Temporal Difference

Konpat Preechakul
Chulalongkorn University
August 2019

Monte Carlo prediction cons

- Needs a “whole episode” to make progress

- Is there a better way? Get a feedback after **each step**

Do we need a whole trajectory?

- Can we start learning as soon as there is a **“surprise”?**

Surprise = |reality - belief|

- We adjust our belief to the reality

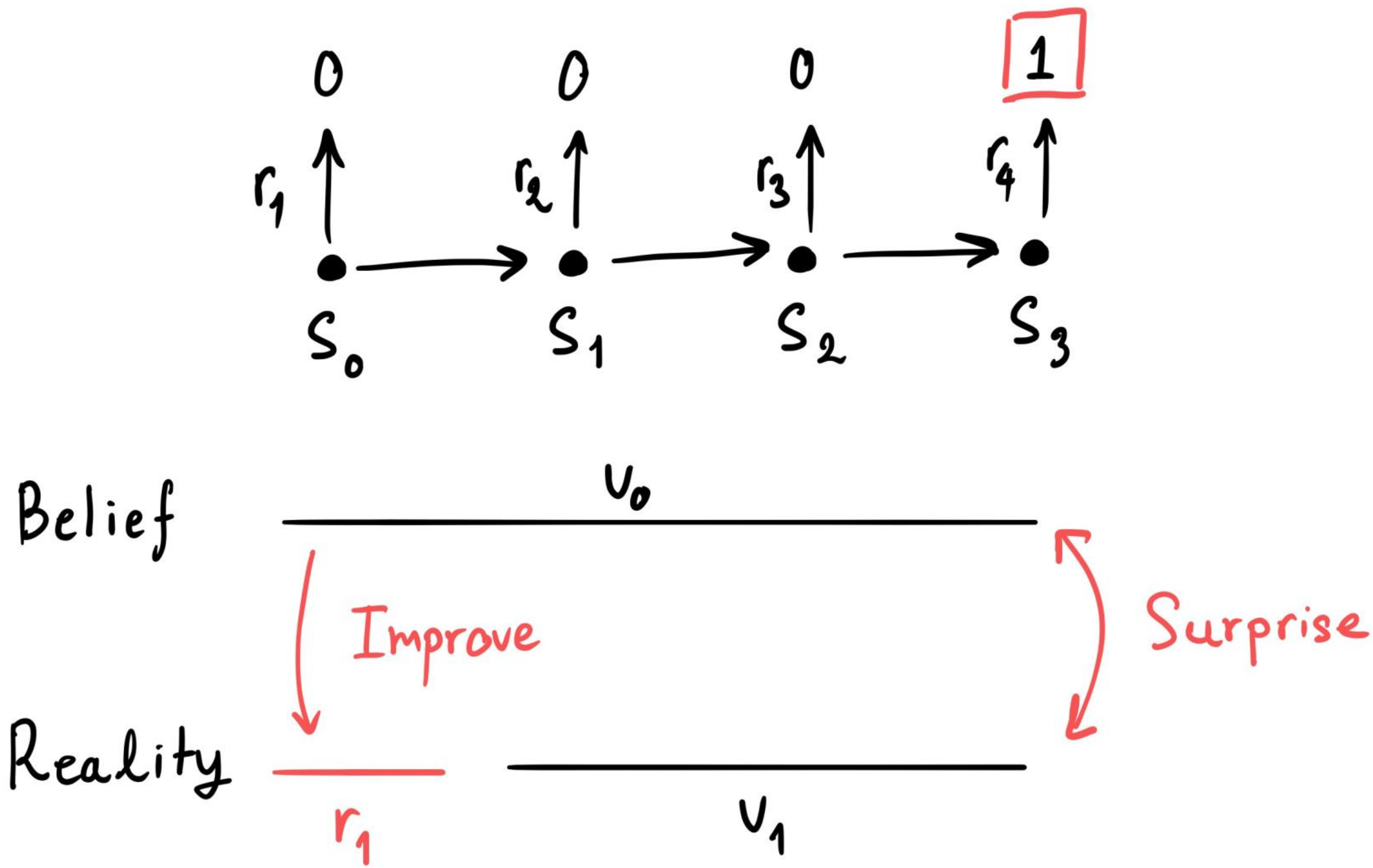
Reality and beliefs

Belief = Current state value (v)

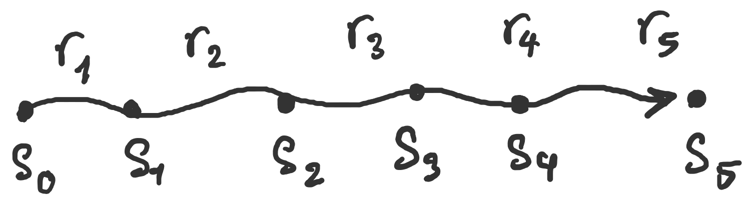
Reality = Sum of reward

How many rewards do we need to get a surprise? Just one.

One step reality



Temporal views



$$v(s_5) = 0$$
$$v(s_4) = r_5 = r_5 + v(s_5)$$
$$v(s_3) = r_4 + r_5 = r_4 + v(s_4)$$
$$v(s_2) = r_3 + r_4 + r_5 = r_3 + v(s_3)$$
$$v(s_1) = r_2 + r_3 + r_4 + r_5 = r_2 + v(s_2)$$
$$v(s_0) = r_1 + r_2 + r_3 + r_4 + r_5 = r_1 + v(s_1)$$

Recursive relationship

- Is it? $v(s_t) = r_{t+1} + v(s_{t+1})$

Bellman equation for v_π

$$v(s_t) = \sum_{a_t} \pi(a_t|s_t)\Big[\underbrace{\sum_{r_{t+1}} p(r_{t+1}|s_t, a_t) r_{t+1}}_{r(s_t,\, a_t)} + \gamma \sum_{s_{t+1}} v(s_{t+1})\Big]$$

Some short hands

- Expected immediate reward

$$r(s, a) = \sum_{r} p(r|s, a) r$$

TD is a "sampling" version of Bellman

Some notes

$$p(r|s,a) \qquad p(s',r|s,a) \qquad p(s'|s,a)$$

All are inside the environment

We don't know anyway

We write the more convenient one!

Temporal difference (TD)

$$v(s_t) \leftarrow r_{t+1} + v(s_{t+1})$$

- Sampling based Bellman equation
- Either environment or policy is stochastic, this will give a wrong result
- We need to average out the randomness

TD with running average

$$v(s_t) \leftarrow v(s_t) + \alpha \big[\underbrace{r_{t+1} + v(s_{t+1}) - v(s_t)}_{\delta_t \,=\, \text{TD error}} \big]$$

- α learning rate (or running average)

Shorthand:

$$v(s_t) \stackrel{\alpha}{\leftarrow} r_{t+1} + v(s_{t+1})$$

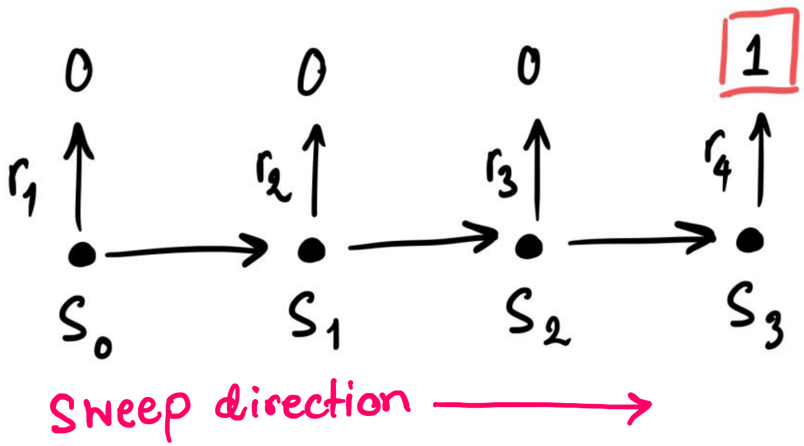
Action value TD q_π

$$q(s_t, a_t) \leftarrow q(s_t, a_t) + \alpha\big[r(s_t, a_t) + v(s_{t+1}) - q(s_t, a_t)\big]$$

Shorthand:

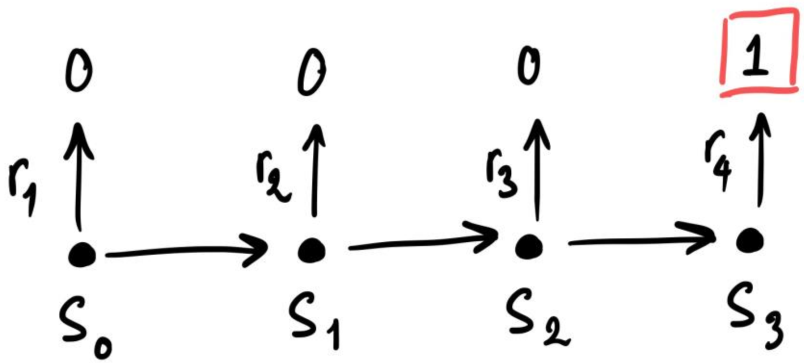
$$q(s_t, a_t) \overset{\alpha}{\leftarrow} r(s_t, a_t) + v(s_{t+1})$$

How TD Works?



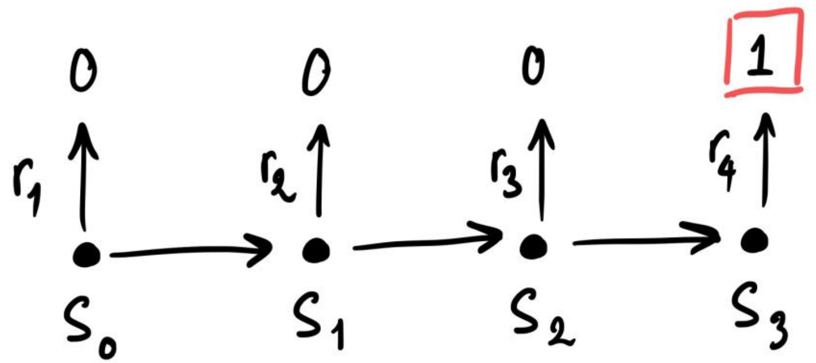
State	Reward	Next V	R + V’	V
0	0			0
1	0			0
2	0			0
3	1	-		0

How TD Works?



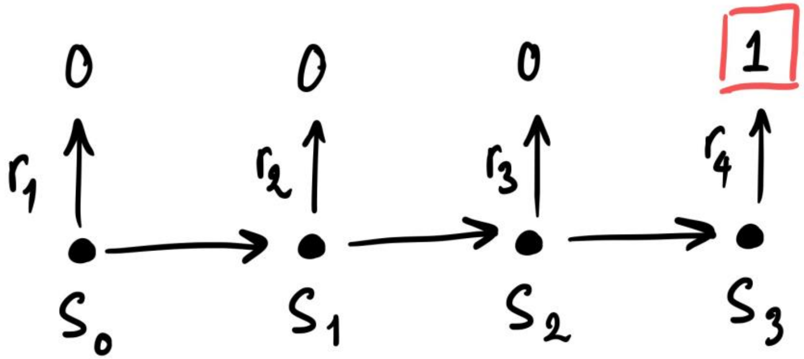
State	Reward	Next V	R + V'	V
0	0			0
1	0			0
2	0			0
3	1	-		1

How TD Works?



State	Reward	Next V	R + V’	V
0	0			0
1	0			0
2	0			1
3	1	-		1

How TD Works?

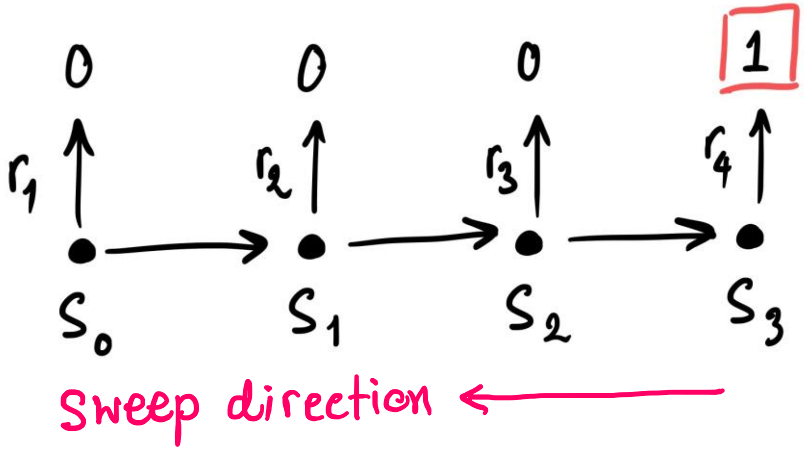


State	Reward	Next V	R + V’	V
0	0			0
1	0			1
2	0			1
3	1	-		1

How TD works?

- The “correctness” is propagated backwards in time
  - From the terminal to beginning states
  - State by state
  - One-step TD
- **At first, TD has high bias**
- During training, it gets closer to the real value
- When there is nothing to update, TD has no bias

Sweeping direction



State	Reward	Next V	R + V’	V
0	0			0
1	0			0
2	0			0
3	1	-		0

The “Right” sweep direction

- It is not obvious what is the right sweep direction in more complex environments
  - Many branches
- If we need to sweep from terminal, what are states before terminals?
- **We usually don’t know what are preceding states**

TD is a bootstrapping technique

- **Bootstrap = prediction on prediction**
- Given a transition (s, a, r, s')
- Return is not available

$$\sum_{t=0} r_{t+1}$$

- We estimate

$$\sum_{t=0} r_{t+1} \approx r_1 + v(s_1)$$

- It is not correct but better than $v(s_0)$

How to store v and q values

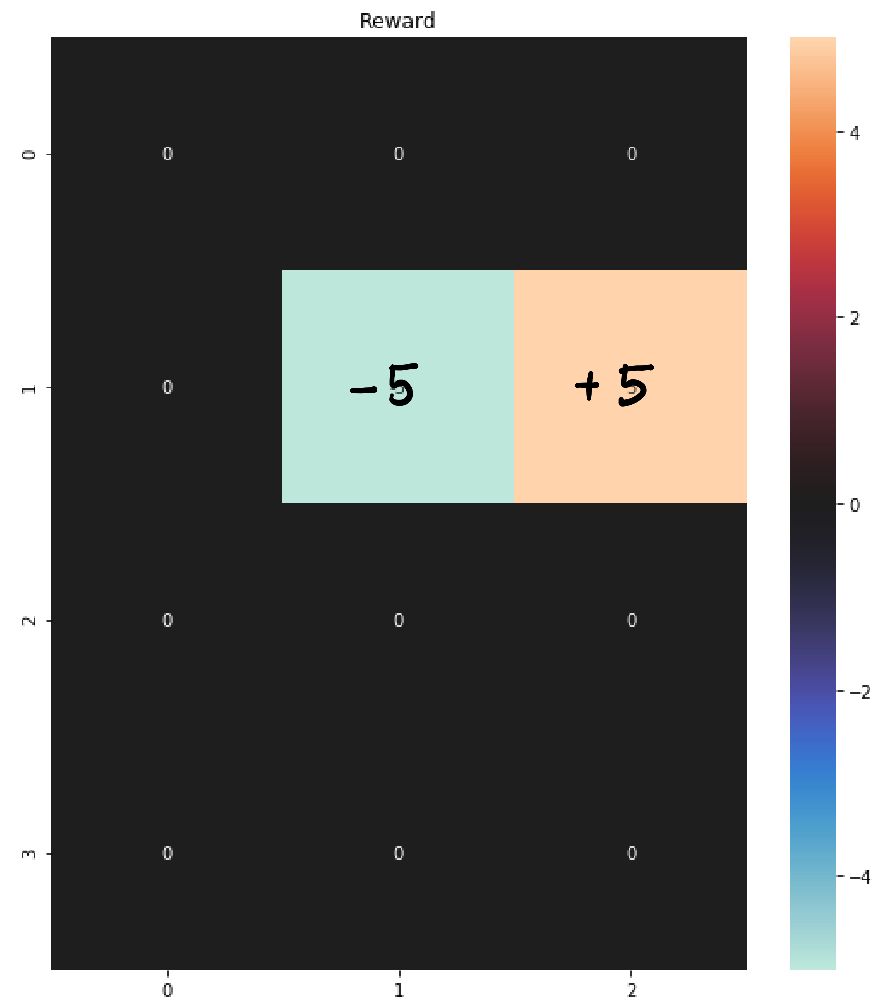
With tables

We use the term "tabular" to this family of algorithms

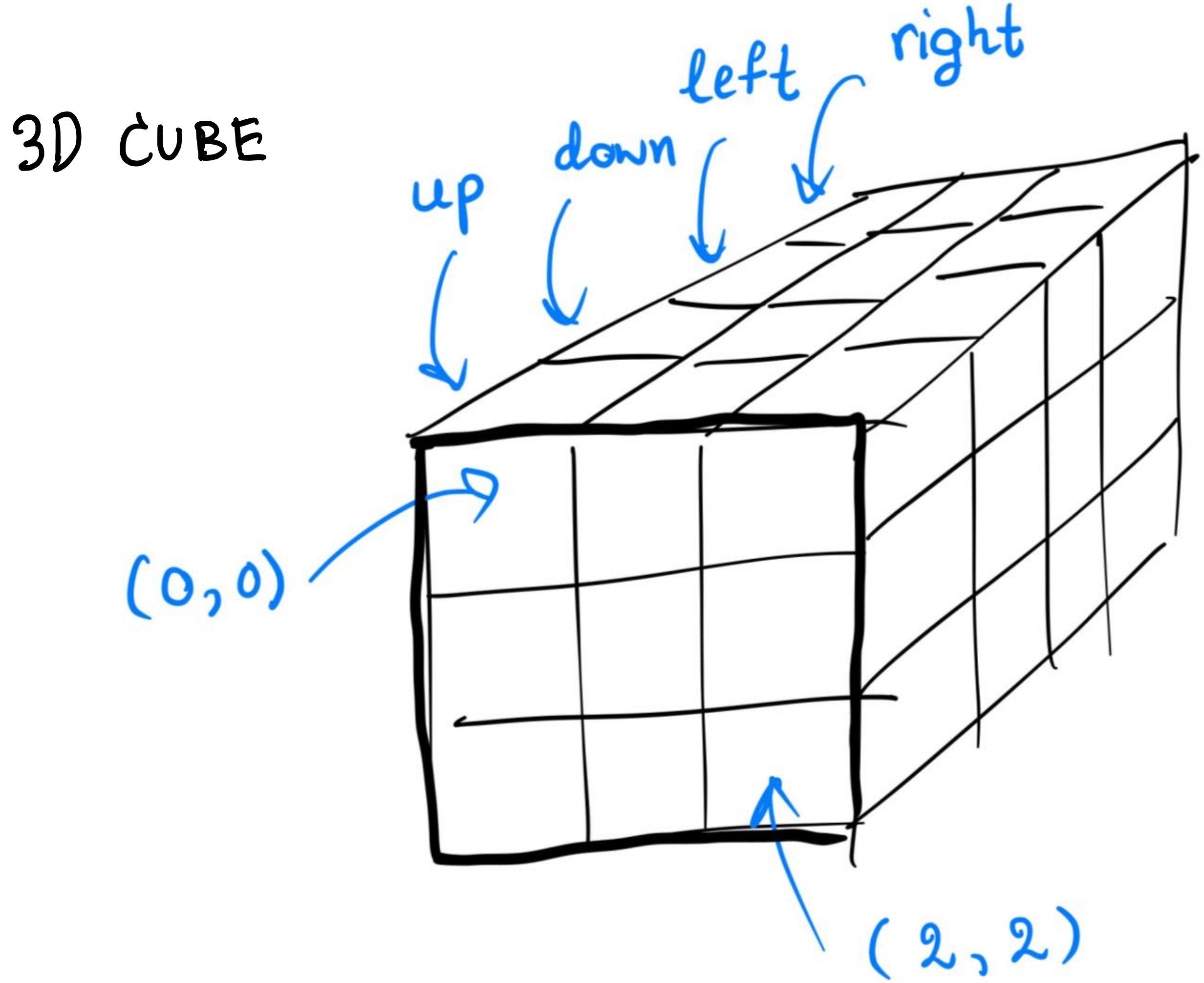
Example: gridworld 2D

Map

Reward



Example: gridworld 2D



Comparing TD and MC for prediction

- TD with moving average

$$v^{TD}(s) \leftarrow v(s) + \alpha\left[r + v(s') - v(s)\right]$$

- MC with moving average

$$v^{MC}(s_t) \leftarrow v(s_t) + \alpha\Big[\underbrace{\sum_{\tau=t} r_{\tau+1}}_{\text{Return}} - v(s_t)\Big]$$

TD with discount factor

$$v^{TD}(s) \leftarrow v(s) + \alpha\big[r + \gamma v(s') - v(s)\big]$$

$$q(s_t, a_t) \leftarrow q(s_t, a_t) + \alpha\big[r(s_t, a_t) + \gamma v(s_{t+1}) - q(s_t, a_t)\big]$$

Bias and variance trade-off

Definition

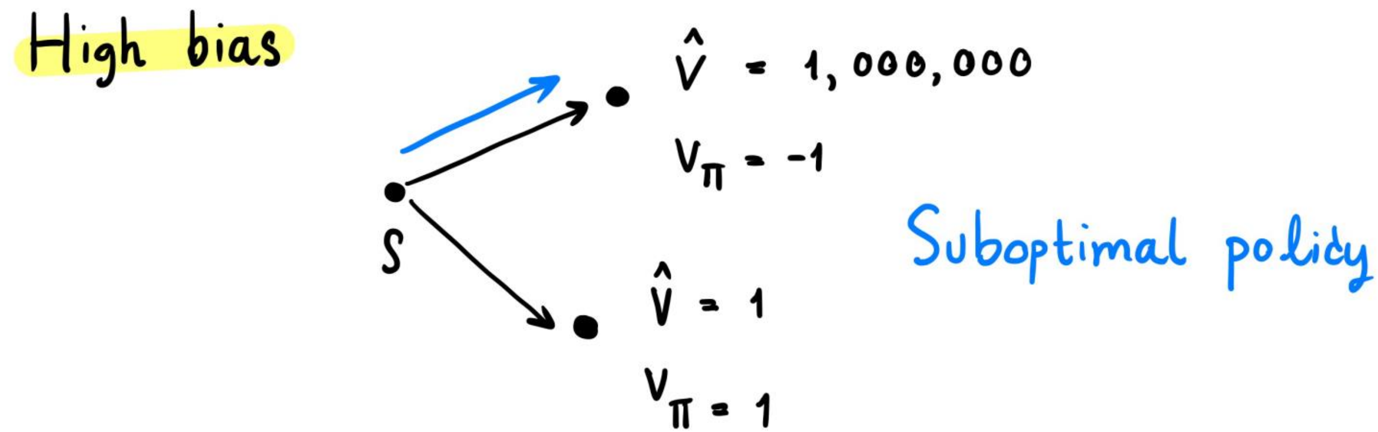
Bias = Inaccuracy of prediction on average

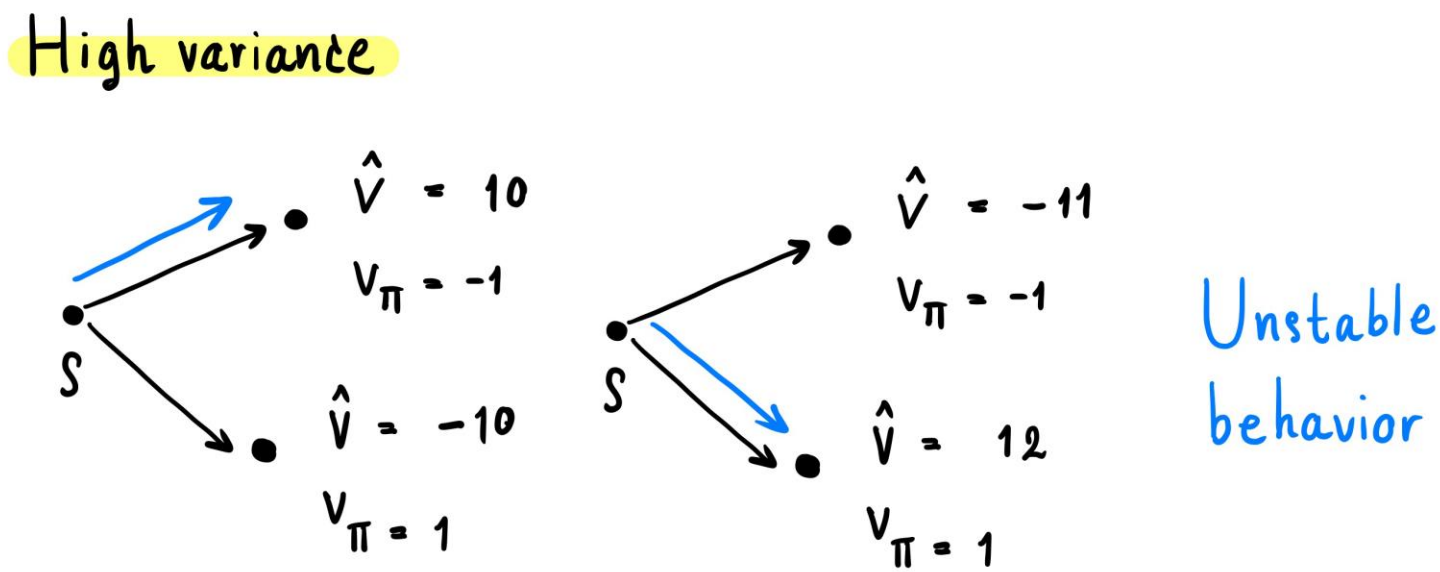
$$\mathbb{E}\left[X - \mathbb{E}\left[X\right]\right]$$

Variance = Fluctuations of the prediction

$$\mathrm{Var}(X)$$

Low bias, low variance the best





Bias and variance trade-off

- *Low* bias *low* variance is best
- TD = **high** bias + *low* variance
- MC = *low* bias + **high** variance
- A dilemma in RL
- **You cannot excel in both**
- Better algorithm gives a better trade-off
- Something in between is available
- Med bias + med variance?

TD could give a different result from MC

Problem statement

Limited Experience

A 1 B 1

B 1

B 1

B 0

What are the state value of MC and TD?

MC and TD solution

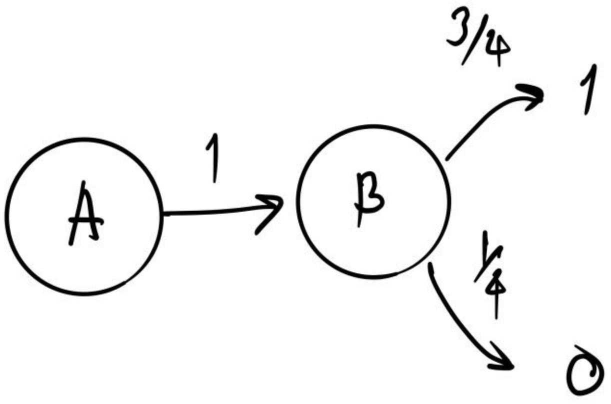
A 1 B 1

B 1

B 1

B 0

MC solution	TD solution
State values A = 2 B = 3/4	**State values** A = 7/4 B = 3/4



MC Solution

- Doesn’t assume MDP
- It gives a solution as-is
- It converges to minimum squared error on training data
- With limited experience, it seems too naive

TD solution

- TD assumes MDP
- It infers the most likely MDP
    - By constructing states, transitions
    - Learning their transition probabilities and rewards
- TD usually gives a favorable result with limited experience
    - If the underlying problem is MDP
- Both TD and MC converge to the same solution with unlimited data

Using TD for policy iteration

Recap what is policy iteration

for until π is stable **do**
    prediction
    improvement
end for

TD for predicting action-value

for until π is stable **do**

  prediction

  improvement

end for

- We will work on action value function **q**
- It is useful for policy improvement

TD for predicting action-value

$$q(s_t, a_t) \leftarrow$$
$$q(s_t, a_t) + \alpha\big[r(s_t, a_t) + q(s_{t+1}, a_{t+1}) - q(s_t, a_t)\big]$$

$$q(s_t, a_t) \overset{\alpha}{\leftarrow} r(s_t, a_t) + q(s_{t+1}, a_{t+1})$$

- We need (s, a, r, s’, a’) to update
- Hence the name **SARSA** 

SARSA Algorithm

for until π is stable **do**

  collect experience (s, a, r, s', a') using π

  $q(s, a) \leftarrow q(s, a) + \alpha\big[r + q(s', a')\big]$

  for s in S **do**

    $\pi(s) \leftarrow \underset{a}{\operatorname{argmax}}\, q(s, a)$

  end for

end for

SARSA

- One-step TD prediction + policy improvement
- SARSA is on-policy
- On-policy = experience must come from itself
- Off-policy will alleviate this constraint!

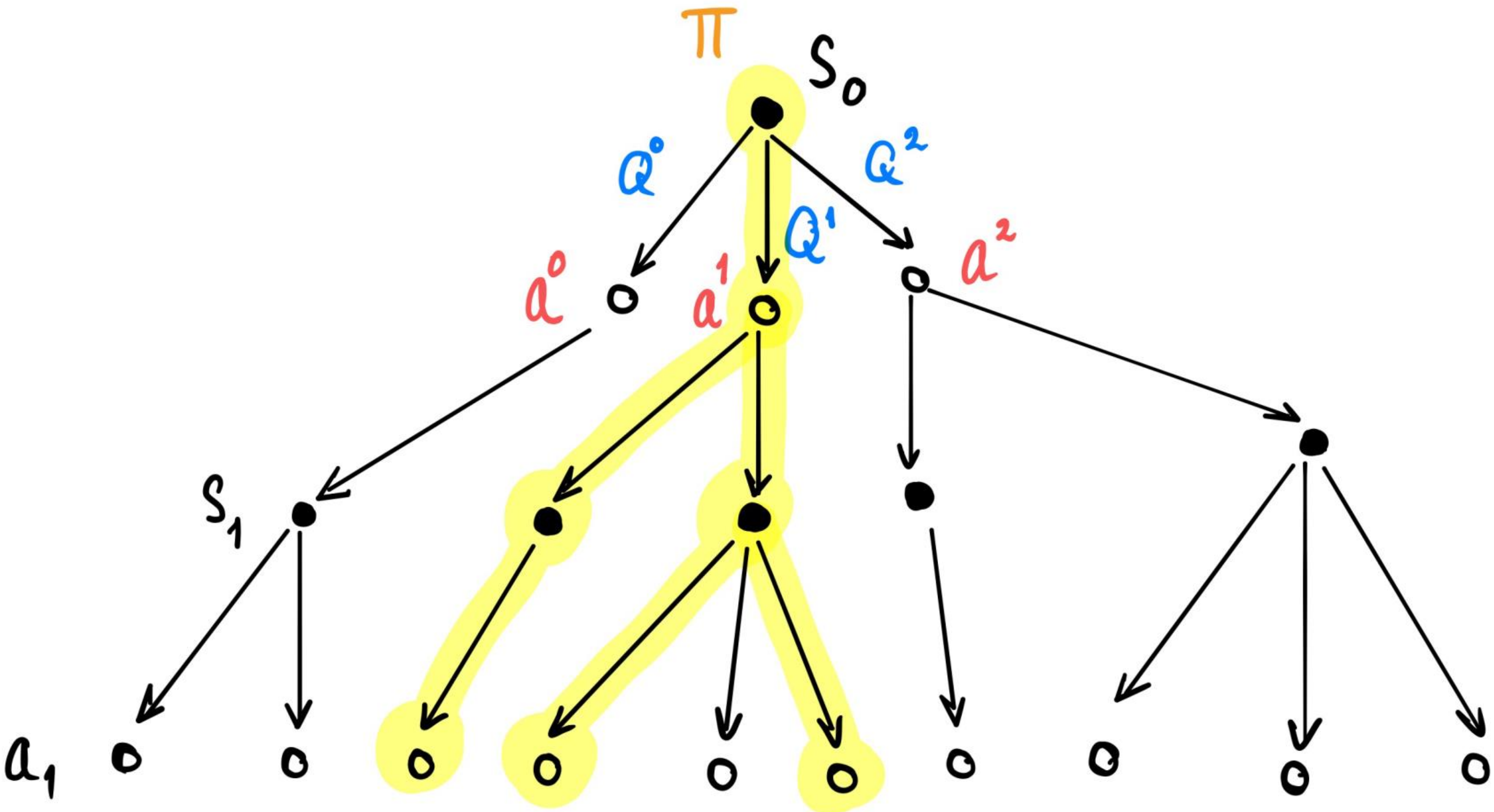
Exploration vs Exploitation

Problem with no exploration

- If you never try an environment
- If you never go a different path
- How would you know what's better?

- So… philosophical

More concrete example



No reward, no learning

- **We learn nothing from zero reward!**
- Learning algorithm might not be a bottleneck

$$v^{TD}(s) \leftarrow v(s) + \alpha \big[r + v(s') - v(s) \big]$$

- Problem then is not with an algorithm but with the exploration

Epsilon greedy

- Once in a while, try something different
- It might turn out to be a new high

$z \sim Uniform(0, 1)$

if z $< \epsilon$ **then**

  random action

else

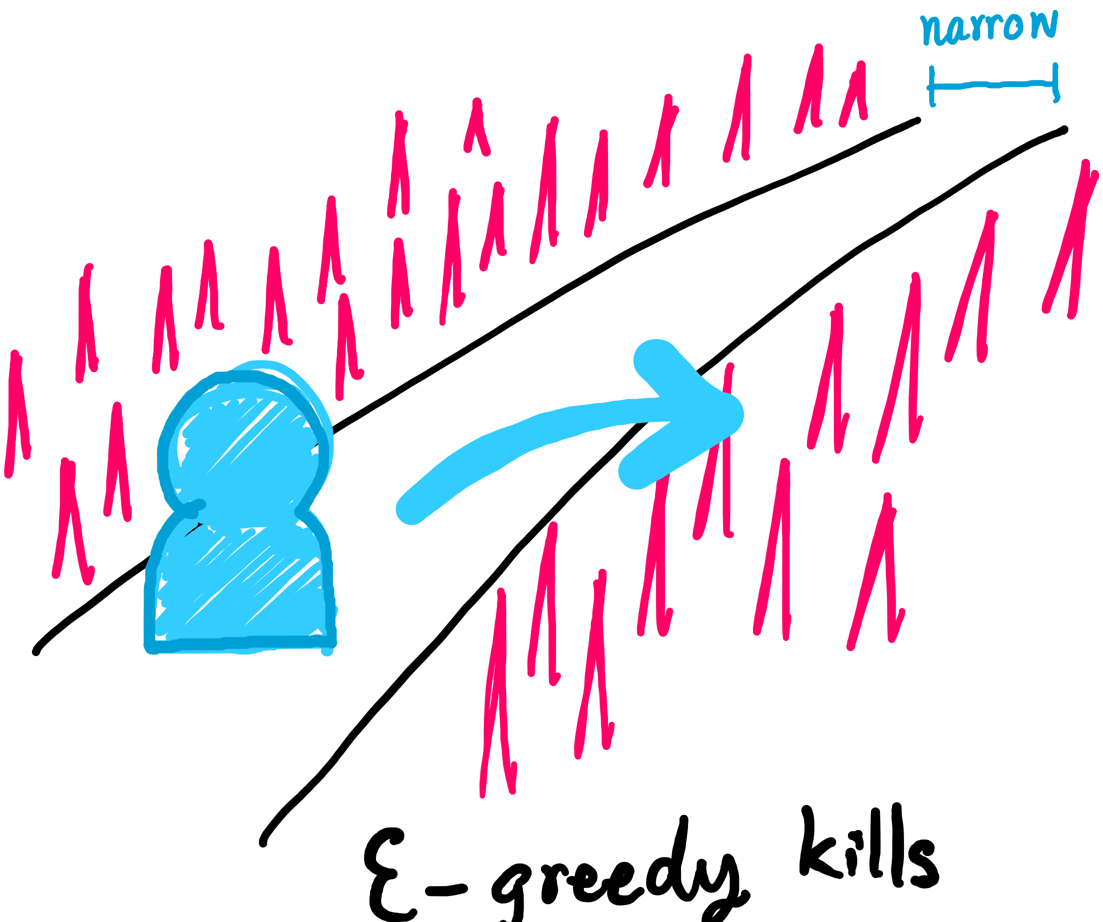
  $a \sim \pi(s, a)$

end if

Epsilon greedy problems

- It could prevent learning optimal policy
- If an environment has “low tolerance” for errors, epsilon greedy is not likely to reach very far
- Annealing epsilon

Environment with low tolerance

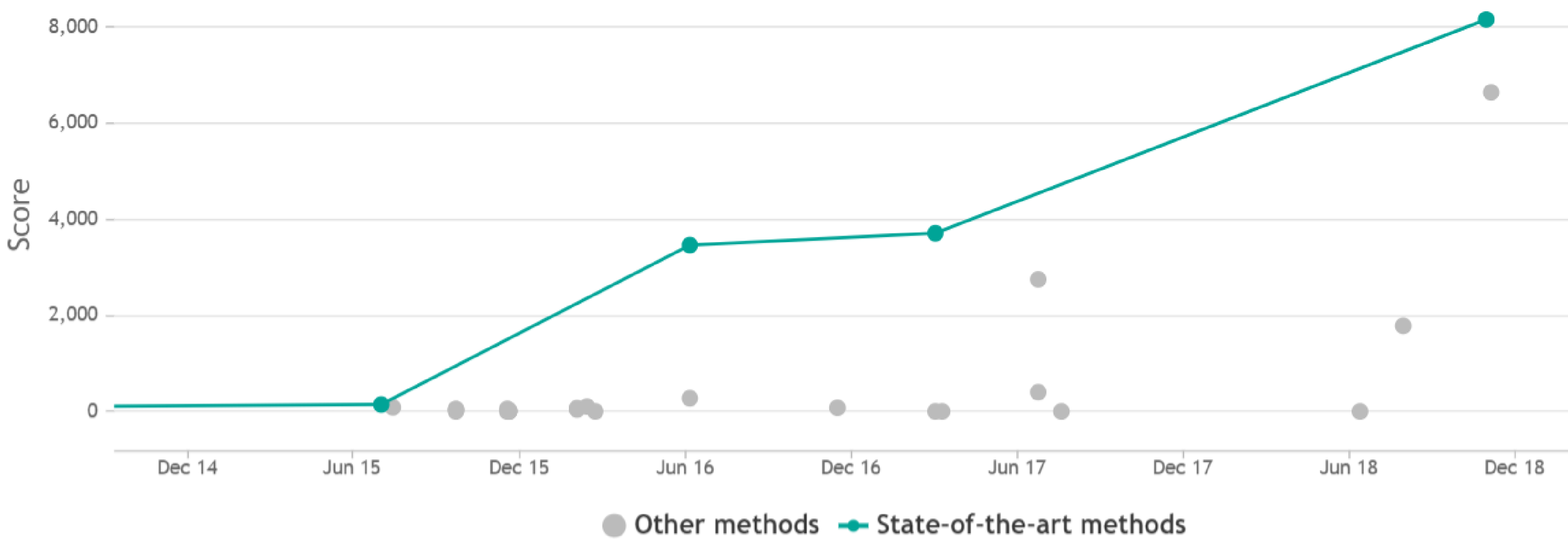


Efficient exploration is a grand challenge

- Better exploration could speed up learning
- It could decide whether an environment is solvable
- Orders of magnitude faster!
- Some environments are just hard to explore thoroughly


0
We trained DQN with
novelty-based rewards.

Atari Games on Atari 2600 Montezuma's Revenge



Rank	Method	Score	Paper Title	Year
1	**RND**	8152	Exploration by Random Network Distillation	2018
2	**A2C+CoEX**	6635	Contingency-Aware Exploration in Reinforcement Learning	2018
3	**DQN-PixelCNN**	3705.5	Count-Based Exploration with Neural Density Models	2017
4	**DDQN-PC**	3459	Unifying Count-Based Exploration and Intrinsic ...	2016

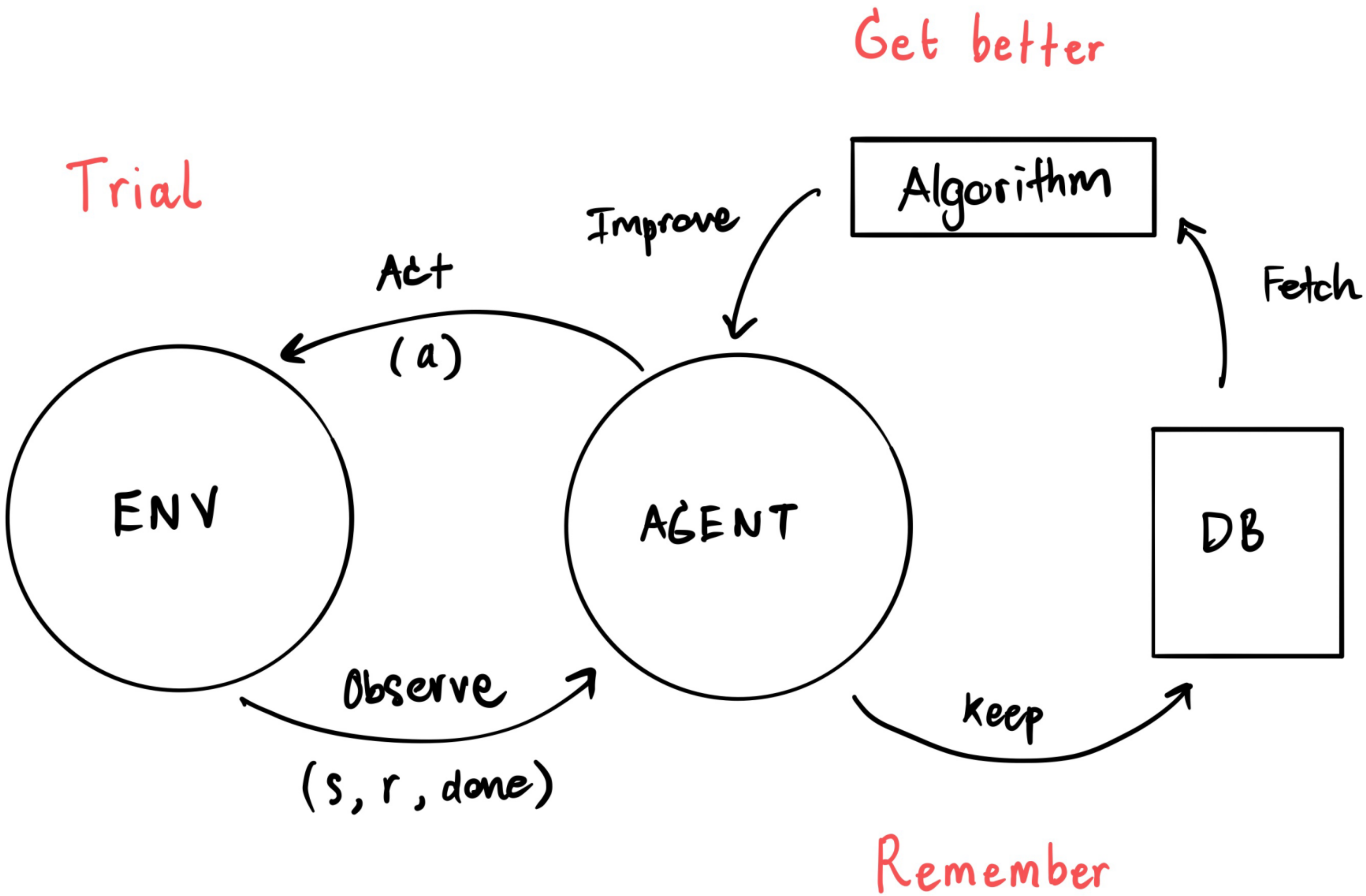
All exploration methods

Rank	Method	Score	Paper Title	Year
1	**RND**	8152	Exploration by Random Network Distillation	2018
2	**A2C+CoEX**	6635	Contingency-Aware Exploration in Reinforcement Learning	2018
3	**DQN-PixelCNN**	3705.5	Count-Based Exploration with Neural Density Models	2017
4	**DDQN-PC**	3459	Unifying Count-Based Exploration and Intrinsic Motivation	2016
5	**Sarsa-φ-EB**	2745.4	Count-Based Exploration in Feature Space for Reinforcement Learning	2017
6	**DQN+SR**	1778.8	Count-Based Exploration with the Successor Representation	2018
7	**Sarsa-ε**	399.5	Count-Based Exploration in Feature Space for Reinforcement Learning	2017
8	**A3C-CTS**	273.7	Unifying Count-Based Exploration and Intrinsic Motivation	2016

Conclusions

RL is trial, remember, get better

- **Trial**
  - Explore a lot
- **Remember**
  - Some states are good (high reward)
  - Some are bad (low reward)
- **Get better**
  - Using Bellman equation to propagate high and low reward
  - Avoid low rewards, gravitate to high rewards


Get better
Trial
Algorithm
Improve
Act
Fetch
(a)
ENV
AGENT
DB
Observe
Keep
(s, r, done)
Remember

Assignments

- Ex 3.0 Interface
- Ex 3.1 Chula RL
- Ex 3.2 MC
- Ex 3.3 SARSA

A tour …

Last assignment …

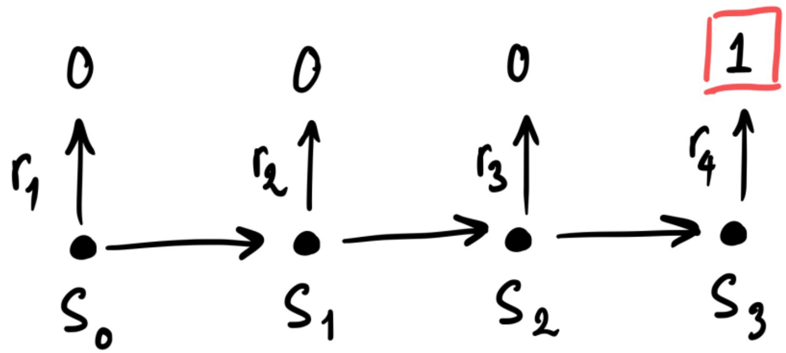
Resources

General RL discussions

- Facebook group “Thailand Reinforcement Learning”

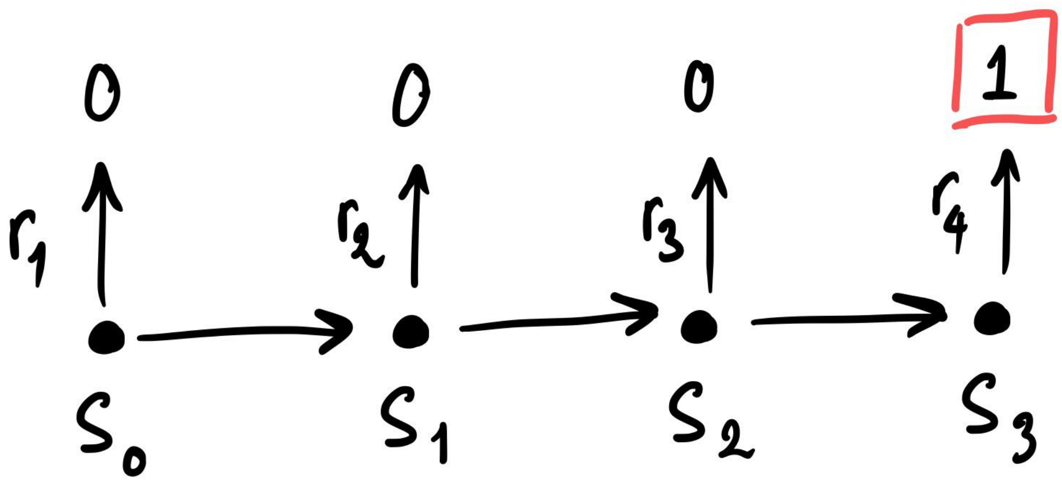
Temporal credit assignment

The problem



- Many actions in sequence lead to a reward
- How do we know which one contributes, which one hinders?
- How do we know what to change to improve?

RL approach

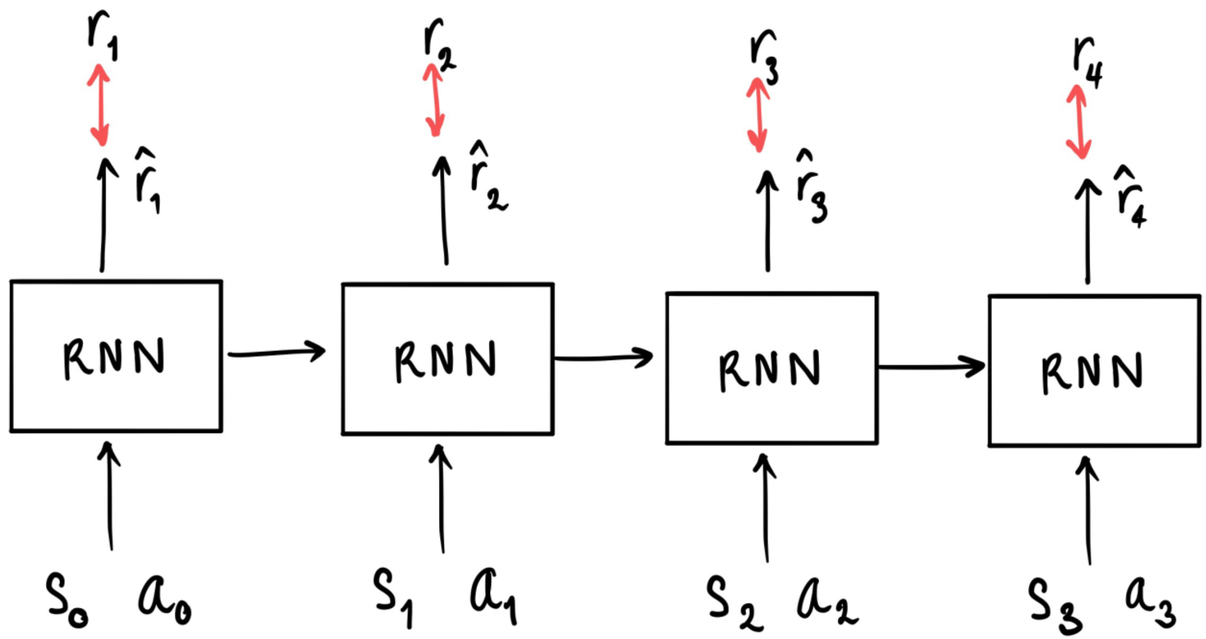


A/S	0	1	2	3
0	1	0	0	-1
1	2	1	0	0
2	0	-1	-2	-3

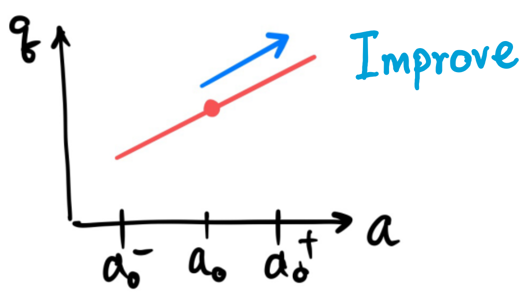
Improve

Backpropagation approach

$$q_\theta(s_0, a_0) \approx \sum_{t=0} \hat{r}_{t+1}$$



$$\frac{\partial q_\theta}{\partial a_0} \Rightarrow$$

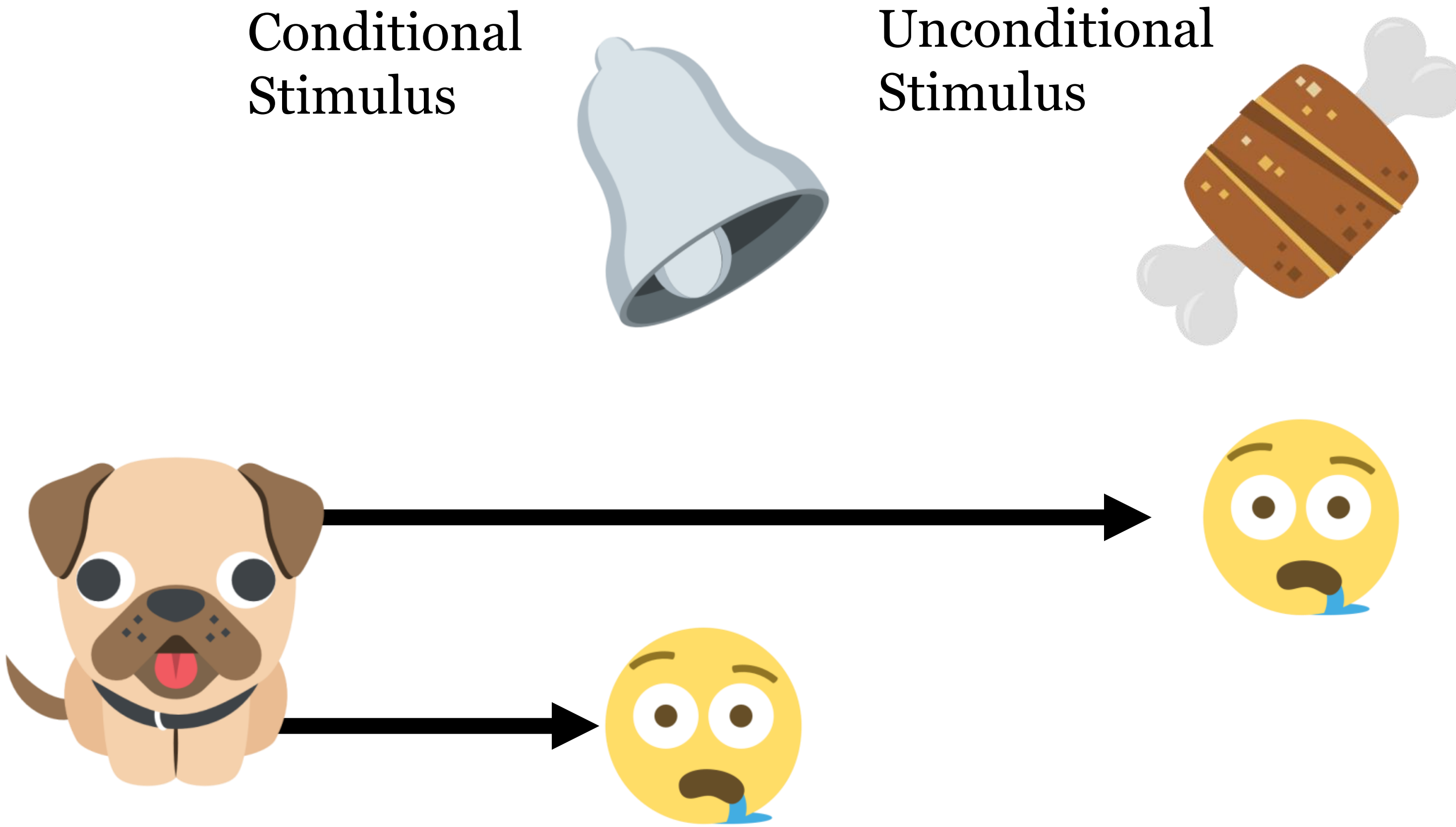


Psychology and Neuroscience

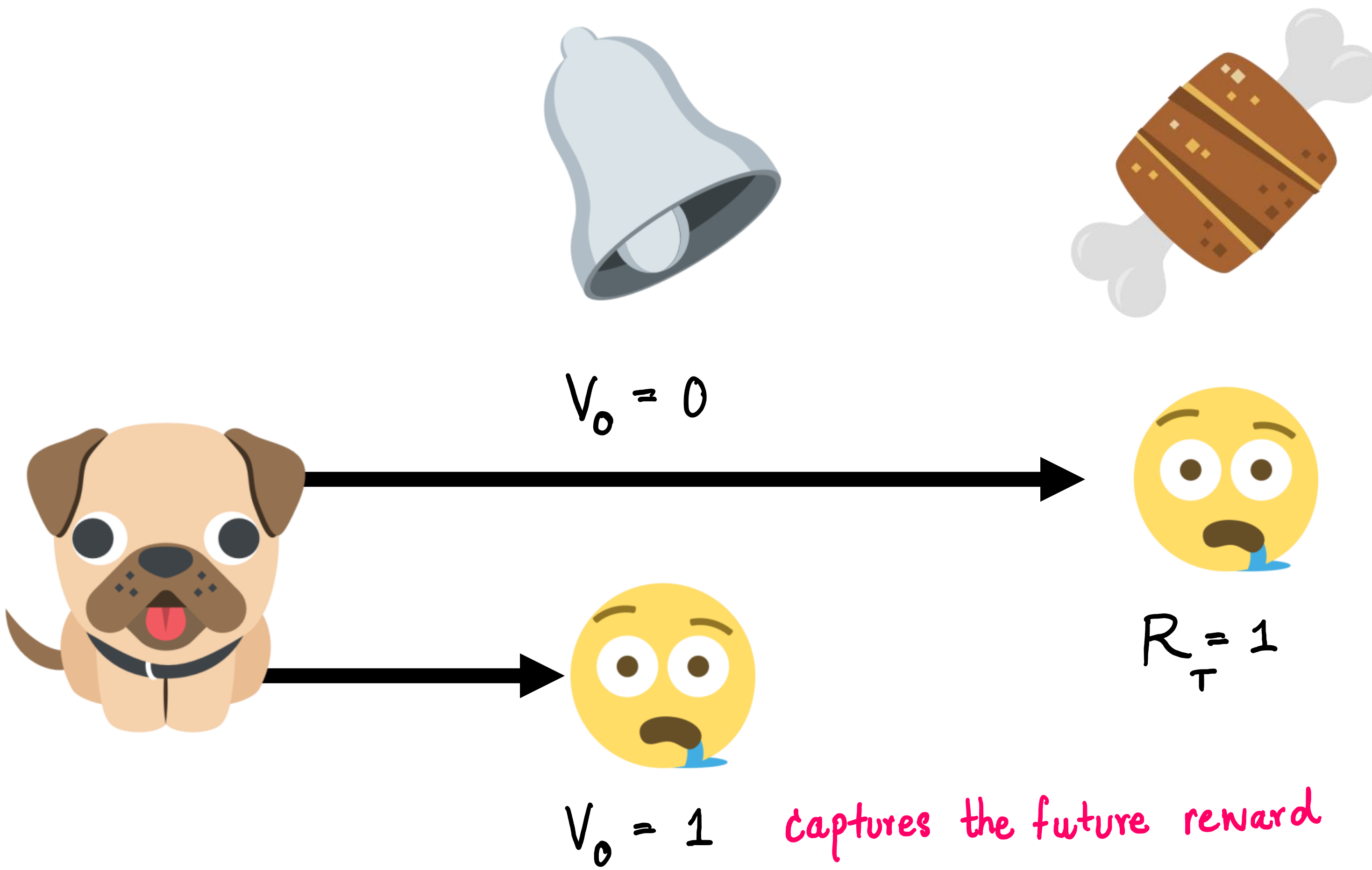
RL and psychology

- Pavlov's classical conditioning
- Thorndike's Law of effect

Classical conditioning



Classical conditioning



Law of effect

"Responses that produce a satisfying effect become more likely to occur, and

responses that produce a discomforting effect become less likely to occur…"

Thorndike, paraphrased

Animals learn policies that maximizes satisfactory

RL and neuroscience

- Dopamine is a neurotransmitter
- Dopamine involves in motivation, learning, action-selection, addiction
- It had been suggested that dopamine activity signals reward
- Recent research suggests that dopamine activity signals *reward prediction error*
- Just like when |reality – belief| > 0 (surprise)
- It suggests that humans utilize some kind of TD