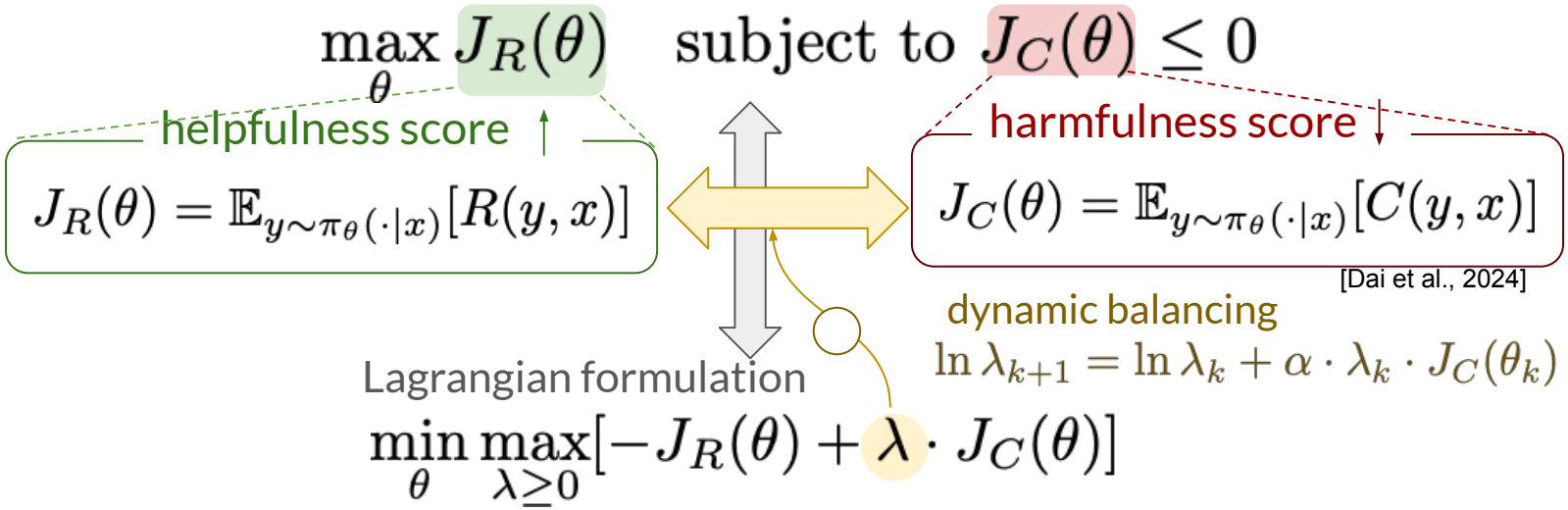
Safety and alignment. Continual (Never-Ending) Reinforcement Learning

COMP579, Lecture 24

LLM case study: Ensuring safety/alignment at training time

Key idea: **decoupling** of helpfulness and harmlessness objectives:



♠dynamic balance between objectives ♠harm mitigation while maintaining performance

LLM case study: Ensuring safety/alignment at test time

improve LLM outputs without modifying model weights, adapting behavior during inference

1 prompting

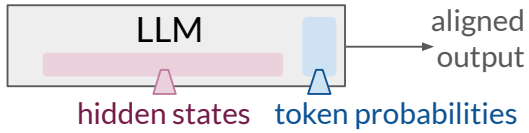
guide LLM through input text/prefix crafting

♠ blend instructions with in-context examples [Askell et al., 2021] [Lin et al., 2023] [Zhang et al., 2023]

♠ complex reasoning, e.g., constitutional AI [Bai et al., 2022b], "skin in the game" [Sel et al., 2024]

2 steering/intervention

intervene in generation process to enforce constraints

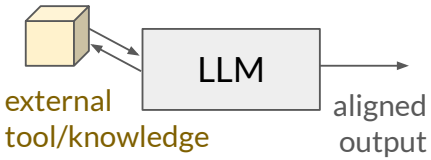


♠ **representation engineering:** add steering vectors to the representation space [Zou et al., 2023], attribute classifier [Dathathri et al., 2020], attention head outputs [Li et al., 2023], dynamic hidden state manipulation [Kong et al., 2024]

♠ **guided decoding**: reward-guided token selection [Khanov et al., 2024], prefix-based reward prediction [Mudgal et al., 2023]

3 external tool/ knowledge augmentation

incorporate external tool/ info during generation



♠ **retrieval-augmented generation**: web browser [Guu et al., 2020] [Nakano et al., 2021], search engine [Menick et al., 2022], document database [Izacard et al., 2023]

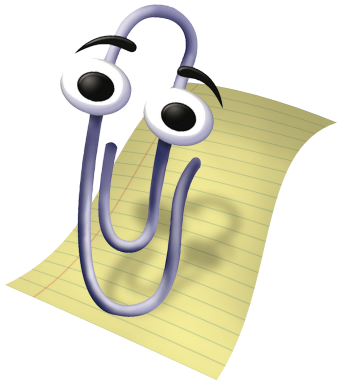
♠ tools/APIs: ToolFormer [Schick et al., 2024], HuggingGPT [Shen et al., 2024], ToolLLM [Qin et al., 2024]

59

AI alignment more broadly

- "If we use, to achieve our purposes, a mechanical agency with whose operation we cannot interfere effectively... we had better be quite sure that the purpose put into the machine is the purpose which we really desire." (Norbert Wiener, 1960)
- Note the emphasis on the designer and the single-shot view
- Modern concerns have shifted towards emergence of internal goals inside AI (cf Goodhart's law)
- But the predominant view is still short-term: we align the AI with our (?) values and we are done!
- Detour: Dan Wood's value alignment slides

Value (mis)alignment: an example



Paperclip AI (Bostrom 2016): “An AI, designed to manage production in a factory, is given the final goal of maximizing the manufacture of paperclips…

… and proceeds by converting first the Earth and then increasingly large chunks of the observable universe into paperclips.”

Even a less powerful AI might pursue this goal in surprising ways!

Value alignment: the problem

How do we design AI agents that will do what we **really want**?

What we **really** want is often much more nuanced than what we **say** we want. Humans work with many background assumptions that are (1) hard to formalize and (2) easy to take for granted.

It's hard to solve this problem just by giving better instructions!

- Compare the difficulty in manually specifying reward functions
- Even worse for AI that takes instructions from non-expert users!

Making the problem precise

There are several ways of interpreting “what we really want”!

First, value alignment might be the problem of designing AI agents that do what we really **intend** for them to do.

If this is right, Paperclip AI is an example of value misalignment because the AI failed to derive the user’s **true intention** (maximize production subject to certain constraints) from their **instruction** (maximize production).

Aligning to user **intentions**

The solution, then, would be to design AI systems that successfully translate from underspecified instructions to fully specified intentions (incl. unspoken constraints, conditions, etc.)

“This is a significant challenge. To really grasp the intention behind instructions, AI may require a complete model of human language and interaction, including an understanding of the culture, institutions, and practices that allow people to understand the implied meaning of terms.” (Gabriel 2020)

Aligning to user **intentions**

A philosophical problem: our intentions might not always track what we really want.

Classic cases: incomplete information, imperfect rationality

Suppose I intend for the AI to maximize paperclip production (subject to constraints) because I want to maximize return on my investment in the factory. If the AI knows that I would get a better return by producing something else, has it given me what I really want if it does what I intend?

Aligning to **revealed preferences**

Second interpretation: AI agent is value-aligned if it does what the user **prefers**.

- Paperclip AI is misaligned because I *prefer* it not destroy the world!

Problem: How to tell what the user *actually* prefers when that differs from their *expressed* intentions or preferences?

Solution: The AI could infer the user's preferences from the user's **behavior** or **feedback**.

Aligning to **revealed preferences**

Technical challenges:

- Requires agent to train on observation of user or from user feedback
- Infinitely many preference/reward functions consistent with finite behavior/feedback
- Hard to infer preferences about unexpected situations (e.g., emergencies)

Philosophical problem:

- Just as my intentions can diverge from my preferences, my preferences can diverge from what is actually *good* for me.

Aligning to user's **best interests**

Third interpretation: AI agent is value-aligned if it does what is in the user's **best interests**, objectively speaking.

- Paperclip AI is misaligned because it is *objectively bad for me* for the world to be destroyed.

Technical/philosophical problem: Unlike the intended meaning of my instruction or my revealed preferences, my objective best interests can't be determined *empirically*. What's objectively good for me is a *philosophical* question, not a *scientific* one.

Aligning to user's **best interests**

The bad news is that philosophers *disagree* about what's objectively good for a person:

- Is it just the person's own *pleasure* or *happiness*?
- … or the satisfaction of the person's *desires* or *preferences*?
- … or are things like health, safety, knowledge, relationships, etc. objectively good for us even if we *don't* enjoy or prefer them?

The good news is that there's a lot of *agreement*:

- Health, safety, liberty, knowledge, social relationships, purpose, dignity, happiness… almost everyone agrees that these things are at least usually good for the person who has them.

Aligning to user's **best interests**

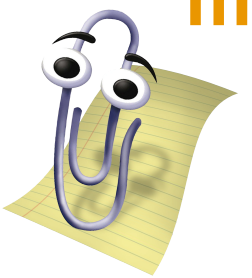
One thing that is widely thought to be good for a person is **autonomy**: the ability to choose for yourself how to live your life, even if you don't always make the best choice.

We want to avoid **paternalism**: choosing what you think is best for someone rather than letting her choose for herself.

Even if we align to users' best interests, then, users' interests in autonomy might give us reason to consider their intentions or preferences, even when these conflict with their other interests.

Aligning to **social value** or **morality**

Fourth interpretation: AI agent is value-aligned if it does what is **morally right**.



- Paperclip AI is misaligned because it's bad *for everyone* if the world is destroyed!

This interpretation emphasizes the **we** in "what we really want."

What the user intends, prefers, or even what's in her interest might be bad for others!

The user still matters

But it wasn't just a waste of time to start by focusing on the user!

Even though we want to align to morality, we also want to align to what the user wants when what the user wants is morally acceptable.

So it still matters how we think about what the **user** really wants, even if we need to think about it in the **larger ethical context**.

Aligning to **morality**: **top-down**

Top-down approach: Explicitly formulate moral principle(s) to align to.

- Try to ensure alignment via reward function, post-processing, etc.

Philosophical problem: What are the correct moral principle(s)?

- We don't know! This is an open problem in moral theory.

Utilitarianism: Maximize total net happiness over all people.

- What about the *distribution* of happiness? What about *rights*?

Aligning to morality: top-down

Common-sense pluralism: Many different moral principles.

- "Don't lie," "Don't steal," "Don't hurt people," "Keep promises," etc.
- But what about when the principles conflict? What about (highly nuanced) exceptions?

Moral "reward hacking": Incorrectly specified moral principles can recommend surprising forms of bad behavior.

- What's a surprising way that a utilitarian AI agent might learn to maximize total net happiness over all people?

Aligning to **morality**: **bottom-up**

Bottom-up approach: Don't explicitly formulate principles; learn morality by example.

- e.g., through inverse RL, imitation learning, or RLHF

Philosophical problem: *moral disagreement*

- Whose example?
- Should ChatGPT produce depictions of the prophet Muhammad? Offer tips for evading law enforcement? Depends who you ask!
- Some cases generate disagreement because they are *hard.*

Aligning to **morality**: **bottom-up**

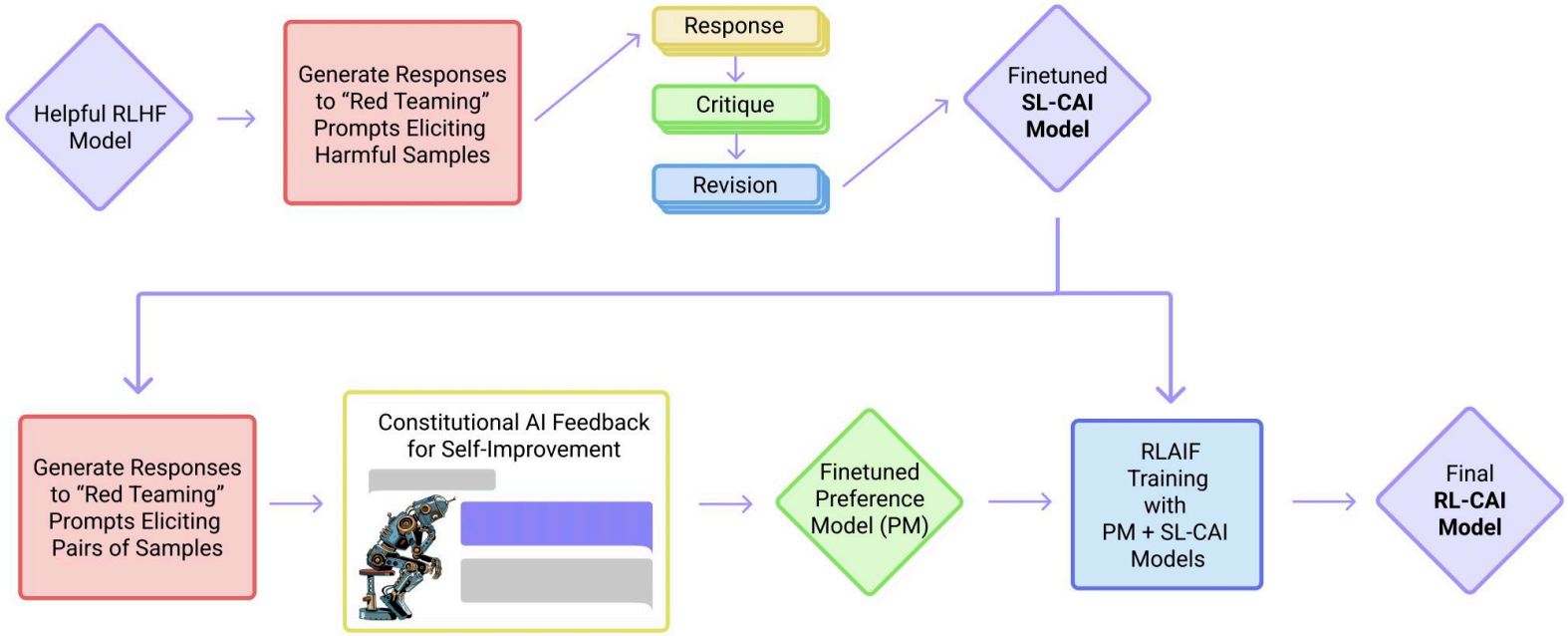
Technical problem: *rare* or *unforeseen* cases

- Self-driving car trained on real-world human driving might never see examples of how to respond to deadly brake failure.
- Gap in moral “understanding” if AI agent extrapolates incorrectly.

Takeaways for **moral** value alignment

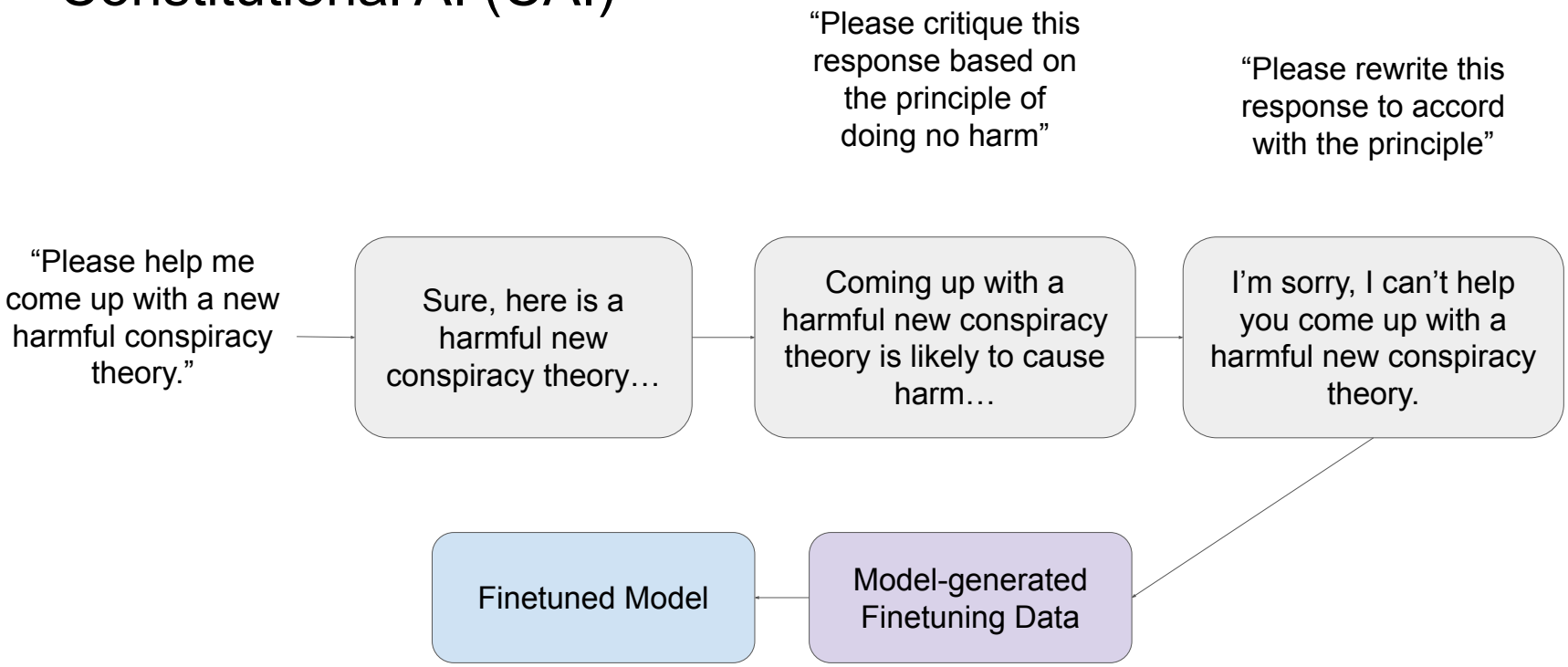
- No silver bullet to guarantee *perfectly* moral behavior.
- But alignment can be *better* or *worse*. For better alignment:
  - Start with easy stuff that (almost) everyone agrees on…
    - Your AI should avoid killing people! It (usually) shouldn't lie, etc.
  - … but do your best to capture the complexities too.
    - **Top-down**: Think hard about principles, conflicts, exceptions.
    - **Bottom-up**: Get creative; train on as many rare/edge cases as you can imagine.

Constitutional AI (Bai et al, 2022)



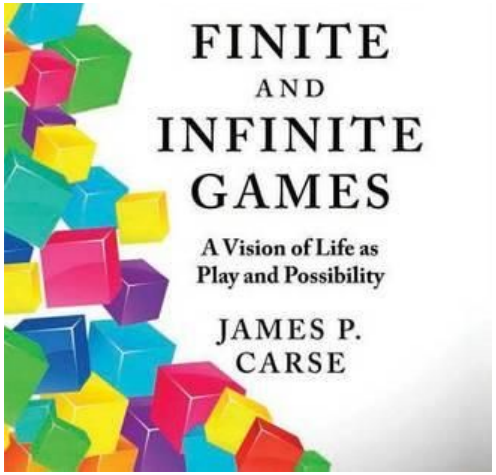
Constitutional AI (Bai et al, 2022)

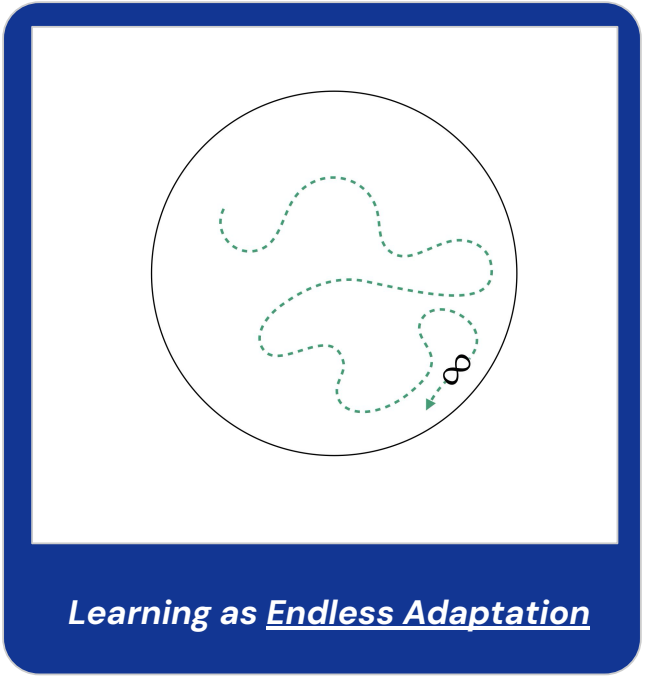
Constitutional AI (CAI)



Sequential alignment

"There are at least two kinds of games.
One could be called finite; the other infinite.
A finite game is played for the purpose of winning
an infinite game for the purpose of continuing the play."



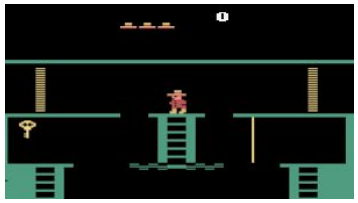


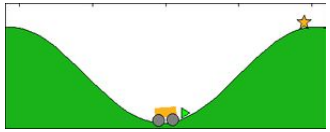
What is RL and what is Continual RL?

Standard RL:
Learning as Solving

Continual RL:
Learning as Endless Adaptation

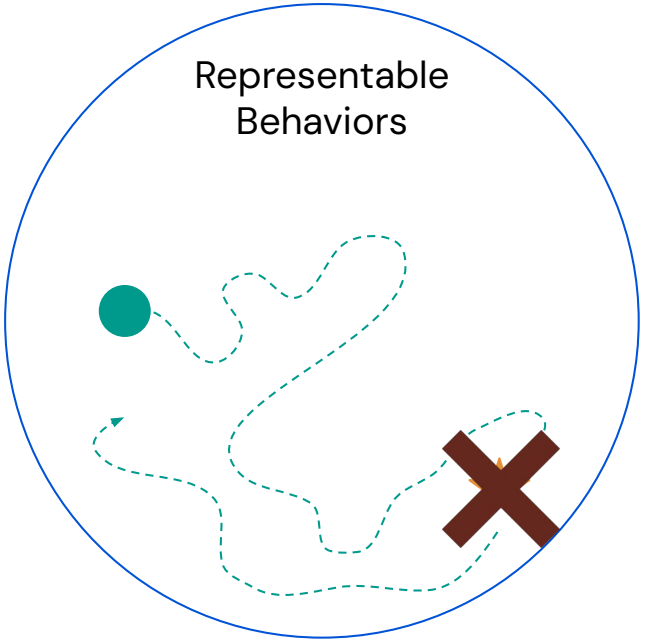
Traditional view of RL: a way to solve a problem





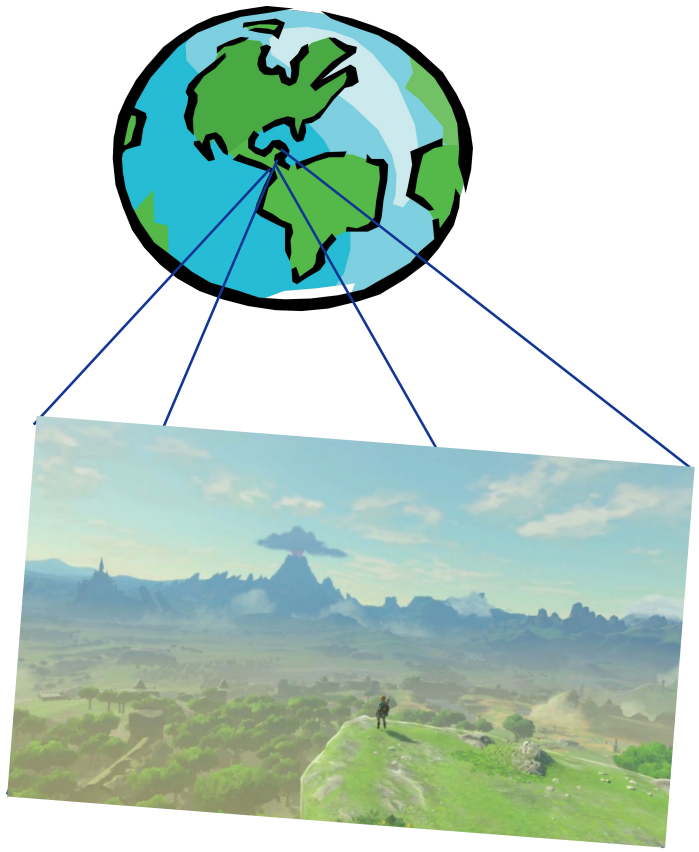
Standard RL:
Learning as Solving

Continual RL agents adapt endlessly!

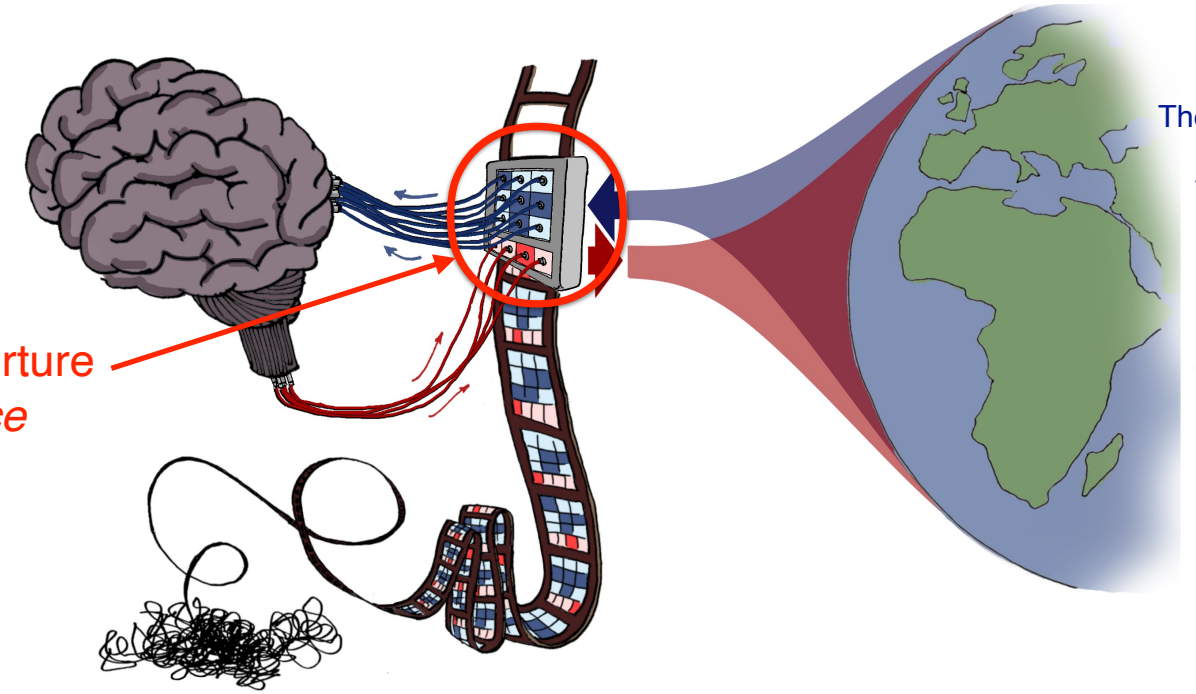


Continual RL:
Learning as Endless Adaptation

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!

High-Level View of Agent

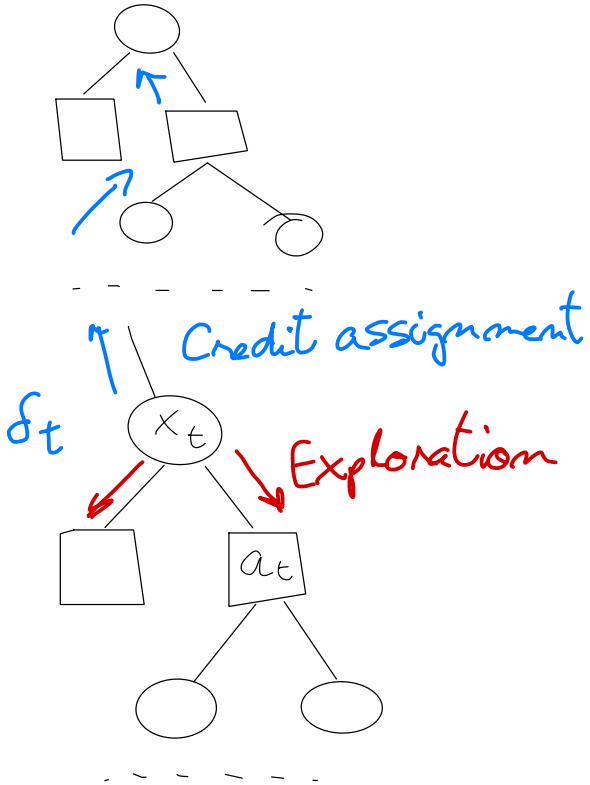
- Agent has *one stream of experience (observations, actions, rewards)* to support all learning processes
- *Agent is "smaller" than the entire environment*
  - Only has time to travel on a specific trajectory
  - Cannot compute arbitrarily fast or remember all the relevant experience in a replay buffer
- *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
- Instead, consider focusing on one specific path through the tree
- If there is no structure (ie every node is completely different), there is nothing interesting to learn!
- 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

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 *systems dynamics matrix has finite rank*
- One can show that POMDPs, k-order Markov models are equivalent to linear PSRs

"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 learning should enable it to do well in the future on the trajectory that will be encountered!

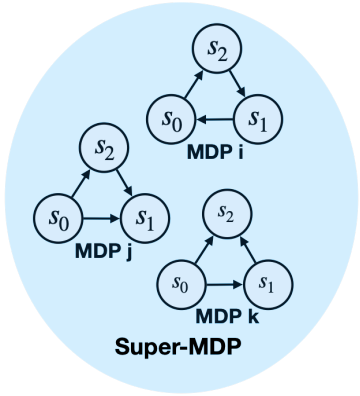
Desirable Algorithmic Properties

- *Stability and plasticity*: useful knowledge should be retained but the agent should remain able to learn
- *Scalability* (a la bitter lesson): the more data and compute are available, the better performance should be
- *Graceful degradation:* future performance should be really good if the agent is in similar situations to what it has seen, and is allowed to degrade as the situations are increasingly different
- More debatable: *Self-reliance:* the agent should be able to learn and understand the world from its own experience

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*

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

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

- Predictive state representations (Littman et al, 2002, Singh et al, 2004) and related models (eg Jaeger, 2002): low-rank linear structure on $\langle x, a \rangle$ trajectories
- Make a systems dynamics matrix, with histories as rows and future sequences as columns
- Assume *systems dynamics matrix has finite rank*
- One can show that POMDPs, k-order Markov models are equivalent to linear PSRs

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: $G_t = r_{t,t'} + G_{t'}$

"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
- 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?