EECS 545: Machine Learning

Lecture 21. Reinforcement Learning

Honglak Lee

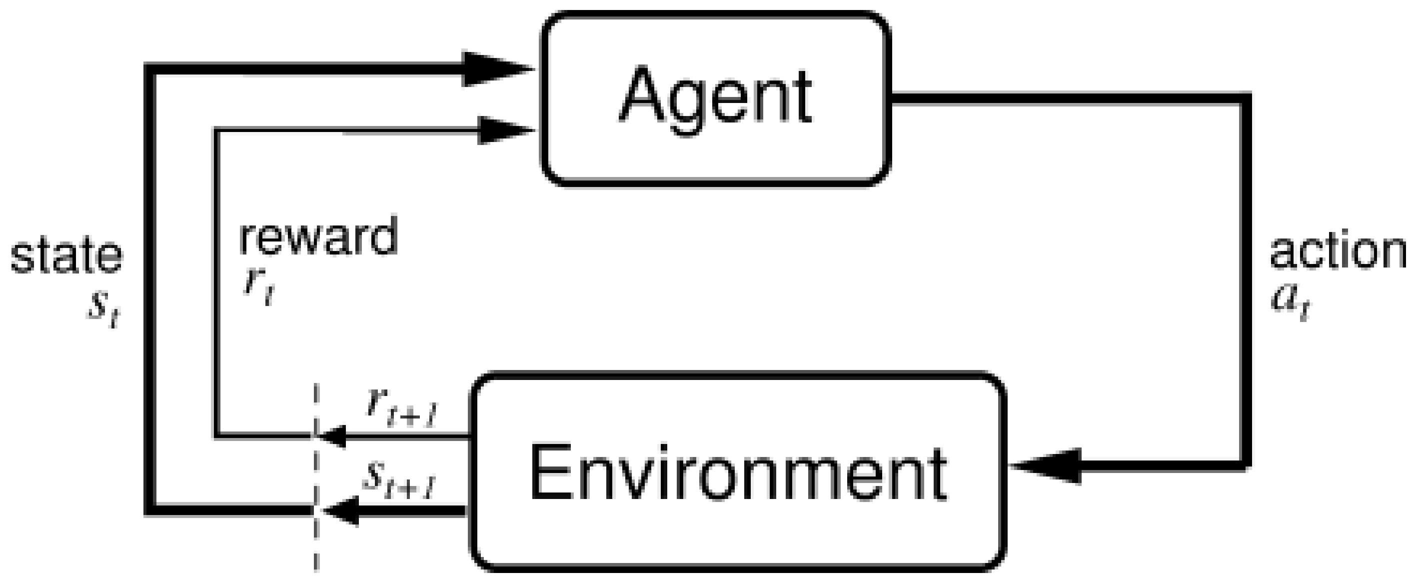
3/28/2011

Outline

- Introduction to Reinforcement Learning

Reinforcement Learning (RL)

- The *reinforcement learning problem* is how an agent in an environment can select its actions to maximize its long-term rewards.

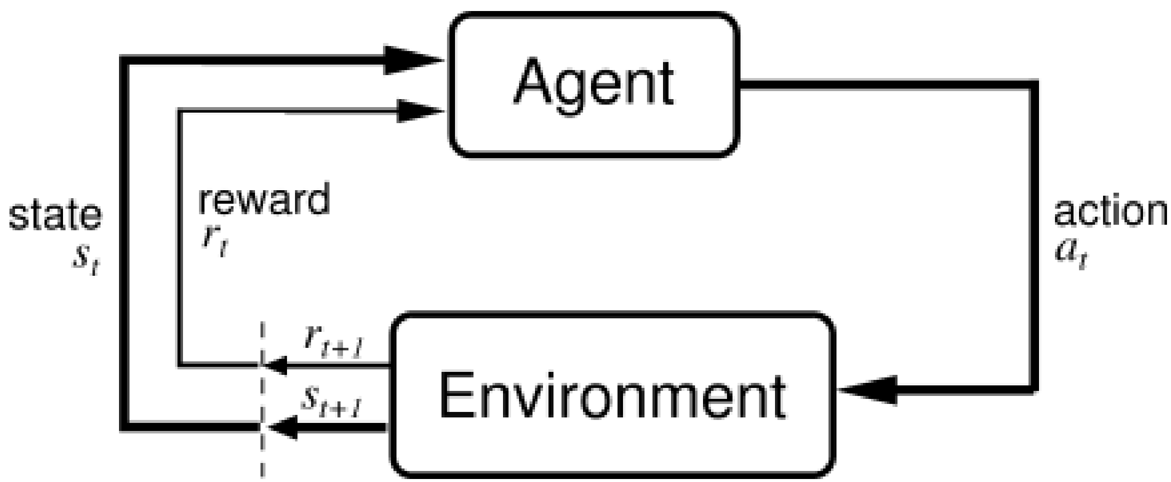


Strengths of the RL Framework

- RL deals with a *complete* (simple) *agent* behaving in an environment.
  - Supervised and unsupervised learning are parts of some larger, unspecified, structure.
- RL makes explicit the *trade-off* between
  - **Exploration:** acting to learn the environment,
  - **Exploitation:** acting to maximize reward.

Formalizing the RL Framework

- At each time t = 0, 1, 2, 3, . . .
- The agent perceives a *state* $s_t \in \mathcal{S}$
- It selects an *action* $a_t \in \mathcal{A}(s_t)$
- Then it receives a *reward* $r_{t+1} \in \Re$
- and finds itself in a new state s_{t+1}



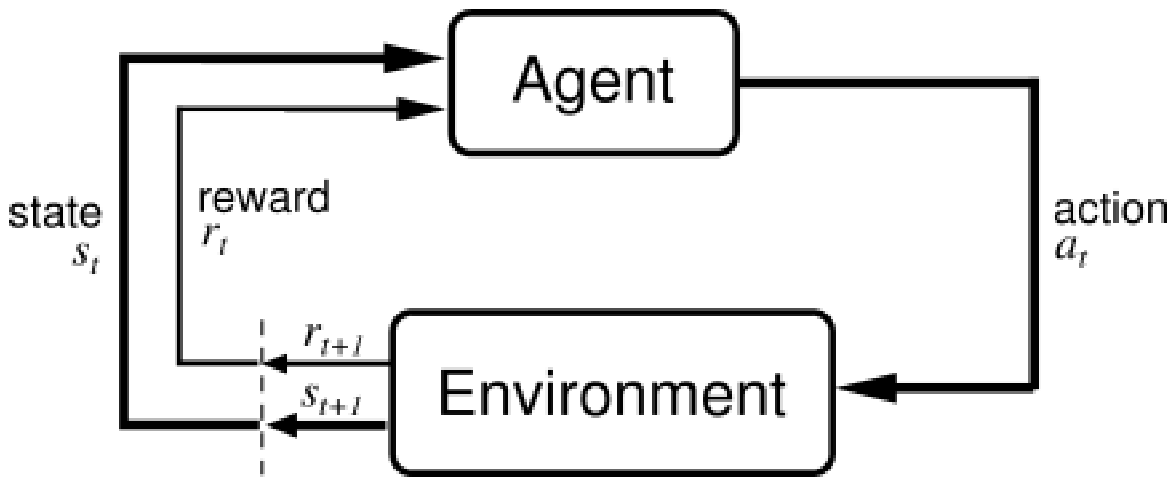
The Environment is Uncertain

- Uncertain result and reward from action.

$$Pr\{s_{t+1} = s', r_{t+1} = r' | s_t, a_t, r_t, s_{t-1}, a_{t-1}, \ldots, r_1, s_0, a_0\}$$

- We usually assume the *Markov property*.

$$Pr\{s_{t+1} = s', r_{t+1} = r' | s_t, a_t\}$$



Transitions & Expected Rewards

- State transition probabilities:

$$\mathcal{P}_{ss'}^{a} = Pr\{s_{t+1} = s' | s_t = s, a_t = a\}$$

- Expected rewards:

$$\mathcal{R}_{ss'}^{a} = E\{r_{t+1} = r' | s_t = s, a_t = a, s_{t+1} = s'\}$$

Expected Future Rewards

- We could just add up future rewards:

$$R_t = r_{t+1} + r_{t+2} + r_{t+3} + \cdots r_T$$

- Typically we *discount* future rewards:

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

- This permits an infinite time-horizon.
  - Near-term rewards are more important than the long-term future.

Consider a Simplified Problem

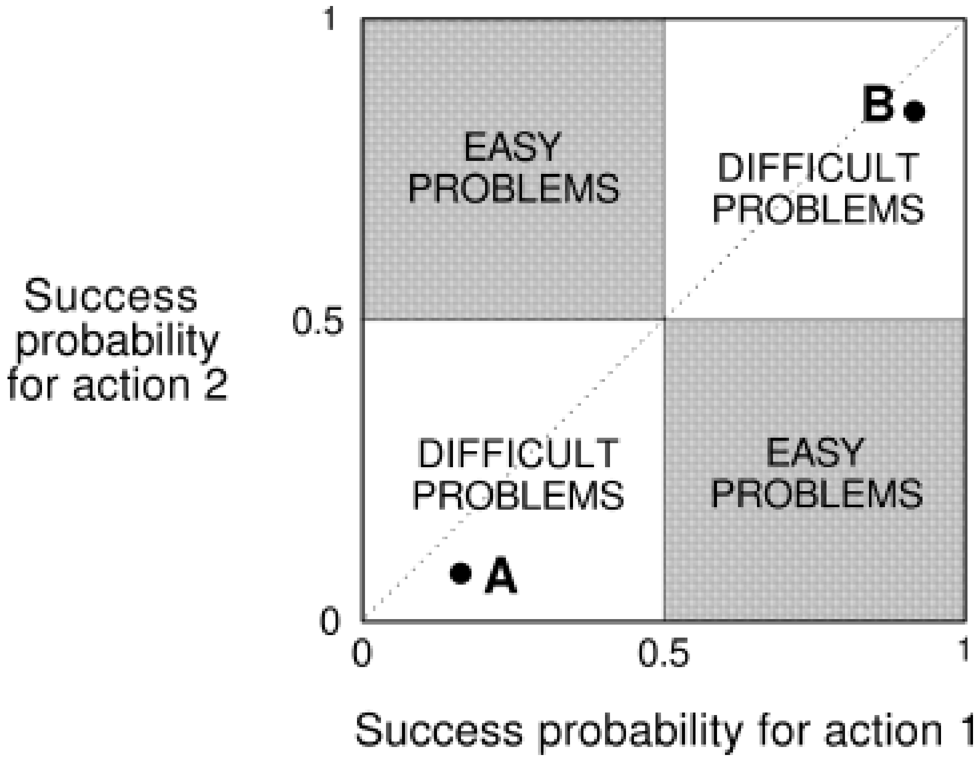
- One state. No state transitions.
  - In the full RL problem, a more complex version of this problem occurs at each state.
- Choice of actions.
  - Uncertain rewards.
- Unknown *distributions* of rewards per action.
  - Exploration versus Exploitation.

The K-Armed Bandit Problem

- Each arm has a different, *unknown*, distribution.
- How do you learn the distribution to maximize payoff?

K-Armed Bandit Problems

- Some versions are easier than others.

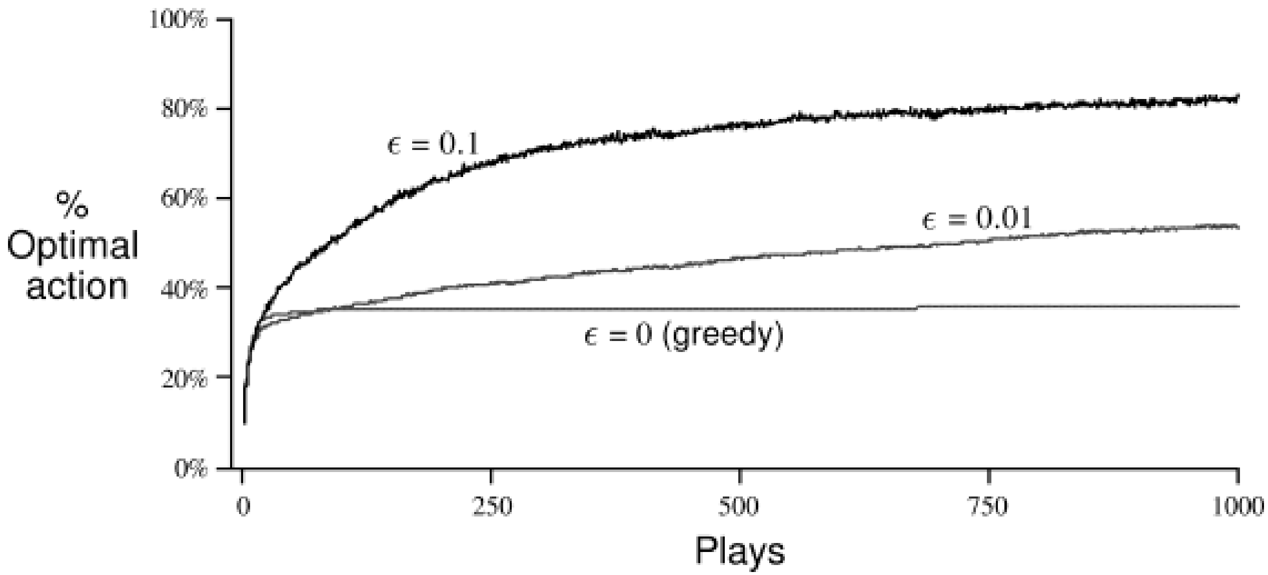


K-Armed Bandit Problems

- Let the true value of action a be $Q^*(a)$.
  - Estimate $$Q_t(a) = \frac{r_1 + r_2 + \cdots + r_{k_a}}{k_a}$$
- Exploitation: the *greedy* strategy always selects action a with the highest $Q_t(a)$.
  - With incomplete knowledge, this may ignore a much better selection.
- Exploration: select action a to improve the estimate $Q_t(a)$. (Randomly?)
  - How much exploration still pays off?

Epsilon-Greedy Methods

- With $p = 1 - \epsilon$
  - Select the greedy action (exploit).
- With $p = \epsilon$
  - Select uniformly across all actions (explore).



Softmax Action Probabilities

- Determine probability of selecting action *a* using *softmax* normalization.

$$\pi_t(a) = \frac{e^{Q_t(a)/\tau}}{\sum_{b=1}^{n} e^{Q_t(b)/\tau}}$$

- High temperature reduces effects of differences (uniform in the limit).
- At low temperaturs, softmax approaches hard max.

The 10-armed bandit testbed

- Create 2000 different 10-armed bandit tasks.
- For each task, select the optimal reward distributions $Q^*(a)$ from $N(0,1)$.
- For each task, do 1000 plays (actions).
- For each action a, select the reward from $N(Q^*(a), 1)$.
- Plot averages over the 2000 tasks.

What should the estimate be?

- Compute estimate $Q_t(a)$ as the mean reward when action *a* was performed.

$$Q_t(a) = \frac{r_1 + r_2 + \cdots + r_{k_a}}{k_a}$$

- This can be computed incrementally.

$$Q_{k+1} = \frac{1}{k+1} \sum_{i=1}^{k+1} r_i = Q_k + \frac{1}{k+1}[r_{k+1} - Q_k]$$

- More generally, the predictor-corrector form:

$$Q_{k+1} = Q_k + \alpha[r_{k+1} - Q_k] \text{ where } 0 < \alpha \leq 1$$

Recency-weighted averaging

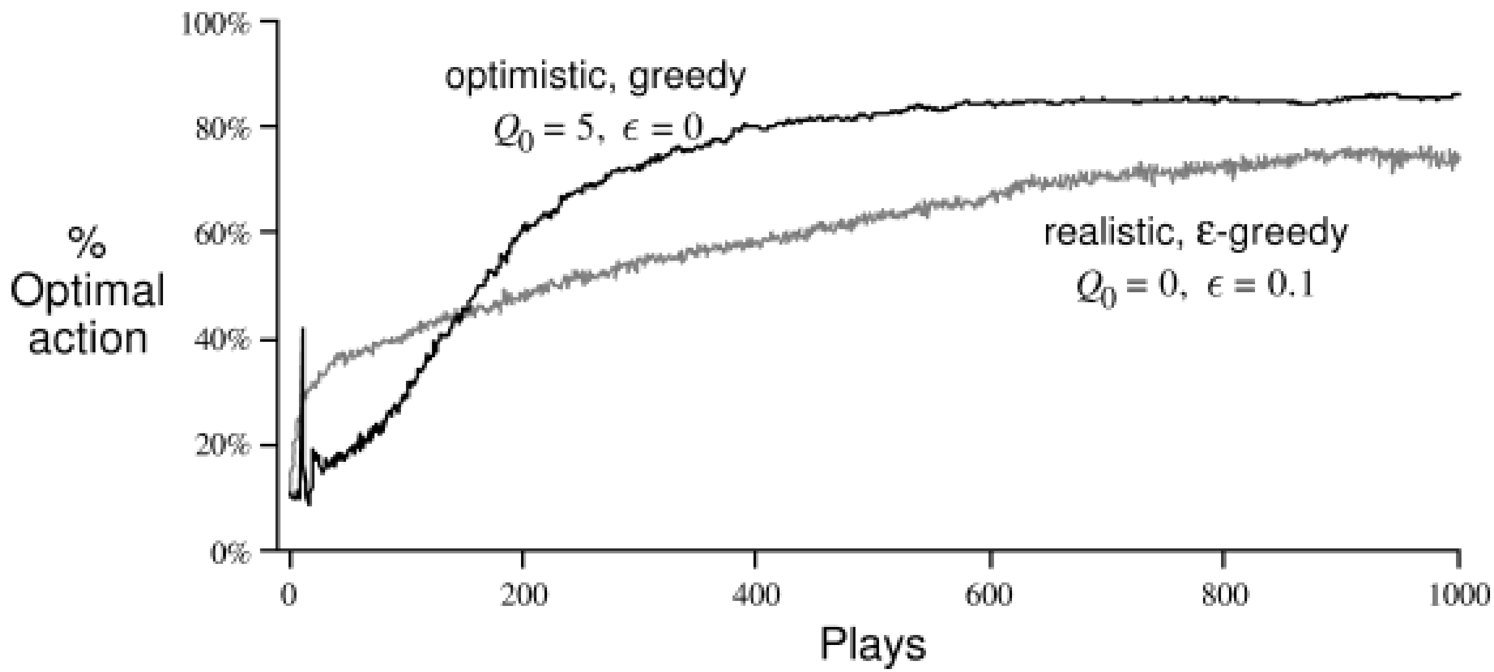
- Suppose we want new rewards to have the most impact, not the oldest rewards.
- With *constant* weight, the recurrence

$$Q_{k+1} = Q_k + \alpha[r_{k+1} - Q_k] \text{ where } 0 < \alpha \leq 1$$

- means that past rewards have exponentially decreasing impact on the estimate $Q_t(a)$.
- In this case, the discount rate is $1 - \alpha$

Encouraging Exploration

- Optimistic initialization: give every action *a* an initial high default value, e.g., $Q_0(a) = +5$.
- Now, *greedy* action selection will explore!



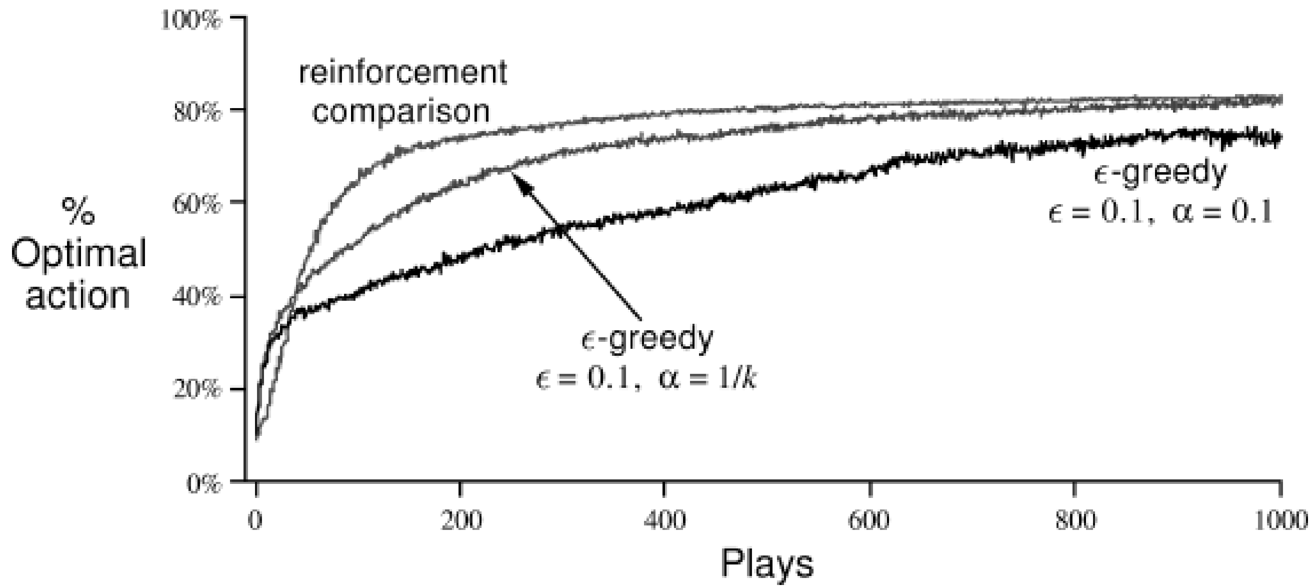
Encouraging Exploration

- Gather evidence about a *reference reward.*

$$\bar{r}_{t+1} = \bar{r}_t + \alpha[r_t - \bar{r}_t]$$

- Prefer actions with above-reference rewards

$$p_{t+1}(a_t) = p_t(a_t) + \beta[r_t - \bar{r}_t]$$

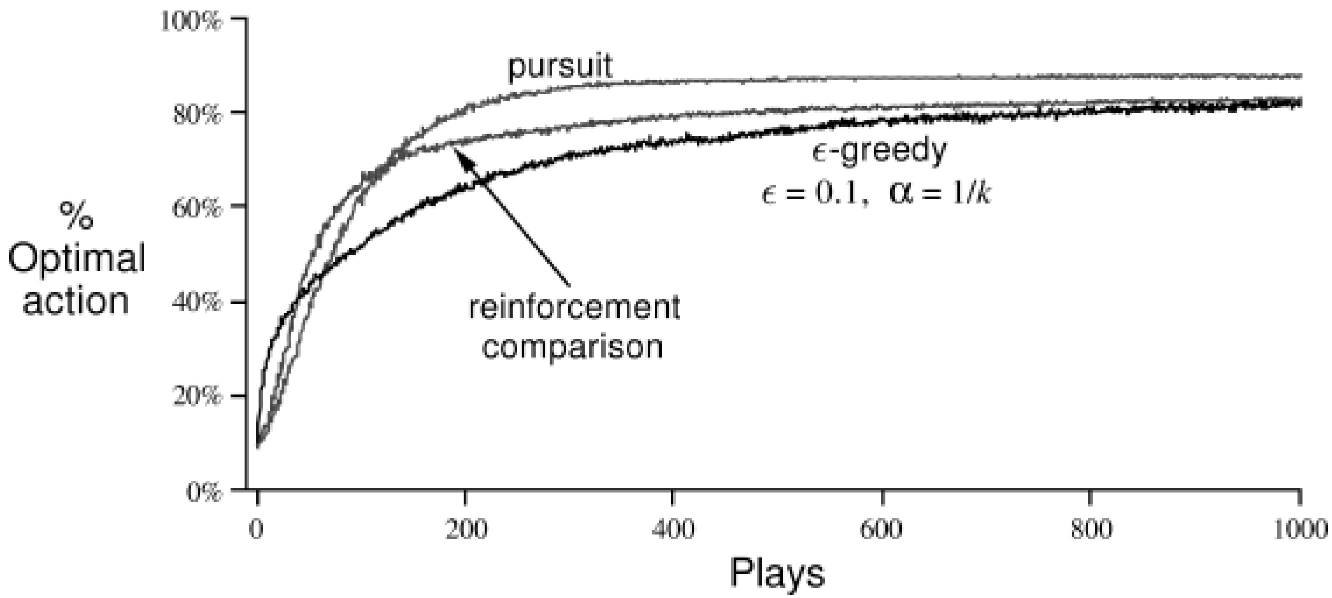


Encouraging Exploration

- Pursuit methods: at each step, move the probability of the greedy action closer to 1.

$$\pi_{t+1}(a^*_{t+1}) = \pi_t(a^*_{t+1}) + \beta[1 - \pi_t(a^*_{t+1})]$$

  – Others closer to zero.



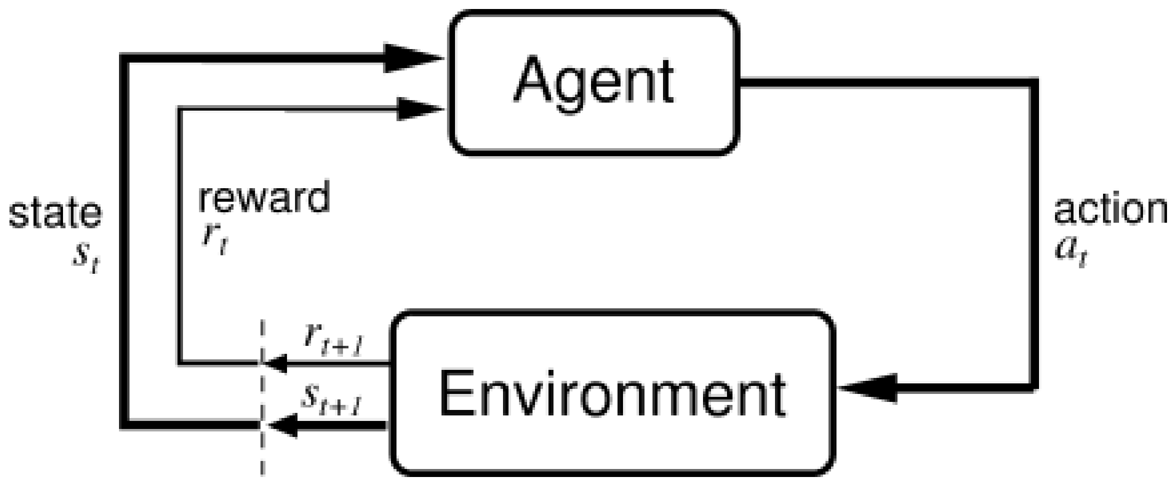
Bandit Problems and RL

- Each state with *k* actions is a *k*-armed bandit problem, to be optimized according to the performance of its own actions.
- In actual RL, the states change, too.

- Performance of an RL algorithm is quite sensitive to choice of parameter values.

Reviewing the RL Framework

- At each time t = 0, 1, 2, 3, . . .
- The agent perceives a *state* $s_t \in \mathcal{S}$
- It selects an *action* $a_t \in \mathcal{A}(s_t)$
- Then it receives a *reward* $r_{t+1} \in \Re$
- and finds itself in a new state s_{t+1}



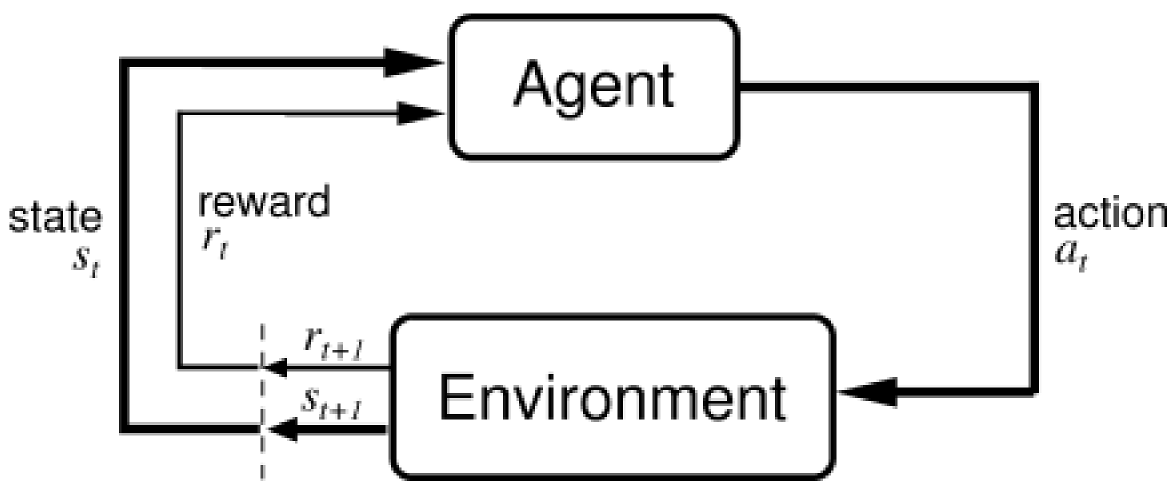
The Environment is Uncertain

- Uncertain result and reward from action.

$$Pr\{s_{t+1} = s', r_{t+1} = r' | s_t, a_t, r_t, s_{t-1}, a_{t-1}, \ldots, r_1, s_0, a_0\}$$

- We usually assume the *Markov property*.

$$Pr\{s_{t+1} = s', r_{t+1} = r' | s_t, a_t\}$$



Transitions & Expected Rewards

- State transition probabilities:

$$\mathcal{P}^{a}_{ss'} = Pr\{s_{t+1} = s' | s_t = s, a_t = a\}$$

- Expected rewards:

$$\mathcal{R}^{a}_{ss'} = E\{r_{t+1} = r' | s_t = s, a_t = a, s_{t+1} = s'\}$$

- This is what we need to specify a Markov Decision Process (MDP).

Expected Future Rewards

- We could just add up future rewards:

$$R_t = r_{t+1} + r_{t+2} + r_{t+3} + \cdots r_T$$

- Typically we *discount* future rewards:

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

- This permits an infinite time-horizon.
  - Near-term rewards are more important than the long-term future.

State-Value Functions

- A policy $\pi(s, a)$ specifies the probability of selecting action *a* when in state *s*.
- The *value* of a state is the expected future return, starting in *s* and following the policy.

$$V^\pi(s) = E_\pi\{R_t | s_t = s\}$$

$$= E\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s\right\}$$

Action-Value Functions

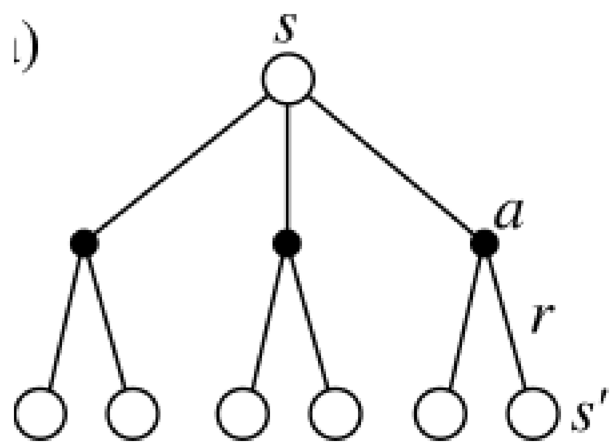
- We describe the value of taking an action a, starting in state s, and following the policy thereafter.

$$
\begin{aligned}
Q^\pi(s,a) &= E_\pi\{R_t | s_t = s, a_t = a\} \\
&= E\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s, a_t = a\right\}
\end{aligned}
$$

The Bellman Equation for *V*

- Expresses the value function at a state as a relationship with its immediate successors.

$$V^{\pi}(s) = \sum_{a} \pi(s,a) \sum_{s'} \mathcal{P}^{a}_{ss'} [\mathcal{R}^{a}_{ss'} + \gamma V^{\pi}(s')]$$



Optimal Value Functions

- There are optimal value functions.
  - Optimal state-value functions:

$$V^*(s) = \max_{\pi} V^{\pi}(s)$$

  - Optimal action-value functions:

$$Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a)$$

- We will use these to find optimal policies.

Next

- The RL problem and the MDP solution approach
- Finding optimal policies: DP and MC
- Finding optimal policies: temporal differences
- Generalization and function approximation