Reinforcement Learning

CS 486/686: Introduction to Artificial Intelligence

Outline

- What is reinforcement learning
- Multiarmed Bandits

What is RL?

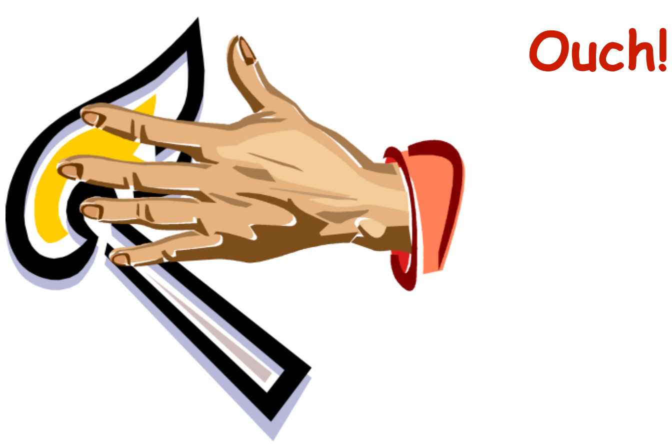
- Reinforcement learning is learning what to do so as to maximize a numerical reward signal
- Learner is not told what actions to take
- Learner discovers value of actions by
  - Trying actions out
  - Seeing what the reward is

What is RL?

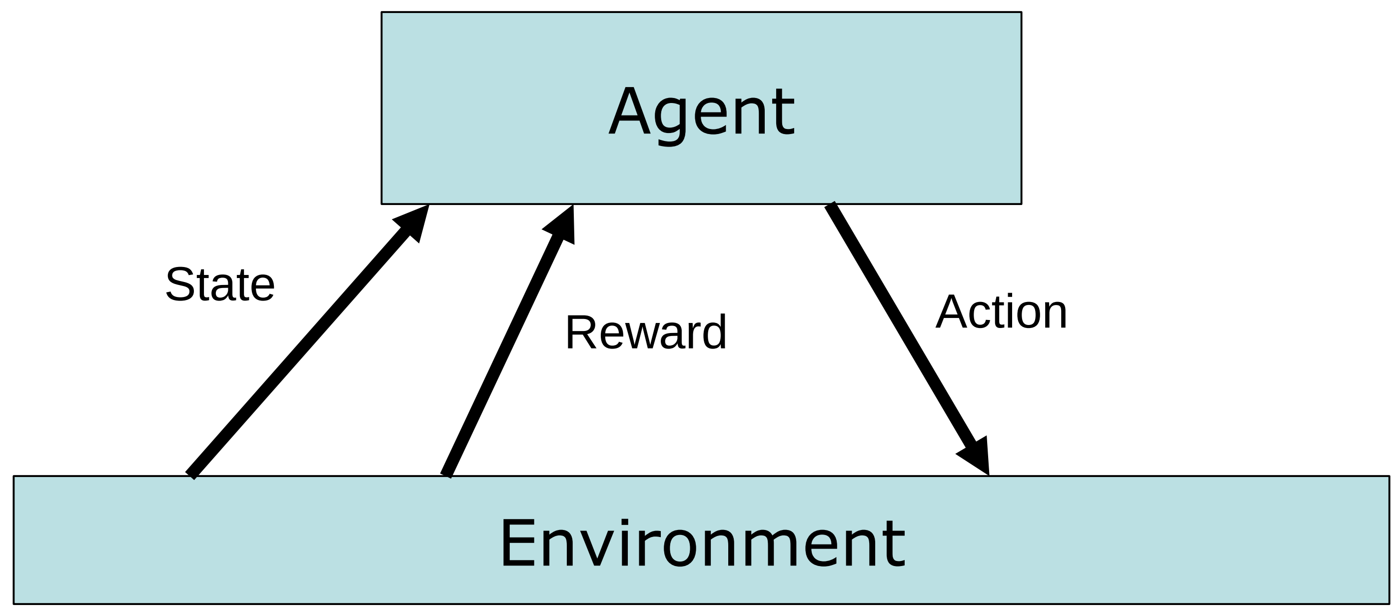
- Another common learning framework is supervised learning (we will see this later in the semester)

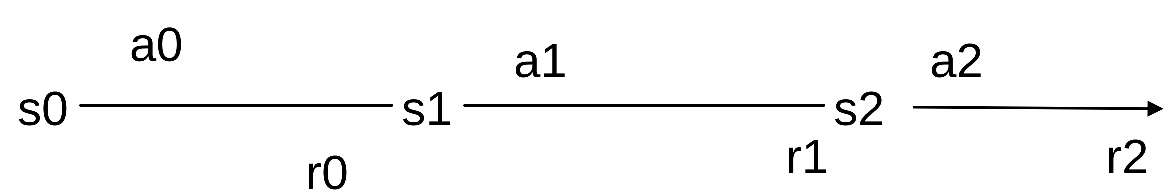
Supervised learning

Reinforcement learning



Reinforcement Learning Problem

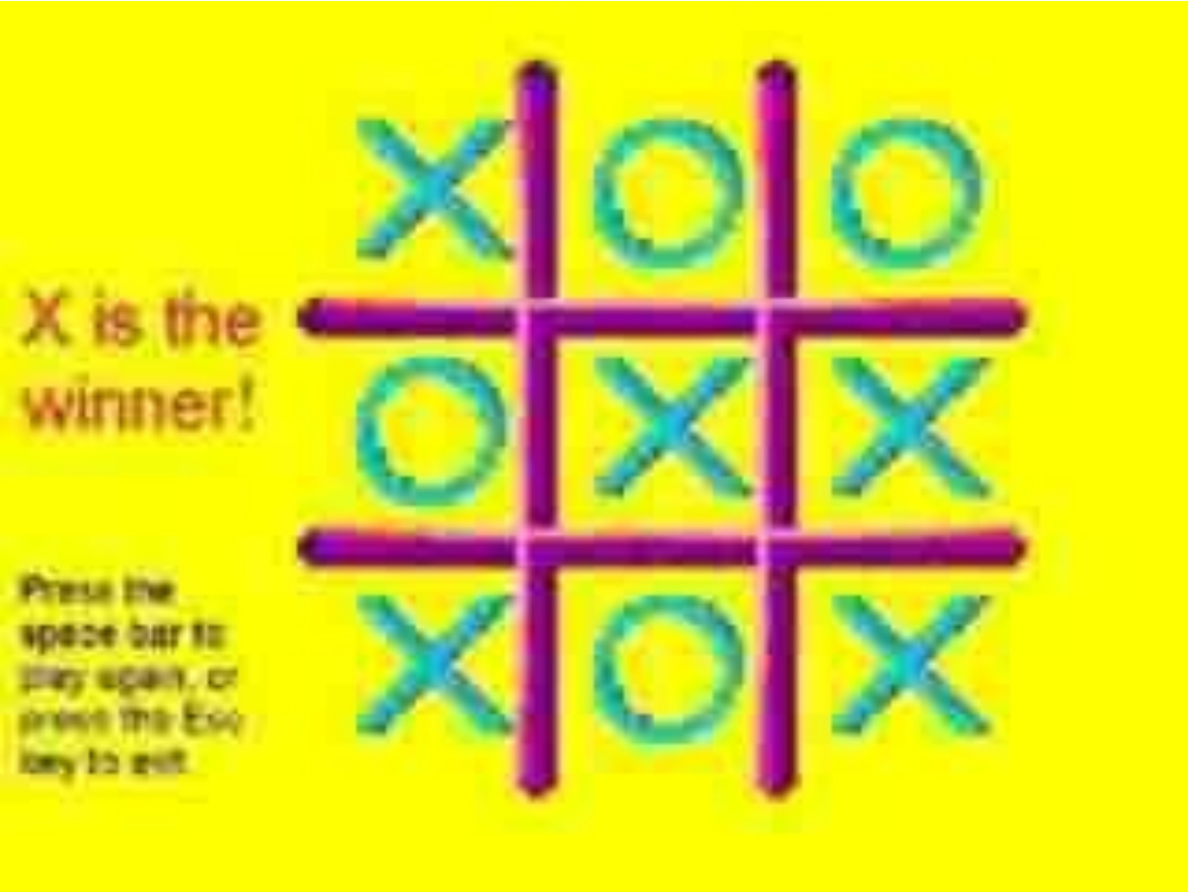




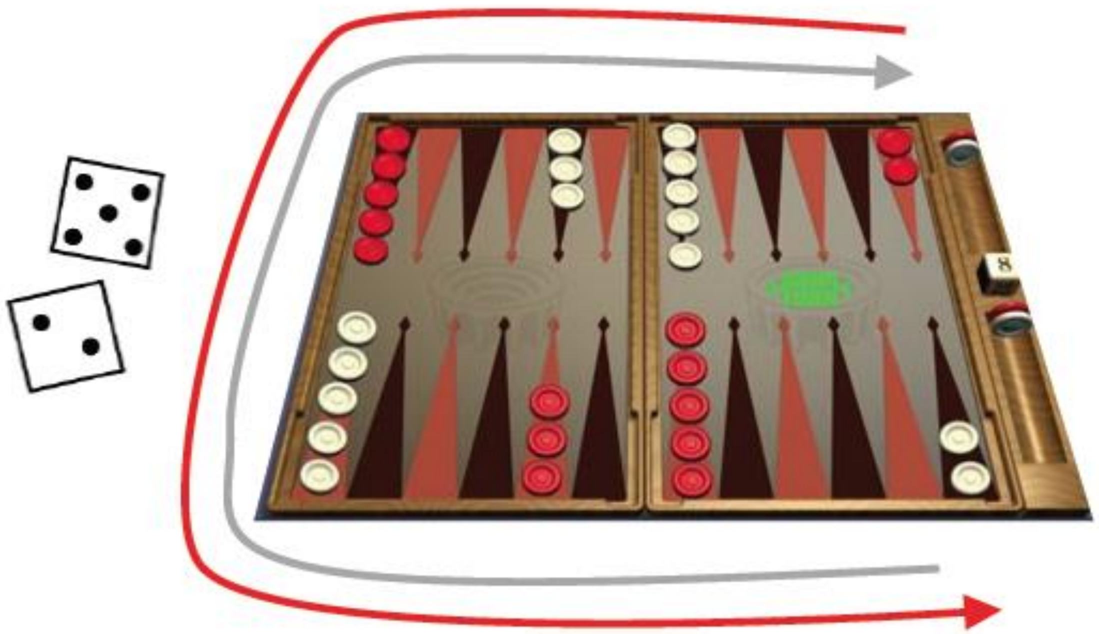
Goal: Learn to choose actions that maximize $r_0+\gamma\ r_1+\gamma^2 r_2+\ldots$, where $0\cdot<\gamma<1$

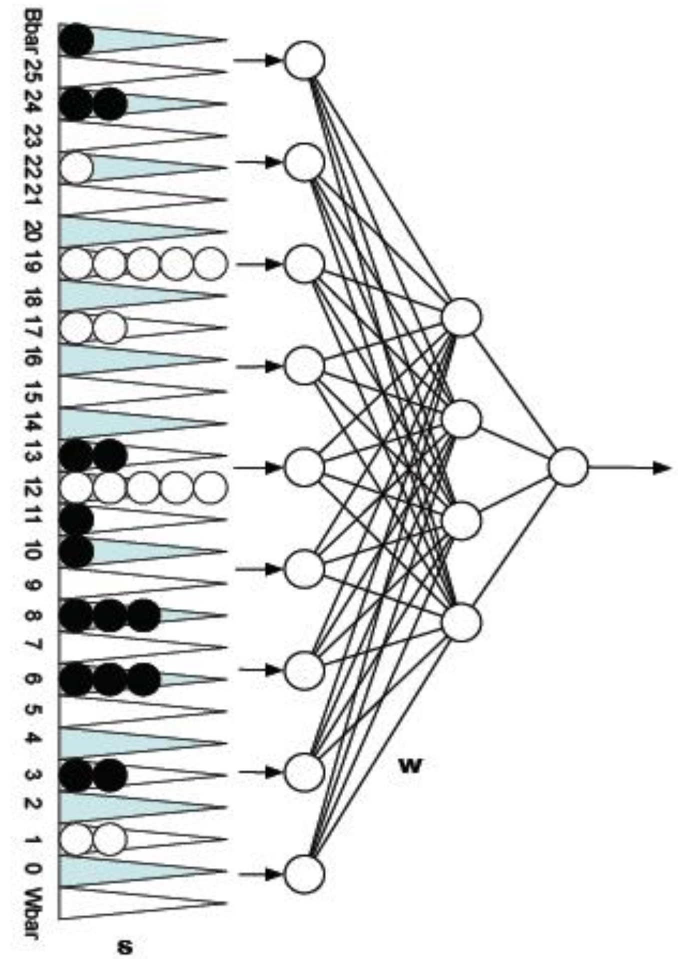
Example: Tic Tac Toe (or Backgammon or Go…)

- **State**: Board configuration
- **Actions**: Next move
- **Reward**: 1 for a win, -1 for a loss, 0 for a draw
- **Problem**: Find π: S →A that maximizes the reward



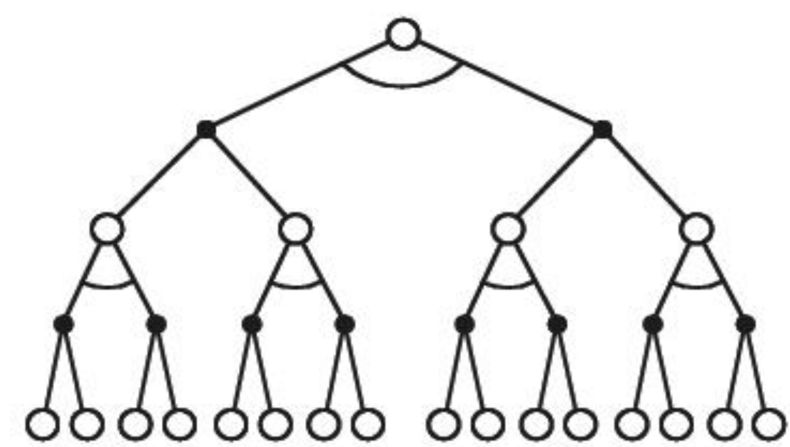
TD-Gammon





estimated state value
(≈ prob of winning)

Action selection
by a shallow search



Start with a random Network

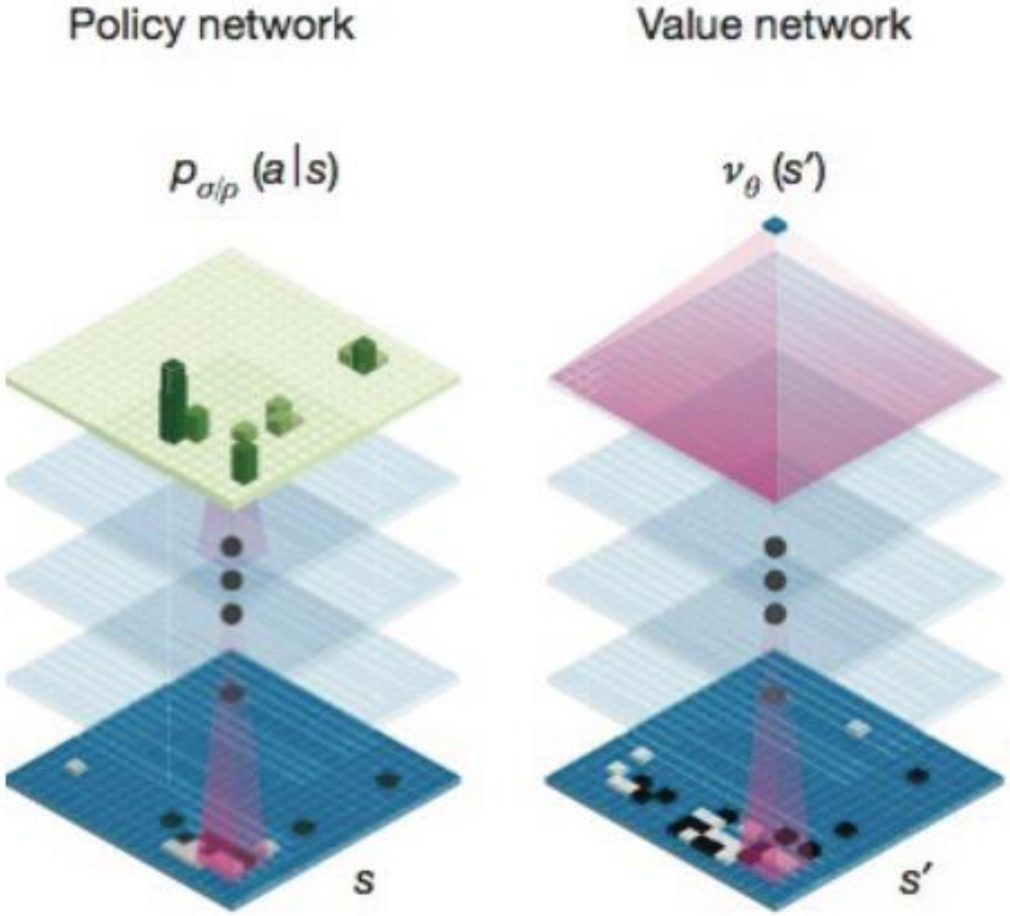
Play millions of games against itself

Learn a value function from this simulated experience

Six weeks later it's the best player of backgammon in the world

Originally used expert handcrafted features, later repeated with raw board positions

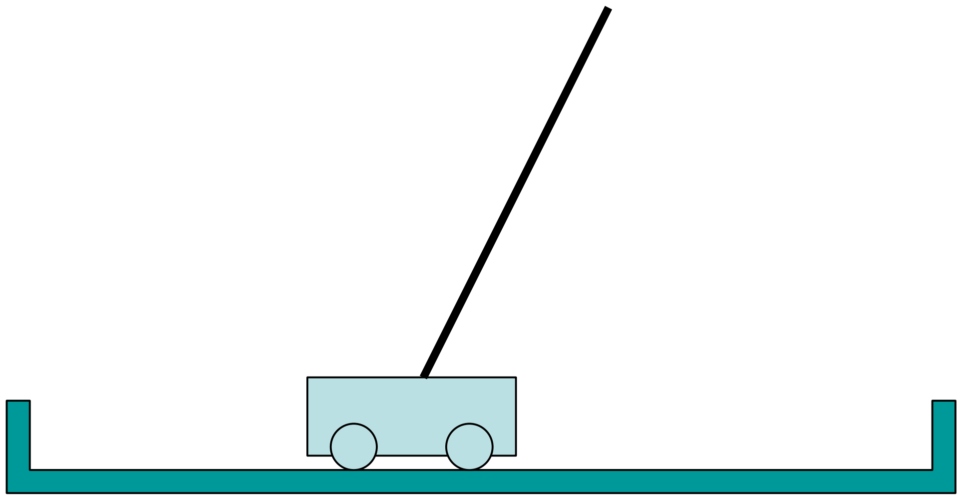
Example: AlphaGo



- Perceptions: state of the board
- Actions: legal moves
- Reward: +1 or -1 at the end of the game
- Trained by playing games against itself
- Invented new ways of playing which seem superior

Example: Inverted Pendulem

- **State**: $x(t)$, $x'(t)$, $\theta(t)$, $\theta'(t)$
- **Actions**: Force F
- **Reward**: 1 for any step where the pole is balanced
- **Problem**: Find $\pi: S \rightarrow A$ that maximizes the reward



RL + Deep Learing Performance on Atari Games

Space Invaders

Breakout

Enduro

Reinforcement Learning Characteristics

- Delayed reward
  - Credit assignment problem
- Exploration and exploitation
- Possibility that a state is only partially observable
- Life-long learning

Reinforcement Learning Model

- Set of states S
- Set of actions A
- Set of reinforcement signals (rewards)
    - Rewards may be delayed

Multi-Armed Bandits

The simplest reinforcement learning problem is the multi-armed bandit

Thanks to D. Precup

K-armed bandit

At each time step t, you choose an action A_t from k possible actions, and receive reward R_t

The reward depends only on the action taken (it is i.i.d)

$$q_*(a) \doteq \mathbb{E}[R_t | A_t = a], \quad \forall a \in \{1, \ldots, k\}$$ ***true values***

These true values are unknown, and the distribution is unknown

Goal: Maximize your total reward

To achieve this goal you need to try actions to learn their values (explore) and prefer those that appear best (exploit)

Exploration/Exploitation Tradeoff

Assume you had estimates

$$Q_t(a) \approx q_*(a), \quad \forall a$$

action-value estimates

Define the greedy action as

$$A_t^* \doteq \arg\max_a Q_t(a)$$

If $A_t = A_t^*$ then you are **exploiting**

If $A_t \neq A_t^*$ then you are **exploring**

You can't do both but you need to do both

Action-Value Methods

Methods that learn action-value estimates and nothing else

Estimate action values as sample averages:

$$Q_t(a) \doteq \frac{\text{sum of rewards when } a \text{ taken prior to } t}{\text{number of times } a \text{ taken prior to } t} = \frac{\sum_{i=1}^{t-1} R_i \cdot \mathbf{1}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbf{1}_{A_i=a}}$$

The same average estimate will converge to true values if the action is taken an infinite number of times

$$\lim_{N_t(a)\to\infty} Q_t(a) \;=\; q_*(a)$$

The number of times action a has been taken by time t

ε-Greedy Action Selection

In greedy action selection you always exploit

In ε-greedy, you are usually greedy but with probability ε you pick an action at random

This is a simple way of balancing exploration and exploitation

A simple bandit algorithm

Initialize, for $a = 1$ to k:

$Q(a) \leftarrow 0$

$N(a) \leftarrow 0$

Repeat forever:

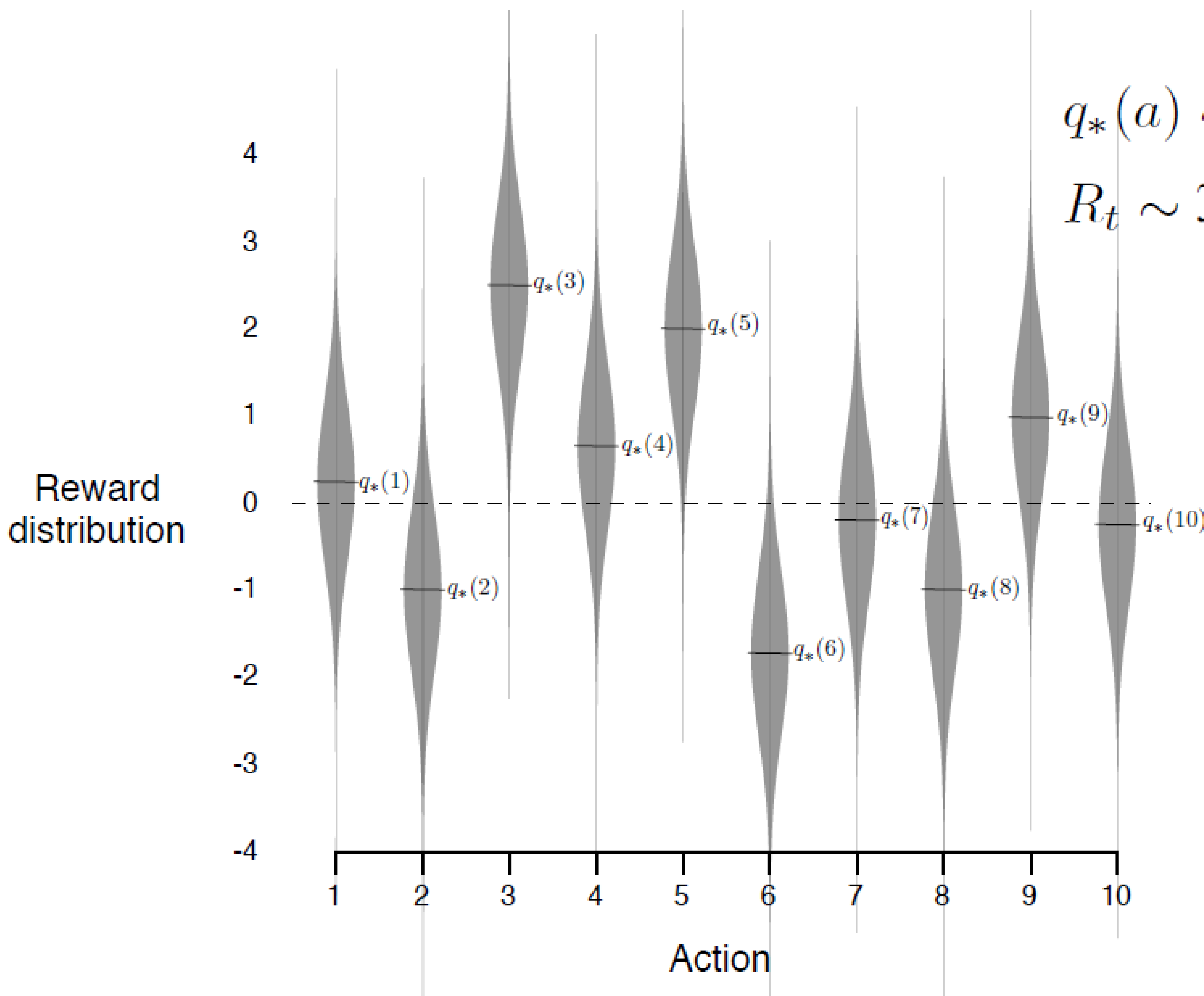
$$A \leftarrow \begin{cases} \arg\max_a Q(a) & \text{with probability } 1-\varepsilon \quad \text{(breaking ties randomly)} \\ \text{a random action} & \text{with probability } \varepsilon \end{cases}$$

$R \leftarrow bandit(A)$

$N(A) \leftarrow N(A) + 1$

$Q(A) \leftarrow Q(A) + \frac{1}{N(A)}\left[R - Q(A)\right]$

A Ten-Armed Example (Sutton and Barto)



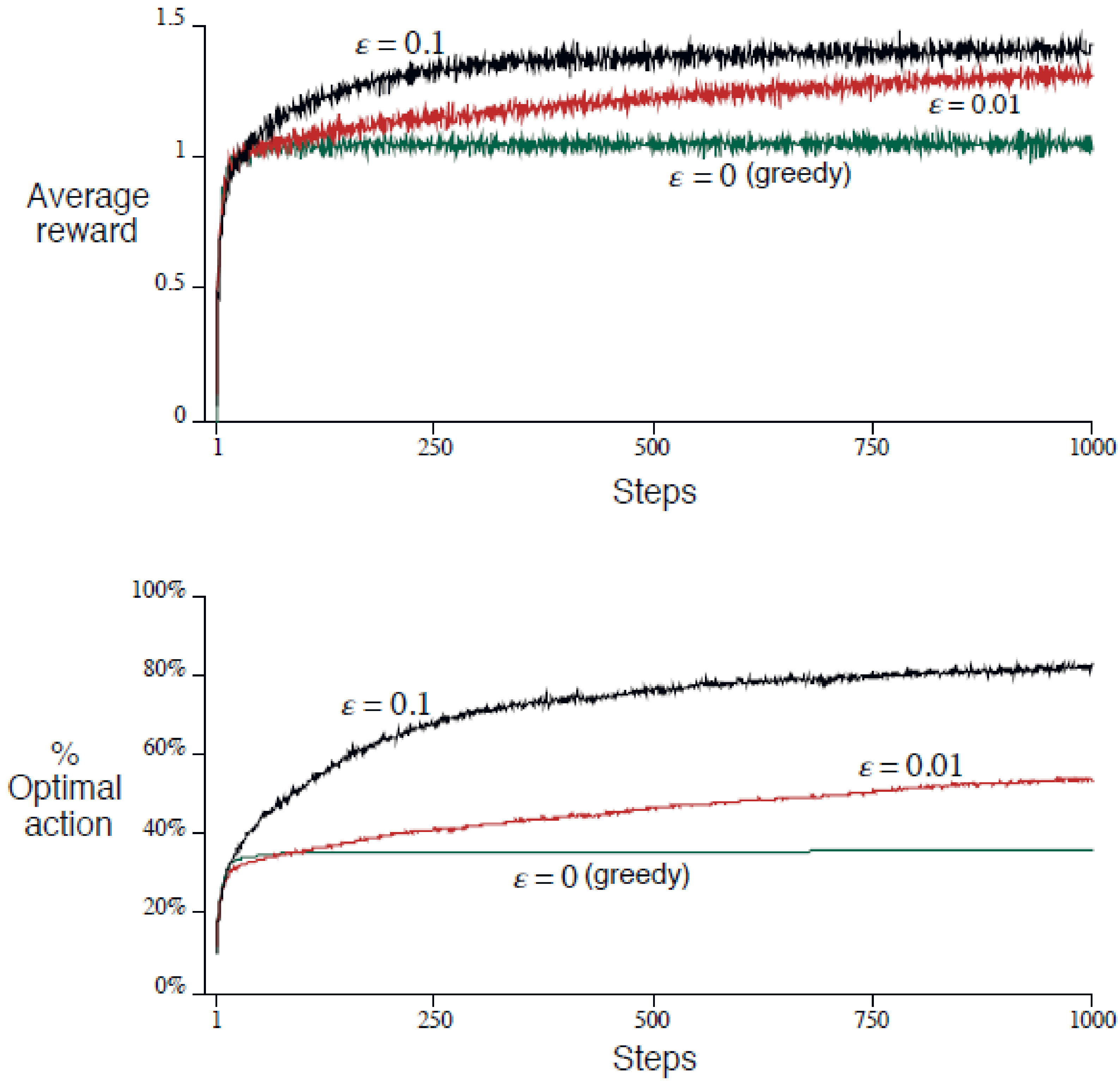
$$q_*(a) \sim \mathcal{N}(0, 1)$$

$$R_t \sim \mathcal{N}(q_*(a), 1)$$

Run for 1000 steps

Repeat the whole thing 2000 times with different bandit tasks

ε-Greedy Action Selection



A simple bandit algorithm

Initialize, for $a = 1$ to k:

$Q(a) \leftarrow 0$

$N(a) \leftarrow 0$

Repeat forever:

$$A \leftarrow \begin{cases} \arg\max_a Q(a) & \text{with probability } 1 - \varepsilon \quad \text{(breaking ties randomly)} \\ \text{a random action} & \text{with probability } \varepsilon \end{cases}$$

$R \leftarrow bandit(A)$

$N(A) \leftarrow N(A) + 1$

$Q(A) \leftarrow Q(A) + \frac{1}{N(A)}\left[R - Q(A)\right]$

Averaging and Learning Rules

Focussing on a single action so as to avoid notation issues

Note that our estimate of the value of taking some action is the average reward collected by taking that action

$$Q_n \doteq \frac{R_1 + R_2 + \cdots + R_{n-1}}{n-1}$$

Can we compute this incrementally without storing all rewards to date?

$$Q_{n+1} = Q_n + \frac{1}{n}\Big[R_n - Q_n\Big]$$

Standard form for learning/update rules

$$NewEstimate \leftarrow OldEstimate \ + \ StepSize \Big[Target - OldEstimate\Big]$$

Derivation of the Update Rule

$$Q_n \doteq \frac{R_1 + R_2 + \cdots + R_{n-1}}{n-1}$$

$$\begin{aligned}
Q_{n+1} &= \frac{1}{n}\sum_{i=1}^{n} R_i \\
&= \frac{1}{n}\left(R_n + \sum_{i=1}^{n-1} R_i\right) \\
&= \frac{1}{n}\left(R_n + (n-1)\frac{1}{n-1}\sum_{i=1}^{n-1} R_i\right) \\
&= \frac{1}{n}\Big(R_n + (n-1)Q_n\Big) \\
&= \frac{1}{n}\Big(R_n + nQ_n - Q_n\Big) \\
&= Q_n + \frac{1}{n}\Big[R_n - Q_n\Big],
\end{aligned}$$

Non-Stationary Problems

What happens when the problem is non-stationary? (i.e. the true values change over time)?

Sample average is not an appropriate technique

Instead, exponential, recency-weighted average

$$
\begin{aligned}
Q_{n+1} &\doteq Q_n + \alpha\Big[R_n - Q_n\Big] \\
&= (1-\alpha)^n Q_1 + \sum_{i=1}^{n} \alpha(1-\alpha)^{n-i} R_i,
\end{aligned}
$$

where α is a constant *step-size parameter*, $\alpha \in (0, 1]$

Convergence Conditions

- To ensure convergence with probability 1, we require

$$\sum_{n=1}^{\infty} \alpha_n(a) = \infty \qquad \text{and} \qquad \sum_{n=1}^{\infty} \alpha_n^2(a) < \infty$$

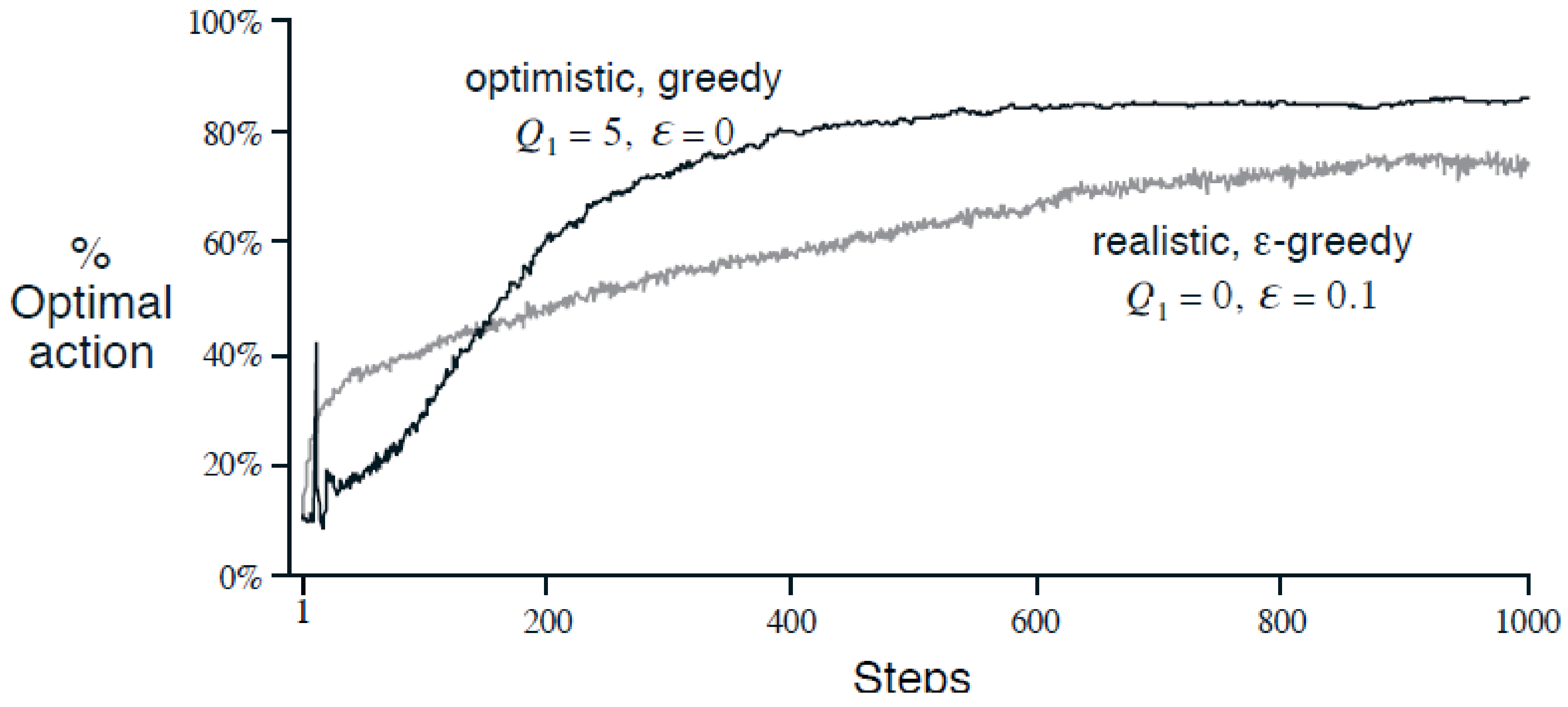
- e.g., $\alpha_n \doteq \frac{1}{n}$
- not $\alpha_n \doteq \frac{1}{n^2}$

if $\alpha_n \doteq n^{-p}, \quad p \in (0,1)$
then convergence is
at the optimal rate:
$O(1/\sqrt{n})$

Optimism and Bandits

- We need to start somewhere with Q1(a), which introduces a bias
  - So far assumed Q1(a)=0

- We could be optimistic and initialize action values differently (Q1(a)=5)



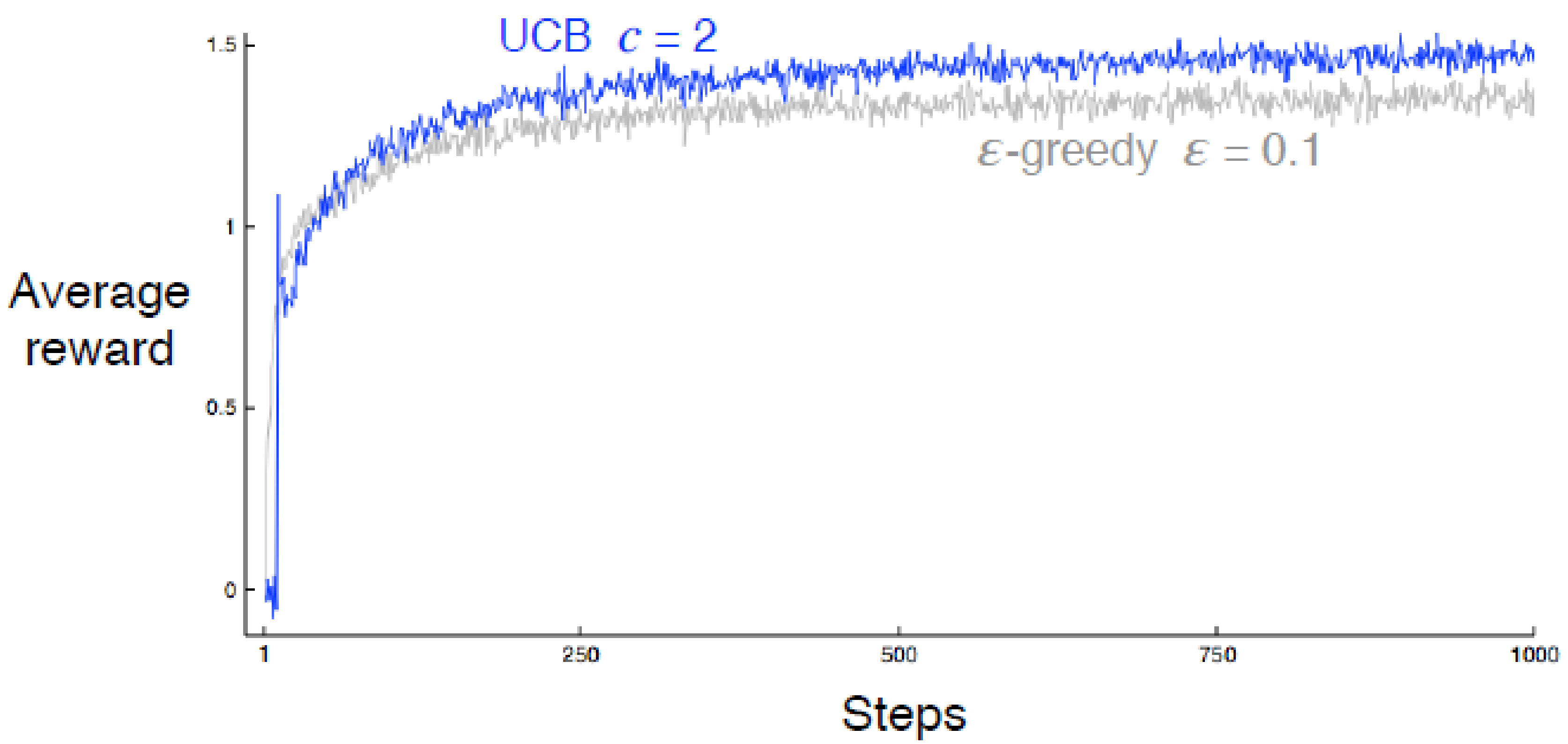
Upper Confidence Bounds

We can reduce exploration over time by using optimism

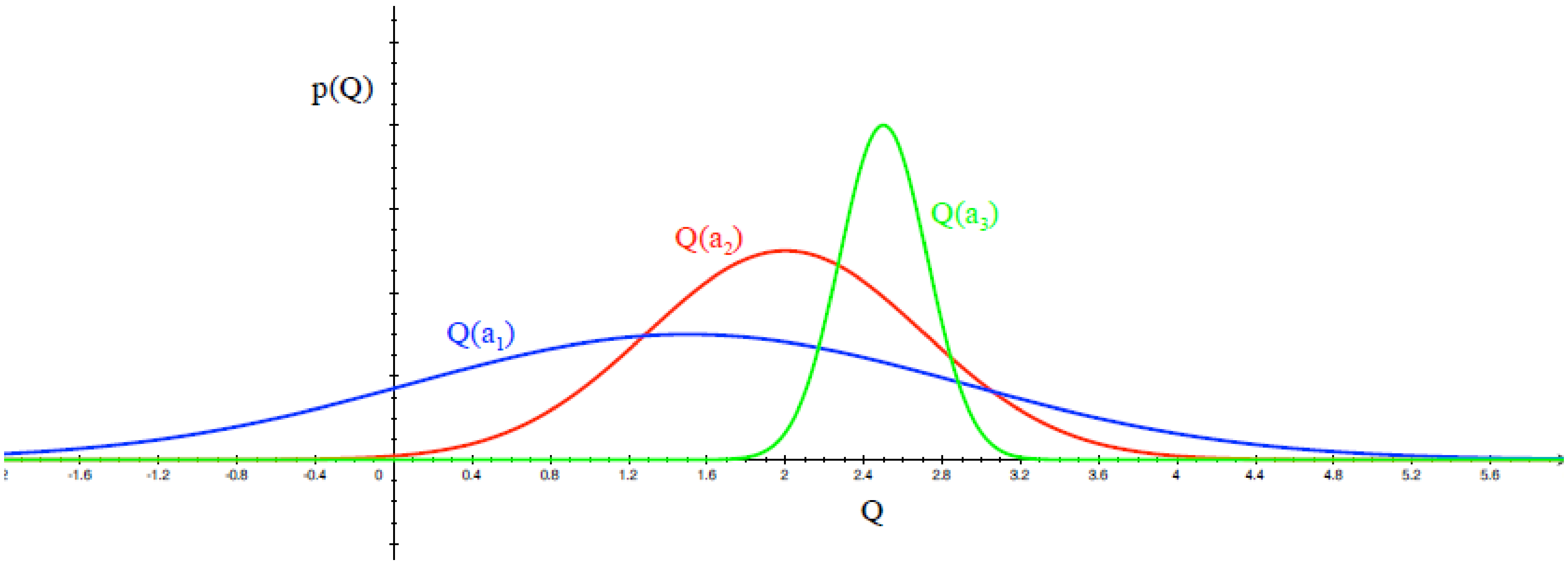
Estimate an upper bound on the true action values

Select action with largest estimated upper bound

$$A_t \doteq \underset{a}{\arg\max} \left[Q_t(a) + c\sqrt{\frac{\log t}{N_t(a)}} \right]$$



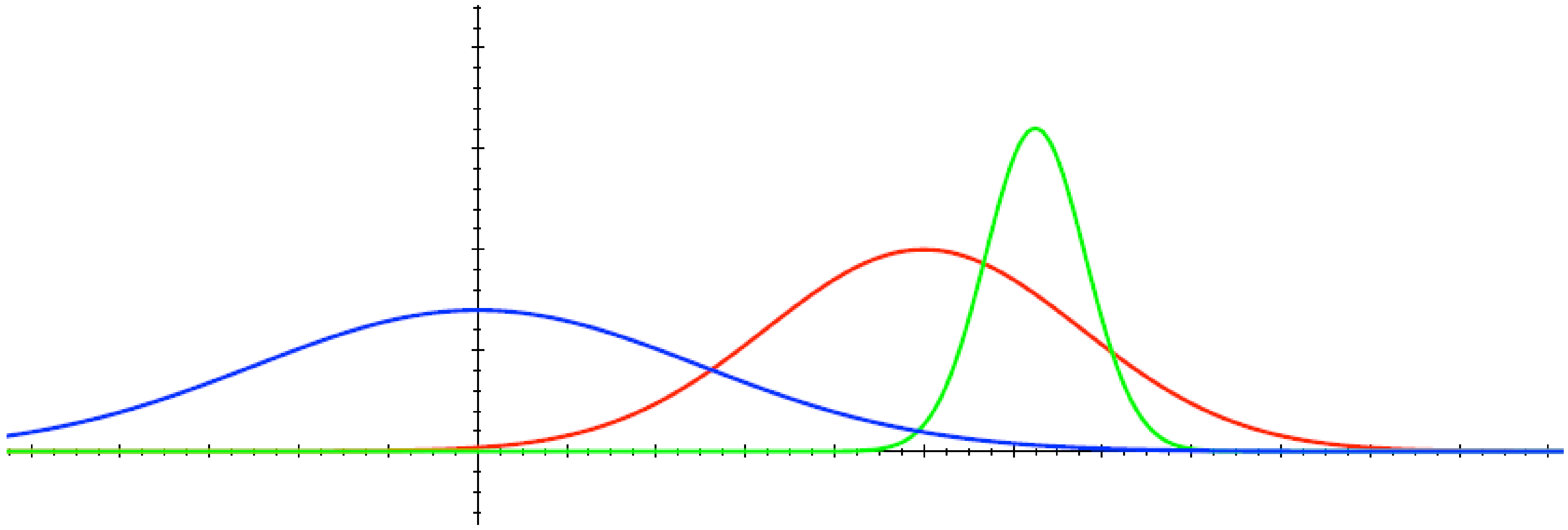
Optimism in the Face of Uncertainty



What action should be picked?
The more uncertain we are about an action-value, the more important it is to explore it since it could turn out to be the best action

D. Silver

Optimism in the Face of Uncertainty



After we have picked the blue action, we are more confident in its value. We are now more likely to pick another action.

D. Silver

Upper Confidence Bounds

- We want to estimate an upper confidence $U_t(a)$ for each action such that $q^*(a) \leq Q_t(a) + U_t(a)$

- This depends on the number of times we have sampled action a
  - Small Nt(a) -> large Ut(a) (estimated value is inaccurate)
  - Large Nt(a) -> small Ut(a) (estimated value is accurate)

- Select action maximizing Upper Confidence Bound

$$A_t = \arg\max \left[Q_t(a) + U_t(a) \right]$$

How to Determine the Bound

Theorem (Hoeffding's Inequality)

Let $X_1, ..., X_t$ be i.i.d. random variables in [0,1], and let $\overline{X}_t = \frac{1}{\tau}\sum_{\tau=1}^{t} X_\tau$ be the sample mean. Then

$$\mathbb{P}\left[\mathbb{E}\left[X\right] > \overline{X}_t + u\right] \leq e^{-2tu^2}$$

Apply bound to bandit awards

$$P[q^*(a) > Q_t(a) + U_t(a)] \leq e^{-2N_t(a)U_t(a)^2}$$

How to Calculate the UCB

Now we pick a probability p that exceeds value of the UCB and solve for Ut(a)

$$e^{-2N_t(a)U_t(a)^2} = p$$

$$U_t(a) = \sqrt{\frac{-\log p}{2N_t(a)}}$$

Reduce p as observe more rewards, e.g. $p=t^{-4}$

$$U_t(a) = \sqrt{\frac{2\log t}{N_t(a)}}$$

Summary

- Bandits are the simplest RL models and we will be building on them
- Key challenge is the balance between exploration and exploitation