Continual (Never-Ending) Reinforcement Learning

COMP579, Lecture 25

A Mystery: Learning Efficiently from Interaction

- People in their lifetime experience: 100 years x 365 days x 24 hours x 3600 seconds x 640 muscle activations/second = 20 trillion motor actions
- How can we learn from only this amount of data? In a very big world that is partially observable, complex, has other agents in it?
- And while consuming around 2000 calories/day?

Can we get AI agents to learn as efficiently?

RL vs Continual RL

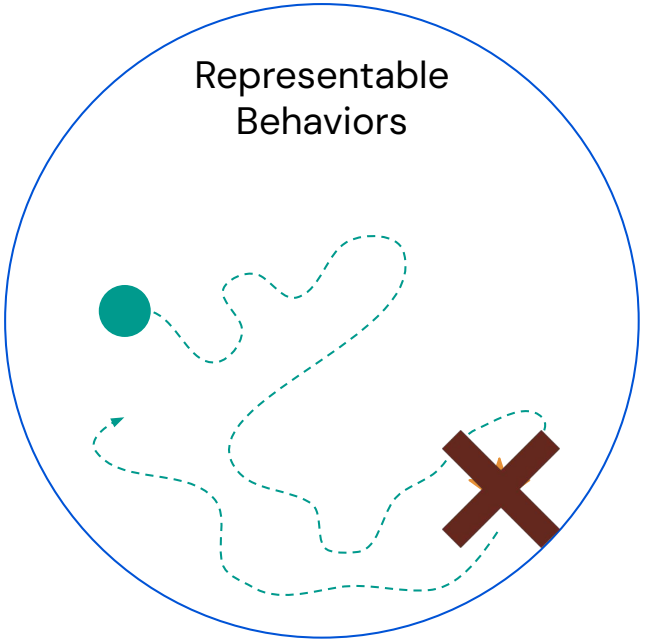
Standard RL:
Learning as Solving

Continual RL:
Learning as Endless Adaptation

Traditional view of RL: a way to solve a problem

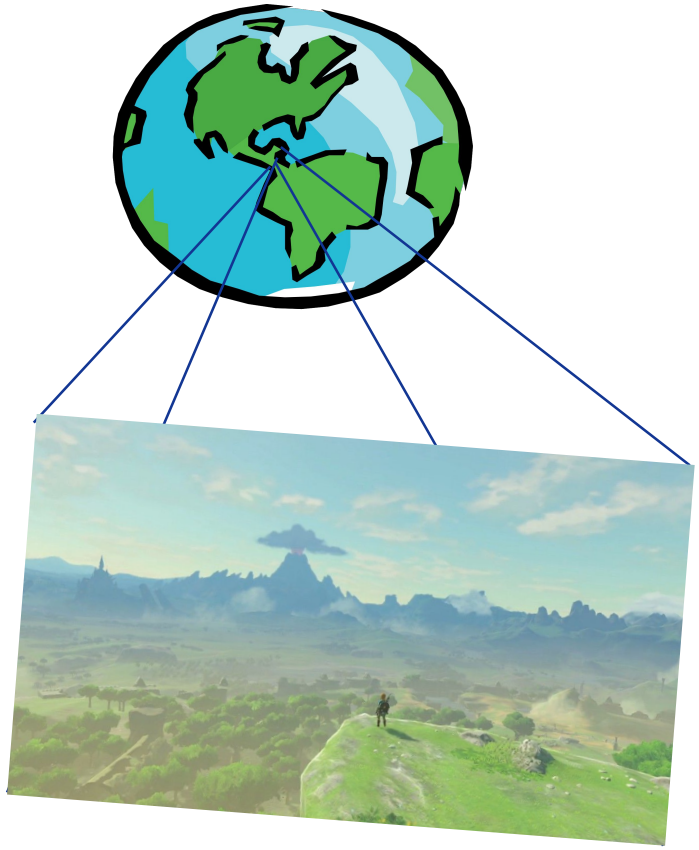
Standard RL:
Learning as Solving

Continual RL agents adapt endlessly!

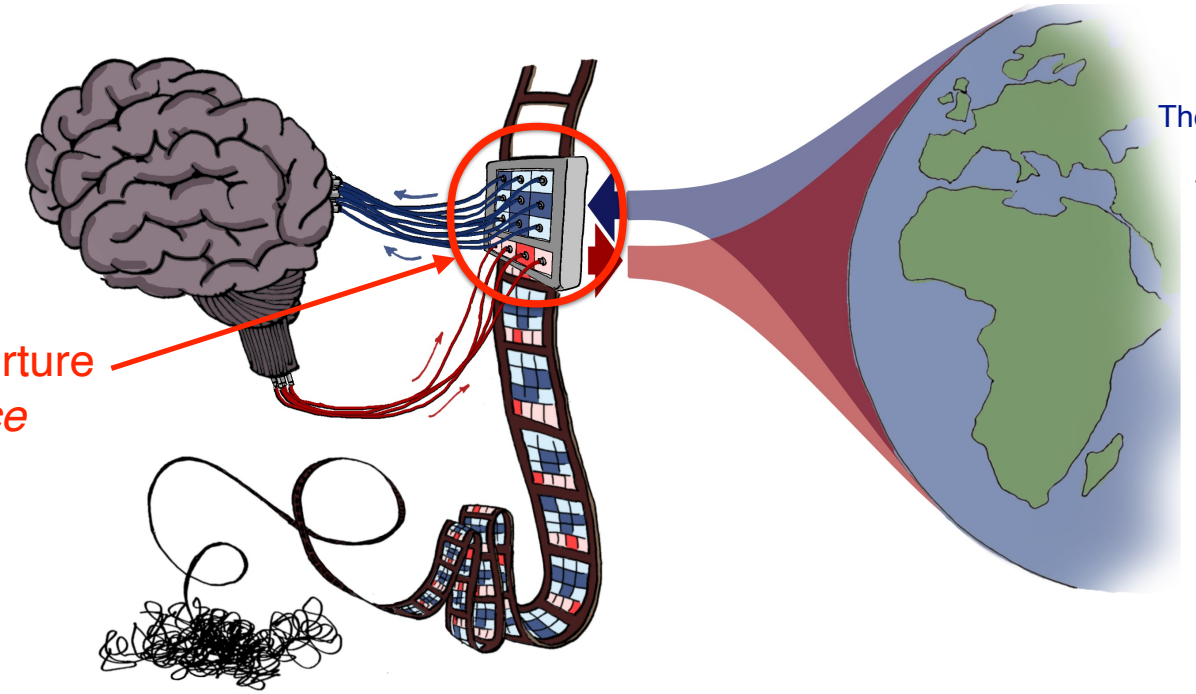


Continual RL:
Learning as Endless Adaptation

Today's Perspective



Aperture principle



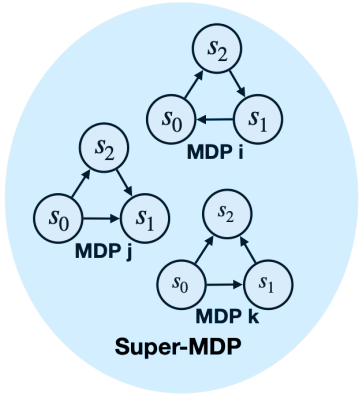
The narrow aperture of *experience*

The Big-World perspective:

The world is much bigger than the agent

It contains many other agents!

Computational and Information Limitations are Important



- If the agent sees the identity of the MDP and the state, it's usual RL
- If the agent sees only the state, we need continual adaptation!
- Cf. Rich Sutton's aperture principle

High-Level View of Agent

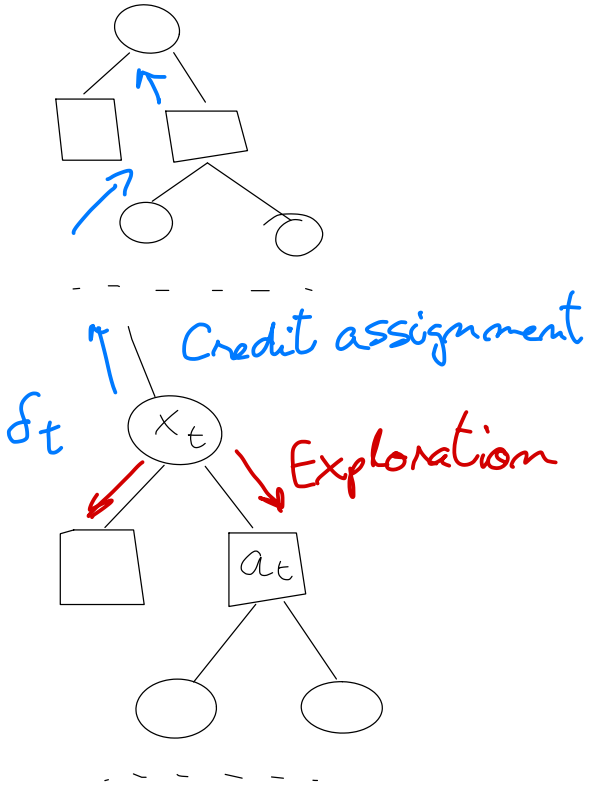
- Agent has *one stream of experience (observations, actions, rewards)* to support all learning processes
- *Agent is "smaller" than the entire environment*
  - Only has time to travel on a specific trajectory
  - Cannot compute arbitrarily fast or remember all the relevant experience in a replay buffer
- *Asynchronous, online learning*
  - The world moves at its own speed
  - Agent has a time scale at which it can perceive, act and learn
  - Agent can also choose the time scale at which it updates its representation

Should We Think This Way?

- Yes!
  - Naturalistic perspective: the conditions in which intelligence has developed in the natural world
  - Realistic perspective: the onus is on the agent to do well *given its current circumstances*
  - Natural for general intelligence, but also consistent with real applications like robotics, health care, energy management...
- No!
  - Are we handicapping ourselves too much?
  - Does this perspective go against the Bitter Lesson?
- Next: explore the implications of this view on algorithmic solutions and theoretical framing

Recall: Cartoon of sequential decision making

- At time t, agent receives an observation from set $\mathcal{X}$ and can choose an action from set $\mathcal{A}$ (think finite for now)
- Goal of the agent is to maximize long-term return



Some observations

- We usually think of the infinite tree of all possible observations and actions
- If there is no structure (ie every node is completely different), there is nothing interesting to learn!
- Bandit assumption: there's a single node
- Markovian assumption: trajectories through the tree *cluster into equivalence classes*, which we call states
- This allows many ways of doing credit assignment: TD(0), TD(λ), Monte Carlo
- Because we cluster an infinite tree into a finite number of clusters, it makes sense to make *recurrence assumptions*: states will be revisited

Sequential Decision Making beyond MDPs

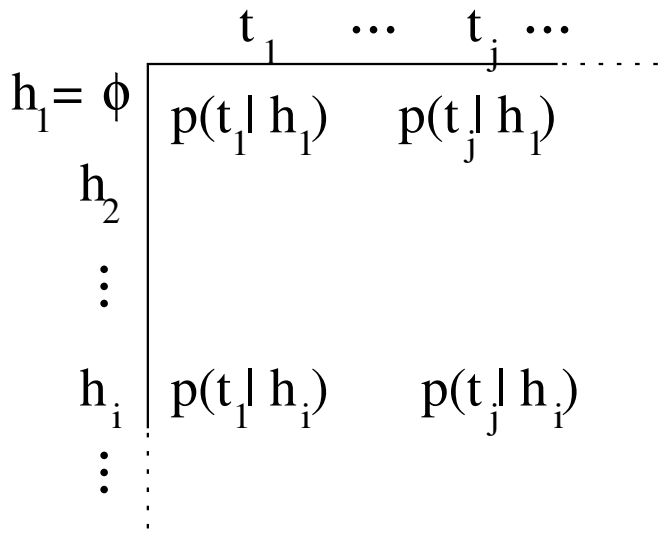
- At decision point t, the agent receives an observation $x_t \in \mathcal{X}$ and chooses an action $a_t \in \mathcal{A}$
- Let t' be the next decision point (as a special case, $t' = t + 1$)
- The agent also receives a reward for this period, with value $r_{t,t'}$, which depends on the agent's action
- There is a designated terminal observation, $\perp$, which ends the agent's trajectory
- Let $t_\perp$ designate the time at which this observation is received
- Assume $t_\perp$ is finite on all trajectories
- *The goal of the agent is to maximize the cumulative return received over its life time, expressed as a sum of rewards*: $\sum r_{t,t'}$ where the first $t = 0$ and the last $t' = t_\perp$
- *A learning algorithm will be evaluated in expectation over instantiations of environment-agent pairs*

Some Interesting Special Cases

- Contextual bandits: x_t always drawn iid from some distribution
- Online regression: the label is the action, the reward is the loss function
- MDPs and POMDPs: $t' = t + 1$, assumptions on how $x_{t'}$ and $r_{t,t'}$ are generated by the environment as a function of x_t and a_t
  - Markovian assumption: trajectories through the tree *cluster into equivalence classes*, which we call states
  - This allows many ways of doing credit assignment: TD(0), TD(λ), Monte Carlo
  - Because we cluster an infinite tree into a finite number of clusters, it makes sense to make *recurrence assumptions*: states will be revisited
- Semi-MDPs: Markovian assumptions on how t', $x_{t'}$ and $r_{t,t'}$ are generated by the environment as a function of x_t and a_t

An example of non-Markovian structure

- Linear predictive state representations (Littman et al, 2001, Singh et al, 2004)
- Make a systems dynamics matrix, with histories as rows and future sequences as columns



- Assume the *systems dynamics matrix has finite rank*

Leveraging finite rank predictive state representations

- Use ideas from linear algebra!
- Incrementally fill the matrix based on the experience, do SVD-like computations to find the rank and basis
- One can show that k-order Markov models and finite partially observable MDPs, are equivalent to linear PSRs of order at most k, which require a polynomial-size matrix
- All probabilities of future queries can be computed using this representation and matrix operations
- Think of the set of core rows/columns as "agent state"

Generalizing to "Small Agent" Perspective

- Agent's trajectory will cover a minuscule fraction of all possible trajectories
- Notions of recurrence like in MDPs no longer make sense (the agent is really transient)
- Yet the agent still needs to do as well as possible *along its current trajectory*
- So it needs to *construct a knowledge representation that allows it to generalize quickly*
- *Agent state:* the internal representation used by the agent to predict and act
- Agent state will have to be learned
- *The representation will inherently be lossy/imperfect*

An Evolution of Ideas

- Dynamic programming: agent needs to find an optimal policy at all states (allowed by Markovian structure)
- Reinforcement learning: agent focuses on states that are actually encountered during its experience

  This is what allows tackling large environments like Go!
- One step further: agent's learned state representation should enable it to do well in the future on the trajectory that will be encountered!

What is useful structure?

- The agent needs to be able to do induction: estimate potential future return from its past history
- We want to continue leveraging the compositionality of returns: $G_t = r_{t,t'} + G_{t'}$

An Evolution of Ideas

- Dynamic programming: agent needs to find an optimal policy at all states
- Reinforcement learning: agent focuses on states that are actually encountered during its experience

  This is what allows tackling large environments like Go!
- One step further: agent's learning should enable it to do well in the future on the trajectory that will be encountered!
- *Optimality is not an absolute notion*, but relative to the agent's circumstances, available data and capacity
- Eg child cooking at home vs chef

Ingredients for characterizing agent performance

- Agent has a particular class of policies (eg due to computational-constraints)
- Kumar, Marklund et al (2023):
  - Learning target is what the agent aims to estimate (eg optimal policy)
  - Regret upper bound is the performance of the best policy given perfect knowledge of the learning target
- Morrill et al (2020): hindsight, sequential rationality (useful for single trajectory)
- *Data-dependent regret* (Abernethy et al, 2008): upper bound depends on the observed data

Can we mix these ingredients into a clear, crisp, optimizable regret notion?