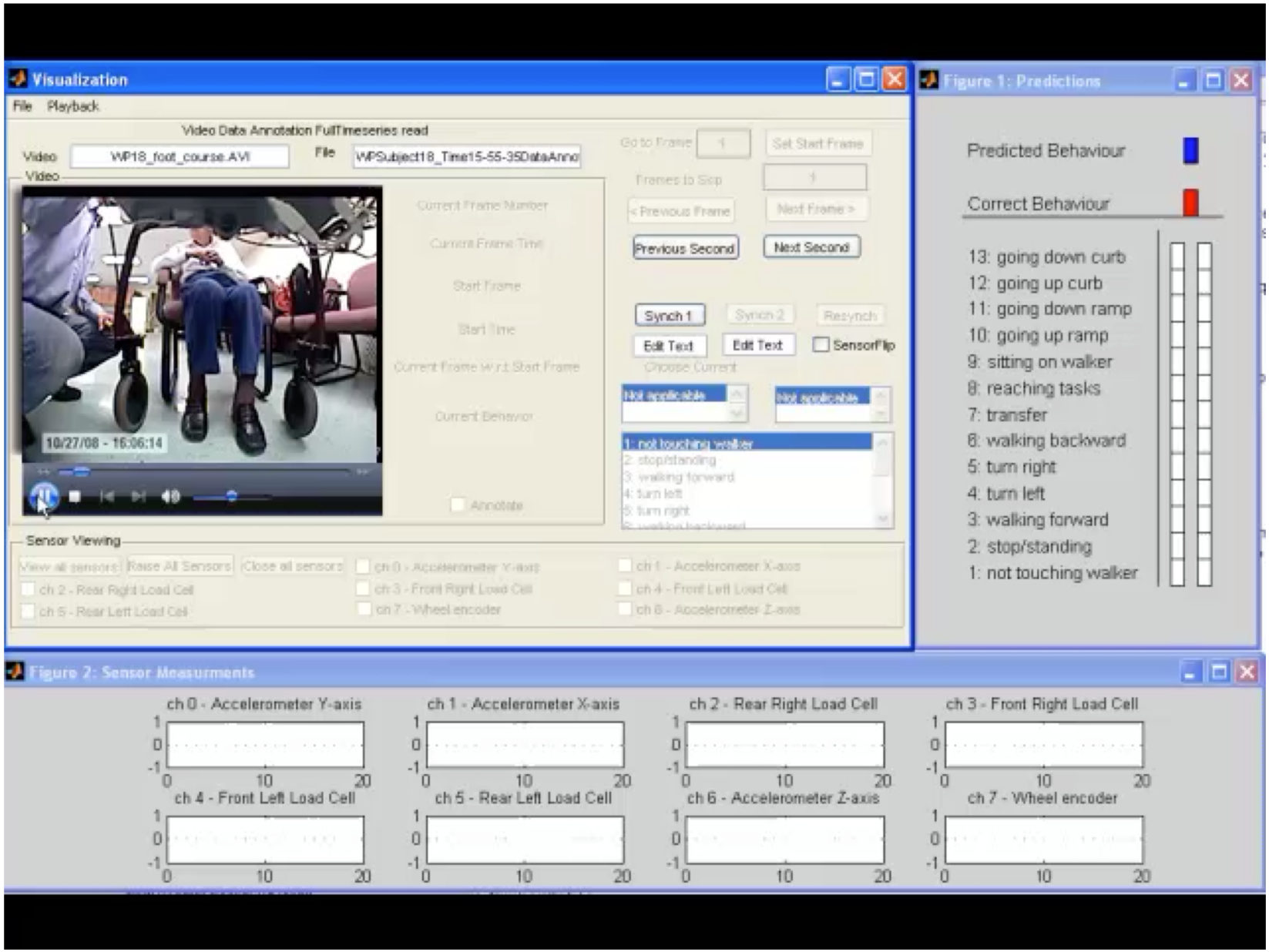
Hidden Markov Model (HMM)



- Parameters
  - Initial state distribution: $\psi_{class} = \Pr(s_1 = class)$
  - Transition probabilities: $\theta_{class'|class} = \Pr(s_{t+1} = class' | s_t = class)$
  - Observation probabilities: $\phi^i_{val|class} = \Pr(o_t^i = val | o_t = class)$

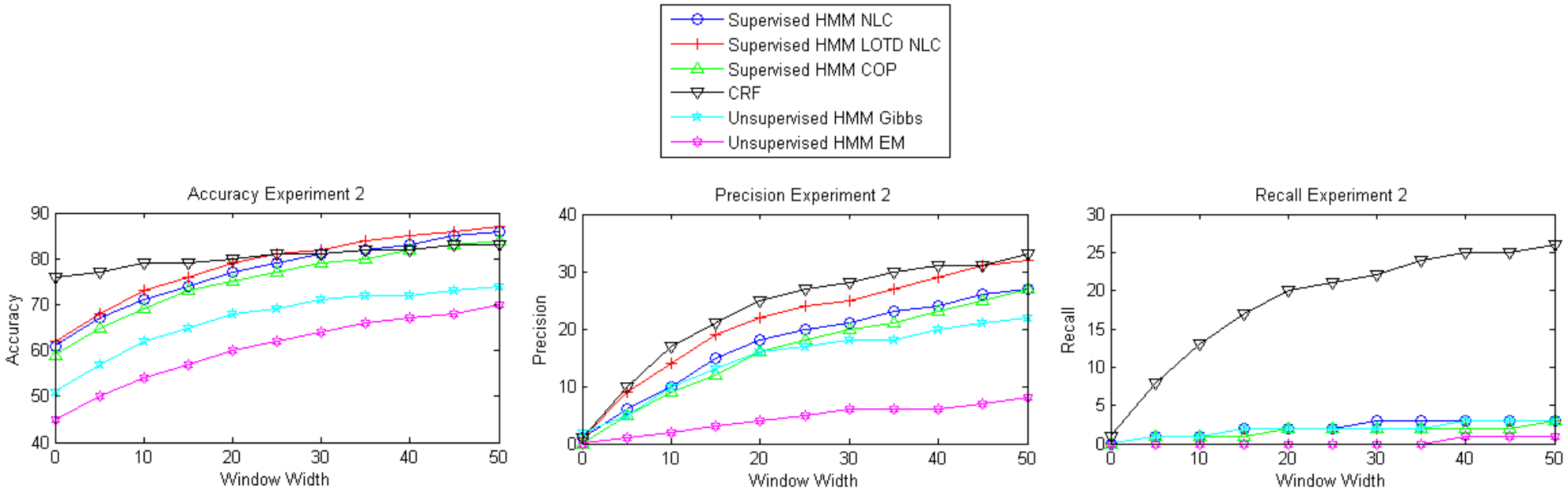
    or $N(val|\mu^i_{class}, \sigma^i_{class}) = \Pr(o_t^i = val | o_t = class)$
- Maximum likelihood:
  - Supervised: $\psi^*, \theta^*, \phi^* = argmax_{\psi,\theta,\phi} \Pr(s_{1:T}, o_{1:T} | \psi, \theta, \phi)$
  - Unsupervised: $\psi^*, \theta^*, \phi^* = argmax_{\psi,\theta,\phi} \Pr(o_{1:T} | \psi, \theta, \phi)$

Demo


Visualization
File Playback
Video Data Annotation FullTimeseries read
Video
WP18_foot_course.AVI
File
WPSubject18_Time15-55-35DataAnno
Video
10/27/08 - 16:06:14
Current Frame Number
Current Frame Time
Start Frame
Start Time
Current Frame w.r.t. Start Frame
Current Behavior
Annotate
Go to Frame
Set Start Frame
Frames to Skip
< Previous Frame
Next Frame >
Previous Second
Next Second
Synch 1
Synch 2
Resynch
Edit Text
Edit Text
SensorFlip
Choose Current
Not applicable
Not applicable
1: not touching walker
2: stop/standing
3: walking forward
4: turn left
5: turn right
Sensor Viewing
View all sensors
Raise All Sensors
Close all sensors
ch 0 - Accelerometer Y-axis
ch 1 - Accelerometer X-axis
ch 2 - Rear Right Load Cell
ch 3 - Front Right Load Cell
ch 4 - Front Left Load Cell
ch 5 - Rear Left Load Cell
ch 6 - Accelerometer Z-axis
ch 7 - Wheel encoder
Figure 1: Predictions
Predicted Behaviour
Correct Behaviour
13: going down curb
12: going up curb
11: going down ramp
10: going up ramp
9: sitting on walker
8: reaching tasks
7: transfer
6: walking backward
5: turn right
4: turn left
3: walking forward
2: stop/standing
1: not touching walker
Figure 2: Sensor Measurments
ch 0 - Accelerometer Y-axis
ch 1 - Accelerometer X-axis
ch 2 - Rear Right Load Cell
ch 3 - Front Right Load Cell
ch 4 - Front Left Load Cell
ch 5 - Rear Left Load Cell
ch 6 - Accelerometer Z-axis
ch 7 - Wheel encoder

Results

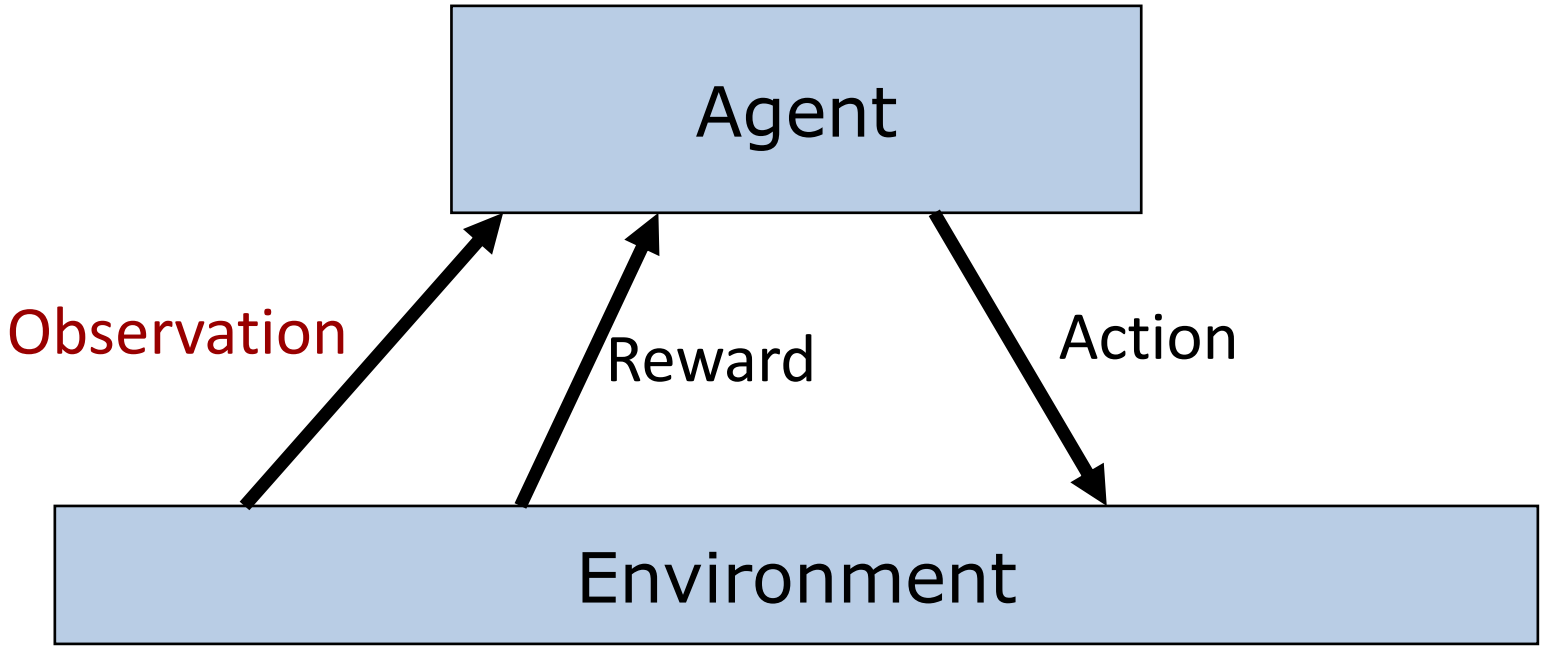
[Omar, Sinn et al. UAI 2010]



Outline

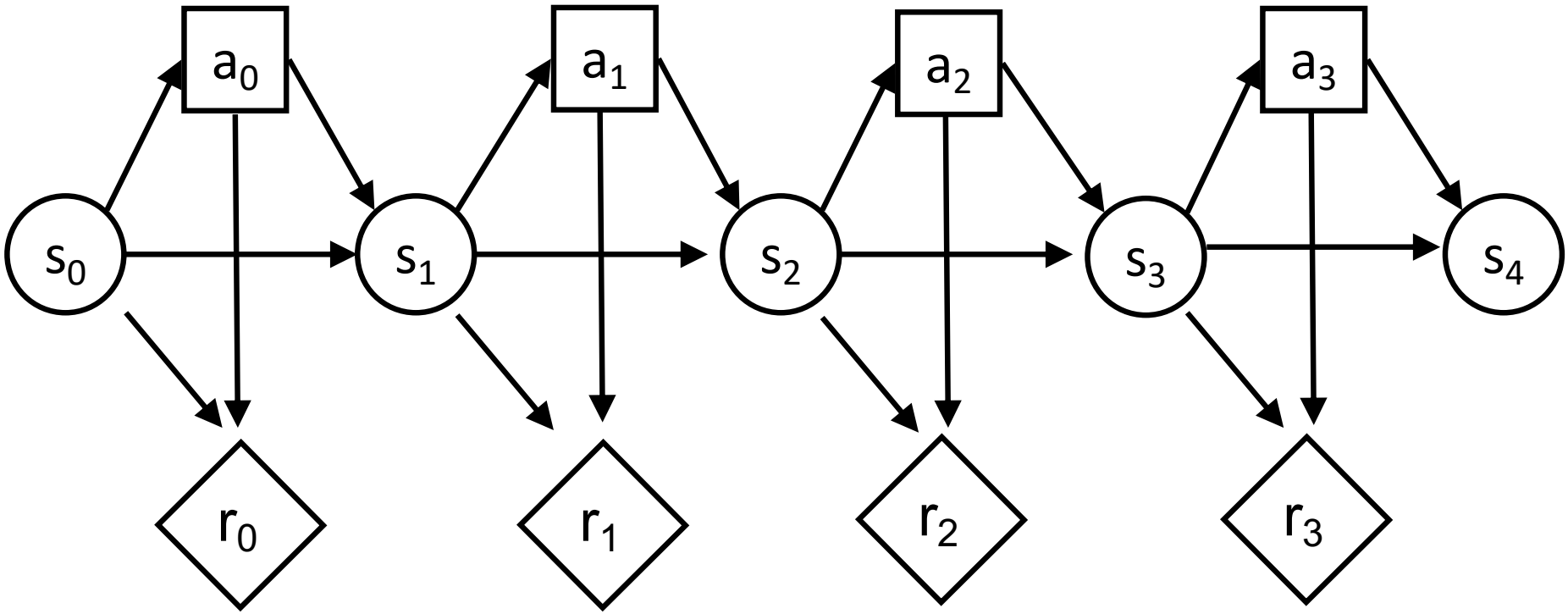
- **Inference** in partially observable domains
  - Hidden Markov models, recurrent neural networks
- **Decision making** in partially observable domains
  - Planning and learning in partially observable MDPs
- **Bayes-optimal exploration**
  - Bayesian RL

Reinforcement Learning Problem



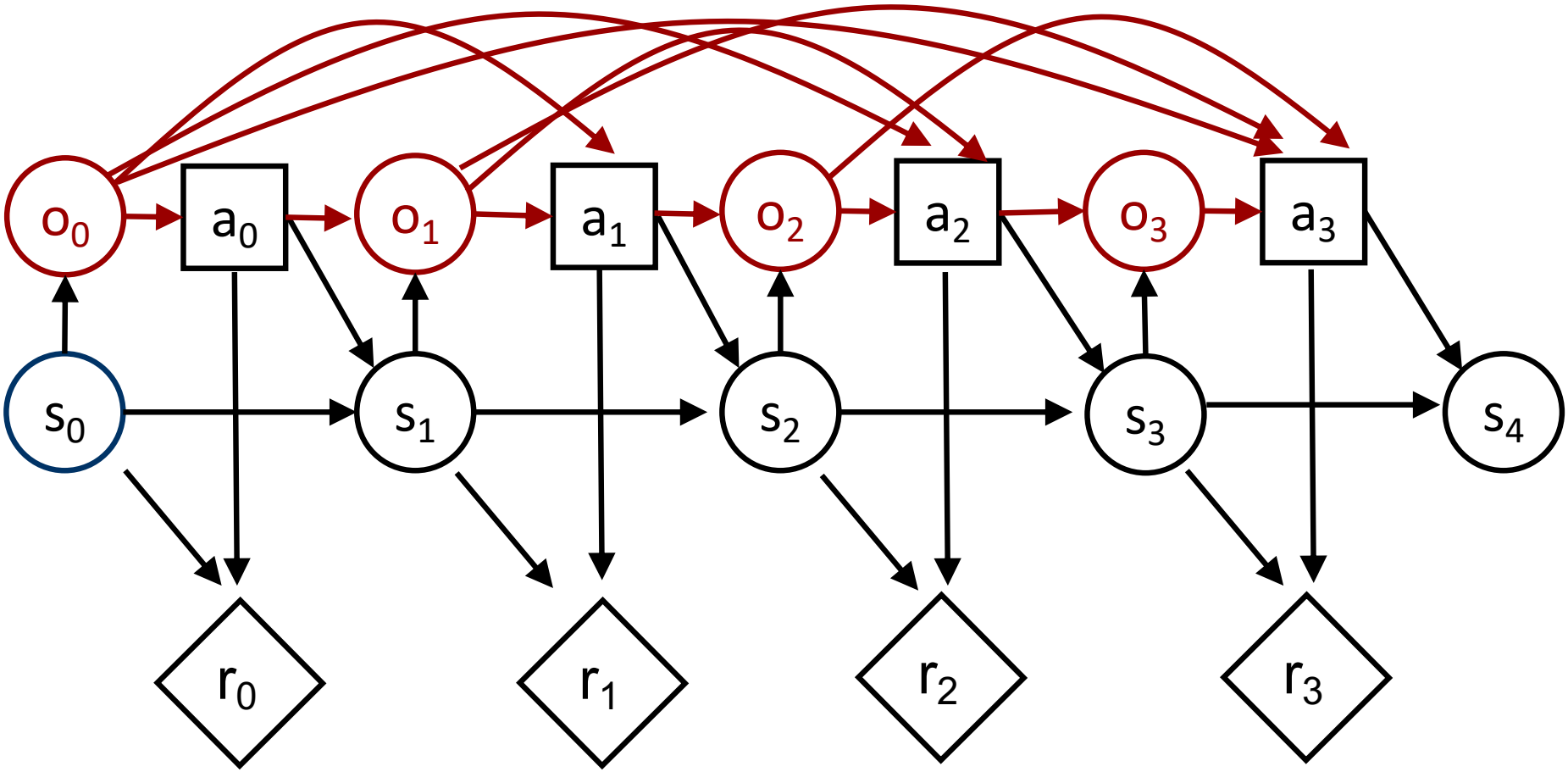
Goal: Learn to choose actions that maximize rewards

(Fully Observable) Markov Decision Process (MDP)



Partially Observable Markov Decision Process (POMDP)

- MDP augmented with observations



POMDP Planning

- Definition
  - States: $s \in S$
  - Observations: $o \in O$
  - Actions: $a \in A$
  - Rewards: $r \in \mathbb{R}$
  - Transition model: $\Pr(s_t|s_{t-1}, a_{t-1})$
  - Observation model: $\Pr(o_t|a_{t-1}, s_t)$
  - Reward model: $\Pr(r_t|s_t, a_t)$
  - Discount factor: $0 \leq \gamma \leq 1$
    - discounted: $\gamma < 1$ undiscounted: $\gamma = 1$
  - Horizon (i.e., # of time steps): h
    - Finite horizon: $h \in \mathbb{N}$ infinite horizon: $h = \infty$

- Goal: find optimal policy π^* such that

$$\pi^* = argmax_\pi \sum_{t=0}^{h} \gamma^t E_\pi[r_t]$$

Partially Observable RL

- Definition
  - States: $s \in S$
  - Observations: $o \in O$
  - Actions: $a \in A$
  - Rewards: $r \in \mathbb{R}$
  - Transition model: $\Pr(s_t|s_{t-1}, a_{t-1})$
  - Observation model: $\Pr(o_t|a_{t-1}, s_t)$
  - Reward model: $\Pr(r_t|s_t, a_t)$

  } unknown model

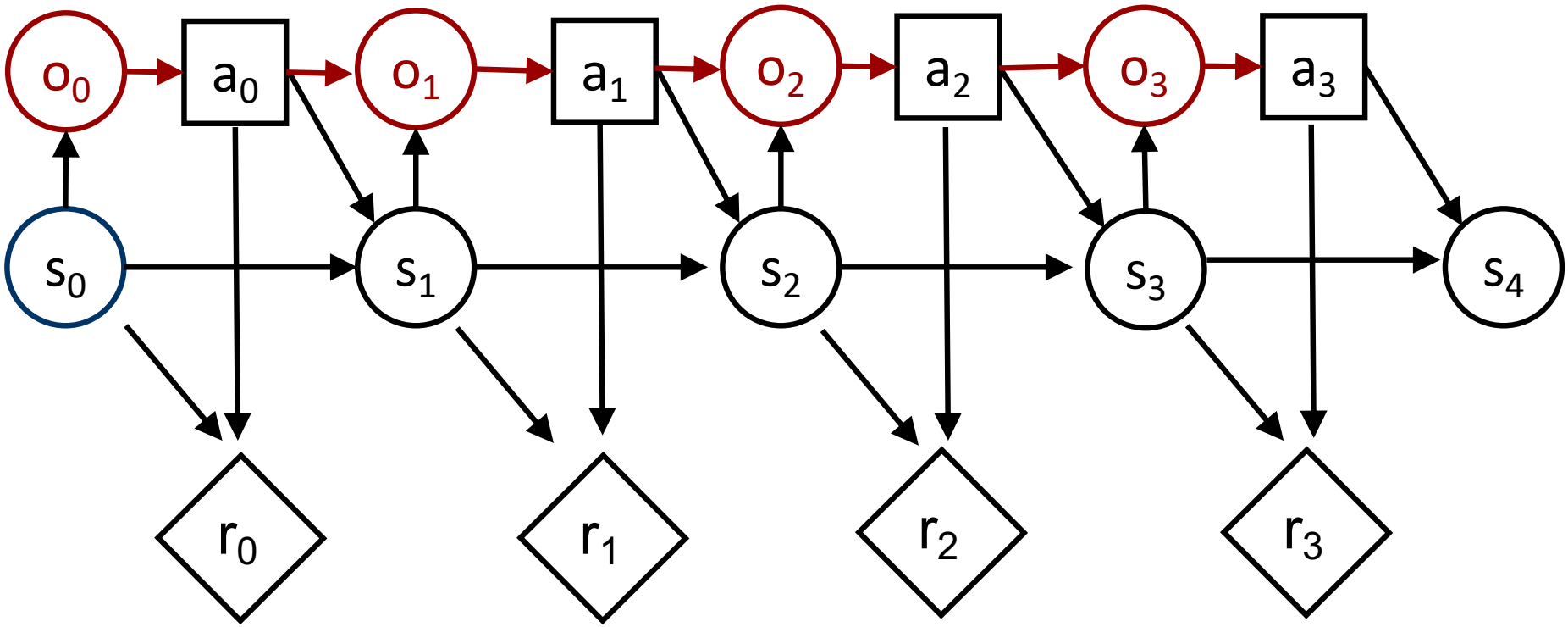
  - Discount factor: $0 \leq \gamma \leq 1$
    - discounted: $\gamma < 1$ undiscounted: $\gamma = 1$
  - Horizon (i.e., # of time steps): h
    - Finite horizon: $h \in \mathbb{N}$ infinite horizon: $h = \infty$

- Goal: find optimal policy π^* such that

$$\pi^* = argmax_\pi \sum_{t=0}^{h} \gamma^t E_\pi[r_t]$$

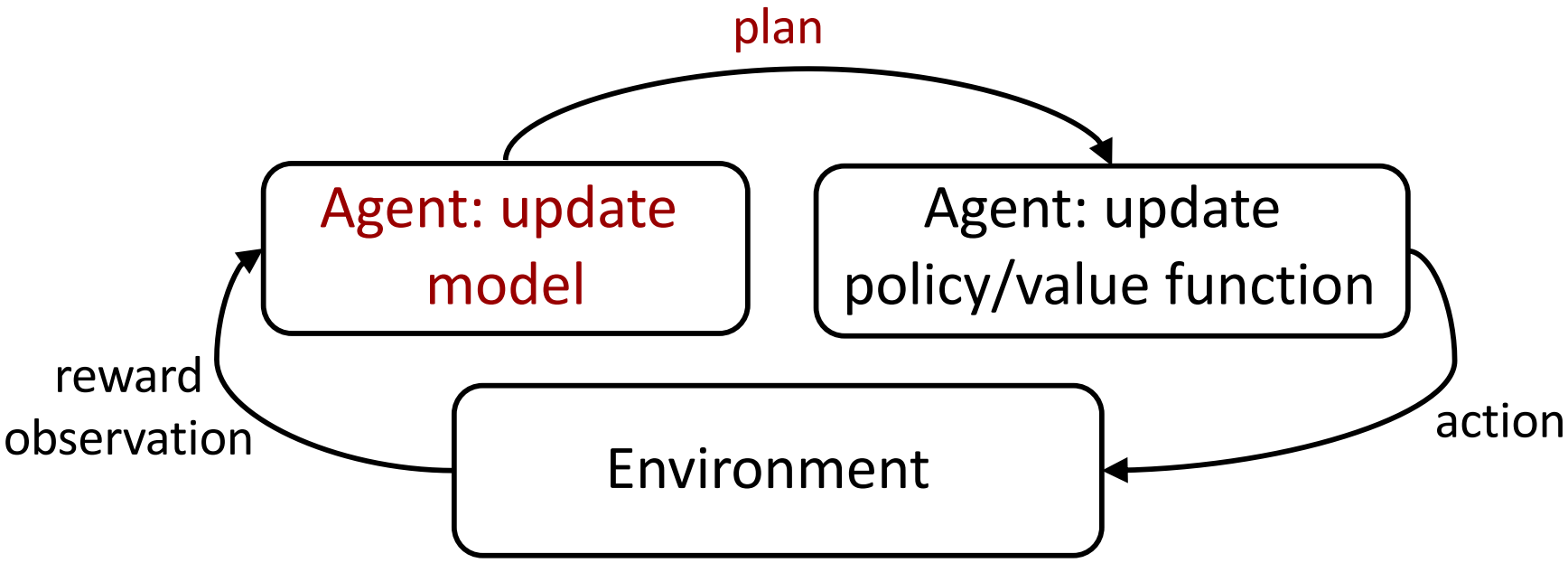
Simple Heuristic

- Approximate s_t by o_t (or finite history: $o_{t-k}, a_{t-k+1}, \dots, o_{t-1}, a_{t-1}, o_t$)



Model-based Partially Observable RL

- Model-based RL
  - Learn model (HMM, RNN) from data
  - Plan by optimizing POMDP policy
    - Value iteration, policy search



Planning

- Idea: summarize previous observations into a distribution about the current unobserved state called **belief**

- History: $h_t = (o_t, a_{t-1}, o_{t-1}, a_{t-2}, \dots, o_2, a_1)$
- Belief: $b_t(s_t) = \Pr(s_t|h_t)$
  - Sufficient statistic: $b_t \equiv h_t$

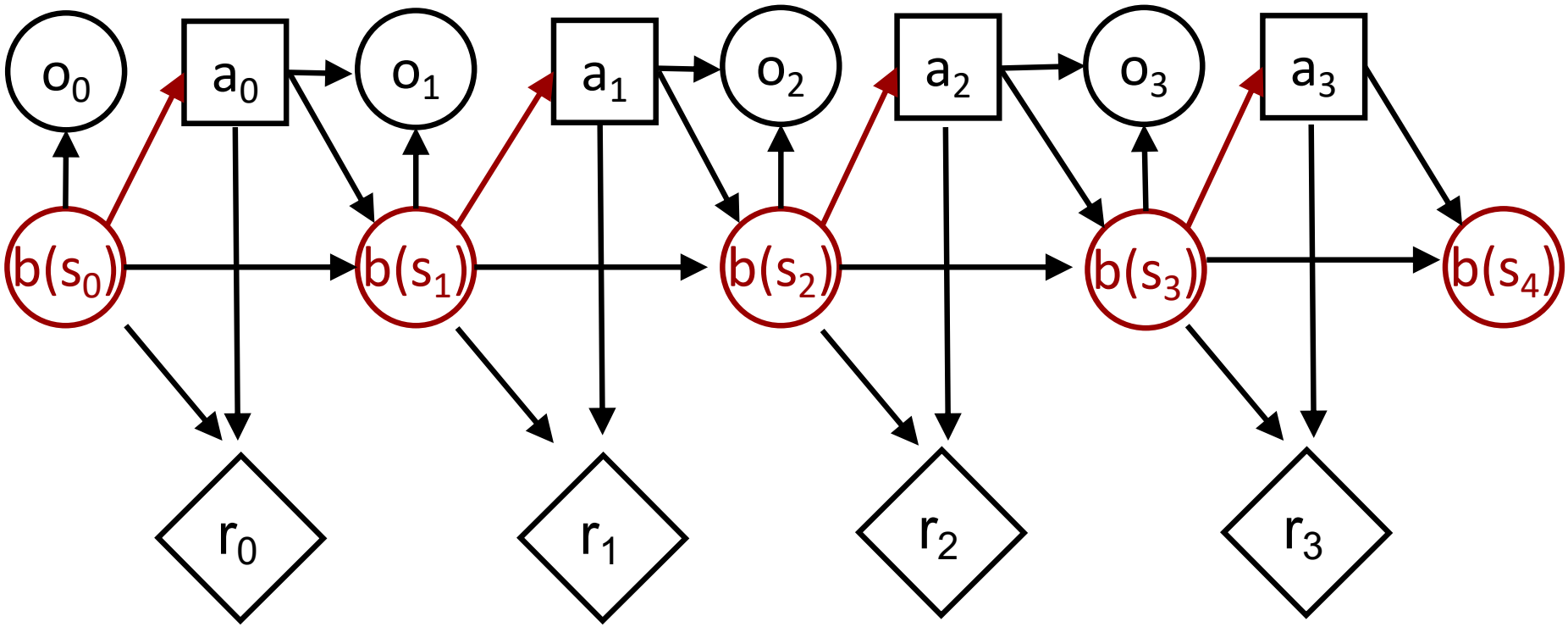
- Belief monitoring:

$$\Pr(s_t|h_t) \propto \Pr(o_t|s_t, a_{t-1}) \sum_{s_{t-1}} \Pr(s_t|s_{t-1}, a_{t-1}) \Pr(s_{t-1}|h_{t-1})$$

$$b_t(s_t) \propto \Pr(o_t|s_t, a_{t-1}) \sum_{s_{t-1}} \Pr(s_t|s_{t-1}, a_{t-1}) b_{t-1}(s_{t-1})$$

Belief MDP

- Replace s_t by $b(s_t)$
- Action depends only on previous belief



Value Iteration Algorithm

valueIteration(beliefMDP)

$V_0^*(b) \leftarrow \max_a R(b,a) \ \forall s$

For $t = 1$ to h do

$V_t^*(b) \leftarrow \max_a R(b,a) + \gamma \sum_{o'} \Pr(o'|b,a) V_{t-1}^*(b^{ao'}) \ \forall b$

Return V^*

Where

$R(b,a) = \sum_s b(s) R(s,a)$

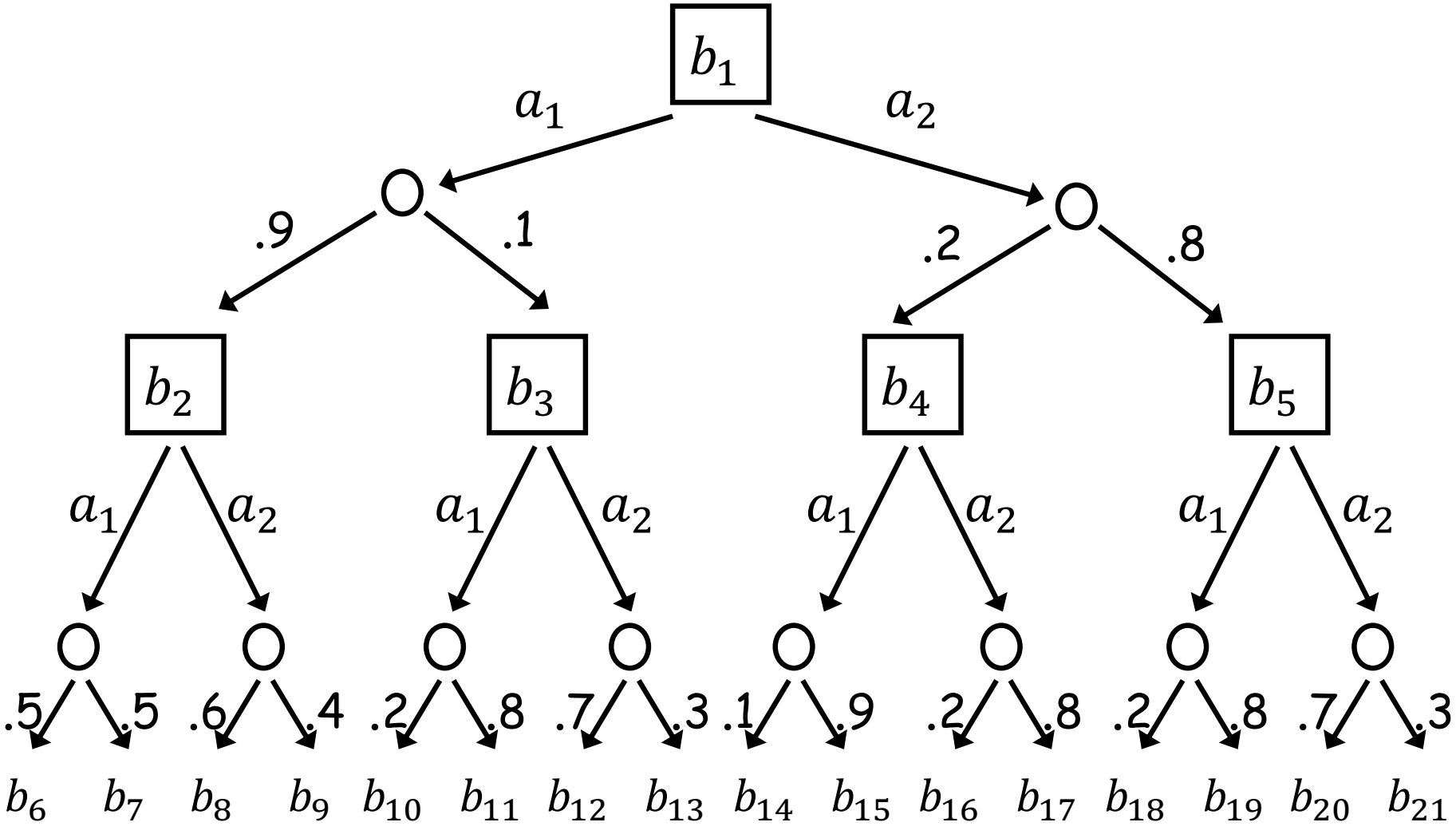
$\Pr(o'|b,a) = \sum_{s'} \Pr(o'|s',a) \sum_s \Pr(s'|s,a) b(s)$

$b^{ao'}(s') = \Pr(s'|b,a,o') \propto \Pr(o'|s',a) \sum_s \Pr(s'|s,a) b(s)$

POMDP Planning

- Offline planning
  - Point-based value iteration [Pineau et al., 2003]
  - Policy search
    - Finite state automata [Hansen 1998]
    - Stochastic satisfiability [Majercik, Littman 1999]
    - Planning as inference [Toussaint, Storkey 2006]
- Online planning
  - Monte-Carlo tree search [Silver, Veness 2010]

Tree Search


b_1
a_1
a_2
.9
.1
.2
.8
b_2
b_3
b_4
b_5
a_1 a_2 a_1 a_2 a_1 a_2 a_1 a_2
.5 .5 .6 .4 .2 .8 .7 .3 .1 .9 .2 .8 .2 .8 .7 .3
b_6 b_7 b_8 b_9 b_{10} b_{11} b_{12} b_{13} b_{14} b_{15} b_{16} b_{17} b_{18} b_{19} b_{20} b_{21}

Tractable Tree Search

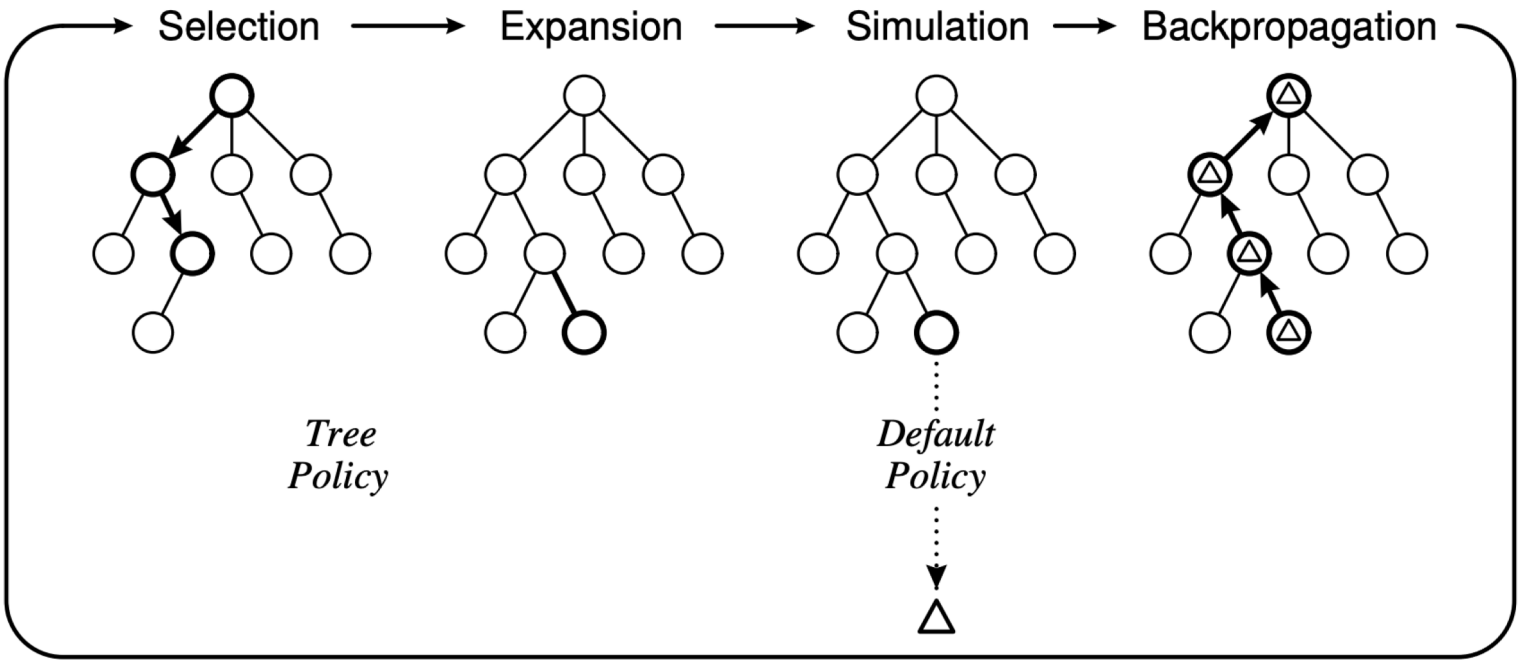
- Combine 3 ideas:
  - Leaf nodes: approximate leaf values with value of default policy π

    $Q^*(b,a) \approx Q^{\pi}(b,a) \approx \frac{1}{n(s,a)} \sum_{k=1}^{n} G_k$ where G_k is value of a rollout
  - Chance nodes: approximate expectation by sampling from transition model

    $$Q^*(b,a) \approx R(b,a) + \gamma \frac{1}{n(b,a)} \sum_{o' \sim \Pr(o'|b,a)} V(b^{ao'})$$
  - Decision nodes: expand only most promising actions

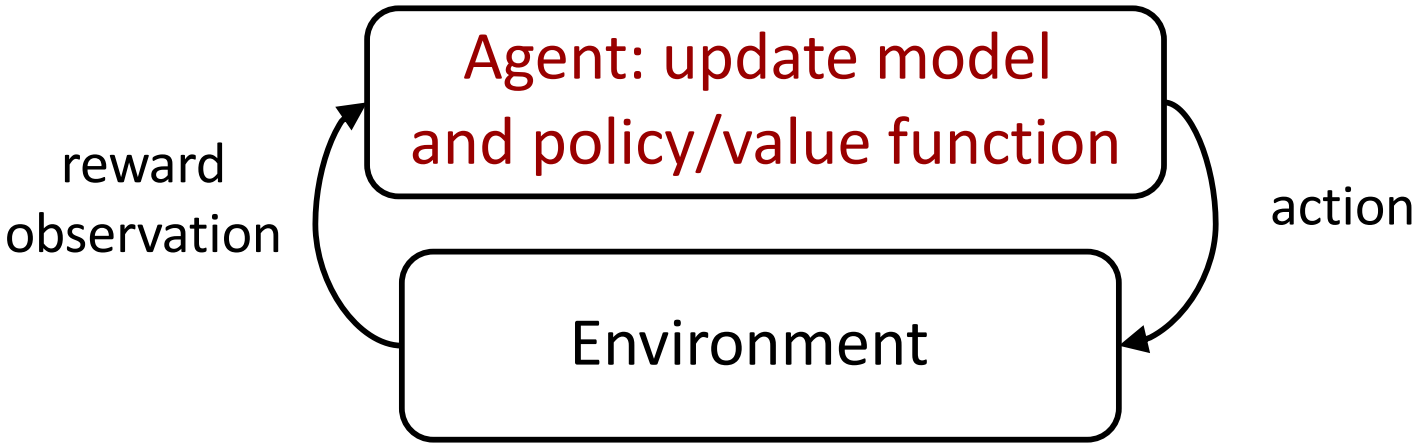
    $a^* = argmax_a Q(b,a) + c\sqrt{\frac{2 \ln n(b)}{n(b,a)}}$ and $V^*(b) = Q(b,a^*)$
- Resulting algorithm: **Monte Carlo Tree Search**

Monte Carlo Tree Search



- **Selection:** $a^* = argmax_a Q(b,a) + c\sqrt{\frac{2\ln n(b)}{n(b,a)}}$ and $V^*(b) = Q(b,a^*)$
- **Simulation:** $Q^*(b,a) \approx Q^\pi(b,a) \approx \frac{1}{n(s,a)} \sum_{k=1}^{n} G_k$
- **Backpropagation:** $Q^*(b,a) \approx R(b,a) + \gamma \frac{1}{n(b,a)} \sum_{o' \sim \Pr(o'|b,a)} V(b^{ao'})$

Simultaneous Learning and Planning

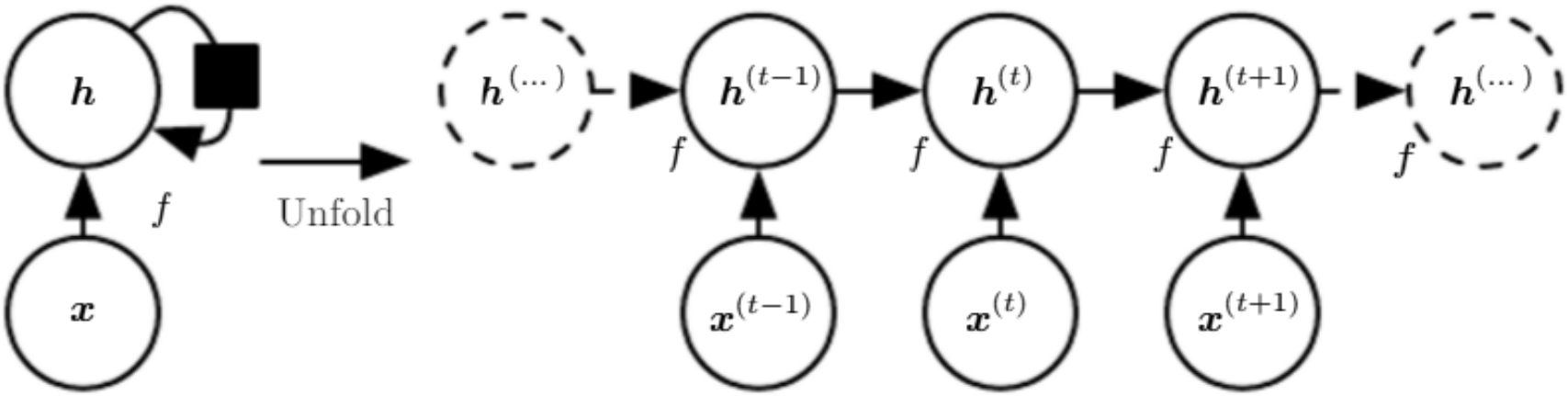


- Deep recurrent Q-networks [Hausknecht & Stone 2016]
  - Transition model: recurrent (LSTM) network
  - Observation model: convolutional neural net
  - Value function: deep neural network

Recurrent Neural Network (RNN)

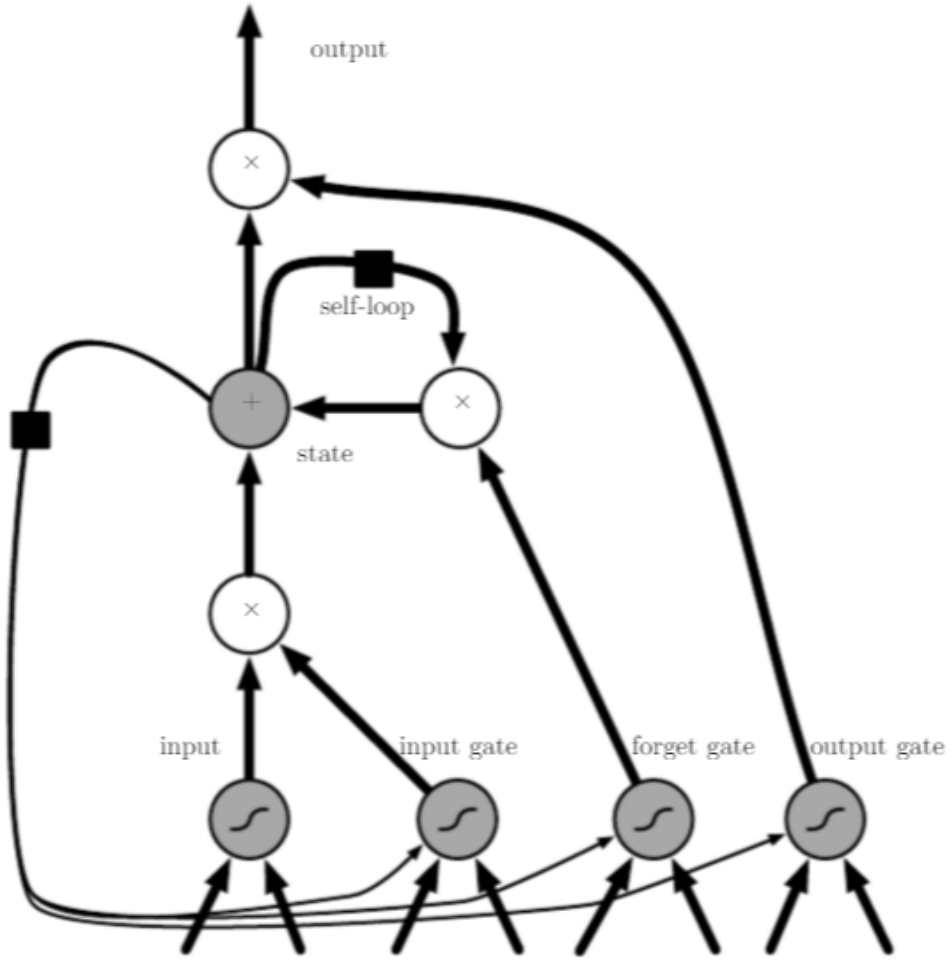
- HMMs can be simulated and generalized by RNNs
- RNNs can be used for belief monitoring

  $\boldsymbol{x}_t$: vector of observations $\boldsymbol{h}_t$: belief state

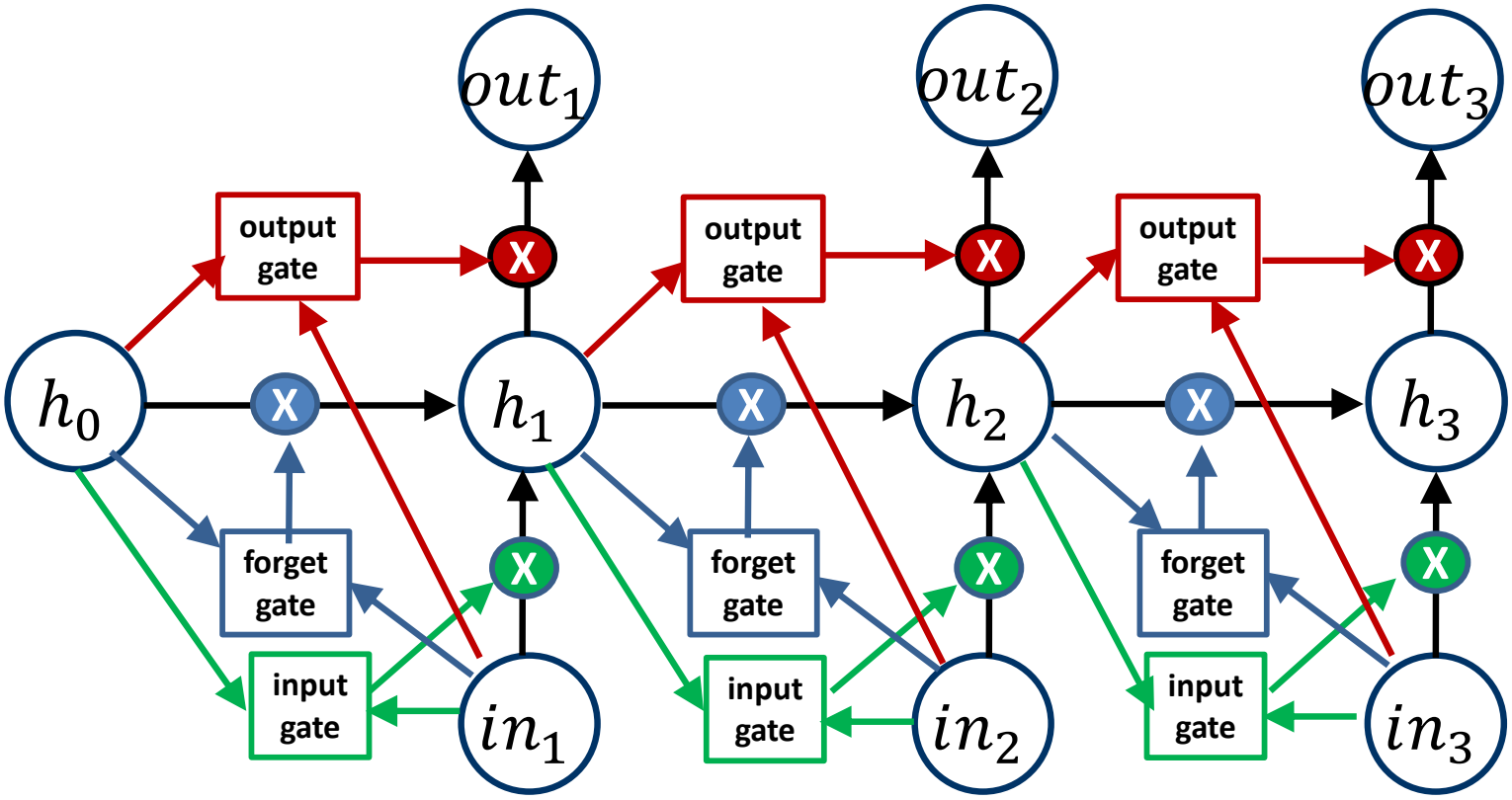


Long Short Term Memory (LSTM)

- Special gated structure to control memorization and forgetting in RNNs
- Mitigate gradient vanishing
- Facilitate long term memory

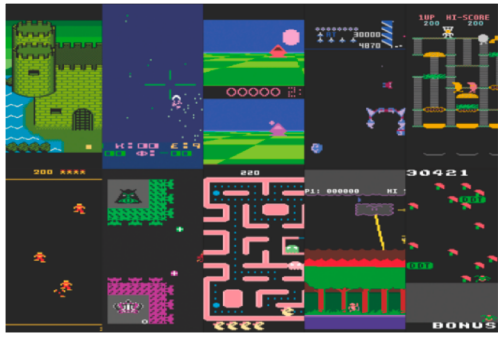


Unrolled long short term memory

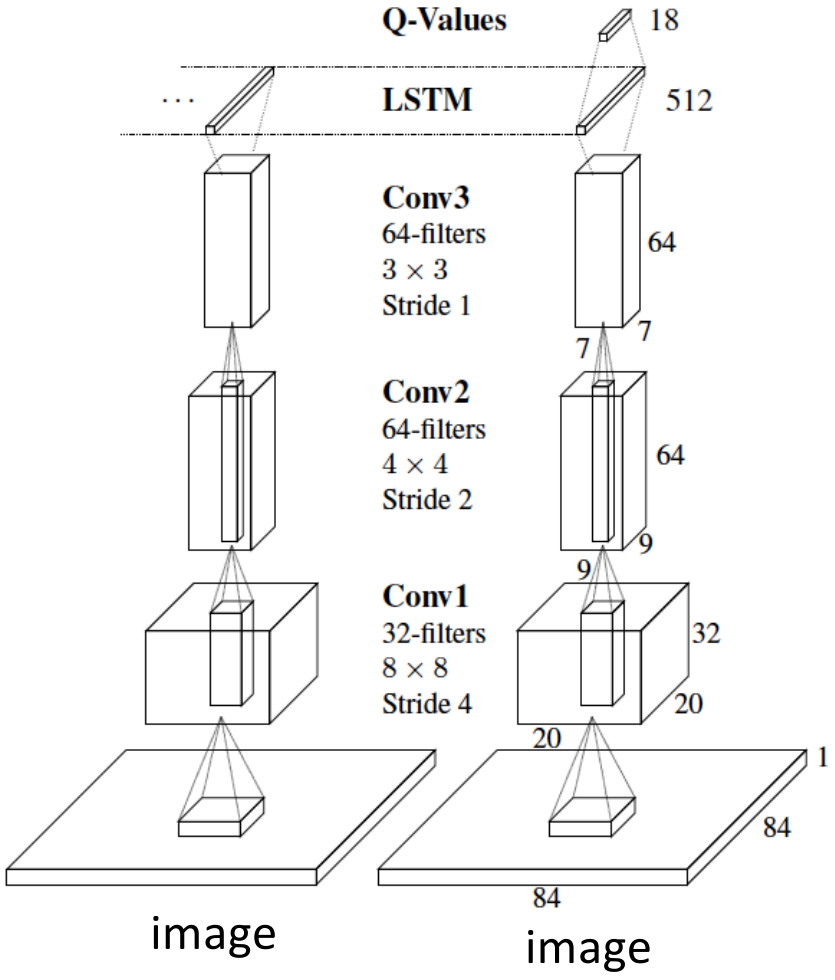


Deep Recurrent Q-Network

- Hausknecht and Stone (2016)
  - Atari games



- Transition model
  - LSTM network
- Observation model
  - Convolutional network



Deep Recurrent Q-Network

Initialize weights $\boldsymbol{w}$ and $\bar{\boldsymbol{w}}$ at random in $[-1,1]$

Observe current state s

Loop

Execute policy for entire episode

Add episode $(o_1, a_1, o_2, a_2, o_3, a_3, \ldots, o_T, a_T)$ to experience buffer

Sample episode from buffer

Initialize h_0

For $t = 1$ till the end of the episode do

$$\frac{\partial Err}{\partial \boldsymbol{w}} = \Big[Q_{\boldsymbol{w}}(RNN_{\boldsymbol{w}}(h_{t-1}\hat{o}_t), \hat{a}_t) - \hat{r} - \gamma \max_{\hat{a}'} \mathrm{Q}_{\bar{\boldsymbol{w}}}(RNN_{\bar{\boldsymbol{w}}}(h_{t-1}\hat{o}_t\hat{o}_{t+1}), \hat{a}_{t+1})\Big] \frac{\partial Q_{\boldsymbol{w}}(RNN_{\boldsymbol{w}}(h_{t-1}\hat{o}_t), \hat{a})}{\partial \boldsymbol{w}}$$

$$h_t \leftarrow RNN_{\bar{\boldsymbol{w}}}(h_{t-1}, \hat{o}_t)$$

Update weights: $\boldsymbol{w} \leftarrow \boldsymbol{w} - \alpha \frac{\partial Err}{\partial \boldsymbol{w}}$

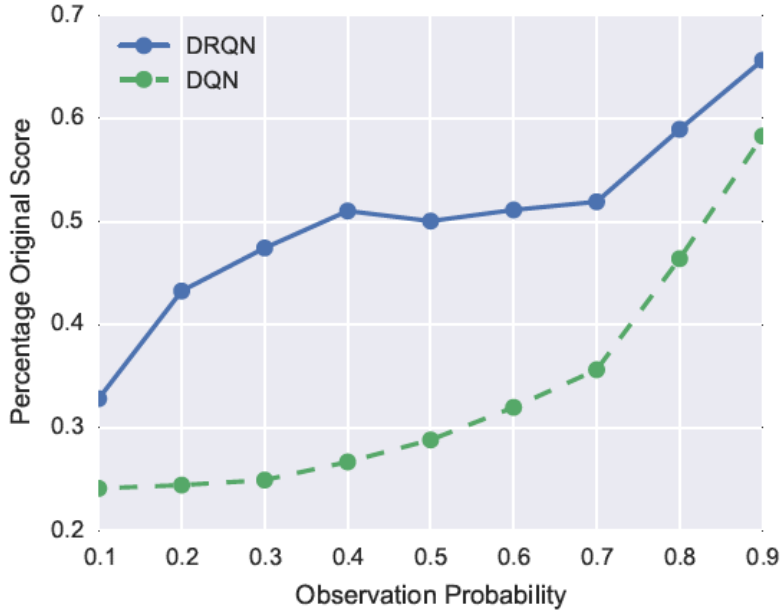
Every c steps, update target: $\bar{\boldsymbol{w}} \leftarrow \boldsymbol{w}$

Results

	DRQN $\pm std$	DQN $\pm std$	
Game		Ours	Mnih et al.
Asteroids	1020 ($\pm$312)	1070 ($\pm$345)	1629 ($\pm$542)
Beam Rider	3269 ($\pm$1167)	**6923** ($\pm$1027)	6846 ($\pm$1619)
Bowling	62 ($\pm$5.9)	72 ($\pm$11)	42 ($\pm$88)
Centipede	3534 ($\pm$1601)	3653 ($\pm$1903)	8309 ($\pm$5237)
Chopper Cmd	2070 ($\pm$875)	1460 ($\pm$976)	6687 ($\pm$2916)
Double Dunk	**-2** ($\pm$7.8)	-10 ($\pm$3.5)	-18.1 ($\pm$2.6)
Frostbite	**2875** ($\pm$535)	519 ($\pm$363)	328.3 ($\pm$250.5)
Ice Hockey	-4.4 ($\pm$1.6)	-3.5 ($\pm$3.5)	-1.6 ($\pm$2.5)
Ms. Pacman	2048 ($\pm$653)	2363 ($\pm$735)	2311 ($\pm$525)

Table 1: On standard Atari games, DRQN performance parallels DQN, excelling in the games of Frostbite and Double Dunk, but struggling on Beam Rider. Bolded font indicates statistical significance between DRQN and our DQN.[5]

Flickering games
(missing observations)



Outline

- **Inference** in partially observable domains
  - Hidden Markov models, recurrent neural networks
- **Decision making** in partially observable domains
  - Planning and learning in partially observable MDPs
- **Bayes-optimal exploration**
  - Bayesian RL

Bayesian RL

- Explicit representation of uncertainty
- Benefits
  - Balance exploration/exploitation tradeoff
  - Mitigate model bias
  - Reduce data needs
- Drawback
  - Complex computation

Traditional RL

- Reinforcement Learning
  - States: $\mathbf{s} \in \boldsymbol{S}$
  - Actions: $\mathbf{a} \in \boldsymbol{A}$
  - Rewards: $\mathbf{r} \in \mathbb{R}$
  - Unknown model: $\mathbf{Pr}(\boldsymbol{r}, \boldsymbol{s}' | \boldsymbol{s}, \boldsymbol{a}; \boldsymbol{\theta})$

- Goal: find policy $\pi: S \rightarrow A$

  and/or value function $Q: S \times A \rightarrow \mathbb{R}$

RL as a POMDP

- Idea: augment state with model parameters
  - Information states: $(\boldsymbol{s}, \boldsymbol{\theta}) \in \boldsymbol{S} \times \boldsymbol{Parameter\ space}$
    - Physical states: $\boldsymbol{s} \in \boldsymbol{S}$
    - **Unknown** model parameters: $\boldsymbol{\theta} \in \boldsymbol{Parameter\ space}$
  - Actions: $\mathbf{a} \in \boldsymbol{A}$
  - Rewards: $\mathbf{r} \in \mathbb{R}$
  - Known model: $\mathbf{Pr}(\boldsymbol{r}, \boldsymbol{s}' | \boldsymbol{s}, \boldsymbol{a}; \boldsymbol{\theta})$
- Goal: find policy $\pi: S \times H \rightarrow A$

  and/or value function $Q: S \times H \times A \rightarrow \mathbb{R}$

RL as a Belief MDP

- Idea: augment state with belief over parameters
  - Information states: $(\boldsymbol{s}, \boldsymbol{b}) \in \boldsymbol{S} \times \boldsymbol{B}$
    - Physical states: $\boldsymbol{s} \in \boldsymbol{S}$
    - **Known** Belief states: $\boldsymbol{b} \in \boldsymbol{B}$ where $b(\theta) = \Pr(\theta)$
  - Actions: $\mathbf{a} \in \boldsymbol{A}$
  - Rewards: $\mathbf{r} \in \mathbb{R}$
  - Known model: $\mathbf{Pr}(\boldsymbol{r}, \boldsymbol{s}', \boldsymbol{b}' | \boldsymbol{s}, \boldsymbol{b}, \boldsymbol{a})$
- Goal: find policy $\pi: S \times B \to A$

  and/or value function $Q: S \times B \times A \to \mathbb{R}$

Model in Bayesian RL

- Claim: the model in Bayesian RL is known!

$$\Pr(r, s', b' | s, b, a) = \underbrace{\Pr(r, s' | s, b, a)}_{\text{Physical model}} \underbrace{\Pr(b' | r, s', s, b, a)}_{\text{belief model}}$$

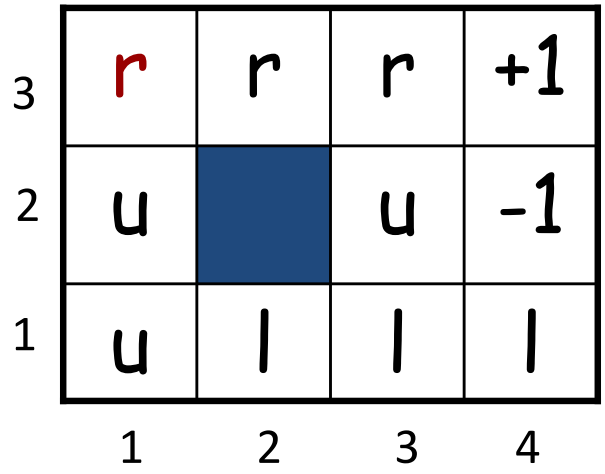
- Idea: integrate out unknown θ

$$\Pr(r, s' | s, b, a) = \int_{\theta} \Pr(r, s' | s, a, \theta)\, b(\theta)\, d\theta$$

- Idea: b' is the posterior belief

$$\Pr(b' | r, s', s, b, a) = \begin{cases} 1 & \text{if } b'(\theta) = b^{s,a,s',r} = b(\theta | s, a, s', r) \\ 0 & \text{otherwise.} \end{cases}$$

Maze Example



$\gamma = 1$

Reward is -0.04 for non-terminal states

Transition model (when ignoring boundaries):

$$\Pr(i', j' | i, j, right, \theta) = \begin{cases} \theta & i' = i + 1 \text{ and } j' = j \\ \frac{1-\theta}{2} & i' = i \text{ and } (j' = j + 1 \; or \; j' = j - 1) \\ 0 & \text{otherwise} \end{cases}.$$

$$\Pr(i', j' | i, j, up, \theta) = \begin{cases} \theta & i' = i \text{ and } j' = j + 1 \\ \frac{1-\theta}{2} & (i' = i + 1 \; or \; i' = i - 1) \text{ and } j' = j. \\ 0 & \text{otherwise} \end{cases}.$$

(similarly for the other actions)

Belief state

- Let's model our uncertainty with respect to θ by a Beta distribution

$$b(\theta) = k\theta^{\alpha-1}(1-\theta)^{\beta-1}$$

- Belief update: Bayes theorem

$$\begin{aligned} b'(\theta) &= b^{s,a,s'}(\theta) \\ &= b(\theta|s,a,s') \\ &\propto b(\theta)Pr(s'|s,a,\theta) \end{aligned}$$

Example belief update

- Prior

$$b(\theta) = Beta(\theta; \alpha, \beta) = k\theta^{\alpha-1}(1-\theta)^{\beta-1}$$

- Posterior for $i, j, up \rightarrow i', j'$ where $i' = i$ and $j' = j + 1$
- Belief update: Bayes theorem

$$b'(\theta) = b^{s,a,s'}(\theta) = b(\theta|s,a,s') = b(\theta|i,j,up,i',j')$$
$$\propto b(\theta)Pr(i',j'|i,j,up,\theta)$$
$$= k\theta^{\alpha-1}(1-\theta)^{\beta-1}\theta$$
$$= k\theta^{\alpha}(1-\theta)^{\beta-1} \propto Beta(\theta; \alpha+1, \beta)$$

Physical Model

- Consider $s = (i, j)$, $a = right$, $s' = (i', j')$
  where $i' = i$ and $j' = j - 1$

- Predictive distribution

$$\Pr(s'|s, b, a) = \int_\theta \Pr(s'|s, a, \theta)\, b(\theta)\, d\theta$$
$$= \int_\theta \Pr(i', j'|i, j, right, \theta)\, Beta(\theta; \alpha, \beta) d\theta$$
$$= \int_\theta \frac{(1-\theta)}{2} k\theta^{\alpha-1}(1-\theta)^{\beta-1} d\theta = \frac{\beta}{2}$$

Planning

- Since the model is known, treat Bayesian RL as an MDP
- Benefits:
  - Solve RL problem by planning (no need to interact with env)
  - Bayes-optimal exploration/exploitation tradeoff
- Drawback:
  - Complex computation
- Bellman's Equation:

$$V^*(s,b) = \max_a E[r|s,b,a] + \gamma \sum_{s'} \Pr(s'|s,a,b)\, V^*(s', b^{s,a,s'}) \;\; \forall s$$

where $E[r|s,b,a] = \int_\theta b(\theta) \int_r pdf(r|s,a,\theta)\, r\, dr\, d\theta$

Value Iteration

- Traditional MDP

valueIteration(MDP)

$V_0^*(s) \leftarrow \max_a E[r|s,a] \ \forall s$

For $t = 1$ to h do

$V_t^*(s) \leftarrow \max_a E[r|s,a] + \gamma \sum_{s'} \Pr(s'|s,a) V_{t-1}^*(s') \ \forall s$

Return V^*

- Information state MDP

valueIteration(BayesianRL)

$V_0^*(s,b) \leftarrow \max_a E[r|s,b,a] \ \forall s$

For $t = 1$ to h do

$V_t^*(s,b) \leftarrow \max_a E[r|s,b,a] + \gamma \sum_{s'} \Pr(s'|s,a,b) V_{t-1}^*(s', b^{s,a,s'}) \ \forall s$

Return V^*

Exploration/exploitation tradeoff

- Dilemma:
  - Maximize immediate rewards (exploitation)?
  - Or, maximize information gain (exploration)?
- **Wrong question!**

- Single objective: max expected total rewards
  - $V^{\pi}(s,b) = \sum_t \gamma^t E[r_t|s_t,b_t]$
  - Optimal policy π^*: $V^{\pi^*}(s,b) \geq V^{\pi}(s,b)$ for all *s,b*
    - **Bayes-optimal exploration/exploitation tradeoff**

Bayesian RL

- Two phases:
  - Offline planning (without the environment)

Find π^* and/or V^*
by policy/value iteration or any other algorithm

  - Online execution (with the environment)

Initialize $s_0, b_0, n \leftarrow 0$
Repeat
    Execute policy $a_n \leftarrow \pi(s_n, b_n)$
    receive s_{n+1} and r_n from the environment
    Belief update: $b_{n+1}(\theta) = b_n^{s_n, a_n, r_n, s_{n+1}}(\theta) = b_n(\theta | s_n, a_n, r_n', s_{n+1}')$
    $n \leftarrow n + 1$

Challenges in Bayesian RL

- Offline planning is notoriously difficult
  - Use function approximators (e.g., Gaussian process or neural net) for model, V^* and π^*
  - Continuous belief space
  - Problem: a good plan should implicitly account for all possible environments, which is intractable

- Alternative: online partial planning
  - Thompson sampling
  - PILCO (Model-based Bayesian Actor Critic)

Thompson Sampling in Bayesian RL

- Idea: Sample models θ_i at each step and plan for the corresponding MDP_{θ_i}'s

ThompsonSamplingInBayesianRL(s,b)
Repeat
Sample $\theta_1, \ldots, \theta_k \sim \Pr(\theta) \ \ \forall a$
$Q^*_{\theta_i} \leftarrow solve(MDP_{\theta_i})$
$\hat{Q}(s,a) \leftarrow \frac{1}{k}\sum_{i=1}^{k} Q^*_{\theta_i}(s,a)$
$a^* \leftarrow \text{argmax}_a \ \hat{Q}(s,a)$
Execute a^* and receive r, s'
$b(\theta) \leftarrow b(\theta)\Pr(r, s'|s, a, \theta)$
$s \leftarrow s'$

Model-based Bayesian Actor Critic

- PILCO: Deisenroth, Rasmussen (2011)
  - $b(\theta)$: Gaussian Process transition model
- Deep PILCO: Gal, McCallister, Rasmussen (2016)
  - $b(\theta)$: Bayesian neural network transition model

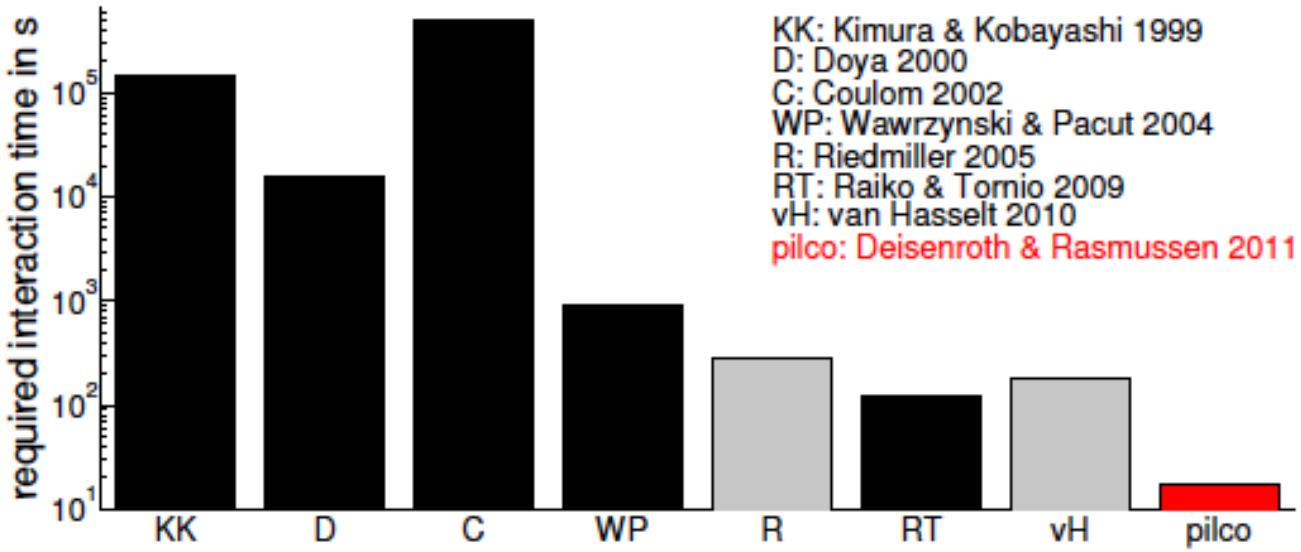
PILCO(s, b, π)
  Repeat
    Repeat
      Critic: $V_b^\pi \leftarrow policyEvaluation(b, \pi)$
      Actor: $\pi \leftarrow \pi + \alpha \, \partial V_b^\pi / \partial \pi$
    $a \leftarrow \pi(s, b)$
    Execute a and receive r, s'
    $b \leftarrow b^{s,a,r,s'}$ and $s \leftarrow s'$

Unprecedented Data Efficiency

Table 1. PILCO's data efficiency scales to high dimensions.

	cart-pole	cart-double-pole	unicycle
state space	$\mathbb{R}^4$	$\mathbb{R}^6$	$\mathbb{R}^{12}$
# trials	≤ 10	20–30	≈ 20
experience	$\approx 20\,\mathrm{s}$	$\approx 60\,\mathrm{s}$–$90\,\mathrm{s}$	$\approx 20\,\mathrm{s}$–$30\,\mathrm{s}$
parameter space	$\mathbb{R}^{305}$	$\mathbb{R}^{1816}$	$\mathbb{R}^{28}$

Cartpole problem



Conclusion

Partially observable domains

- Inference: belief monitoring (HMM, RNN)
- Decision making: Partially observable RL
- Bayes-optimal exploration: Bayesian RL

Some exciting directions for future work

- Data efficient RL, robust RL, self-supervised RL, constrained RL, object-oriented RL, mean-field RL