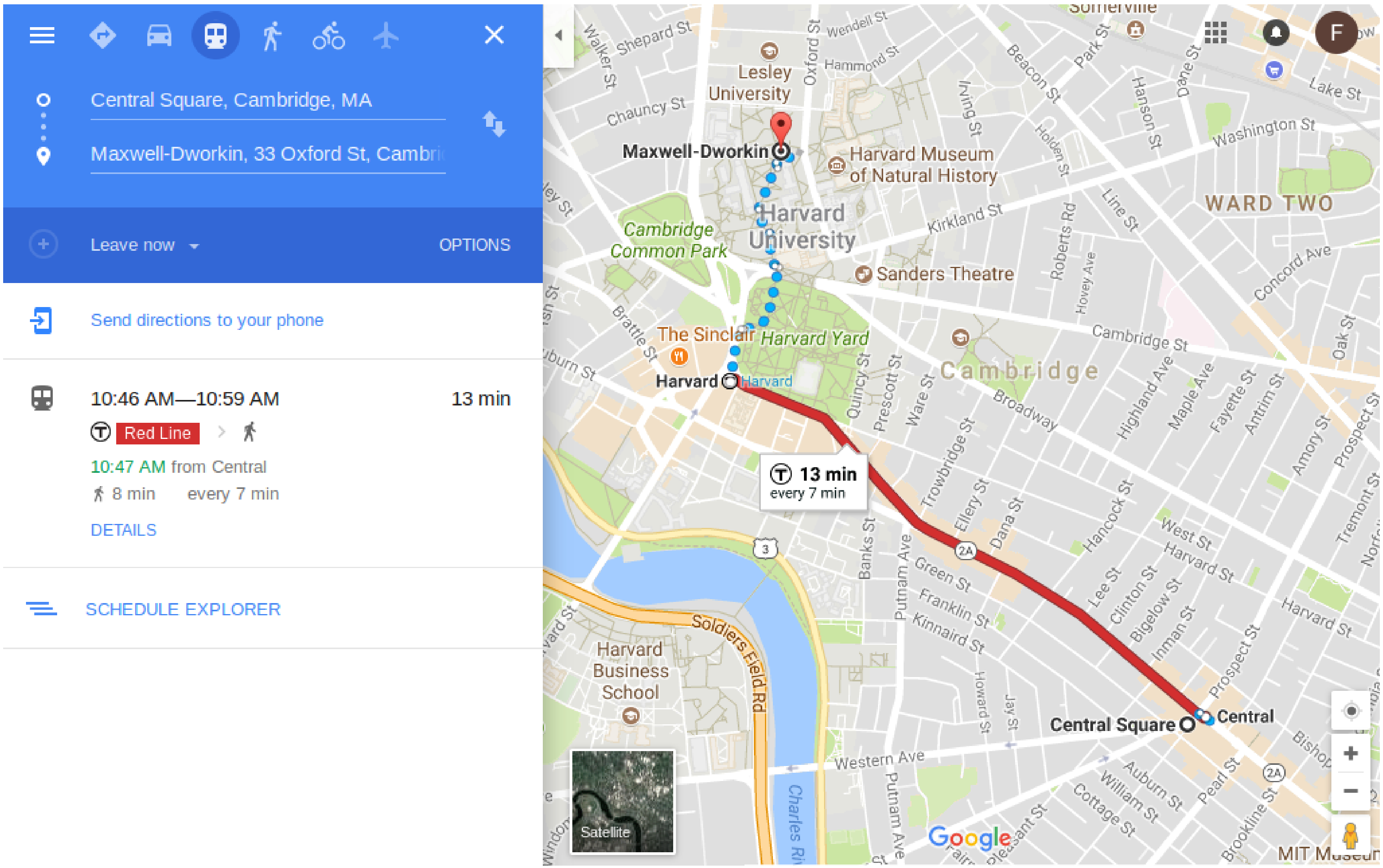
Introduction to Reinforcement Learning

Finale Doshi-Velez
Harvard University

Buenos Aires MLSS 2018

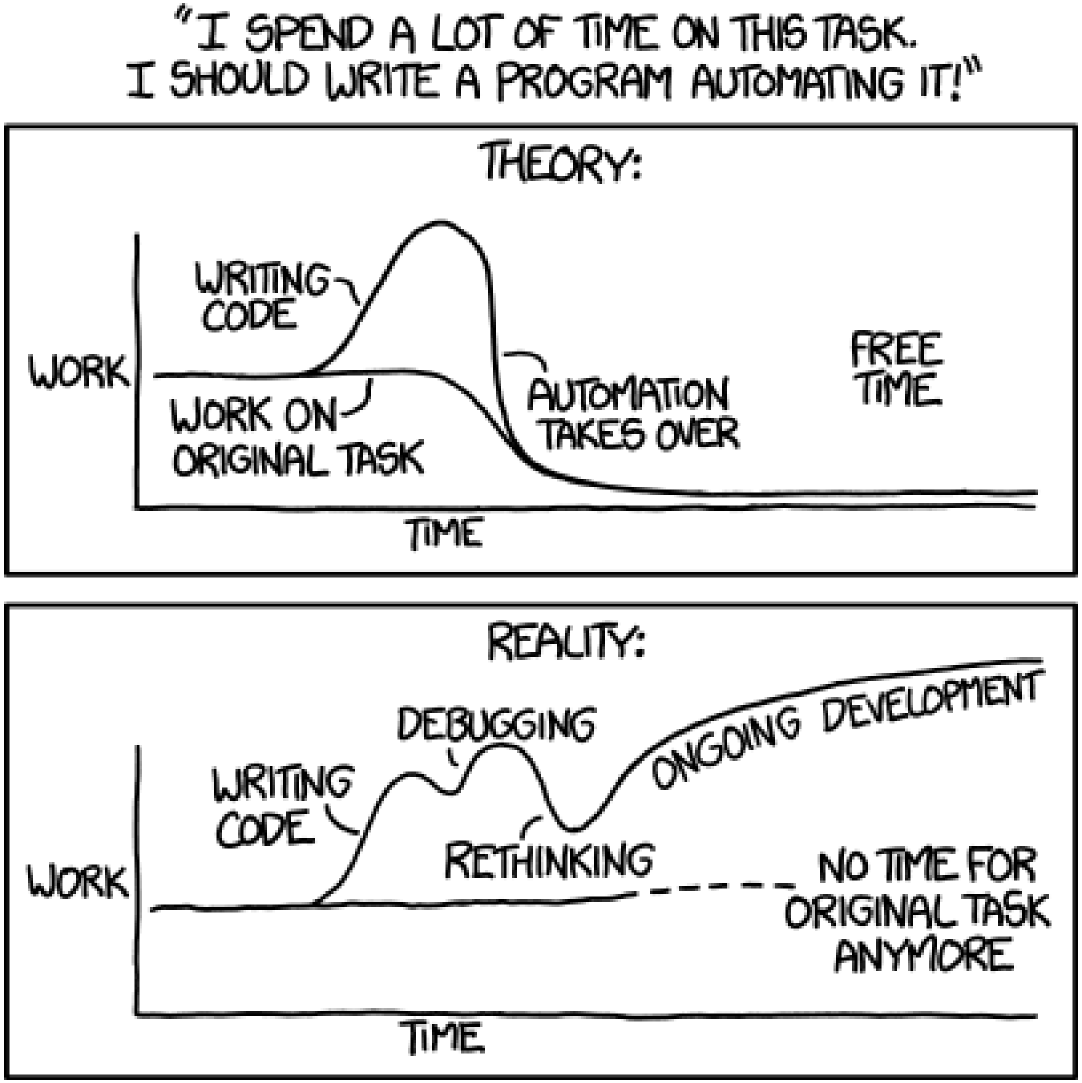
We often must make decisions under uncertainty.

How to get to work, walk or bus?



We often must make decisions under uncertainty.

What projects to work on?



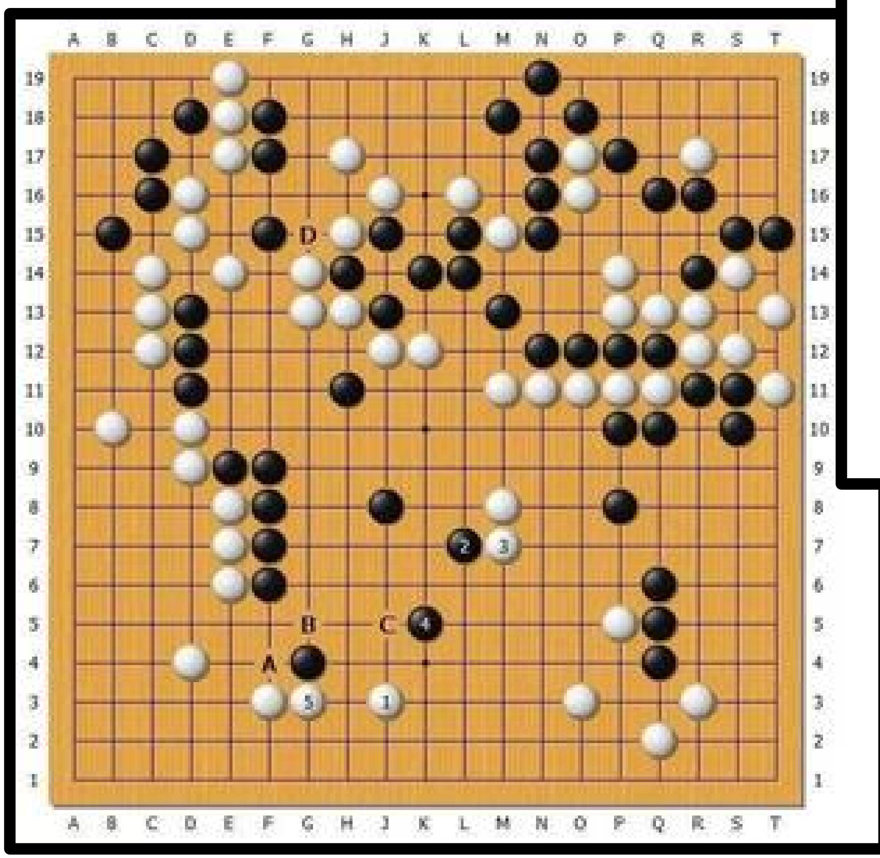
https://imgs.xkcd.com/comics/automation.png

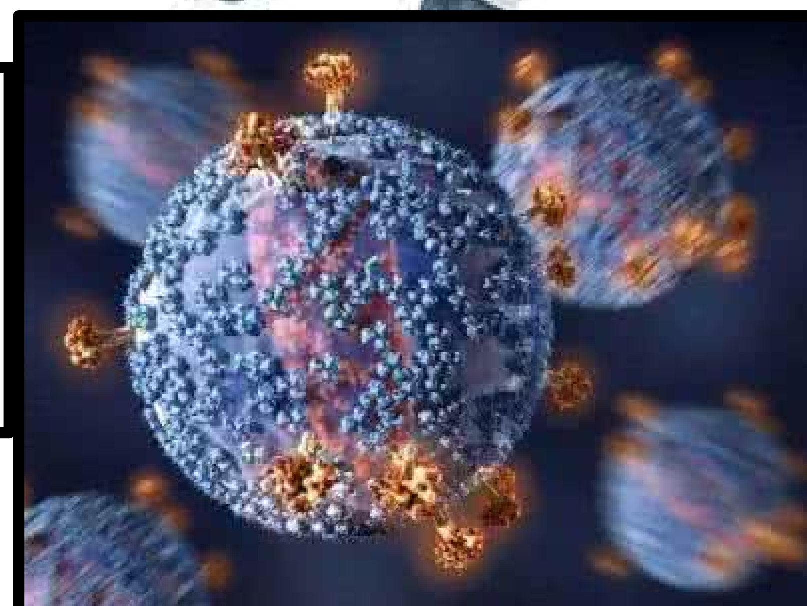
We often must make decisions under uncertainty.

How to improvise with a new recipe?

https://s-media-cache-ak0.pinimg.com/originals/23/ce/4b/23ce4b2fc9014b26d4b811209550ef5b.jpg

Some Real Applications of RL

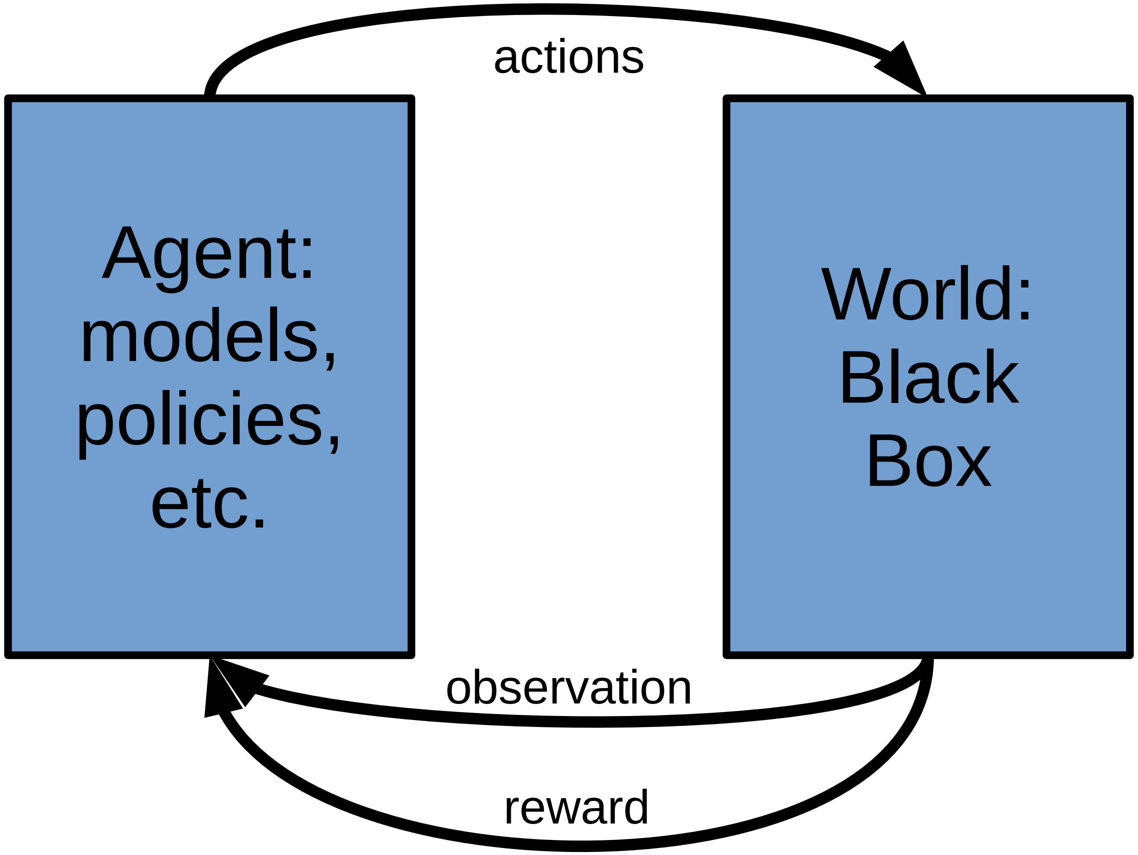




Why are these problems hard?

- Must learn from experience (may have prior experience on the same or related task)
- Delayed rewards/actions may have long term effects (delayed credit assignment)
- Explore or exploit? Learn and plan together.
- Generalization (new developments, don’t assume all information has been identified)

Reinforcement learning formalizes this problem



Objective: Maximize $E[\sum_t \gamma^t r_t]$

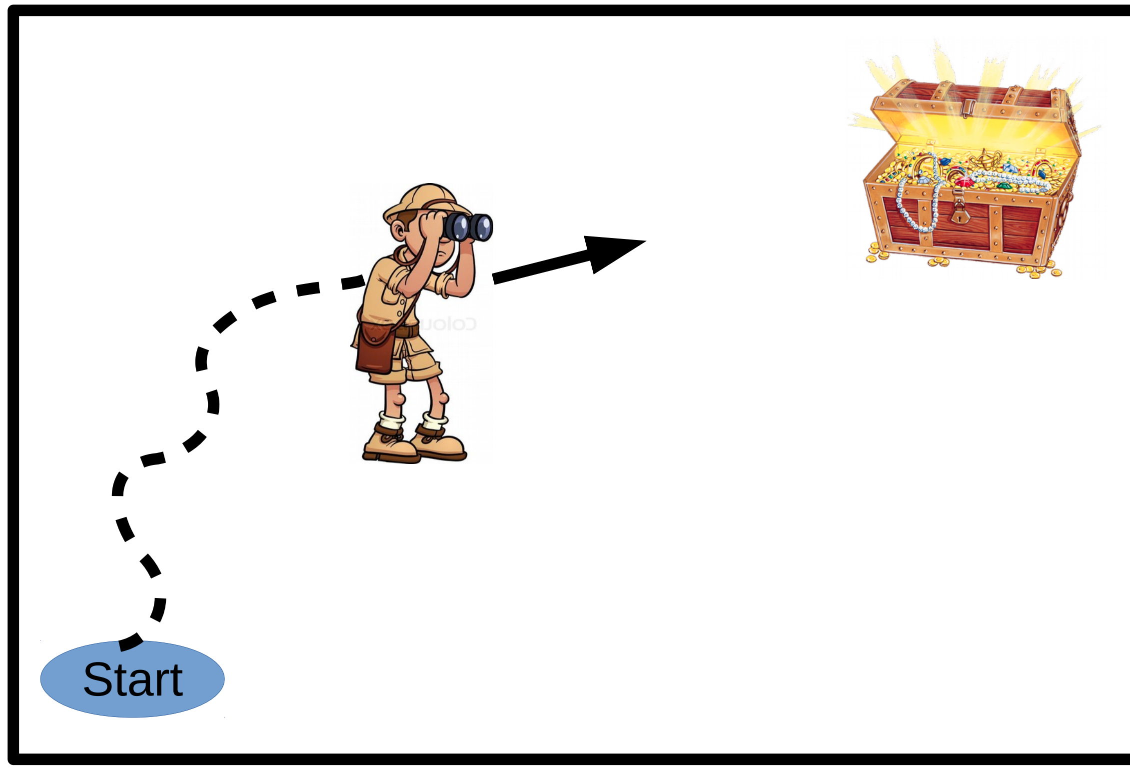
(finite or infinite horizon)

Concept Check: Reward Adjustment

- If I adjust every reward r by r + c, does the policy change?
- If I adjust every reward r by c*r, does the policy change?

Key Terms

- Policy $\pi(s,a)$ or $\pi(s) = a$
- State s
- History {s0,a0,r0,s1,a1…}



Markov Property:

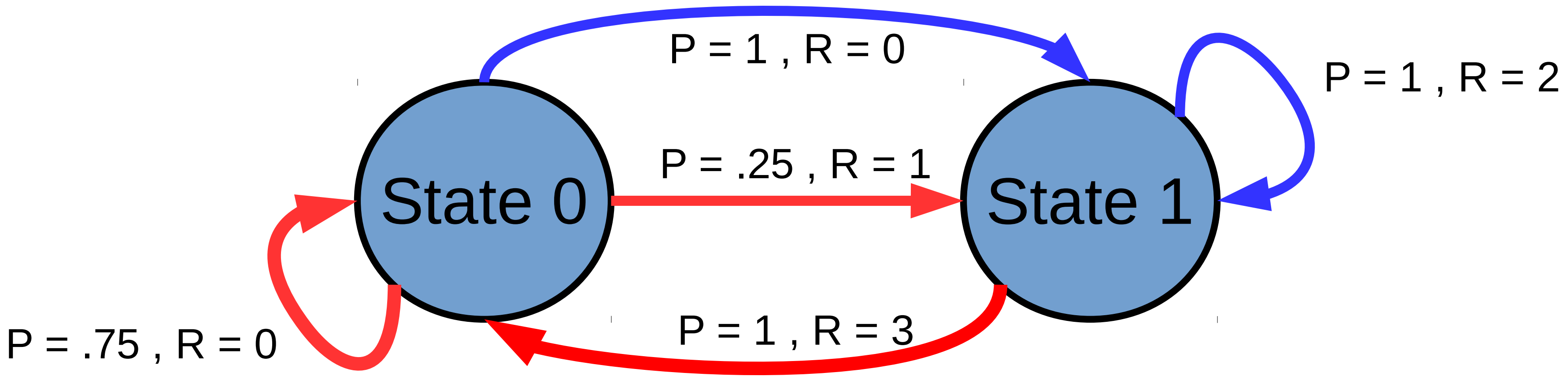
$p(s_{t+1} \mid h_t) = p(s_{t+1} \mid h_{t-1}, s_t, a_t) = p(s_{t+1} \mid s_t, a_t)$

… we'll come back to identifying state later!

Markov Decision Process

- T(s' | s , a) = Pr(state s' after taking action a in state s)
- R(s , a , s ') = E[reward after taking action a in state s and transitioning to s']

… but may depend on less, e.g. R(s , a) or even R(s)



Notice given a policy, we have a Markov chain to analyze!

How to Solve an MDP: Value Functions

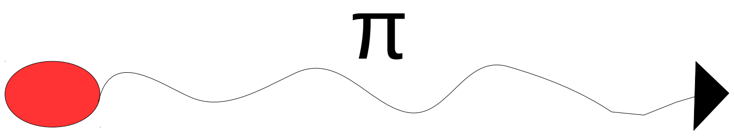
Value: $V_\pi(s) = E_\pi[\ \Sigma_t\ \gamma^t\ r_t \mid s_0 = s\ ]$

… in s, follow π

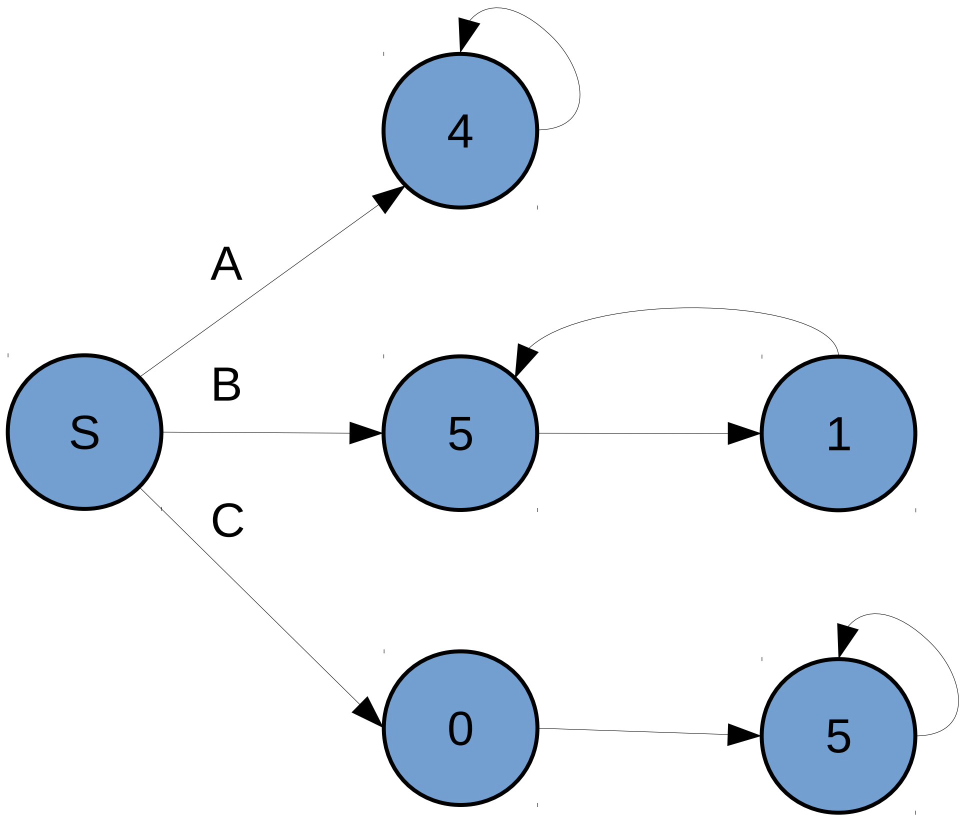
How to Solve an MDP: Value Functions

Value: $V_\pi(s) = E_\pi[\ \Sigma_t\ \gamma^t\ r_t \mid s_0 = s\ ]$

… in s, follow π



Concept Check: Discounts



(1) In functions of γ, what are the values of policies A, B, and C?

(2) When is it better to do B? C?

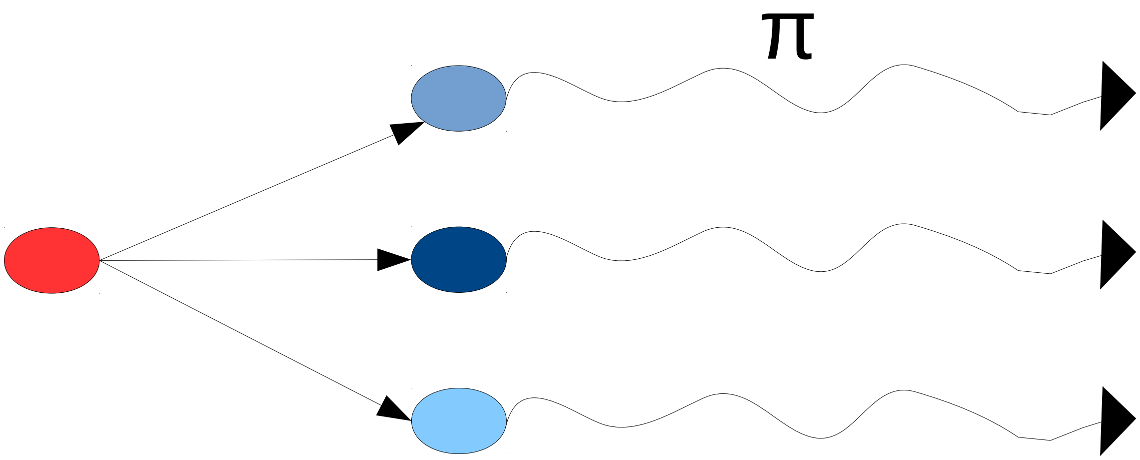
How to Solve an MDP: Value Functions

Value: $V_\pi(s) = E_\pi[\ \Sigma_t\, \gamma^t\, r_t \mid s_0 = s\ ]$

… in s, follow π

Action-Value: $Q_\pi(s,a) = E_\pi[\ \Sigma_t\, \gamma^t\, r_t \mid s_0 = s,\ a_0 = a\ ]$

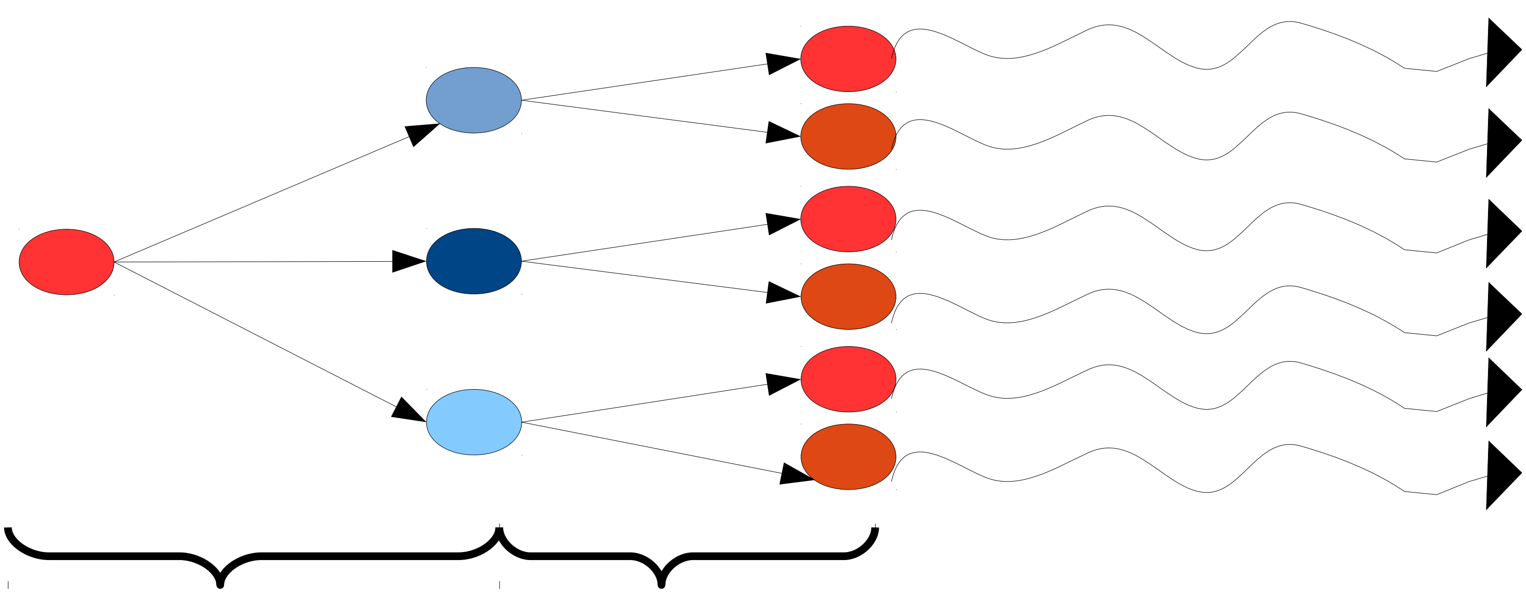
… in s, do a, follow π



Expanding the expression...

$$V_\pi(s) = E_\pi[\sum_t \gamma^t r_t | s_0 = s]$$

$$V_\pi(s) = \underbrace{\sum_a \pi(a|s)}_{\text{Next action}} \sum_{s'} \underbrace{T(s'|s,a)}_{\text{Next state}} [\underbrace{r(s,a,s')}_{\text{Next reward}} + \underbrace{\gamma E_\pi[\sum_t \gamma^t r_t | s_0 = s']}_{\text{Discounted future rewards}}]$$



action: you choose state: world chooses

Expanding the expression...

$$V_\pi(s) = E_\pi[\sum_t \gamma^t r_t | s_0 = s]$$

$$V_\pi(s) = \underbrace{\sum_a \pi(a|s)}_{\text{Next action}} \sum_{s'} \underbrace{T(s'|s,a)}_{\text{Next state}} [\underbrace{r(s,a,s')}_{\text{Next reward}} + \underbrace{\gamma E_\pi[\sum_t \gamma^t r_t | s_0 = s']}_{\text{Discounted future rewards}}]$$

Expanding the expression...

$$V_\pi(s) = E_\pi[\sum_t \gamma^t r_t | s_0 = s]$$

$$V_\pi(s) = \underbrace{\sum_a \pi(a|s)}_{\text{Next action}} \sum_{s'} \underbrace{T(s'|s,a)}_{\text{Next state}} [\underbrace{r(s,a,s')}_{\text{Next reward}} + \underbrace{\gamma E_\pi[\sum_t \gamma^t r_t | s_0 = s']}_{\text{Discounted future rewards}}]$$

$$V_\pi(s) = \sum_a \pi(a|s) \sum_{s'} T(s'|s,a)[r(s,a,s') + \gamma V_\pi(s')]$$

Expanding the expression...

$$V_\pi(s) = E_\pi[\sum_t \gamma^t r_t | s_0 = s]$$

$$V_\pi(s) = \underbrace{\sum_a \pi(a|s)}_{\text{Next action}} \sum_{s'} \underbrace{T(s'|s,a)}_{\text{Next state}} [\underbrace{r(s,a,s')}_{\text{Next reward}} + \underbrace{\gamma E_\pi[\sum_t \gamma^t r_t | s_0 = s']}_{\text{Discounted future rewards}}]$$

$$V_\pi(s) = \sum_a \pi(a|s) \sum_{s'} T(s'|s,a) [r(s,a,s') + \gamma V_\pi(s')]$$

Exercise: Rewrite in finite horizon case, making the rewards and transitions depend on time t… notice how thinking about the future is the same as thinking backward from the end!

Optimal Value Functions

Don't average, take the best!

$$V(s) = max_a Q(s,a)$$
$$V(s) = max_a \sum_{s'} T(s'|s,a)[r(s,a,s') + \gamma V(s')]$$

Q-table is the set of values Q(s,a)

Note: we still have problems – system must be Markov in s, the size of {s} might be large

Can we solve this? Policy Evaluation

$$V_\pi(s) = \sum_a \pi(a|s) \sum_{s'} T(s'|s,a)[r(s,a,s') + \gamma V_\pi(s')]$$

This is a system of linear equations!

Can we solve this? Policy Evaluation

$$V_\pi(s)=\sum_a \pi(a|s)\sum_{s'} T(s'|s,a)[r(s,a,s')+\gamma V_\pi(s')]$$

This is a system of linear equations!

We can also do it iteratively:

$$V_\pi^0(s)=c$$

$$V_\pi^k(s)=\sum_a \pi(a|s)\sum_{s'} T(s'|s,a)[r(s,a,s')+\gamma V_\pi^{k-1}(s')]$$

Will converge because the Bellman iterator is a contraction – the initial value $V^0(s)$ is pushed into the past as the “collected data” r(s,a) takes over.

Can we solve this? Policy Evaluation

$$V_\pi(s) = \sum_a \pi(a|s) \sum_{s'} T(s'|s,a)[r(s,a,s') + \gamma V_\pi(s')]$$

This is a system of linear equations!

We can also do it iteratively:

$$V_\pi^0(s) = c$$

$$V_\pi^k(s) = \sum_a \pi(a|s) \sum_{s'} T(s'|s,a)[r(s,a,s') + \gamma V_\pi^{k-1}(s')]$$

Will converge because the Bellman iterator is a contraction – the initial value $V^0(s)$ is pushed into the past as the "collected data" r(s,a) takes over.

Finally, can apply Monte carlo: many simulations from s, and see what $V_\pi(s)$ is.

Policy Improvement Theorem

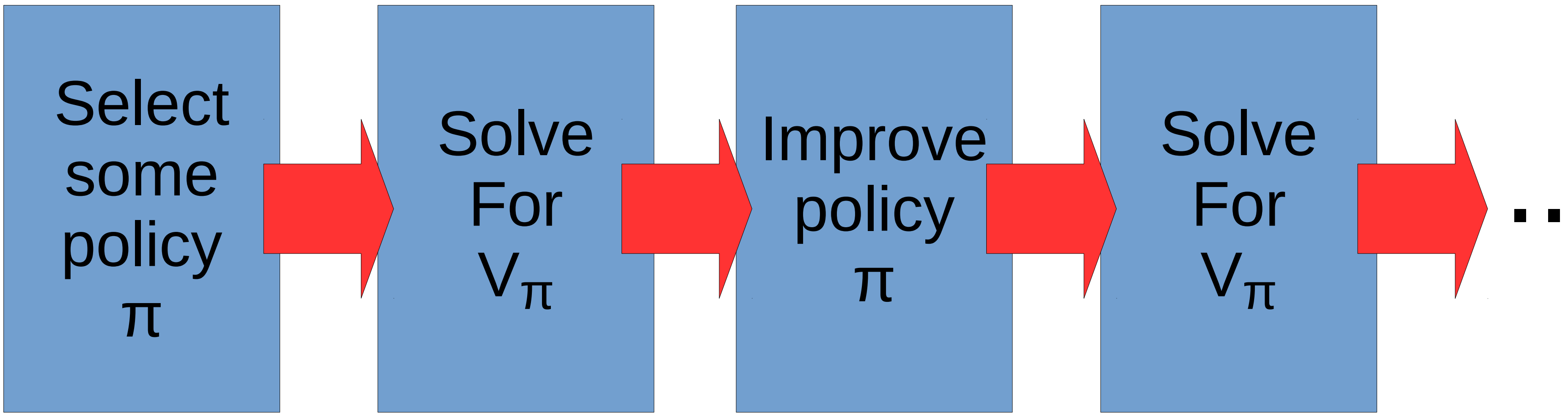
Let π, π' be two policies that are the same except for the action that they recommend at state s.

If $\quad Q_{\pi}(s, \pi'(s)) \; > \; Q_{\pi}(s, \pi(s))$

Then $\quad V_{\pi'}(s) \; > \; V_{\pi}(s)$

Gives us a way to improve policies: just be greedy with respect to Q!

Policy Iteration



Will converge; each step requires a potentially expensive policy evaluation computation

Value Iteration

$$V^k(s) = \underbrace{max_a}_{\text{Policy Improvement}} \underbrace{\sum_{s'} T(s'|s,a)[r(s,a,s') + \gamma V^{k-1}(s')]}_{\text{Policy Evaluation}}$$

Also converges (contraction)

Note that in the tabular case, this is a bunch of inexpensive matrix operations!

Linear programming

$$\min \sum_s V(s)\mu(s)$$

$$s.t.\ V(s) \geq \sum_{s'} T(s'|s,a)[r(s,a,s') + \gamma V(s')]\ \forall a,s$$

For any μ; equality for the best action at optimality

Learning from Experience: Reinforcement Learning

Now, instead of the transition T and reward R, we assume that we only have histories. Why is this case interesting?

- May not have the model
- Even if have model (e.g. rules of go, or Atari simulator code), focuses attention on right place

Taxonomy of Approaches

- Forward Search/Monte Carlo: Simulate the future, pick the best one (with or without a model).
- Value function: Learn $V(s)$
- Policy Search: parametrize policy $\pi_\theta(s)$ and search for the best parameters θ, often good for systems in which the cardinality of θ is small.

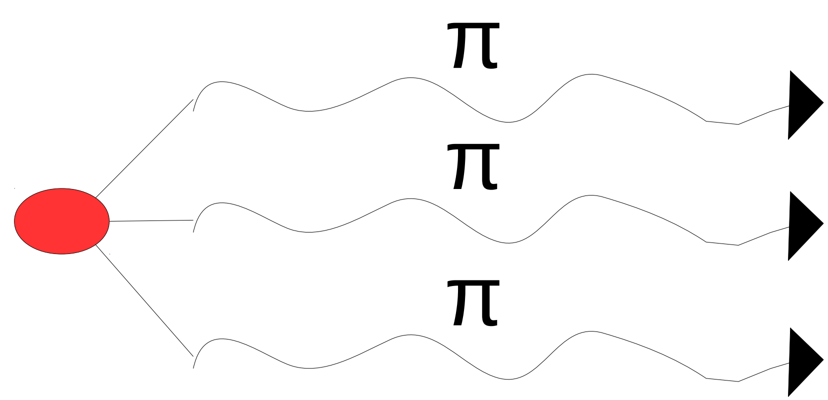
Taxonomy of Approaches

- Forward Search/Monte Carlo: Simulate the future, pick the best one (with or without a model).
- Value function: Learn V(s)
- Policy Search: parametrize policy $\pi_\theta(s)$ and search for the best parameters θ, often good for systems in which the cardinality of θ is small.

Monte Carlo Policy Evaluation

1) Generate N sequences of length T from state s_0 to estimate $V_\pi(s_0)$.
2) If π has some randomness, or we do s_0, a_0, then π, can do policy improvement.

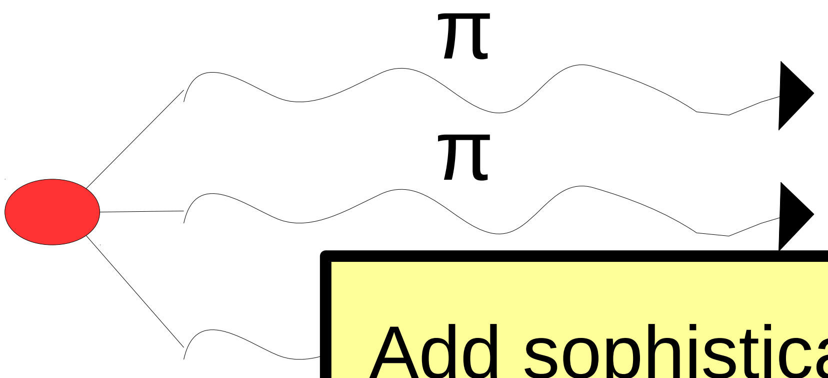
… might need a lot of data! But okay if we have a blackbox simulator.



Monte Carlo Policy Evaluation

1) Generate N sequences of length T from state s_0 to estimate $V_\pi(s_0)$.
2) If π has some randomness, or we do s_0, a_0, then π, can do policy improvement.

… might need a lot of data! But okay if we have a blackbox simulator.



Add sophistication: UCT, MCTS

Taxonomy of Approaches

- Forward Search/Monte Carlo: Simulate the future, pick the best one (with or without a model).
- Value function: Learn V(s)
- Policy Search: parametrize policy $\pi_\theta(s)$ and search for the best parameters θ, often good for systems in which the cardinality of θ is small.

Temporal Difference

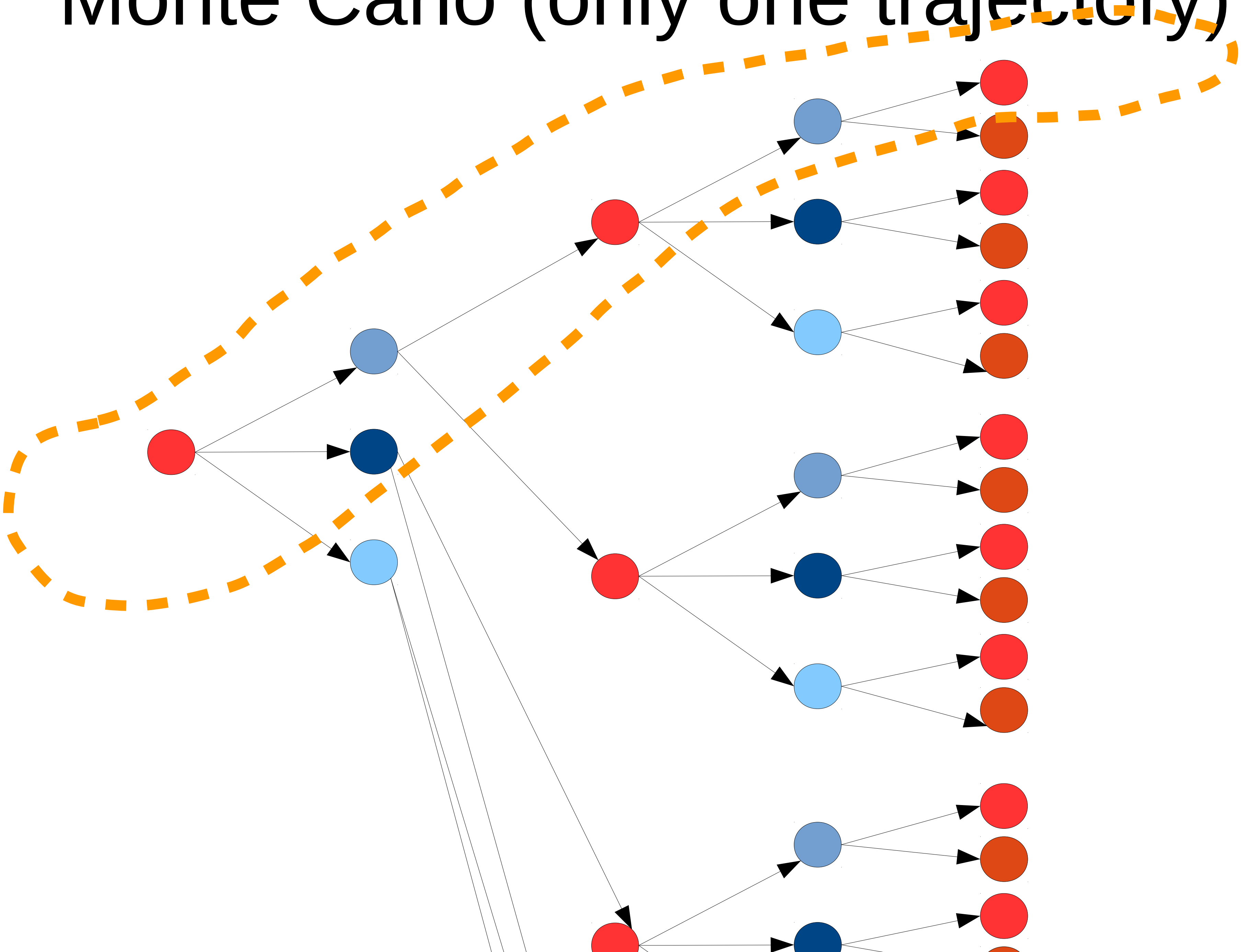
$$V_\pi(s) = E_\pi[\underbrace{\textstyle\sum_t \gamma^t r_t | s_0 = s}_{\text{Monte Carlo Estimate}}] = E_\pi[\underbrace{r_0 + \gamma V_\pi(s')}_{\text{Dynamic Programming}}]$$

TD: Start with some V(s), do π(s), and update:

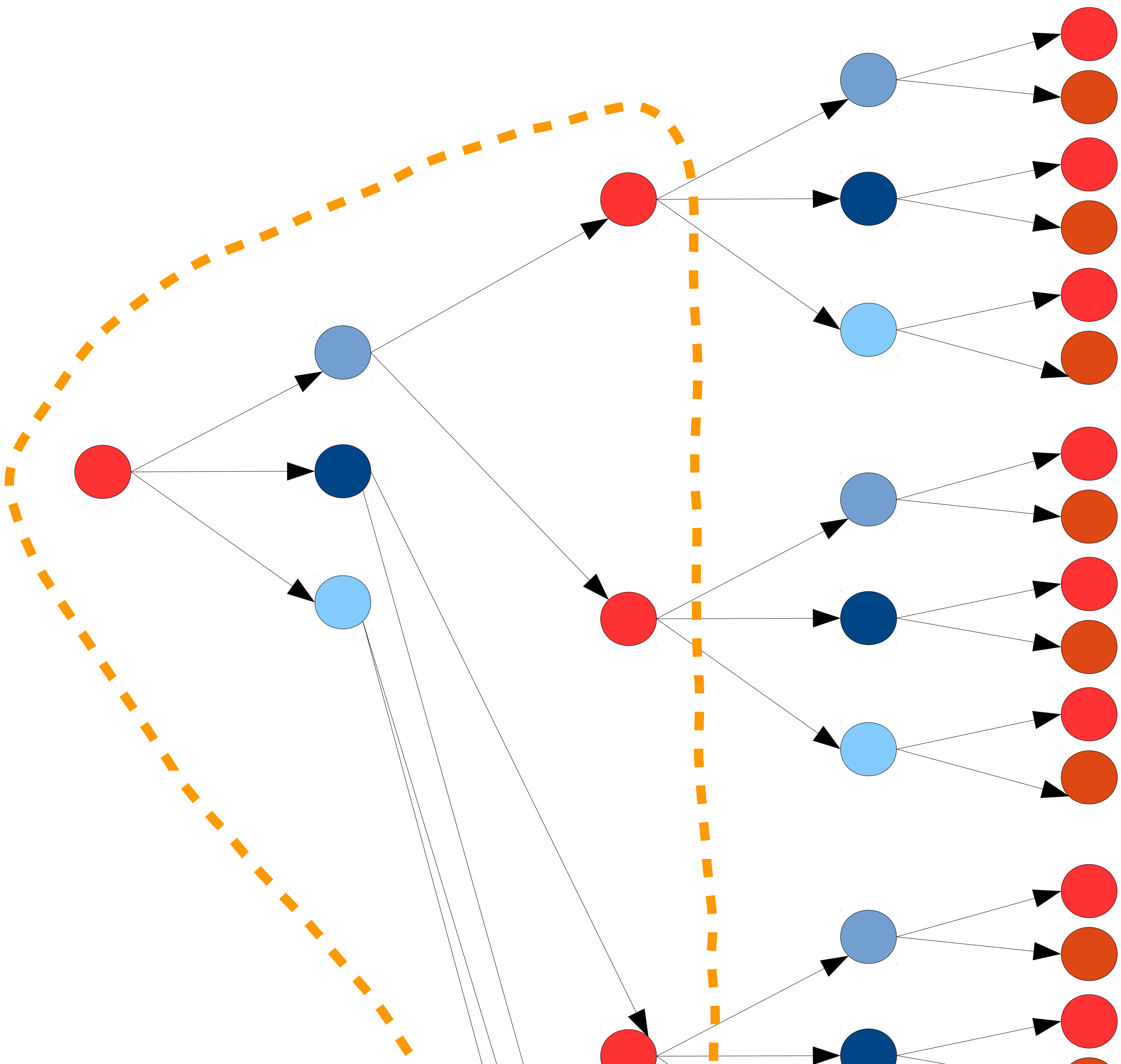
$$V_\pi(s) \leftarrow \underbrace{V_\pi(s)}_{\text{Original Value}} + \alpha_t \underbrace{(r_0 + \gamma V_\pi(s') - V_\pi(s))}_{\text{Temporal Difference: Error between the sampled value of where you went and the stored value}}$$

Will converge if $\sum_t \alpha_t \rightarrow \infty, \sum_t \alpha_t^2 \rightarrow C$

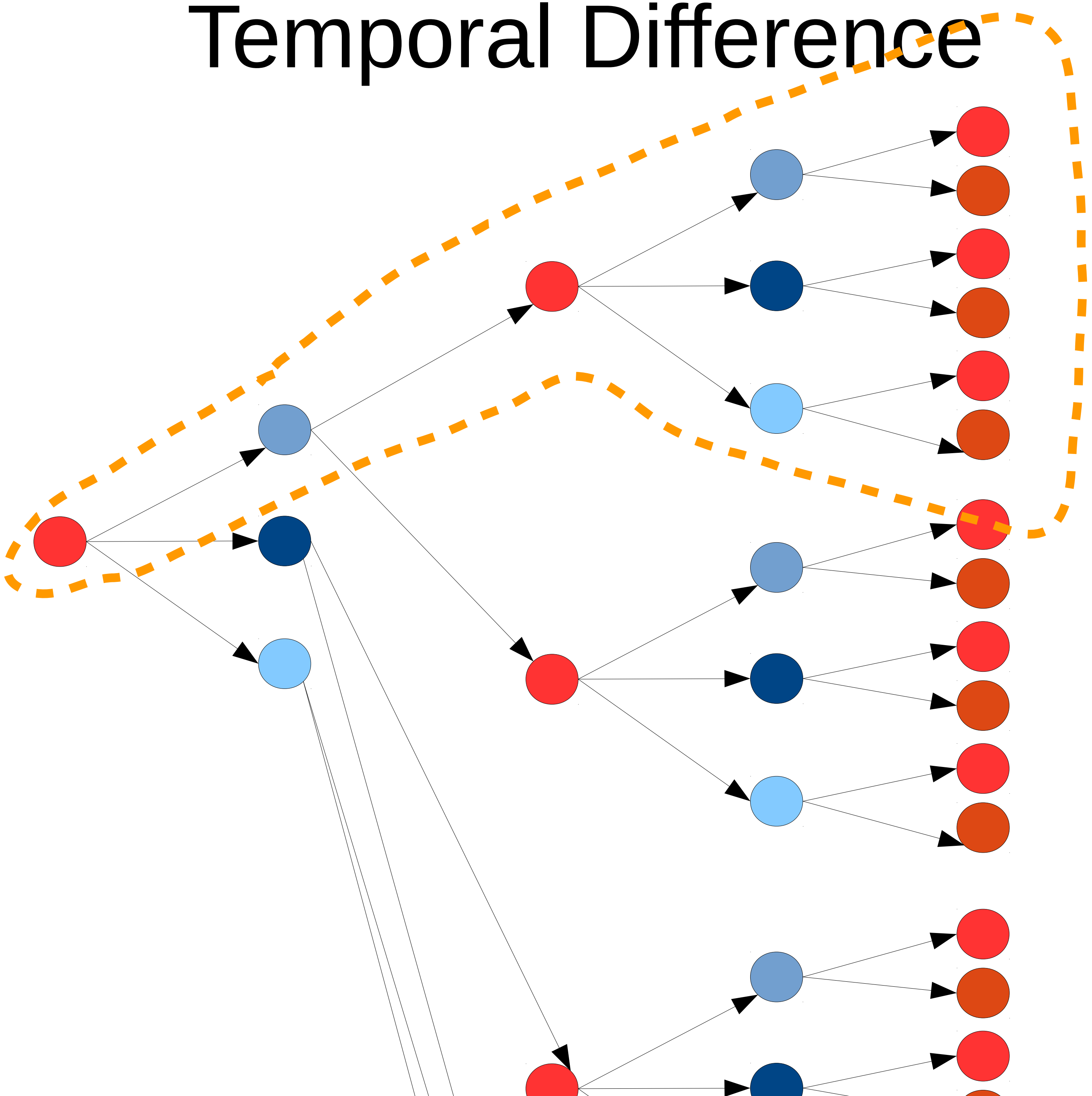
Monte Carlo (only one trajectory)



Value Iteration (all actions)



Temporal Difference



Example (S&B 6.4, Let $\gamma = 1$)

Two states (A,B). Two rewards (0,1).
Suppose we have seen the histories:

A0B0
B1
B1
B1
B1
B1
B1
B0

MC estimate of V(B)?
TD estimate of V(B)?

Example (S&B 6.4, Let $\gamma = 1$)

Two states (A,B). Two rewards (0,1).
Suppose we have seen the histories:

A0B0
B1
B1
B1
B1
B1
B1
B0

MC estimate of V(B)? $V_{MC}(B) = \frac{3}{4}$

TD estimate of V(B)? $V_{TD}(B) = \frac{3}{4}$

Example (S&B 6.4, Let $\gamma = 1$)

Two states (A,B). Two rewards (0,1).
Suppose we have seen the histories:

A0B0	MC estimate of V(B)? $V_{MC}(B) = 3/4$
B1	TD estimate of V(B)? $V_{MC}(B) = 3/4$
B1	MC estimate of V(A)?
B1	TD estimate of V(A)?
B1	
B1	
B1	
B0	

Example (S&B 6.4, Let $\gamma = 1$)

Two states (A,B). Two rewards (0,1).
Suppose we have seen the histories:

A0B0
B1
B1
B1
B1
B1
B1
B0

MC estimate of V(B)? $V_{MC}(B) = 3/4$

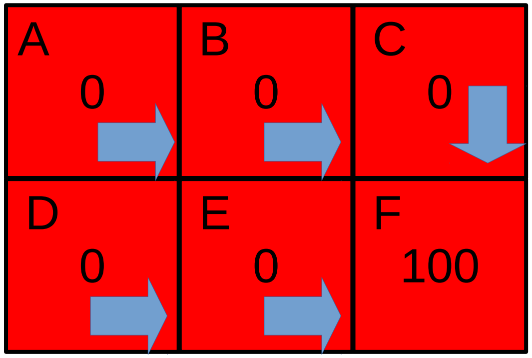
TD estimate of V(B)? $V_{TD}(B) = 3/4$

MC estimate of V(A)? $V_{MC}(A) = 0$

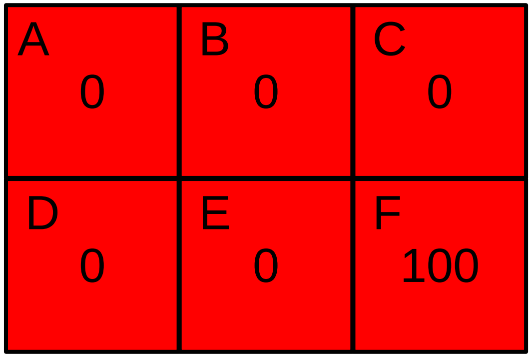
TD estimate of V(A)? $V_{TD}(A) = 3/4$

(because $A \rightarrow B$)

Concept Check: DP, MC, TD



Initialize Values with rewards:



(1) What would one round of value iteration do?

(2) What would MC do after ABCF?

(3) What would TD do after ABCF? (α=1)

From Policy Evaluation to Optimization

SARSA: On-policy

$$Q(s,a) \leftarrow Q(s,a) + \alpha_t \left(r_t + \gamma Q(s',a') - Q(s,a)\right)$$

Improve what you did

From Policy Evaluation to Optimization

SARSA: On-policy

$$Q(s,a) \leftarrow Q(s,a) + \alpha_t \left(r_t + \gamma Q(s',a') - Q(s,a) \right)$$

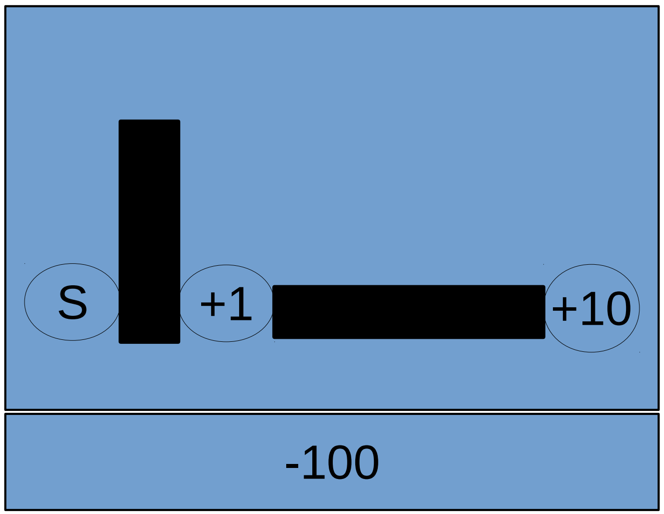
Improve what you did

Q-learning: Off-policy

$$Q(s,a) \leftarrow Q(s,a) + \alpha_t \left(r_t + \gamma \, max_{a'} Q(s',a') - Q(s,a) \right)$$

Improve what you could do

Concept Check



Let δ be the transition noise.
All actions cost -0.1

(1) What is the optimal policy for (γ=.1,δ=.5)? (γ=.1,δ=0)? (γ=.99,δ=.5)? (γ=.99,δ=0)?

(2) Using a ε-greedy policy with ε=.5, γ=.99, δ=0: What will SARSA learn? Q-learning learn?

MC + TD: Eligibility Traces

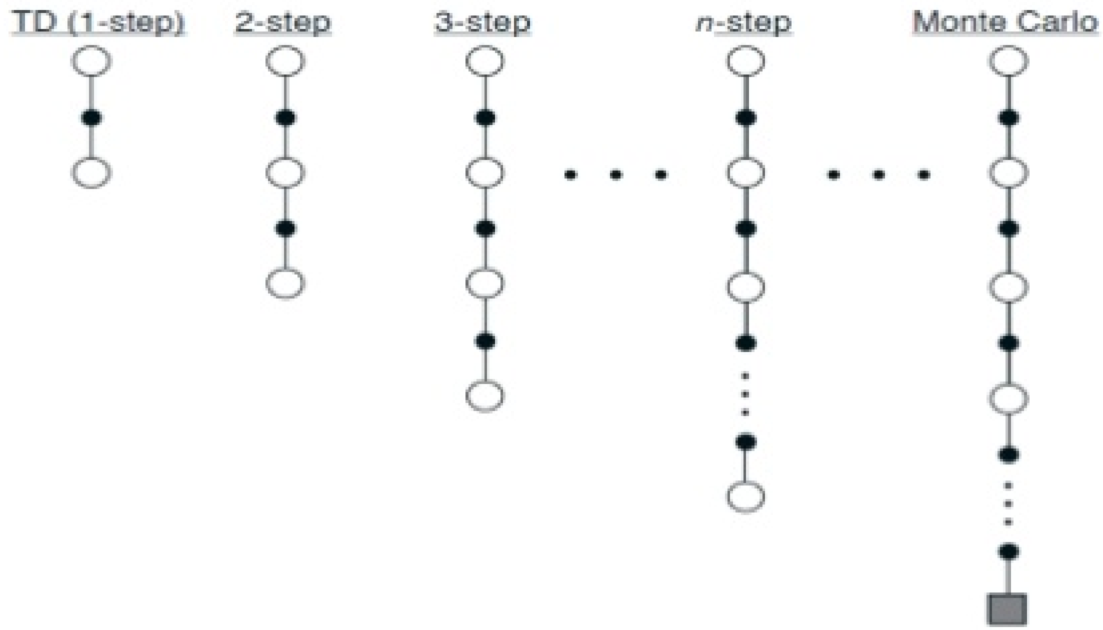
$$TD(0): V(s) \leftarrow V(s) + \alpha_t \big(r_t + \gamma \underbrace{V(s')}_{\text{Biased estimate of future}} - V(s)\big)$$

$$TD(1): V(s) \leftarrow V(s) + \alpha_t \big(r_t + \gamma \underbrace{r_{t+1} + \gamma^2 V(s'')}_{\text{Less bias, more variance}} - V(s)\big)$$

...

Until we get to MC
(all variance, no bias)

Eligibility traces average over all backups

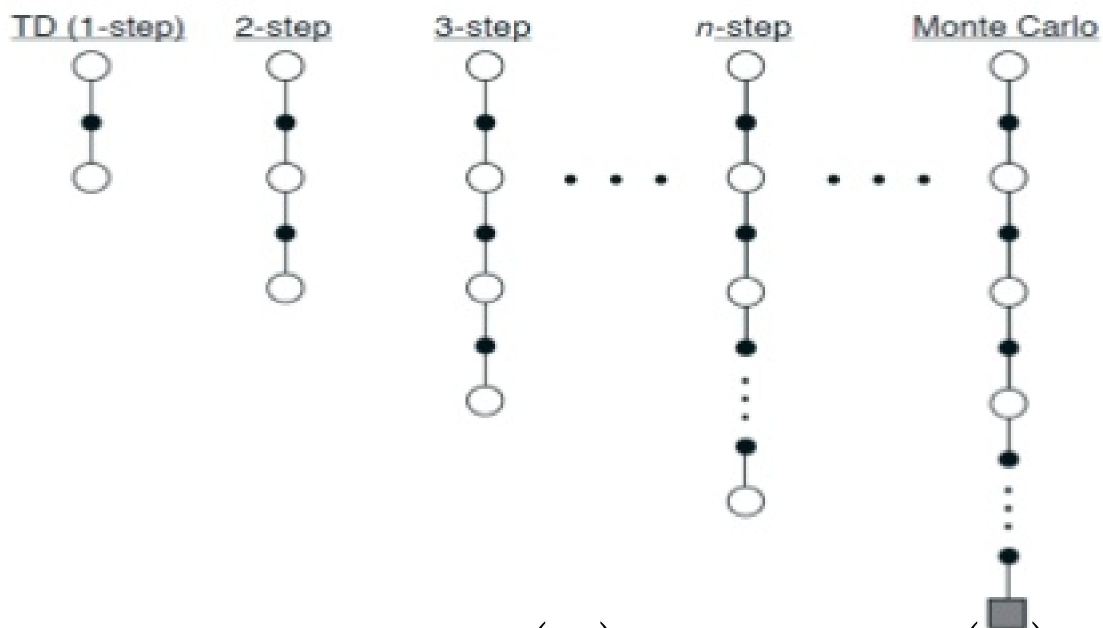


Forward view (can't implement):

$$(1-\lambda)\sum_n \underbrace{\lambda^{n-1}}_{\text{average}}\underbrace{[r_t+\gamma r_{t+1}+\gamma^2 r_{t+2}+\ldots+\gamma^n V(s_{t+n})]}_{\substack{\text{n-step return} \\ \text{(we don't know all these future values)}}}$$

Image from S&B

Eligibility traces average over all backups



Backward view: Let $z_t(s) = \gamma \lambda z_{t-1}(s)$ for all s, except s_t: $z_t(s_t) = 1 + \gamma \lambda z_{t-1}(s_t)$

$$\forall s, V(s) \leftarrow V(s) \alpha_t z_t(s) \left(r_t + \gamma V(s_{t+1}) - V(s_t)\right)$$

Credit assignment back in time.

Image from S&B

Interlude: What about actions??

Given some $Q(s,a)$, how do you choose the action to take? Want to balance exploration with exploitation.

Two simple strategies:

- Epsilon-greedy: take $\text{argmax}_a\, Q(s,a)$ with probability $(1-\varepsilon)$, else take a random action
- Softmax: take actions with probability proportional to $\exp(\, \tau\, Q(s,a)\,)$.

More general principles

Lots of research about curiosity, value of future information, etc. Important ideas:

- Learning has utility (succeed-or-learn)
- Optimism under uncertainty

Examples: interval exploration, UCB/UCT, E3, RMAX. Recent advances in PSRL.

Taxonomy of Approaches

- Forward Search/Monte Carlo: Simulate the future, pick the best one (with or without a model).
- Value function: Learn $V(s)$
- Policy Search: parametrize policy $\pi_\theta(s)$ and search for the best parameters θ, often good for systems in which the cardinality of θ is small.

Next Speaker: Sergey Levine

Practical time!

Clone code from

https://github.com/dtak/tutorial-rl.git

Follow instructions in tutorial.py