University of California, Irvine

CS 295: Optimal Control and Reinforcement Learning

Winter 2020

Lecture 11: Partial Observability Methods

Roy Fox
Department of Computer Science
Bren School of Information and Computer Sciences
University of California, Irvine

Today's lecture

- Partially Observable Markov Decision Processes (POMDPs)
- History- and memory-based policies
- Belief-state MDPs
- Recurrent Neural Networks (RNNs)

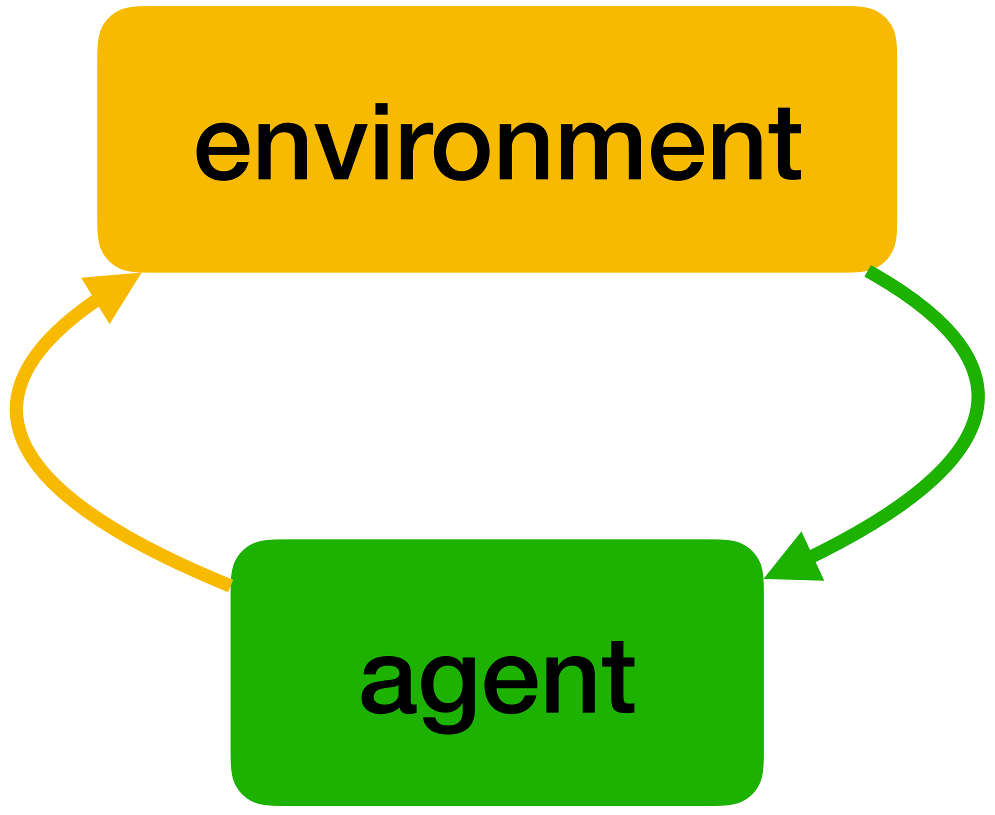
What does the policy depend on?

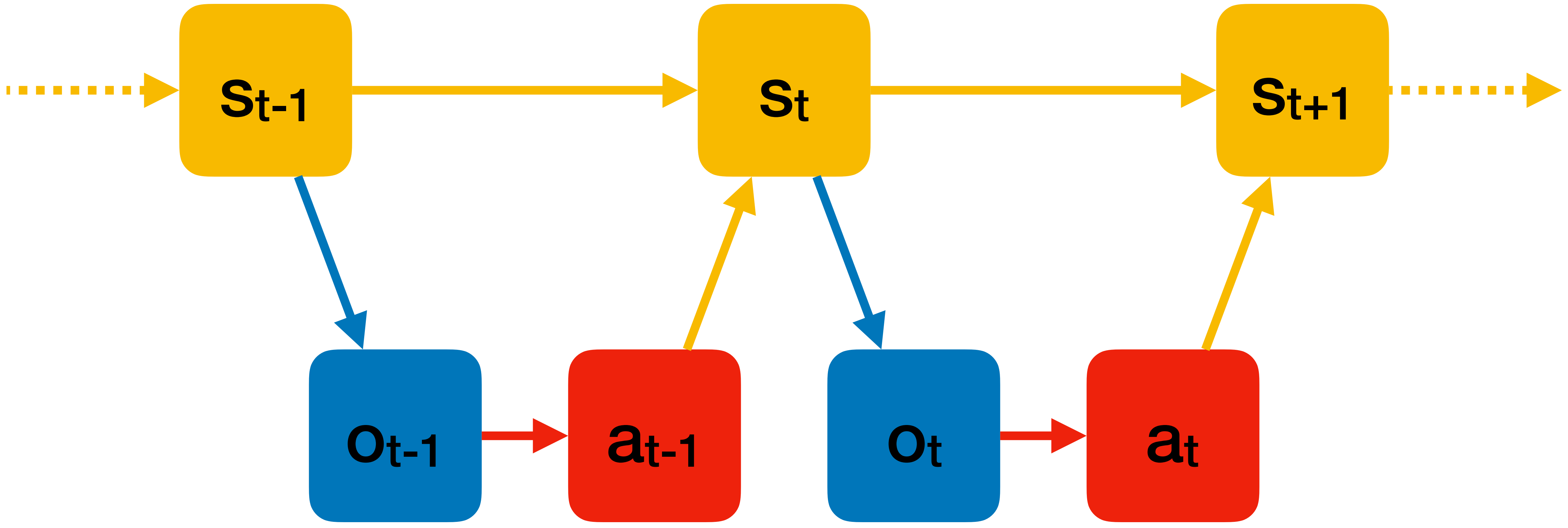
- Minimally, nothing
  - Just an open-loop sequence of actions $a_0, a_1, \ldots$
    - Except, even this depends on a clock
- Typically, the current state $\pi(a_t|s_t)$
- What if the state is not fully observable to the agent's sensors?
  - Completely unobservable → forced open loop
  - Partially observable → $\pi(a_t|o_t)$?

Partially Observable Markov Decision Process (POMDP)

- States $\mathcal{S}$
- Actions $\mathcal{A}$
- Observations $\mathcal{O}$
- Transitions $p(s_{t+1}|s_t, a_t)$
- Emissions $p(o_t|s_t)$
- Rewards $r(s_t, a_t)$

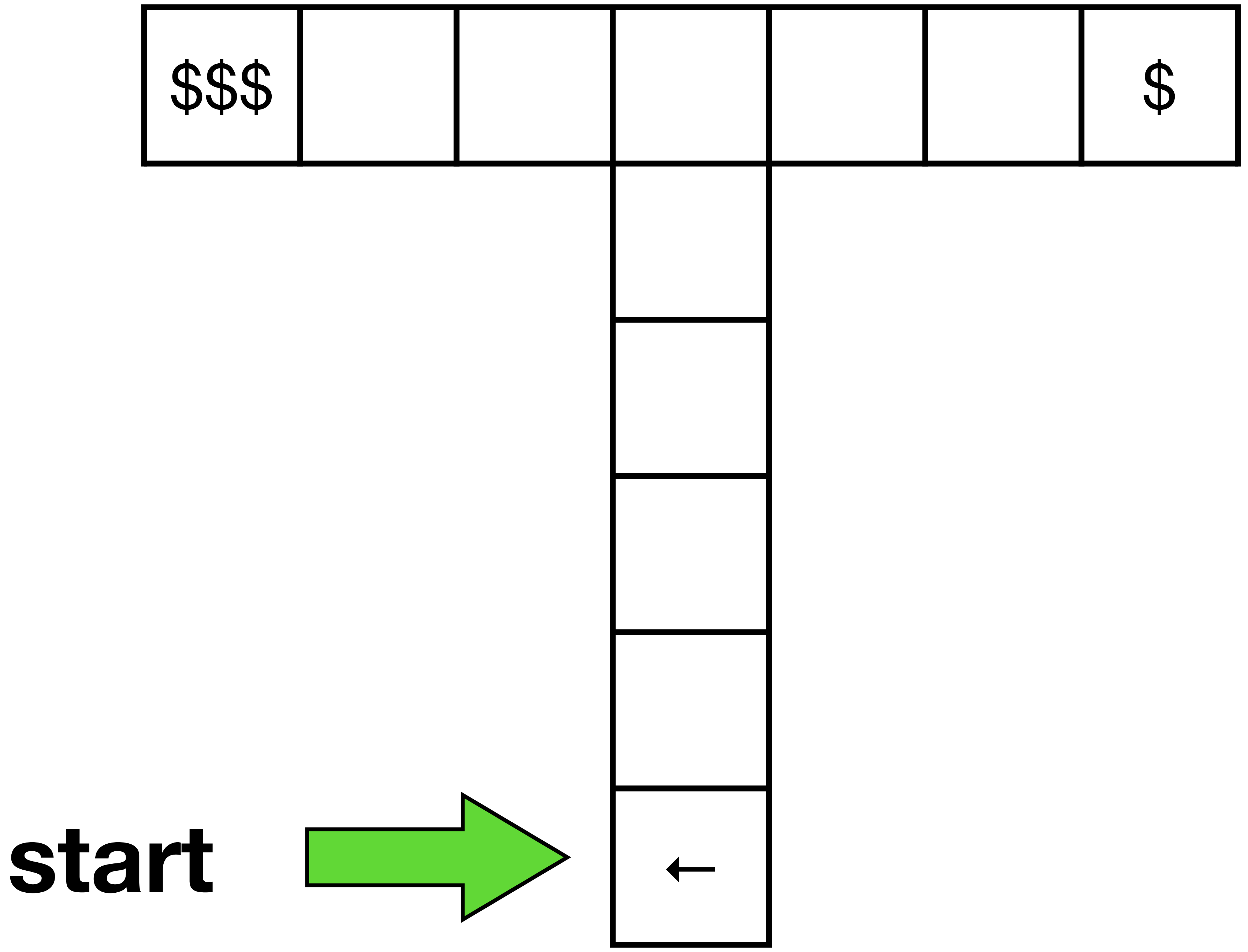
Agent–environment interaction





T-maze domain

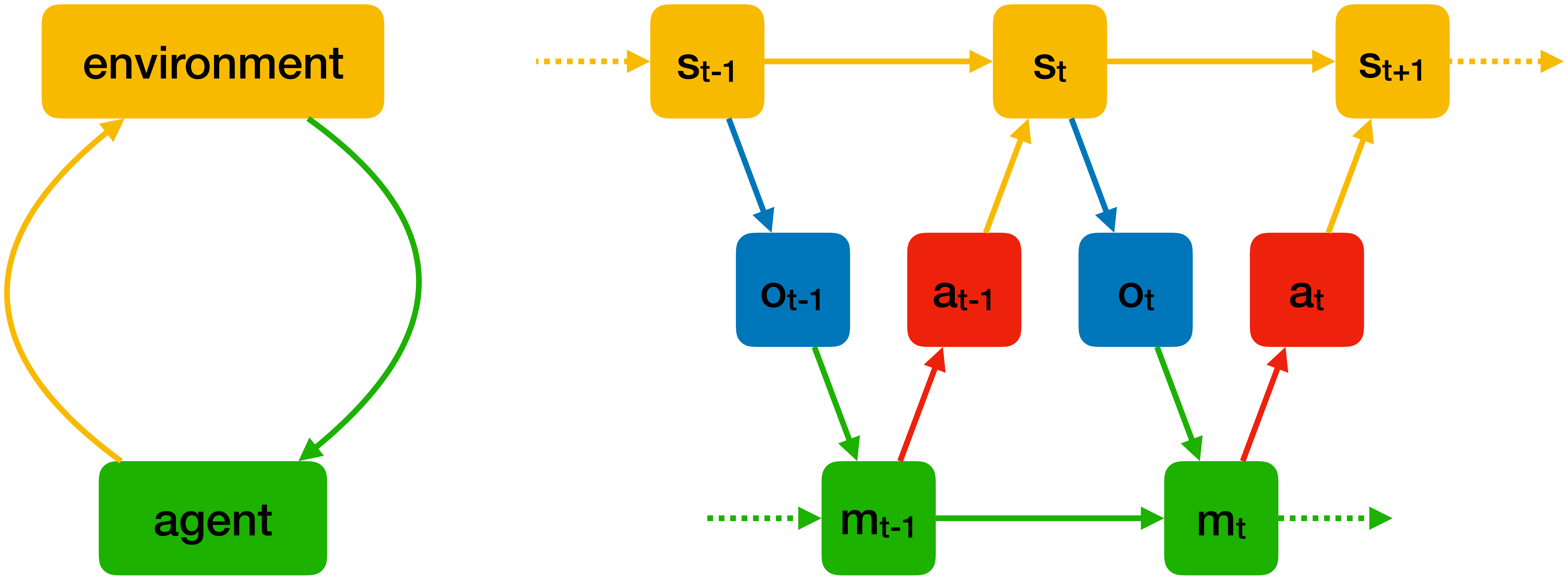
- Observation: current cell
- Memory is needed



What does the policy depend on? (revisited)

- Maximally, the entire observable history $\pi(a_t|h_t = (o_0, a_0, \ldots, o_t))$
  - Do we have to remember past actions?
    - In a deterministic policy, only for computational reasons
    - In a stochastic policy, yes
- Problem: we can't have unbounded memory
- Solution 1: keep a window of observable history $\pi(a_t|o_{t-k+1}, \ldots, o_{t-k})$
- Solution 2: keep a statistic of the observable history $\pi(a_t|m_t)$, with $\pi(m_t|h)$
  - Ideally, update the **memory** statistic sequentially $\pi(m_t|m_{t-1}, o_t)$

Agent–environment interaction



- For simplicity, no edge from a_{t-1} to m_t
  - Either make a_{t-1} explicitly observable in o_t, or roll all the stochasticity into $\pi(m'|m, o)$

So what is memory?

- There's no Markov property in the observable process alone
  - Past observations are informative of future actions
- Filter the observable past to provide more information about the hidden state
- No less important: plan for the future
  - Previously, we needed to trade off short-term with long-term rewards
  - Now we also need to trade off with information-gathering = **active perception**
- In multi-agent: state of the world is incomplete without other agent's memory
  - **Theory of mind**

Tiger domain

- 2 states: which door leads to a tiger (-100 reward) and which to $$$ (+10)

- You can stop and listen: $p(o_t = s_t | s_t) = 0.8$

$p(s_0 = \text{left}) = 0.5; \quad \mathbb{E}[r(s_0, \text{left})] = -45$ → listen → $o_1 = \text{right}$

$p(s_1 = \text{left}) = 0.2; \quad \mathbb{E}[r(s_1, \text{left})] = -12$ → listen → $o_2 = \text{left}$

$p(s_2 = \text{left}) = 0.5; \quad \mathbb{E}[r(s_2, \text{left})] = -45$ → listen → $o_3 = \text{right}$

$p(s_3 = \text{left}) = 0.2; \quad \mathbb{E}[r(s_3, \text{left})] = -12$ → listen → $o_4 = \text{right}$

$$p(s_4 = \text{left}) = \frac{0.04}{0.04 + 0.64} \approx 0.06; \quad \mathbb{E}[r(s_4, \text{left})] \approx 3.5$$

$$p(s_5 = \text{left}) = \approx 0.015; \quad \mathbb{E}[r(s_5, \text{left})] \approx 8.3$$

Sufficient statistics

- A statistic of h is independent of all else given h
  - Satisfying the Markov chain $s - h - m$, and so by the DPI $\mathbb{I}[s;m] \leqslant \mathbb{I}[s;h]$
- A *sufficient* statistic of h for s additionally has $s - m - h$
  - Equivalently, $p(s|m) = p(s|h)$

- A **belief** is a distribution over the state $b(s)$
- The **Bayesian belief** $b(s) = p(s|h)$ is a sufficient statistic of h for s

Computing the Bayesian belief

- In the linear–Gaussian case: the Kalman filter
  - Bayesian belief is Gaussian, precomputed covariance and updated mean

$$\hat{x}_t = \hat{x}'_t + K_t e_t \qquad e_t = y_t - C\hat{x}'_t \qquad \hat{x}'_t = A\hat{x}_{t-1} + Bu_{t-1}$$

- More generally:

$$b'_t(s_{t+1}|b_t, a_t) = \sum_{s_t} b_t(s_t) p(s_{t+1}|s_t, a_t)$$

$$b_t(s_t|o_t) = \frac{b'_t(s_t)p(o_t|s_t)}{p(o_t)} = \frac{b'_t(s_t)p(o_t|s_t)}{\sum_{\bar{s}_t} b'_t(\bar{s}_t)p(o_t|\bar{s}_t)}$$

- This is a deterministic belief-state update, given the observations

Belief-state MDP

- In the linear–quadratic–Gaussian case: **certainty equivalence**
  - Plan using $\hat{x}_t$ as if it was x_t
- More generally, though vastly less useful: belief-state MDP
- States: $\Delta(\mathcal{S})$ Actions: $\mathcal{A}$ Rewards: $r(b_t, a_t) = \sum_{s_t} b_t(s_t) r(s_t, a_t)$
- Transitions: each possible observation o_{t+1} contributes its probability

$$p(o_{t+1}|b_t, a_t) = \sum_{s_t, s_{t+1}} b_t(s_t) p(s_{t+1}|s_t, a_t) p(o_{t+1}|s_{t+1})$$

to the total probability that the belief that follows (b_t, a_t) is

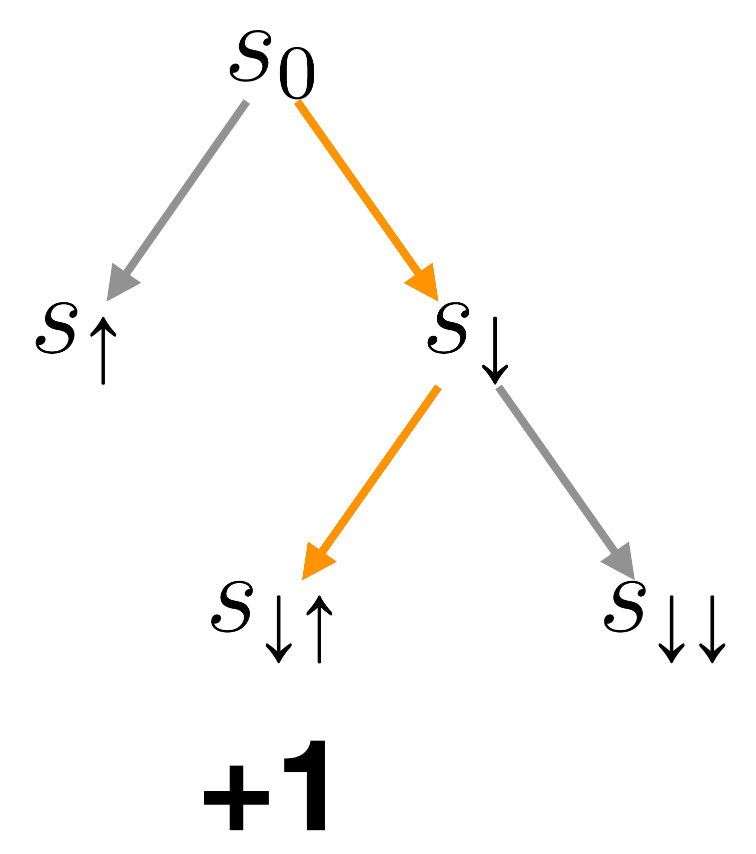
$$b_{t+1}(s_{t+1}) = p(s_{t+1}|b_t, a_t, o_{t+1}) = \sum_{s_t} b_t(s_t)p(s_{t+1}|s_t,a_t)p(o_{t+1}|s_{t+1}) \big/ p(o_{t+1}|b_t,a_t)$$

Why is this hard

- Belief space is continuous, as high-dimensional as the state space
  - **Curse of dimensionality**
  - Beliefs can be multi-modal — how do we even represent them?
- The number of reachable beliefs may grow exponentially with time
  - **Curse of history**
- As we'll see, belief-value function very complex, hard to approximate
- There may not exist optimal stationary deterministic policy

Stationary deterministic policy counterexample

- Assume no observability
- No stationary deterministic policy gets any reward
- Non-stationary policy: ↓, ↑; expected return: +1
    - But non-stationary = observability of a clock
- Stationary stochastic policy: ↓ / ↑ with equal prob.; expected return: +0.25



- Open problem: advantage of non-stationarity → instability of Bellman backup

Filtering with function approximation

- Instead of the Bayesian belief, compute a memory update $m_t = \pi_\theta(m_{t-1}, o_t)$
- Then the action policy can be $\pi_\theta(a_t|m_t) = \pi_\theta(a_t|h_t)$
  - With sequential structure of the history dependence: **Recurrent Neural Network**
- We can back-propagate gradients through the whole sequence
- Unfortunately, gradients tend to vanish / explode for such deep structures
  - Reflecting the challenge of long term coordination of memory updates + actions
  - RNN must remember information to use it, but no memory gradient unless used

Deep RL with RNN policies

- Most Deep RL approaches don't use RNNs
  - Hoping that the current or k recent observations are informative enough
- In principle: RNNs are easy to use with on-policy methods
  - Roll out a complete episode
  - Compute $\nabla_\theta \log \pi_\theta(a_t|h_t)$, with backprop all the way to start of episode
- In practice: episodes may not fit memory, gradients may vanish / explode
- For off-policy methods, using replay buffer or offline data:
  - Use n-step experience, initialize RNN state from buffer (ignoring off-policy effects)

Deep RL as partial observability

- Memory-based policies fail us in Deep RL, where we need them most:
  - Deep RL is inherently partially observable
- Consider what deeper layers get as input
  - High-level / action-driven state features are not Markov!
- Memory management is a huge open problem in Deep RL
  - Actually, in other areas of ML: NLP, time-series analysis, video processing, ...

Recap

- Let policies depend on observable history through memory
- Memory update: Bayesian, approximate, or learned
  - Learning to update memory is one of the biggest open problems in all of ML
- Let policy be stochastic
  - Should memory be stochastic? interesting research question...
- Let policies be non-stationary if possible, otherwise learning may be unstable
  - Time-dependent policies for finite-horizon tasks
  - Periodic policies for periodic tasks