Introduction to Reinforcement Learning:

Urtzi Ayesta and Matthieu Jonckheere

CNRS-IRIT & Ikerbasque-Univ. Basque Country

Lecture 4

Outline

- Prediction: How to learn the performance
- Control: How to learn the optimal control?
- Approximate control

Policy Evaluation: Monte Carlo Methods

- Let π be a fixed policy. The goal is to evaluate the value function:

$$V(s) = \mathbb{E}(\sum_{k=0}^{T} \gamma^k R_{t+k+1} | S_t = s)$$

- Monte-Carlo policy evaluation uses *empirical mean return* instead of *expected* return.

Monte-Carlo for episodial (cont.)

Assume we operate or simulate the system under policy π.

- After visiting state s at time t_s, we add-up the total cost until target is reached:

$$\hat{v}(s) = \sum_{t=t_s}^{T} R_t$$

- After k visits to s, we obtain a sequence of total-cost estimates:

$$\hat{v}_1(s), \ldots, \hat{v}_k(s),$$

- The estimate is

$$\hat{V}_k(s) = \frac{1}{k} \sum_{i=1}^{k} \hat{v}_i(s)$$

- LLN ensures that as $k \to \infty$, $\hat{V}_k(s) \to V(s) \quad a.s.$

State Counting Options

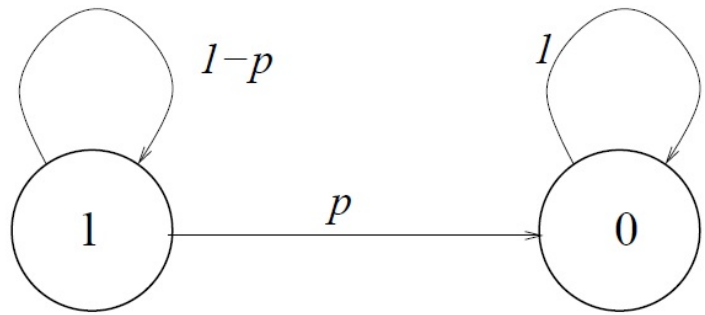
There are several options for state counting:

- (a) Compute $\hat{v}(s)$ only for initial states $(s_0 = s)$.
- (b) Compute $\hat{v}(s)$ each time s is visited.
- (c) Compute $\hat{v}(s)$ only on first visit of s each trial.

Method (b) gives largest number of samples, but are correlated.
Convergence $\hat{V}_k(s) \to V(s)$ $a.s.$ is still guaranteed.

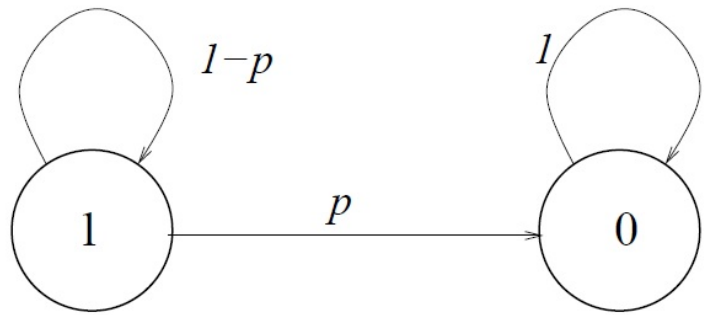
Example: Which method is best?

$$\mathbb{E}((\hat{V} - V)^2) = \underbrace{(\mathbb{E}(\hat{V}) - V)^2}_{\text{Bias}^2} + \underbrace{\mathbb{E}((\hat{V} - \mathbb{E}(\hat{V}))^2)}_{\text{Variance}}$$



Example: Which method is best?

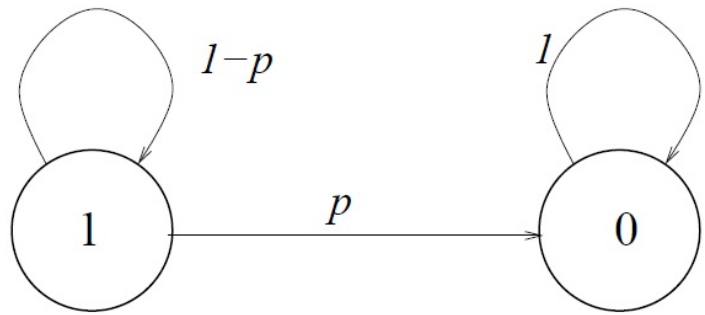
$$\mathbb{E}((\hat{V} - V)^2) = \underbrace{(\mathbb{E}(\hat{V}) - V)^2}_{\text{Bias}^2} + \underbrace{\mathbb{E}((\hat{V} - \mathbb{E}(\hat{V}))^2)}_{\text{Variance}}$$



Reward 1 in state 1, and 0 otherwise
Trajectories of type $(x_0, x_1, \ldots, x_K)$ with $x_k = 1$ for $0 \leq k < K$ and $x_K = 0$.
K is a geometric R.V. with mean $\mathbb{E}(K) = 1/p$.

Example: Which method is best?

$$\mathbb{E}((\hat{V} - V)^2) = \underbrace{(\mathbb{E}(\hat{V}) - V)^2}_{\text{Bias}^2} + \underbrace{\mathbb{E}((\hat{V} - \mathbb{E}(\hat{V}))^2)}_{\text{Variance}}$$



Reward 1 in state 1, and 0 otherwise
Trajectories of type $(x_0, x_1, \ldots, x_K)$ with $x_k = 1$ for $0 \leq k < K$ and $x_K = 0$.
K is a geometric R.V. with mean $\mathbb{E}(K) = 1/p$.
Note that $V^\pi(1) = 1 + (1 - p)V^\pi(1) = 1/p$

Comparing First Visit and Every Visit

- MC First Visit. The reward over one trajectory is K, so it's not biased

$$\mathbb{E}(K) = \frac{1}{p} = V^{\pi}(1)$$

The mean square error is

$$\mathbb{E}((K - \frac{1}{p})^2) = \frac{1}{p^2} - \frac{1}{p}$$

Comparing First Visit and Every Visit (cont.)

- MC Multiple Visits. Over one trajectory, reward is $\frac{1}{K}\sum_{k=0}^{K-1}(K-k) = \frac{K+1}{2}$. In expectation

$$\mathbb{E}(\frac{K+1}{2}) = \frac{1+p}{2p} \neq V^{\pi}(1)$$

However, over multiple trajectories, the empirical mean will converge to $1/p$ (Consistency)
Quadratic error is

$$\mathbb{E}((\frac{K+1}{2} - \frac{1}{p})^2) = \frac{1}{2p^2} - \frac{3}{4p} + \frac{1}{4}$$

Every Visit is a consistent estimator (cont.)

$$\frac{\sum_{i=1}^{n}\sum_{k=0}^{K_i-1} K_i - k}{\sum_{i=1}^{n} K_i} = \frac{\sum_{i=1}^{n} K_i(K_i+1)}{2\sum_{i=1}^{n} K_i} \xrightarrow{p.s.} \frac{\mathbb{E}[K^2]+\mathbb{E}[K]}{2\mathbb{E}[K]} = \frac{1}{p} = V^{\pi}(1)$$

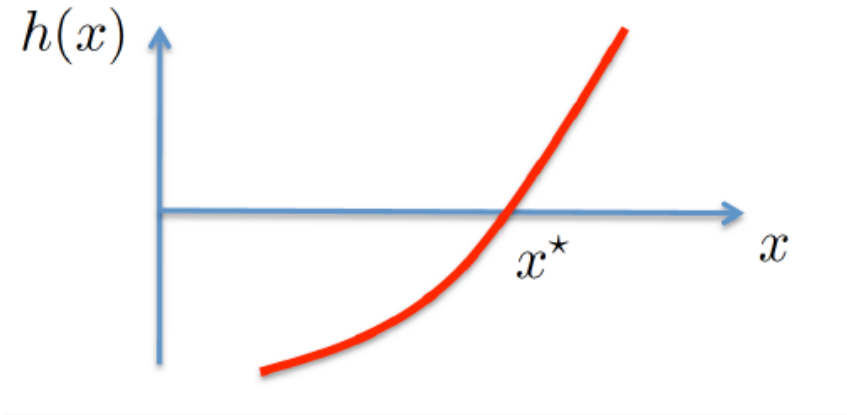
Incremental mean

The empirical mean of $\mu_1, \mu_2, \ldots$ of a sequence $x_1, x_2, \ldots$ can be computed incrementally

$$\begin{aligned}
\mu_k &= \frac{1}{k}\sum_{j=1}^{k} x_j \\
&= \frac{1}{k}\left(x_k + \sum_{j=1}^{k-1} x_j\right) \\
&= \frac{1}{k}\left(x_k + (k-1)\mu_{k-1}\right) \\
&= \mu_{k-1} + \frac{1}{k}\left(x_k - \mu_{k-1}\right)
\end{aligned}$$

Convergence of iterative algorithms

Particular instance of stochastic approximation



Find the root of $h(x) = 0$ from noisy measurements
$y_n = h(x_n) + M_n$, with $\mathbb{E}(M_n) = 0$
Robbins-Monro Algorithm: $x_{n+1} = x_n - \alpha_n(h(x_n) + M_n)$

Convergence of iterative algorithms (cont.)

Theorem If

$$\sum_{n=1}^{\infty} \alpha_n = \infty$$

and

$$\sum_{n=1}^{\infty} \alpha_n^2 < \infty$$

then the SA iterations converge, i.e.:

$$x_n \to x^* \qquad w.p.1$$

Incremental mean is a particular case

$$\theta_{n+1} = \theta_n + \frac{1}{n+1}(x_n - \theta_n)$$

Particular case of SA with $\alpha_n = 1/n$, and Let $y_n = x_n - \theta_n$.
We have that $x_n = \mu + M_n$, and

$$\begin{aligned} y_n &= x_n - \theta_n \\ &= \underbrace{\mu - \theta_n}_{h(\theta_n)} + M_n \end{aligned}$$

Sketch of Proof for $\gamma_n = n^{-1}$

By Chernoff-Hoeffding bound we have:

$$\mathbb{P}(|\mu_n - \mu| \geq \epsilon) \leq 2e^{-2n\epsilon^2}$$

Sketch of Proof for $\gamma_n = n^{-1}$

By Chernoff-Hoeffding bound we have:

$$\mathbb{P}(|\mu_n - \mu| \geq \epsilon) \leq 2e^{-2n\epsilon^2}$$

Apply Borel-Cantelli to events $E_n = \{|\mu_n - \mu| \geq \epsilon\}$
$\Rightarrow$ $\sum_{n\geq 1} \mathbb{P}(E_n)$ is finite with prob. 1.

Sketch of Proof for $\gamma_n = n^{-1}$

By Chernoff-Hoeffding bound we have:

$$\mathbb{P}(|\mu_n - \mu| \geq \epsilon) \leq 2e^{-2n\epsilon^2}$$

Apply Borel-Cantelli to events $E_n = \{|\mu_n - \mu| \geq \epsilon\}$
$\Rightarrow$ $\sum_{n\geq 1} \mathbb{P}(E_n)$ is finite with prob. 1.
Finite number of instants such that $|\mu_n - \mu| \geq \epsilon$
For all ϵ_k we can find n_k such that $\mathbb{P}(\forall n \geq n_k, |\mu_n - \mu| \leq \epsilon_k) = 1$

$$\mathbb{P}(\forall k, \exists n_k, \forall n \geq n_k, |\mu_n - \mu| \leq \epsilon_k) = 1$$

which is the definition of $\lim_{n\to\infty} \mu_n = \mu$

Sketch proof: ODE method

Related asymptotic behavior SA algorithm to an ODE:

$$\frac{d}{dt}\theta(t) = h(\theta(t))$$

If we can show that as $n \to \infty$, the estimates $\{\theta_n\}$ follow the ODE, then

- Stationary points of the ODE are given by θ^* s.t. $h(\theta^*) = 0$.
- An obvious requirement for $\theta_n \to \theta^*$ is $\theta(t) \to \theta^*$ $\Longrightarrow \theta^*$ is globally asymptotically stable equilibrium

  Necessary condition for convergence, also sufficient under additional assumptions.

Sketch proof: ODE method (cont.)

Given $\{\theta_n, \alpha_n\}$, definte continuous process $\theta(t)$: $t_n = \sum_{k=0}^{n-1} \alpha_k$ and $\theta(t_n) = \theta_n$

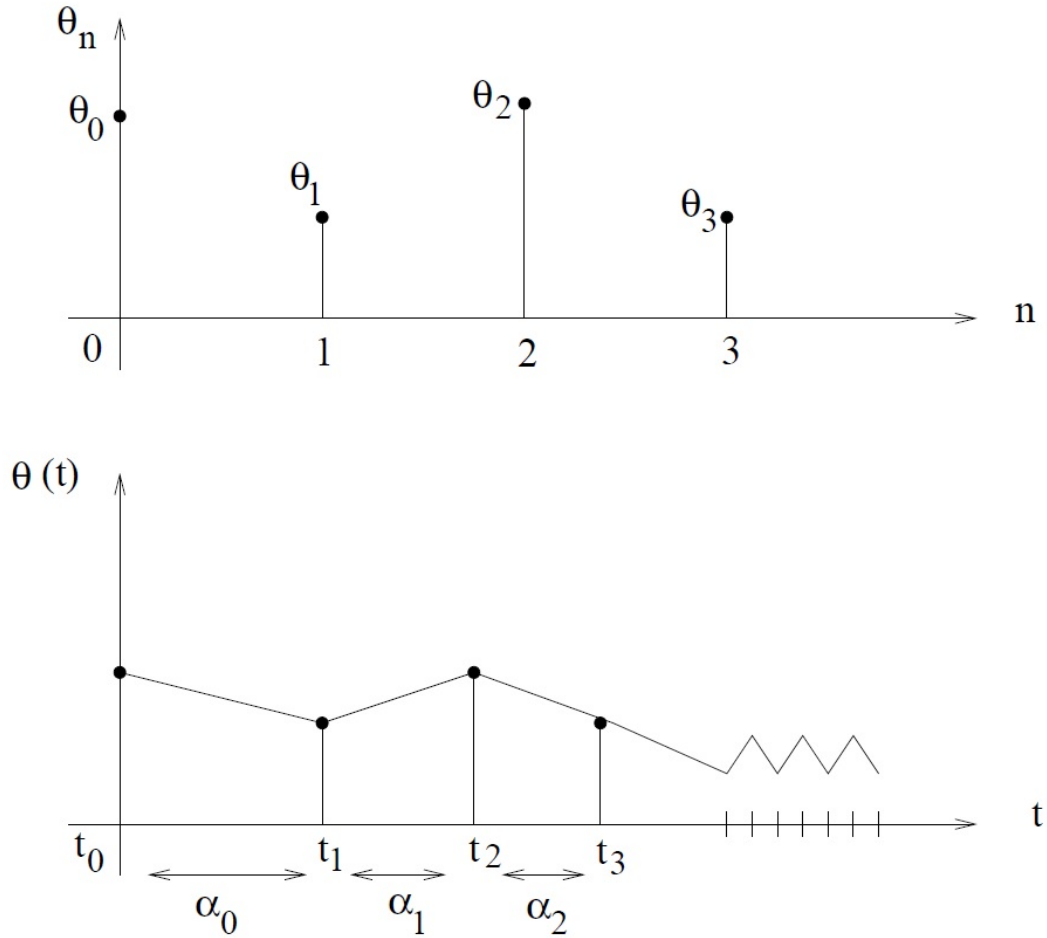


Figure: Time axis t is rescaled according to gains $\{\alpha_n\}$

Sketch proof: ODE method (cont.)

By substituting:

$$\theta(t+\Delta t) = \theta(t) + \sum_{k \in K(t,\Delta t)} \alpha_k \left[h(\theta_n) + M_n\right]$$

Over a fixed Δt, the total gain is approx const:
$\sum_{k \in K(t,\Delta t)} \alpha_k \approx \Delta t$, where $K(t, \Delta t) = \{k : t \leq t_k < t + \Delta t\}$

- For t large, α_k small and many in the summation, noise is averaged out: $\sum \alpha_k M_k \to 0$
- For Δt small, θ_k is approx constant over $K(t, \Delta t)$: $h(\theta_k) \approx h(\theta(t))$

We thus have

$$\theta(t+\Delta t) \approx \theta(t) + \Delta t \cdot (\theta(t)),$$

and for $\Delta t \to 0$, this reduces to the ODE: $\frac{d}{dt}\theta(t) = h(\theta(t))$

Iterative MC

$$\hat{V}_(S_t) = \hat{V}(S_t) + \alpha_t(\hat{G}_t - \hat{V}(S_t))$$

with $\alpha_t = \frac{1}{t}$, will converge to $V(s)$ a.s.

Temporal-Difference Learning

- TD methods learn directly from episodes of experience
- TD is model-free: no knowledge of MDP transitions / rewards
- TD learns from incomplete episodes, by bootstrapping
- TD updates a guess towards a guess

MC and TD

Goal: Learn V^π online from experience under policy π

- Incremental MC

$$\hat{V}_(S_t) = \hat{V}(S_t) + \alpha_t(\hat{G}_t - \hat{V}(S_t))$$

- TD(0): Simplest temporal-difference learning algorithm
  - $$V(S) \longleftarrow V(S) + \alpha(R + \gamma V(S') - V(S))$$
  - $R + \gamma V(S')$ is called the TD target
  - $\delta = R + \gamma V(S') - V(S)$ is called the TD error

TD(0) algorithm

- $R + \gamma V(S')$ is an unbiased estimate for the right-hand side of Value Iteration
- Noisy, due to randomness in R and S', thus

$$\hat{V}(S) = (1 - \alpha_n)\hat{V}(S) + \alpha_n(R + \hat{V}(S'))$$

$$\hat{V}(S) \longleftarrow \hat{V}(S) + \alpha(R + \gamma\hat{V}(S') - \hat{V}(S))$$

- $\hat{V}(S)$ is updated based on estimate $\hat{V}(S')$
  $\Rightarrow$ Bootstrapping

Convergence of TD(0)

Convergence follows from general SA result:

$$\hat{V}_{n+1}(S) = \hat{V}_n(S) + \alpha_n(R + \gamma \hat{V}_n(S') - \hat{V}_n(S))$$

Let $V(s) = \mathbb{E}(R + \gamma \hat{V}_n(S'))$. We have

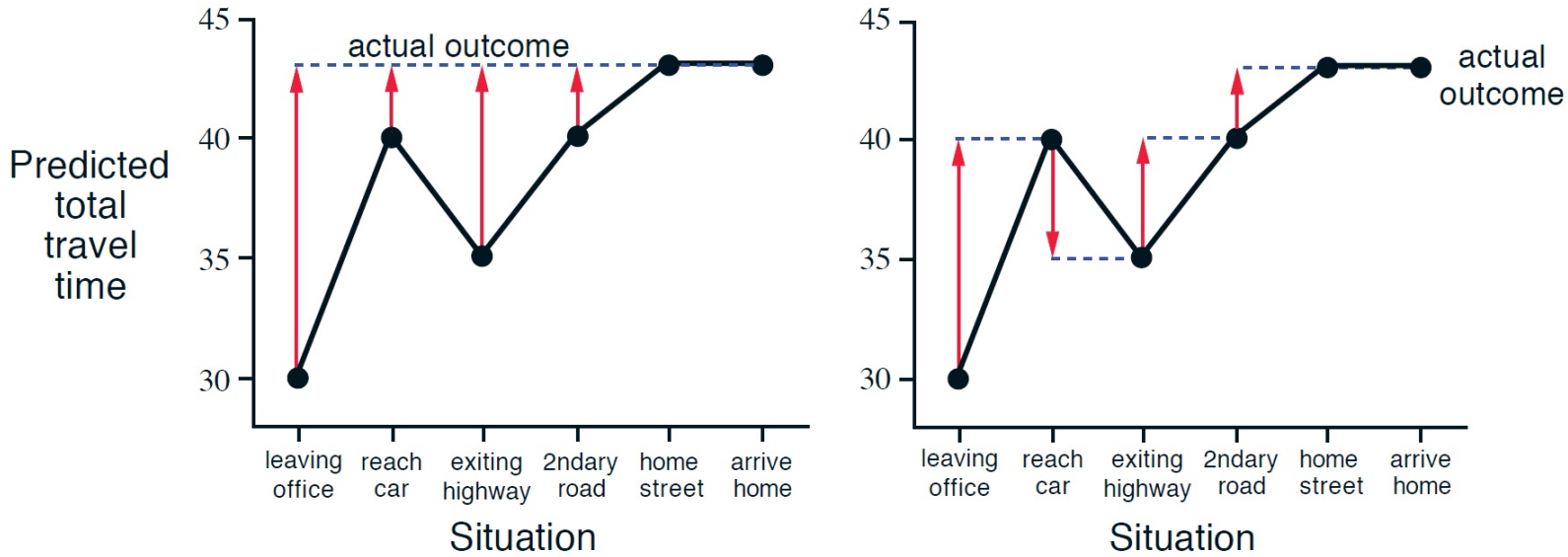
$$y_n = \underbrace{V(S) - \hat{V}_n(S)}_{h(\hat{V}_n(S))} + \underbrace{(R + \gamma \hat{V}_n(S') - V(S))}_{M_n}$$

- Convergence: If $\alpha_n \to 0$ at suitable rate, then $\hat{V} \to V^\pi$ w.p. 1
- For example ($\alpha_n \approx 1/\text{number of visits to } S$),

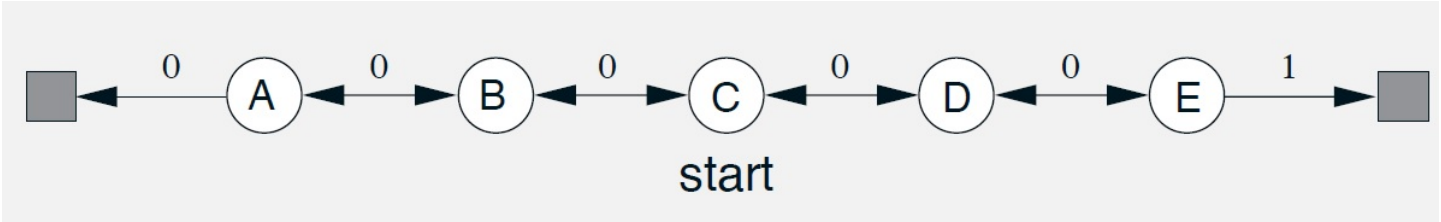
Example: Driving Home Example

State	Elapsed Time (minutes)	Predicted Time to Go	Predicted Total Time
leaving office	0	30	30
reach car, raining	5	35	40
exit highway	20	15	35
behind truck	30	10	40
home street	40	3	43
arrive home	43	0	43

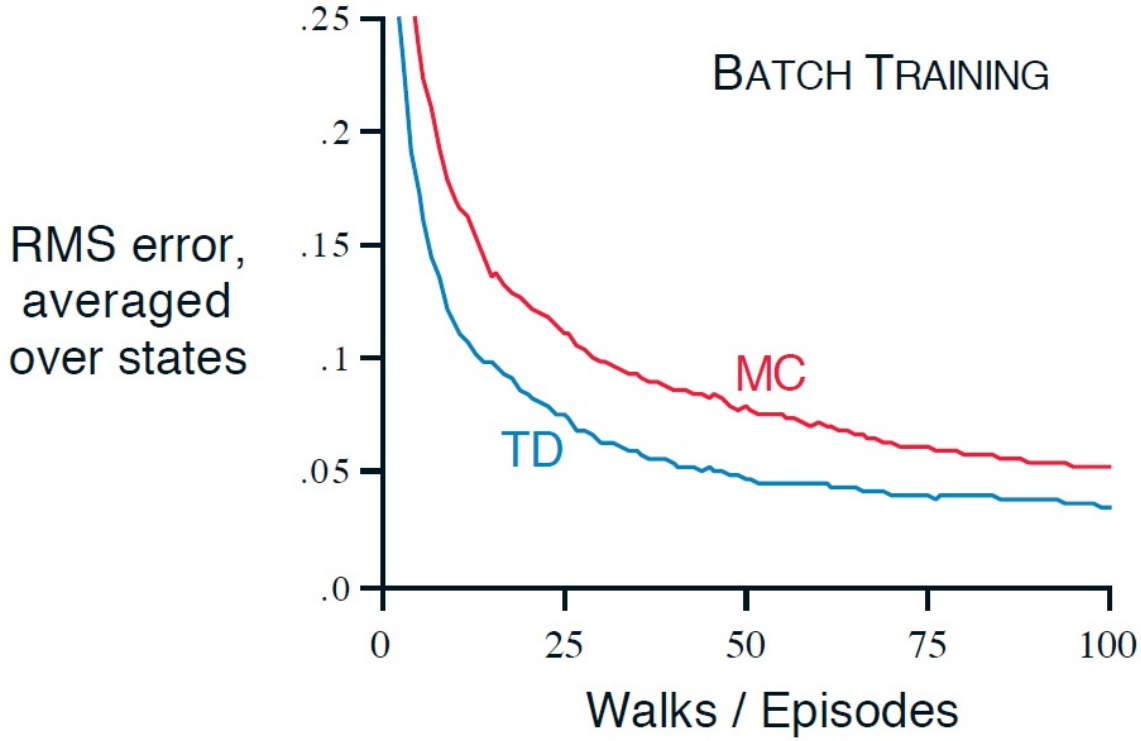
Example: Driving Home Example



Optimality of TD(0)



$V(s)$ is equal to probability of finishing on the right.



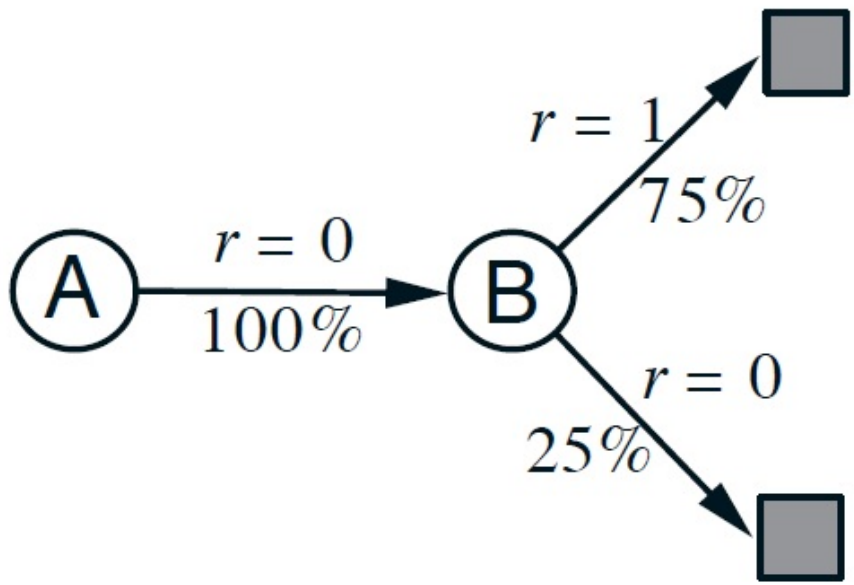
Optimality of TD(0) (cont.)

You're the predictor
The AB example.
Consider the following transitions and rewards:
A, 0, B, 0
B, 1
B, 1
B, 1
B, 1
B, 1
B, 1
B, 0
What is $V(A)$, $V(B)$?

Optimality of TD(0) (cont.)

We can argue that $V(B) = 3/4$. But what about $V(A)$?

- A once, and reward 0, so $V(A) = 0$
- 100% of times, from A we move to B, so $V(A) = 3/4$. This is the result by TD(0)



MC better with existing data, but TD(0) lower error on future data

Advantages vs. Disadvantages of MC vs. TD

- TD can learn before knowing the final outcome
  - TD can learn online after every step
  - MC must wait until end of episode before return is known

Advantages vs. Disadvantages of MC vs. TD

- TD can learn before knowing the final outcome
  - TD can learn online after every step
  - MC must wait until end of episode before return is known
- TD can learn without the final outcome
  - TD can learn from incomplete sequences
  - MC can only learn from complete sequences
  - TD works in continuing (non-terminating) environments
  - MC only works for episodic (terminating) environments

Advantages vs. Disadvantages of MC vs. TD (2)

- MC has high variance, zero bias
  - Good convergence properties (even with function approximation)
  - Not very sensitive to initial value
  - Very simple to understand and use

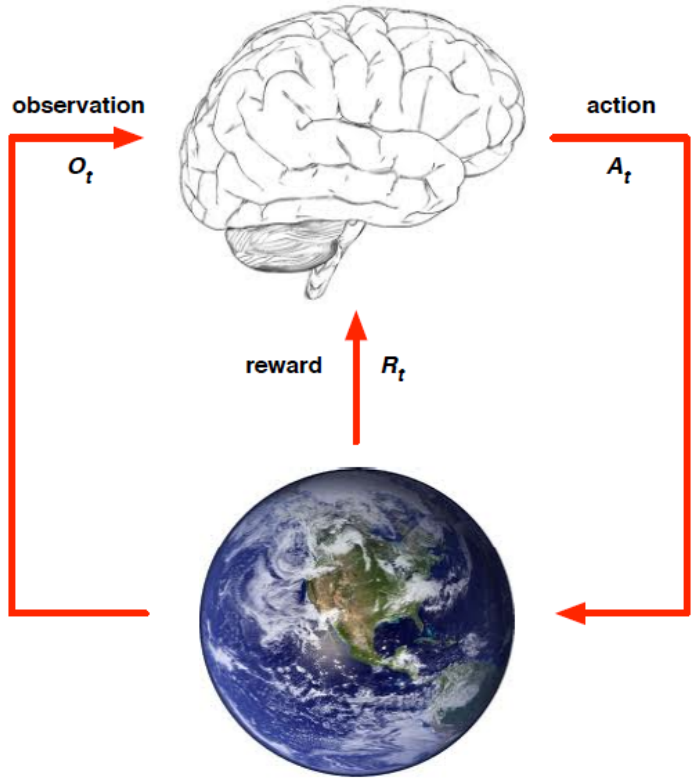
Advantages vs. Disadvantages of MC vs. TD (2)

- MC has high variance, zero bias
  - Good convergence properties (even with function approximation)
  - Not very sensitive to initial value
  - Very simple to understand and use
- TD has low variance, some bias
  - Usually more efficient than MC
  - TD(0) converges to $V_\pi(s)$
  - not always with function approximation
  - More sensitive to initial value

Theoretical Advantages vs. Disadvantages of MC vs. TD (3)

- MC does not take into account the Markovian structure.
  It is the best estimator in the sense of mean squares.
  Explores full trajectories.
- TD supposes a Markovian structure.
  Best estimator in the sense of maximum likelyhood (supposing MDP).
  Explores steps.

Control: Exact Solution Methods



The aim is to choose A_t in order to maximize
$R_{t+1} + \alpha R_{t+2} + \alpha^2 R_{t+3} + \cdots$
The **action-value** function says how good is a state-action pair

$$q_\pi(s, a) = \mathbb{E}\left(R_{t+1} + \alpha R_{t+2} + \alpha^2 R_3 + \cdots | S_t = s, A_t = a, A_{t+1:\infty} \sim \pi\right)$$

Optimal policies

A policy π_* is optimal if it maximizes the action-value function

$$q_*(s, a) = q_{\pi_*}(s, a) = \max q_\pi(s, a)$$

Given the optimal action-value function, the optimal action is

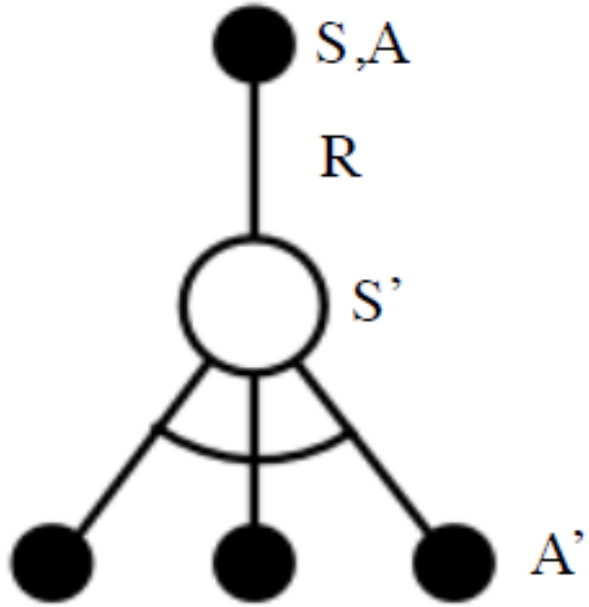
$$\pi_*(s) = \text{argmax}_a q_*(s, a)$$

Question: How to learn $q_*(s, a)$?

On and Off-policy learning

- On-policy Learning
  - Learn on the go
  - Learn about policy π from experience sampled from π
- Off-policy learning
  - look over someone's shoulder
  - Learn about policy π from experience sampled from μ

Off-policy: Q-learning



$$Q(S,A) \leftarrow Q(S,A) + \alpha \left(R + \gamma \max_{a'} Q(S', a') - Q(S, a) \right)$$

Theorem: For appropriate choice of step γ, Q converges to q_*

Q learning

Initialize $Q(s,a), \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(\textit{terminal-state}, \cdot) = 0$
Repeat (for each episode):
  Initialize S
  Repeat (for each step of episode):
    Choose A from S using policy derived from Q (e.g., ε-greedy)
    Take action A, observe R, S'
    $Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma \max_a Q(S',a) - Q(S,A)\big]$
    $S \leftarrow S'$;
  until S is terminal

Learning long-term optimal behavior without any model of the environment

Proof of convergence (deterministic case for simplicity)

Start with some $\hat{Q}$ values. Let: $\Delta_0 = \max_{s,a} |\hat{Q}(s,a) - Q(s,a)|$.

Define a "Q-interval" to be an interval in which every (s,a) is tried at least once.

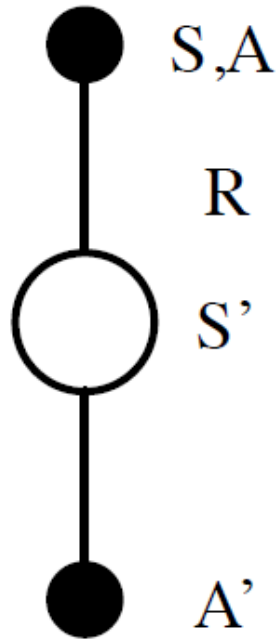
Claim: after the ith Q-interval, $\max_{s,a} |\hat{Q}(s,a) - Q(s,a)| \leq \Delta_0 \gamma^i$. In addition, during the ith Q-interval, this maximum difference will be at most $\Delta_0 \gamma^{i-1}$.

Proof: Prove by induction. Base case (the very beginning) is OK. After an update in interval i of $\hat{Q}(s,a)$, (let's say that action a takes you from s to s') we have:

$$\begin{aligned}
|\hat{Q}(s,a) - Q(s,a)| &= |(r + \gamma \max_{a'} \hat{Q}(s',a')) - (r + \gamma \max_{a'} Q(s',a')| \\
&= \gamma |\max_{a'} \hat{Q}(s',a') - \max_{a'} Q(s',a')| \\
&\leq \gamma \max_{a'} |\hat{Q}(s',a') - Q(s',a')| \\
&\leq \gamma \cdot \gamma^{i-1} \Delta_0. \quad \blacksquare
\end{aligned}$$

So, to get convergence, pick actions so that #intervals $\to \infty$.

How to learn the action-value of a policy: SARSA



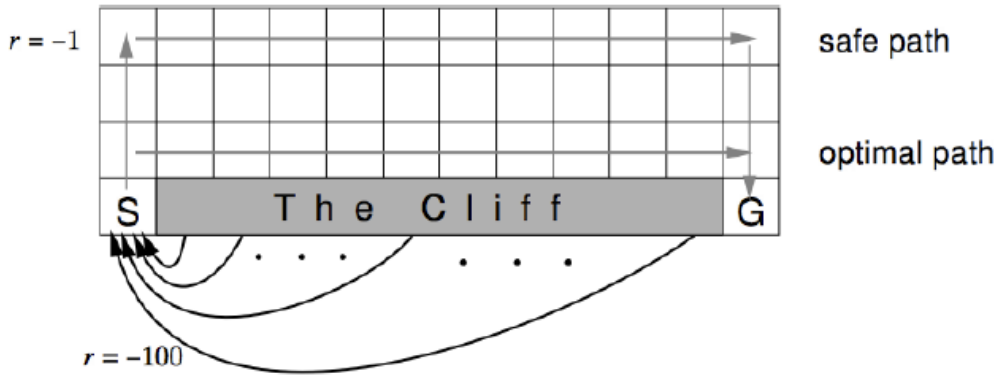
$$Q(S, A) \leftarrow Q(S, A) + \alpha \left(R + \gamma Q(S', A') - Q(S, a) \right)$$

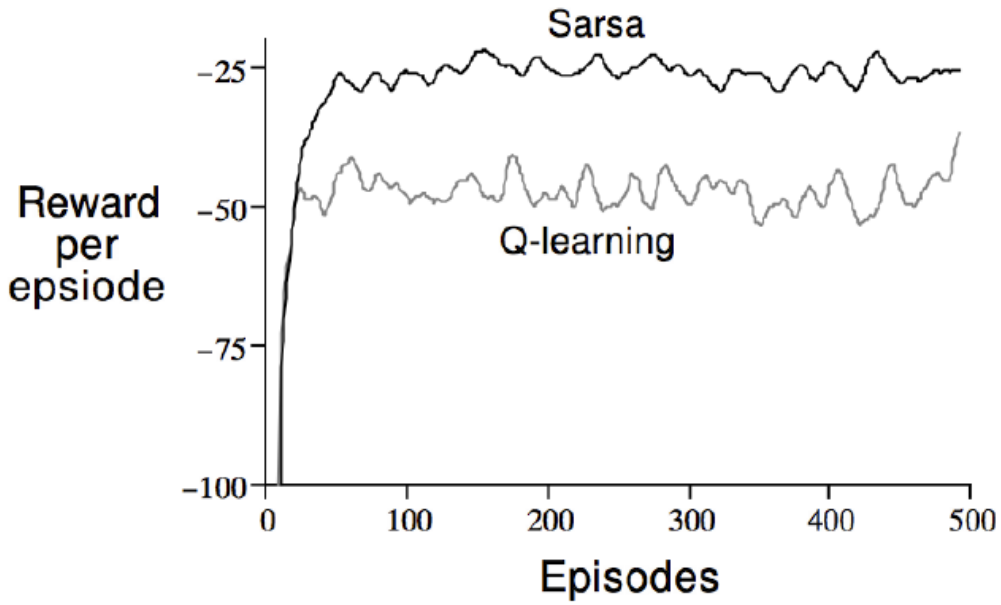
SARSA algorithm

Initialize $Q(s,a), \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(\textit{terminal-state}, \cdot) = 0$
Repeat (for each episode):
  Initialize S
  Choose A from S using policy derived from Q (e.g., ε-greedy)
  Repeat (for each step of episode):
    Take action A, observe R, S'
    Choose A' from S' using policy derived from Q (e.g., ε-greedy)
    $Q(S,A) \leftarrow Q(S,A) + \alpha\left[R + \gamma Q(S',A') - Q(S,A)\right]$
    $S \leftarrow S'$; $A \leftarrow A'$;
  until S is terminal

$$Q(S,A) \leftarrow Q(S,A) + \alpha\left(R + \gamma Q(S',A') - Q(S,a)\right)$$

On vs. Off policie





On-policy methods perform better, but find poorer policies

Model-Based and Model-Free RL

Model-Free RL

- No model
- Learn value function and policy from experience

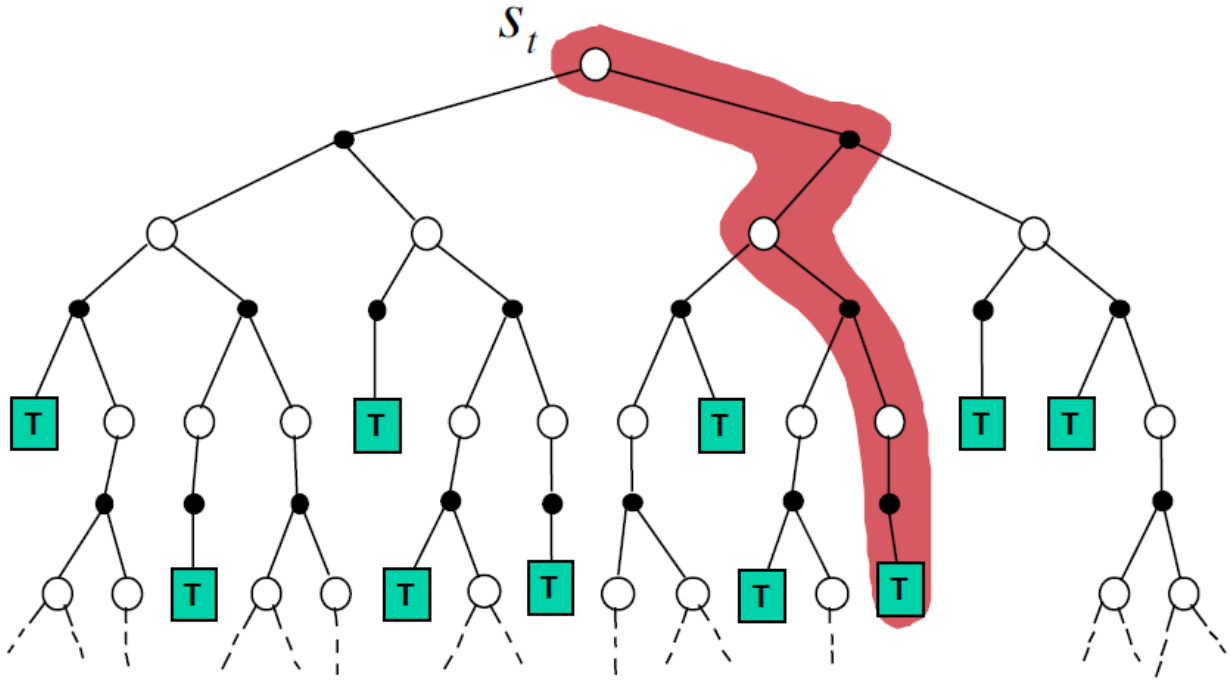
Model-based RL

- Learn a model from experience
- Plan from model

$$\hat{\mathbb{P}}(S_{t+1}, R_{t+1}|S_t, A_t) \approx \mathbb{P}(S_{t+1}, R_{t+1}|S_t, A_t)$$

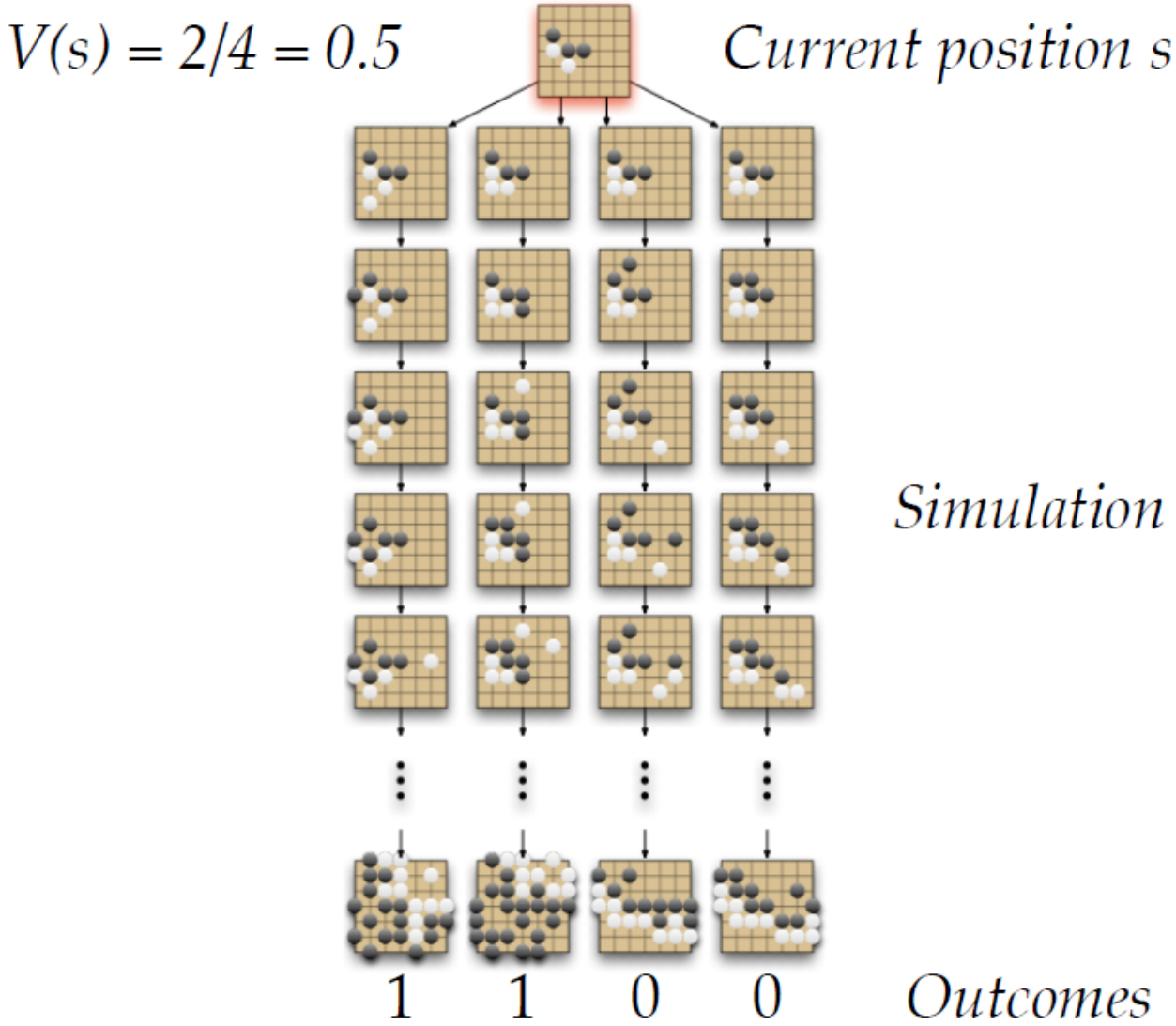
SImulation-based search

Simulate using the model
Apply model-free RL to simulated experience



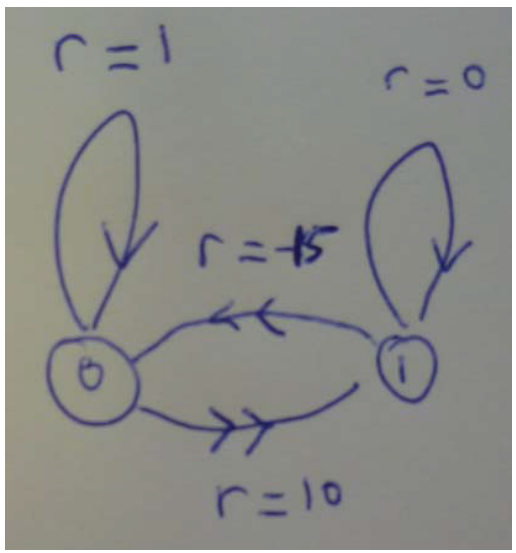
Go

Learning from simulation is effective method to search



Exercise 1 : Learning version of problem from TD

Let us consider the same Markov Chain as in the TD. We now assume that this model is not known to us. This Markov Chain wlll now represent the "environment".
Note : In this lecture, γ is the learning parameter, and α is the discounting factor!



Exercise 1 : Learning version of problem from TD

Evaluation:. We will fix a policy, so the MDP reduces to a Markov Process (recall slide 22 in Lecture 2). As a baseline policy, we take the policy that chooses (in both states) both actions with probability $1/2$.

Planning Assuming the model is known (so we know how the Markov Chain "looks like"), calculate $V(1)$ and $V(2)$.
Help: You may use the equation in $18/43$ in Lecture 2.

Learning We will now assume that the model is not known to us. We're the "agent", and we only observe the sequence of states and rewards. Implement the algorithm $TD(0)$, and verify that the algorithm learns the correct values of the value function, i.e., those derived in the first item above.
Help: $TD(0)$ is described in slide $15/45$. As $\gamma(s)$ you can take $\gamma(s) = 1/(number \quad visits \quad state \quad s)$

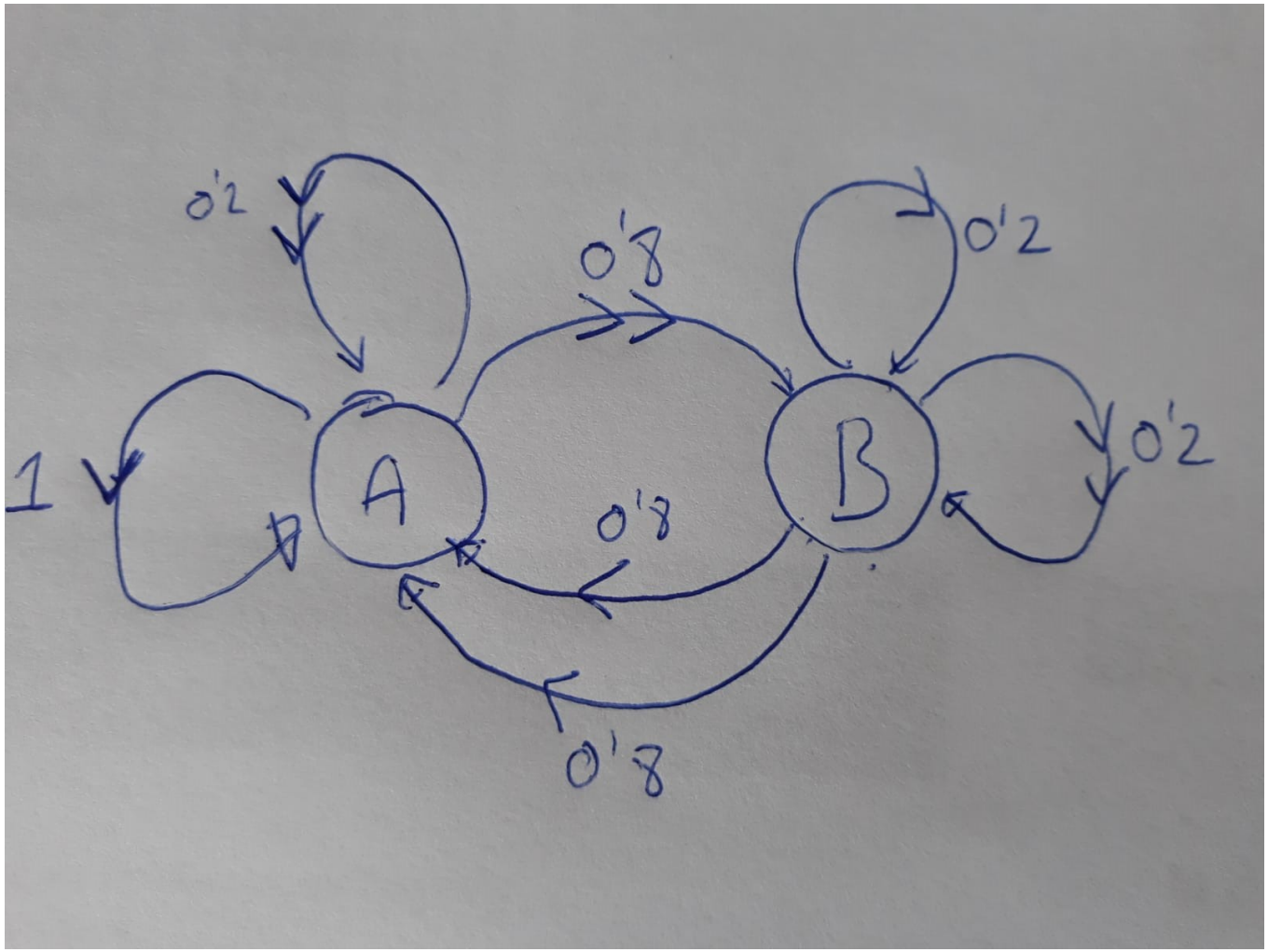
Exercise 1 : Learning version of problem from TD

Control: We will now learn the optimal policy. In order to do that, we will implement the Q-learning algorithm.

Help 1: As $\gamma(s,a)$ you can take $\gamma(s,a) = 1/(number \quad visits \quad (s,a))$
Help 2: Fifth line in slide $24/44$. The action policy to take action A can be arbitrary, the only condition is that it needs to keep exploring. For convergence, we need that $(number \quad visits \quad (s,a)) \to \infty$
Help 3: Consider two policies: Uniform at random and ϵ-greedy. With Uniform at random, in both states, we take each action with probability $1/2$. With ϵ-greedy, with probability $\epsilon = 0.9$ we take the action that according our current $Q(s,a)$ estimates is the optimal, and with probability $1-\epsilon$ takes an action randomly.

- Implement Q-learning (slide $24/44$) in order to learn the optimal control policy.
- Verify that Q-learning learns the optimal policy as expected. Verify numerically with which policy (uniform of ϵ-greedy) the learning is fastest.

Exercise 2



$r(A,1) = 10; r(A,2) = -10; r(B,1) = +40, r(B,2) = 20$

Exercise 2

- Known Model: Write the Value Iteration algorithm to find the optimal value function
- Unknown model: Write the algorithm that finds the best action-value function $q^*(s, a)$