Aalto University
School of Electrical
Engineering

ELEC-E8125 Reinforcement learning
Partially observable Markov decision processes

Joni Pajarinen

21.11.2023

Today

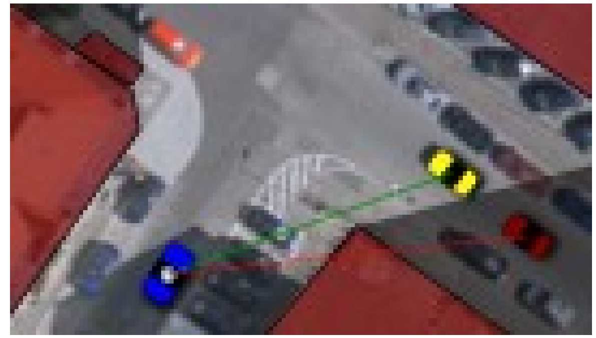
- Partially observable Markov decision processes

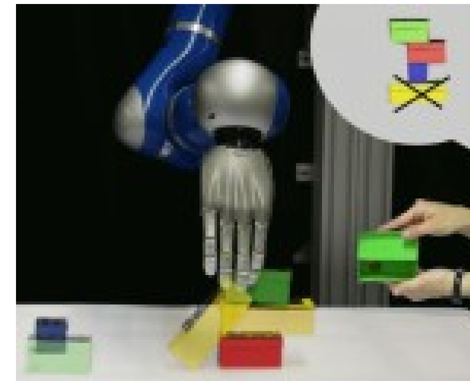
Learning goals

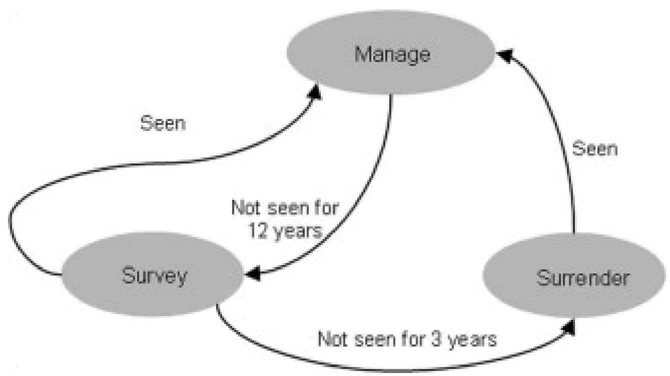
- Understand POMDPs and related concepts
- Be able to explain why solving POMDPs is difficult

Motivation: POMDP application examples

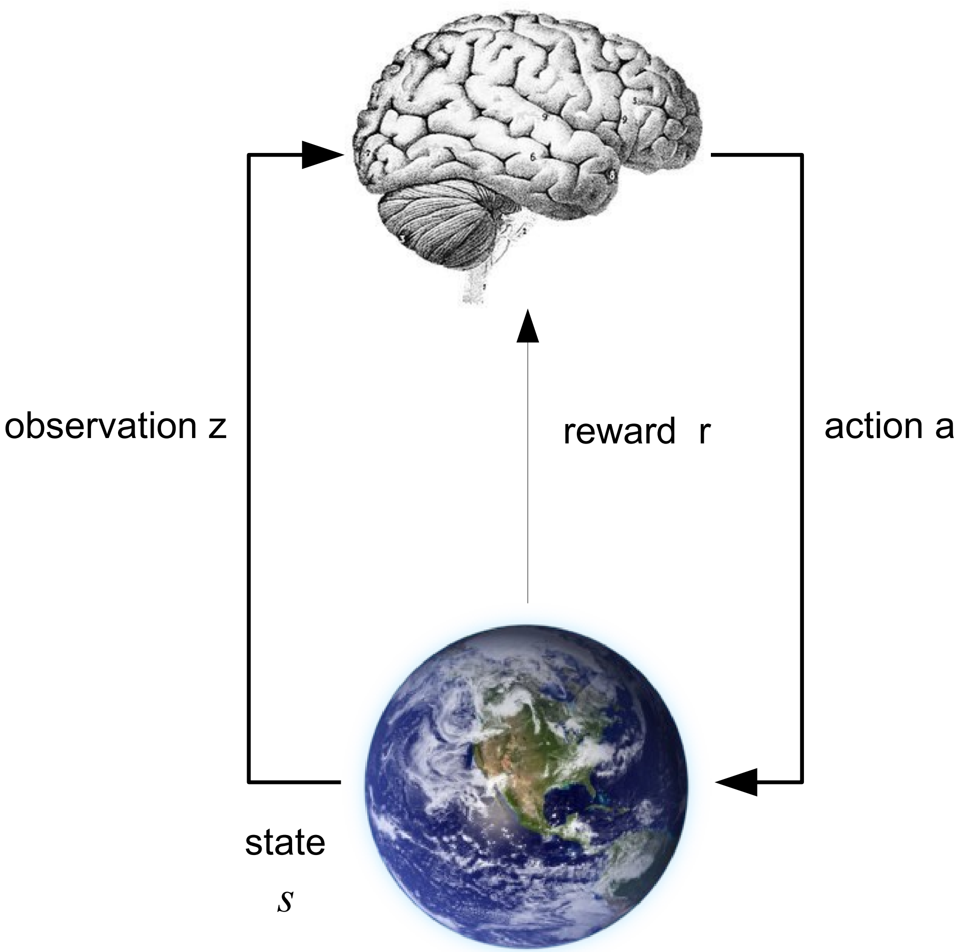
- Autonomous driving
- Human-robot interaction
- Tiger reservation
- Robotic manipulation
- Teaching systems
- Target tracking
- Localization and Navigation
- Handwashing for dementia patients







Markov decision process (MDP)



MDP

Environment observable

$z = s$

Defined by dynamics

$P(s_{t+1} | s_t, a_t)$

And reward function

$r_t = r(s_t, a_t)$

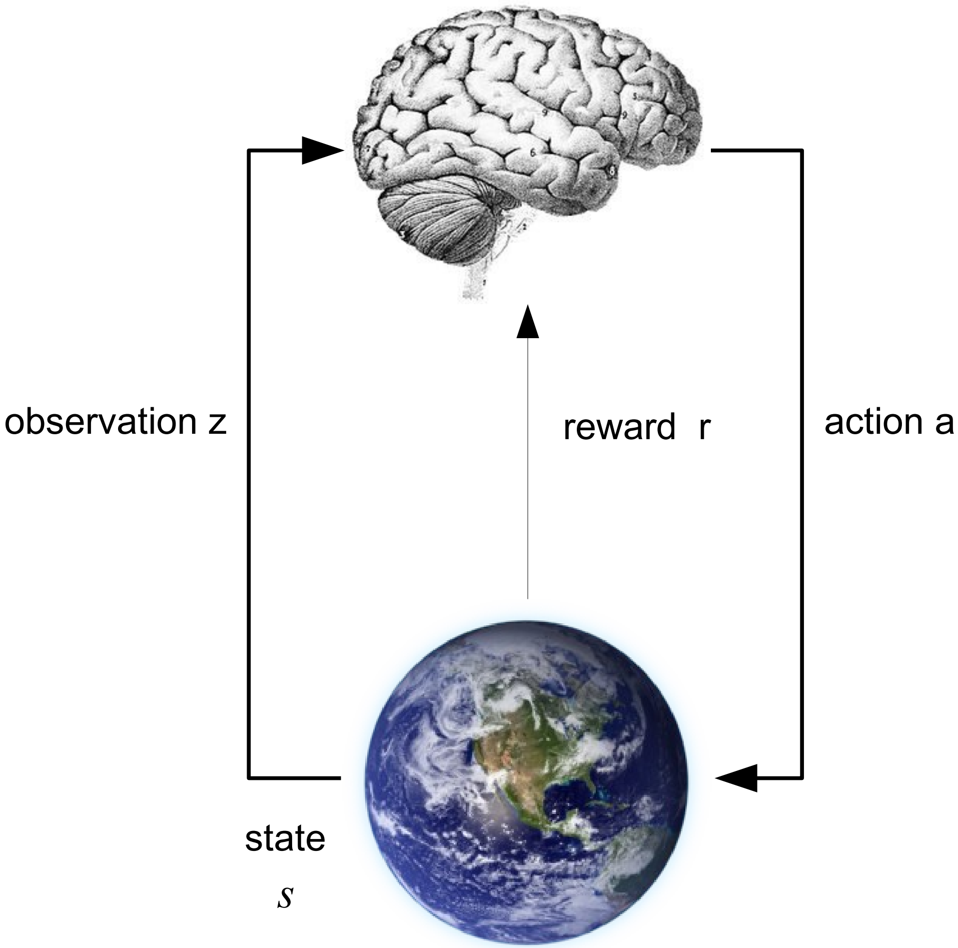
Solution, for example

$a^*_{1,\ldots,T} = \arg\max_{a_1,\ldots,a_T} \sum_{t=1}^{T} r_t$

Represented as policy

$a = \pi(s)$

Partially observable MDP (POMDP)



POMDP
Environment not directly observable

Defined by dynamics

$$P(s_{t+1}|s_t, a_t)$$

reward function

$$r_t = r(s_t, a_t)$$

and observation model

$$\boxed{P(z_{t+1}|s_{t+1}, a_t)}$$

Solution, for example

$$a^*_{1,\ldots,T} = arg\,max_{a_1,\ldots,a_T} E\left[\sum_{t=1}^{T} r_t\right]$$

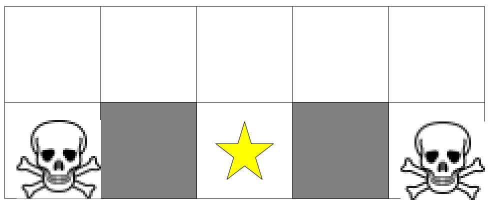
Agent does not directly observe environment state!

Example of partial observability

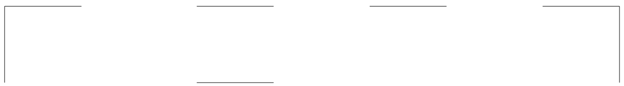
- Observe only adjacent walls
- Starting state unknown, in upper row of grid
- Assume perfect actions

- Give a policy as function of observations!

- Any problems?



Observations:



Can you present a (time-dependent) optimal policy as a tree?

History and information state

- *History* (= Information state) is the sequence of actions and observations until time *t*.

- Information state is Markovian, i.e.,

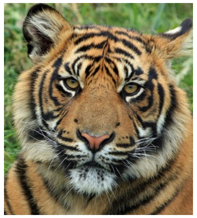
$$P_I\left(I_{t+1} \middle| z_{t+1}, a_t, I_t\right) = P_I\left(I_{t+1} \middle| z_{t+1}, a_t, I_t, I_{t-1}, \ldots, I_0\right)$$

- POMDP thus corresponds to an Information state MDP

What is the largest feline species in the world?

Example: Tiger problem

r=10



r=-100

S = {Tiger left (TL), Tiger right (TR)}

A = {open right, open left, listen}

O = {Hear left (HL), Hear right (HR)}

P(HL|TL)=0.85
P(HR|TL)=0.15
P(HL|TR)=0.15
P(HR|TR)=0.85

?

What kind of policy would be reasonable?

Policy depends on history of observations and actions = information state.

Belief state, belief space MDP

- Belief state = distribution over states
  - Compresses information state

- Belief $b_t(s) \equiv p(s_t = s | I_t)$ ← Can be represented as a vector $\boldsymbol{b} = (b(s_1), b(s_2), \ldots)$

- POMDP corresponds to belief space MDP
- POMDP solution can be structured as
  - State estimation (of belief state) +
  - Policy on belief state

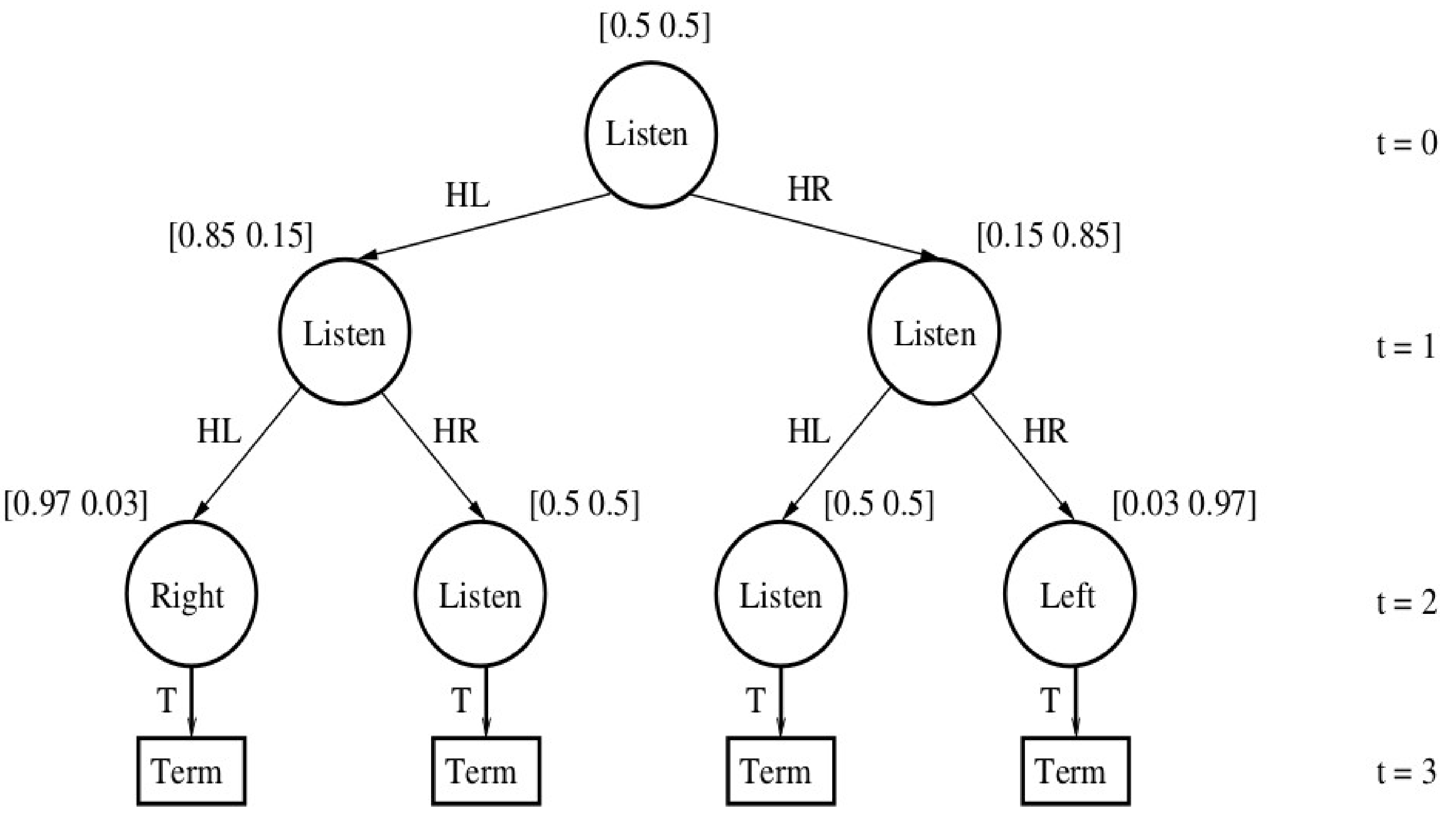
Belief update

Belief updates can be done using state estimation techniques, for example, using a Kalman filter or particle filter. Here we look at updates with discrete states, actions, observations.

$$b_t(s|a,z) = b_{t+1}(s') = \frac{\overbrace{\boxed{P(z|s',a)}}^{\text{"measurement update"}}\ \overbrace{\boxed{\sum_s P(s'|s,a)\, b_t(s)}}^{\text{"prediction"}}}{\underbrace{\boxed{\sum_{s',s} P(s'|s,a)\, P(z|s',a)\, b_t(s)}}_{\text{Normalization factor}}}$$

Tiger example update

Tree search starting from known belief



Computational complexity

- For a known starting belief state and horizon *H*, the size of a full policy tree is $\left(|A||Z|\right)^H$

- Infinite horizon POMDPs thus not possible to optimally solve in general

- Note: Linear systems with Gaussian uncertainty optimally solvable by Kalman filter + optimal control

Value iteration on belief states

- For discrete actions, observations and states, value iteration in principle possible
- No trivial closed form solution (similar to MDP tabulation) because *V(b)* is a function of a continuous variable. In a POMDP, value function is a set of “alpha”-vectors (value function is piecewise linear): $V_t^*(b)=max_i \sum_s \alpha_t^i(s) b(s)$
- Bellman backup for a specific belief b(s) using alpha vectors:

$$V_t^*(b)=max_a E_{b(s)}\left[r(s,a)+\gamma \sum_z \sum_{s'} P(z|s',a) P(s'|s,a) V_{t+1}^*(b_z^a)\right]$$

$$V_T^*(b)=max_a E_{b(s)}\left[r(s,a)\right]=max_a \sum_s b(s) r(s,a)=max_a \sum_s \alpha_T^a(s) b(s)$$

$$V_t^*(b)=max_a E_{b(s)}\left[r(s,a)+\gamma \sum_z max_i \sum_{s'} P(z|s',a) P(s'|s,a) \alpha_{t+1,z}^i(s')\right]$$

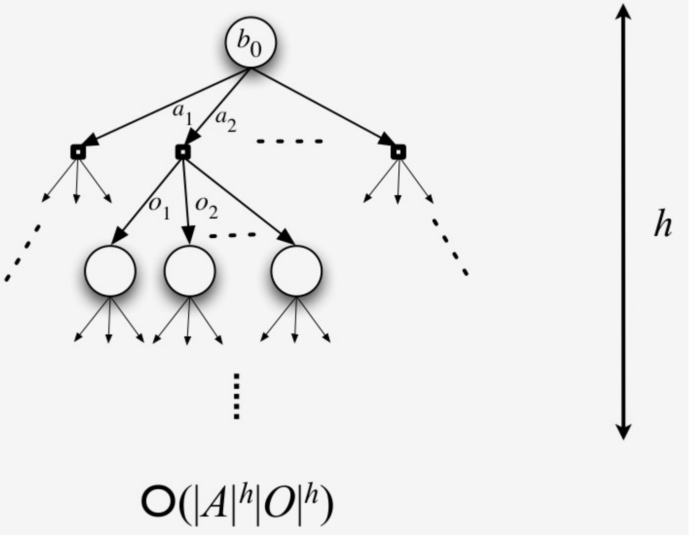
$$= \quad E_{b(s)}\left[\alpha_t^j(s)\right]$$

Backup for belief b(s) creates a new alpha vector j

- Details in https://www.pomdp.org/tutorial

“Curses” of POMDP

- Curse of dimensionality
  - Number of states exponential in number of state variables (similar to MDPs)
  - Complexity of accurate discretization exponential in belief dimensionality, that is, number of states
- Curse of history
  - Complexity exponential in length of history



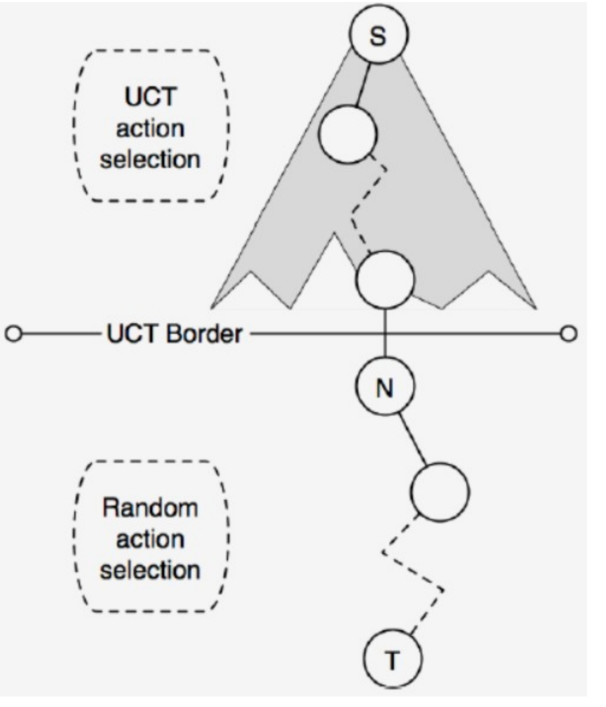
On-line planning with tree search

- Build a search tree from current belief
  - Start from a tree with one node corresponding to current belief
  - Choose a node to expand
  - Choose an action based on (optimistic) heuristic
  - Choose an observation based on another heuristic (or sample randomly)
  - Expand tree and backup back to root
  - Repeat
- Execute the best action
- Update belief
- Repeat

Does search sound familiar?
Have we seen something similar on the course?

Reminder: Monte-Carlo tree search

- From start node *S* choose actions to walk down tree until reaching a leaf node
- Choose an action and create a child node *N* for that action
- Perform a **random** roll-out (take random actions) until end of episode (or for a fixed horizon)
- Record returns as value for *N* and back up value to root



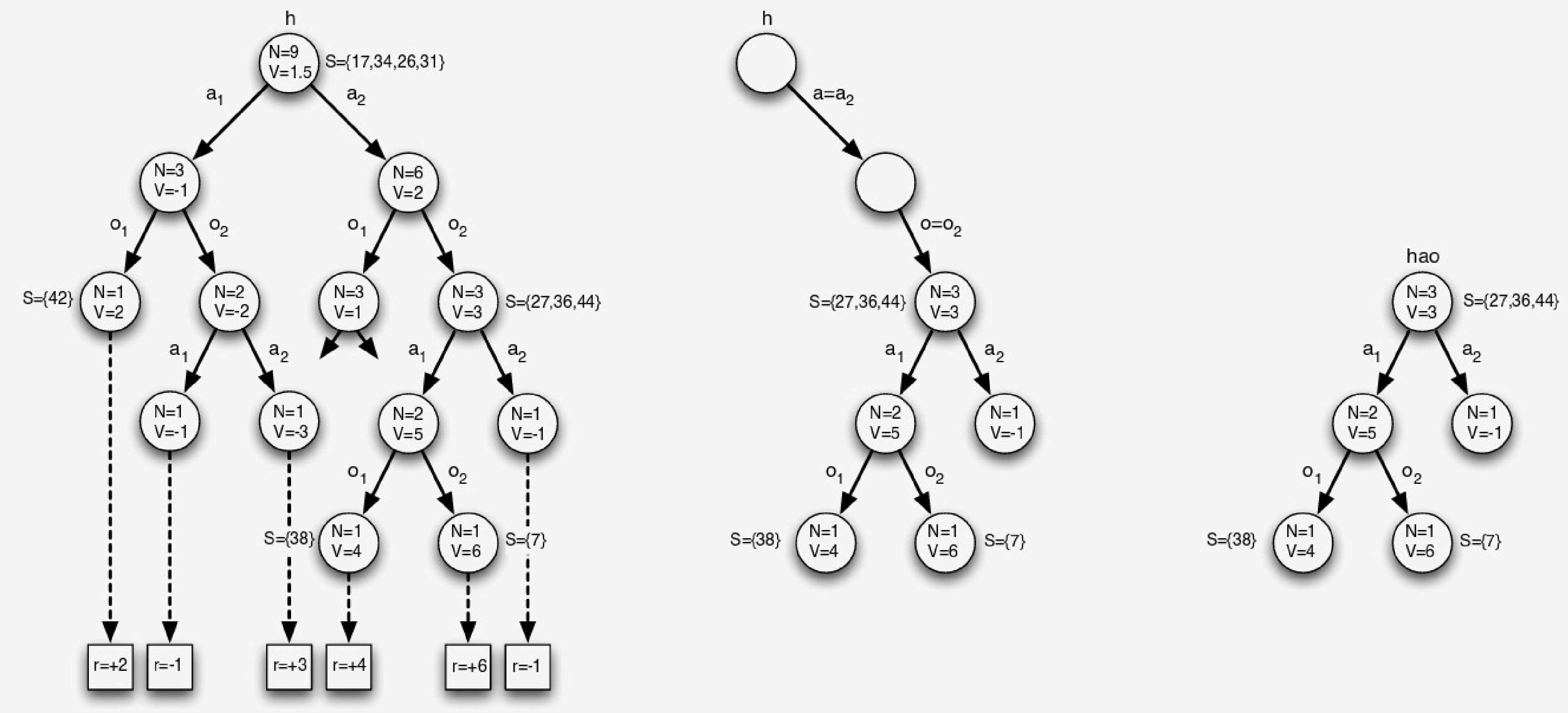
Remember MDPs!

In POMDPs, we do not know the state → how to use MCTS with POMDPs?

From MCTS to POMCP (Silver & Veness, 2010)

- Extension of MCTS to POMDPs
- Search tree node represents a *history* (actions and observations) instead of a state
- Belief state approximated by a *particle filter*
  - After taking an action, update belief by sampling particles by using simulation and keeping ones with true observation
- Each node has visitation count, mean value and particles (states)

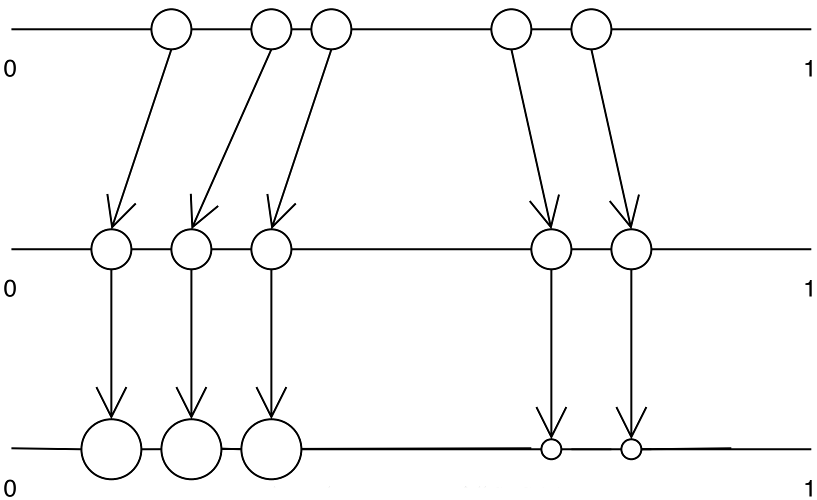
POMCP example



Silver & Veness, 2010

Particle filter for belief updates

Main idea: update *belief*, represented by a *finite set of states (= particles)*, using *action* and *observation*



Using *action* sample next states from current belief

Weight sampled states using observation probabilities and normalize weights

If desired, resample particles to get rid of particles with very small probabilities

POMDPs with large action and observation spaces

- How to handle POMDPs with continuous observations and actions?
- How to handle POMDPs with high-dimensional, e.g. image, observations?
- Possible solutions:
  - Kalman filter + optimal control
  - Discretization / simplification of continuous / complex values
  - Policy gradient / value iteration / actor-critic (Lectures 1 – 8) but how?

Reinforcement learning with POMDPs

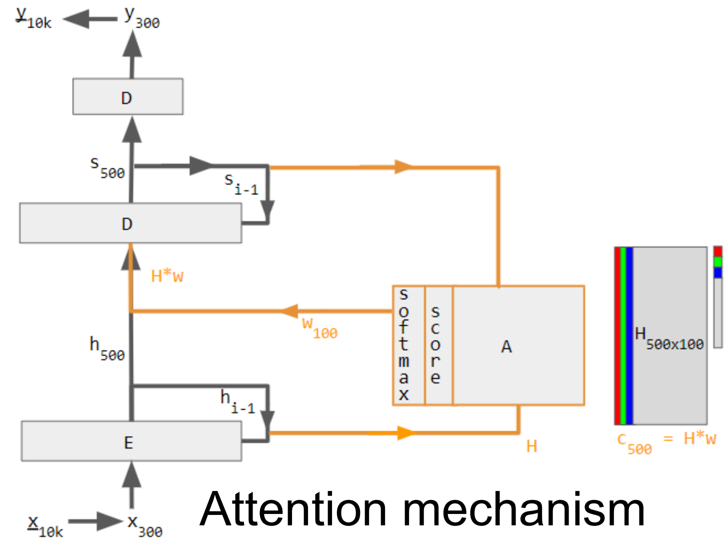
- Sufficient statistics for optimal decision making in POMDPs:
  - Belief, a probability distribution over states $b(s)$
  - Full history of actions and observations $a_0, z_1, \ldots, a_{t-1}, z_t$
- Problems:
  - Belief computation requires dynamics/observation model
  - History grows with each time step
- Solution:
  - Put history into a "*memory representation*" q
  - Replace $\pi(s), V(s), Q(s,a)$ with $\pi(q), V(q), Q(q,a)$ and apply policy gradient, value iteration, actor-critic, or other methods

What kind of memory representation q to use?

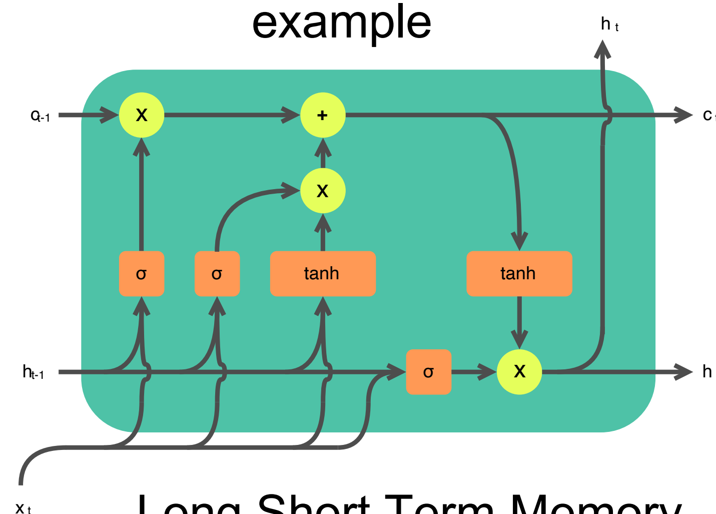
Memory representations

- Direct mapping: $q_t = f(a_1, z_1, \ldots, a_t, z_t)$
  - Truncated history

    $$q_t = f(a_{t-N}, z_{t-N}, \ldots, a_t, z_t)$$
  - Look at only parts of the history: *attention*
- Recurrent memory: $q_t = f(a_t, z_t, q_{t-1})$
  - Memory state part of neural network
  - External memory state
- Many others



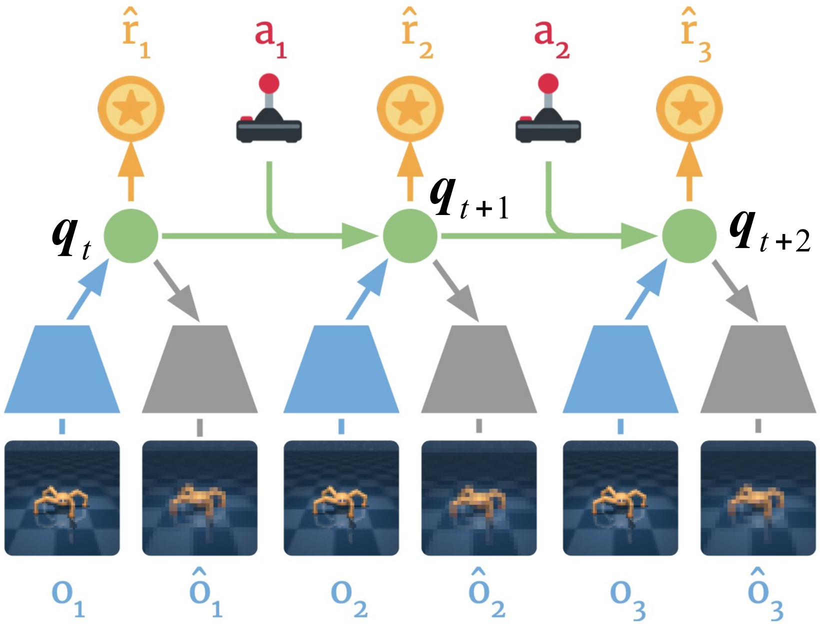
Attention mechanism example



Long Short Term Memory

Remember? Learning latent dynamics

- For real world data tuples $(\boldsymbol{o}_t, \boldsymbol{a}_t, \boldsymbol{r}_t)$ update latent state using
  $f(\boldsymbol{q}_t | \boldsymbol{q}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{o}_{t-1})$
- and to match real world data update latent models:
  $f(\boldsymbol{q}_t | \boldsymbol{q}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{o}_{t-1})$
  $f(\boldsymbol{q}_t | \boldsymbol{q}_{t-1}, \boldsymbol{a}_{t-1})$
  $r(r_t | \boldsymbol{q}_t)$
- Latent state $\boldsymbol{q}_t$
  is a POMDP memory state!



Picture adapted from Dream to Control: Learning Behaviors by Latent Imagination [Hafner et al., ICLR 2019]

Note: $\boldsymbol{o}_t$ and z_t are both commonly used for observations: $z_t = \boldsymbol{o}_t$

Summary

- Partially observable MDPs are MDPs with observations that depend stochastically on state
- POMDP integrates optimal information gathering to optimal decision making
- POMDP = belief-state estimation + belief-state MDP
- POMDPs computationally challenging
  - Bellman equation
  - Tree search
  - Action-observation history / memory representations in reinforcement learning

Current directions in reinforcement learning (RL)

- Challenges: sample efficiency, computational efficiency, safety
- Offline RL
- Hierarchical RL
- Model-based RL
- Exploration in RL
- Multi-agent RL
- Safe RL
- POMDPs
- Deep RL
- Combining different approaches: offline/online, model-free/model-based, planning
- Many other topics