Reinforcement Learning

Multi-armed Bandits

U Kang
Seoul National University

In This Lecture

- K-armed Bandit problem
- Action-value methods
- Gradient bandits
- Ways of balancing exploration and exploitation

Outline

- ➡ □ **K-armed Bandit Problem**
- □ Action-value Methods
- □ Incremental Implementation
- □ Tracking a Nonstationary Problem
- □ Optimistic Initial Values
- □ UCB Action Selection
- □ Gradient Bandit
- □ Contextual Bandits
- □ Conclusion

K-armed Bandit

- Repeatedly choose among k different options, or actions
- After each choice you receive a numerical reward chosen from a stationary probability distribution
- Goal: maximize the expected total reward over some time period

en.wikipedia.org/wiki/Multi-armed_bandit#/media/File:Las_Vegas_slot_machines.jpg

K-armed Bandit

- Example 1: Casino
  - Play of one of the slot machine’s levers
  - Maximize your winnings by concentrating your actions on the best levers

- Example 2: patient treatment
  - A doctor needs to choose between experimental treatments for a series of seriously ill patients
  - Each action is the selection of a treatment, and each reward is the survival or well-being of the patient

K-armed Bandit

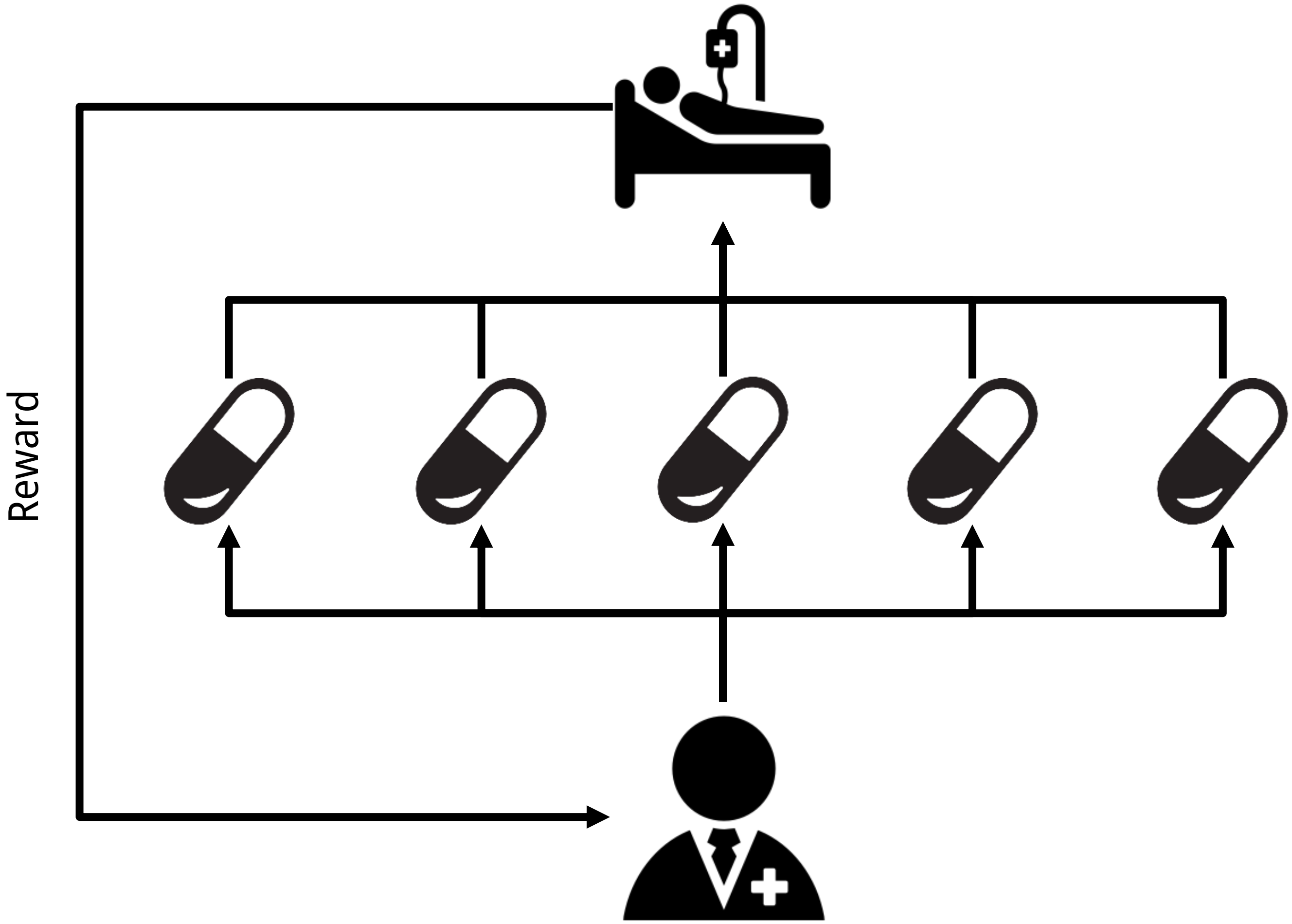
- Value of an action a: expected reward given that the action a is selected
  - $q_*(a) = E[R_t | A_t = a]$

- If we know the value of each action, then we can always select the action with the highest value
- Since we do not know the exact value, we estimate it
  - $Q_t(a)$: our estimated value of action a at time t
  - We want $Q_t(a)$ to be close to $q_*(a)$

Exploitation vs. Exploration

- Greedy action: choose the action with the greatest estimated value
- Exploitation: select greedy action
- Exploration: select nongreedy actions

- Exploitation vs Exploration
  - Exploitation gives the maximum reward in one step. However, exploration may produce the greater total reward in the long run


Reward

Exploitation vs. Exploration

- Whether it is better to explore or exploit depends in a complex way on the precise values of the estimates, uncertainties, and the number of remaining steps

- Balancing exploitation and exploration is a key challenge in RL

Outline

- ☑ K-armed Bandit Problem
- ➡ □ **Action-value Methods**
- □ Incremental Implementation
- □ Tracking a Nonstationary Problem
- □ Optimistic Initial Values
- □ UCB Action Selection
- □ Gradient Bandit
- □ Contextual Bandits
- □ Conclusion

Action-value Methods

- Action-value methods: estimate the values of actions, and use them to select actions

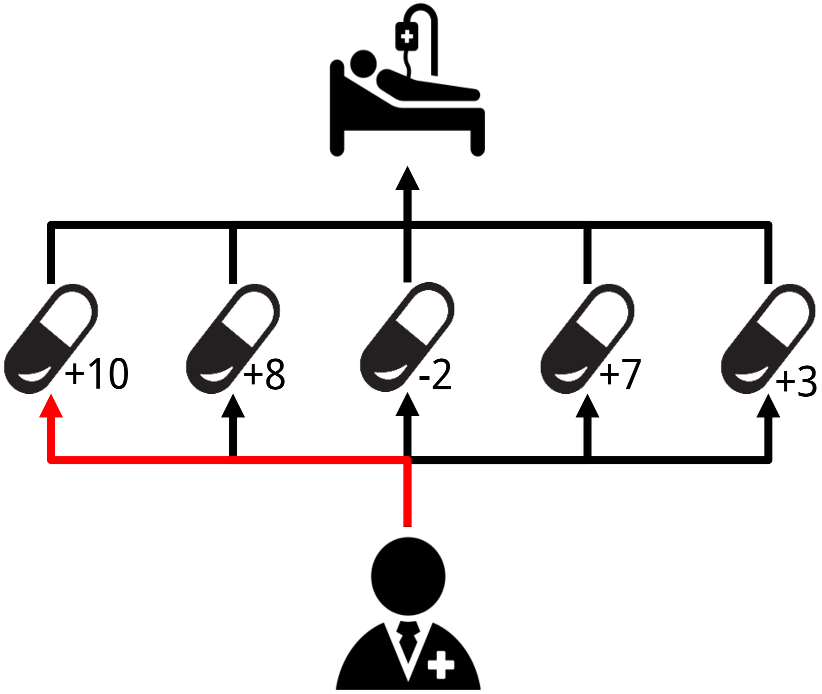
- Sample-average method: estimation by averaging
  - $Q_t(a) = \frac{\sum_{i=1}^{t-1} R_i \mathbf{1}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbf{1}_{A_i=a}}$

- Greedy action selection
  - $A_t = argmax_a Q_t(a)$

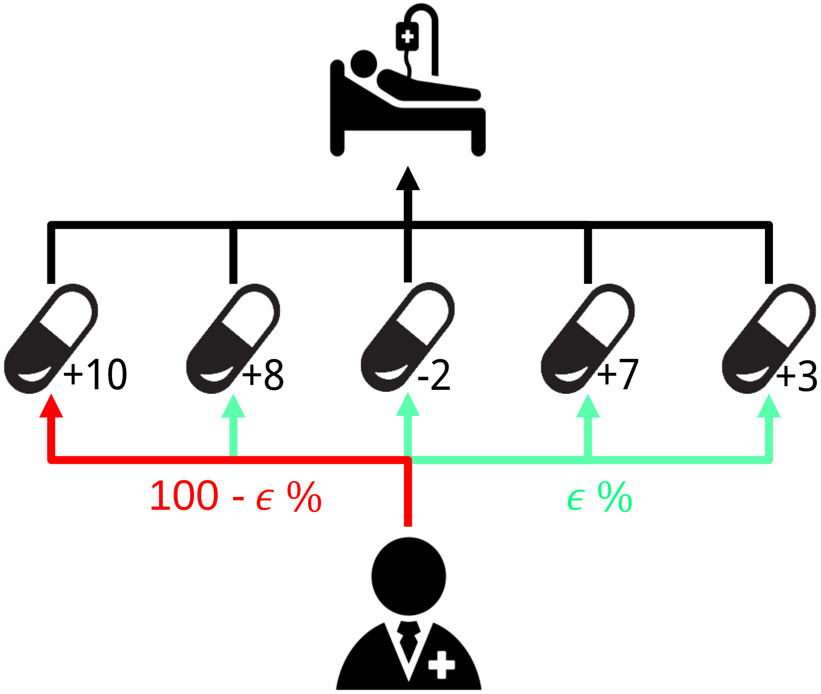
Action-value Methods

- Greedy method: always exploits current knowledge to maximize immediate reward; it spends no time for exploration

- ϵ-greedy method: behave greedily most of the time, but select a random action with prob. ϵ
  - Ensures that $Q_t(a)$ converges to $q_*(a)$ in the limit, since every action will be sampled many times
  - This also implies that the probability of selecting the optimal action converges to greater than $1 - \epsilon$, or to near certainty



Greedy

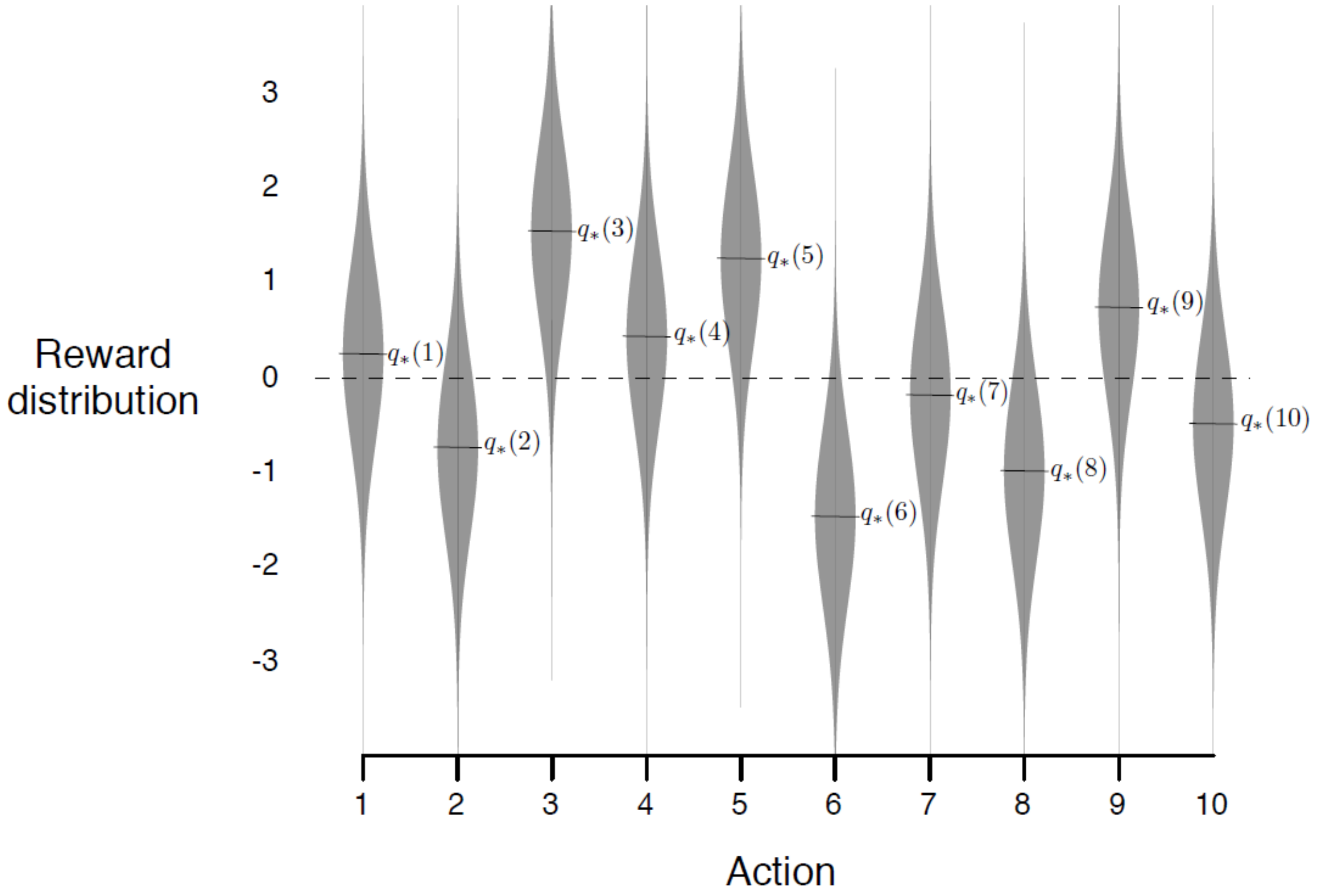


ϵ -greedy

10-armed Testbed

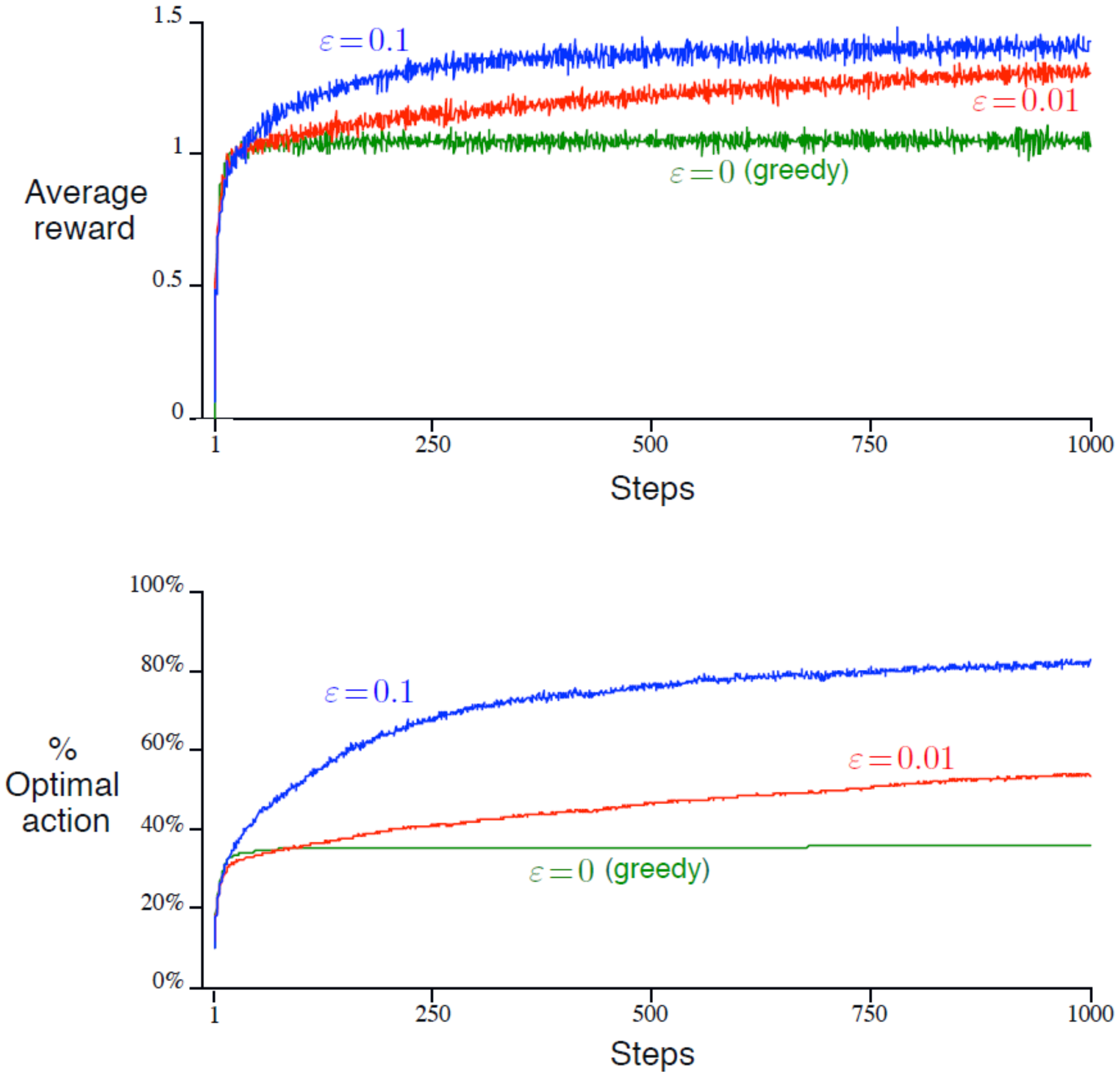
- Goal: compare greedy and ϵ -greedy methods
- 2000 randomly generated 10-armed bandit problems
- The action values $q_*(a)$, a=1, …, 10, were selected according to a Gaussian $N(0;\ 1)$
- The actual reward R_t from action A_t at time t was selected from a Gaussian $N(q_*(A_t); 1)$
- 1 run = performance over 1000 time steps

10-armed Testbed



Sutton and Barto,
Reinforcement
Learning, 2018

10-armed Testbed



Sutton and Barto,
Reinforcement
Learning, 2018

ϵ–greedy vs. greedy

- Depends on the task
- Suppose the reward variance is 10 (not 1)
  - ϵ-greedy method would outperform greedy, since it takes more exploration to find the optimal action
- Suppose the reward variance is 0 (not 1)
  - Greedy method would outperform ϵ–greedy, since the greedy method would know the true value of each action after trying it once

ϵ–greedy vs. greedy

- Non-stationary task
  - The true values of the actions changed over time
  - ϵ-greedy is advantageous to consider the changed true values
  - Nonstationarity is the case most commonly encountered in RL

- RL requires a balance between exploration and exploitation

Outline

☑ K-armed Bandit Problem

☑ Action-value Methods

➡ □ **Incremental Implementation**

□ Tracking a Nonstationary Problem

□ Optimistic Initial Values

□ UCB Action Selection

□ Gradient Bandit

□ Contextual Bandits

□ Conclusion

Incremental Implementation

- So far, action values are averages of observed rewards

- How to update action values efficiently with constant memory and constant per-time-step computation?

Incremental Implementation

- Consider an action a
- R_i: the reward received after the i th selection of this action
- Q_n: estimate of the action value of a after it has been selected n-1 times
  - $Q_n = \frac{R_1+R_2+\cdots+R_{n-1}}{n-1}$

- Naïve method: keep all R_i and compute Q_n
  - Disadvantage: memory and computation grow over time

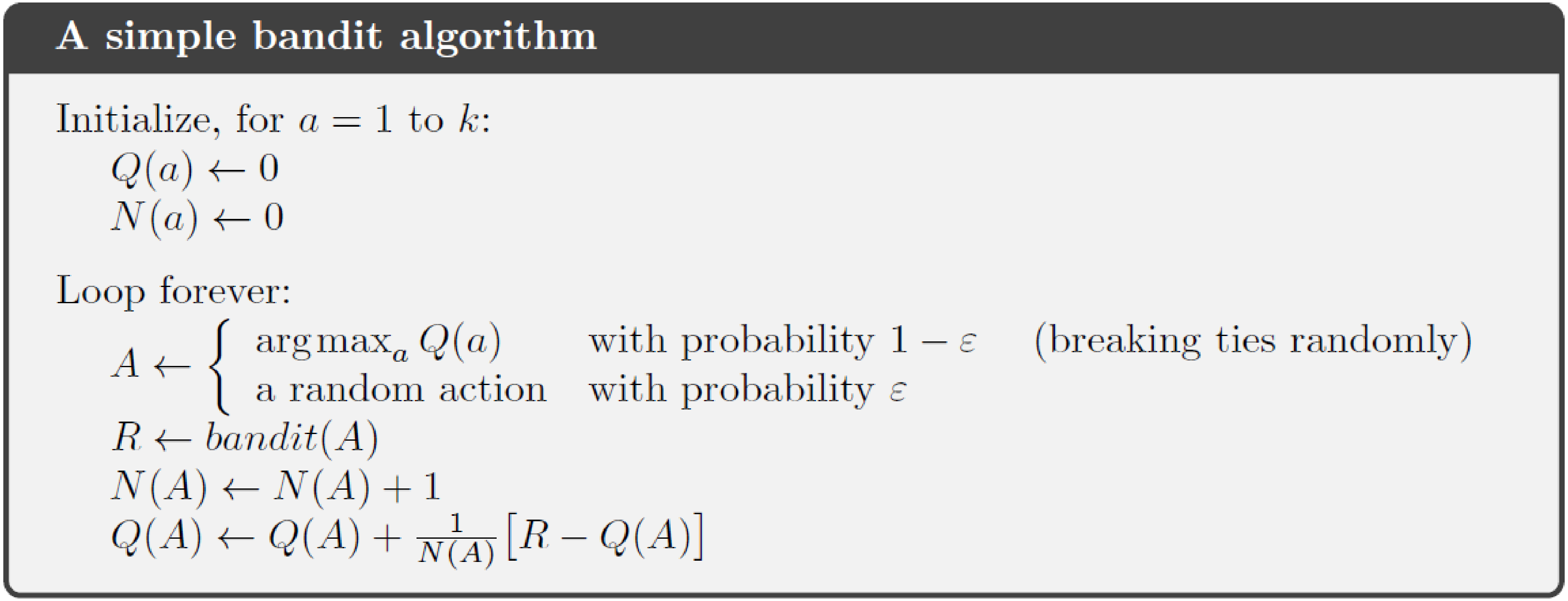
Incremental Implementation

- It's possible to incrementally update Q_n, with constant memory and computation
- $$\begin{aligned} Q_{n+1} &= \frac{1}{n}\sum_{i=1}^{n} R_i \\ &= \frac{1}{n}\left(R_n + \sum_{i=1}^{n-1} R_i\right) \\ &= \frac{1}{n}\left(R_n + (n-1)\frac{1}{n-1}\sum_{i=1}^{n-1} R_i\right) \\ &= \frac{1}{n}\left(R_n + (n-1)Q_n\right) \\ &= \frac{1}{n}\left(R_n + nQ_n - Q_n\right) \\ &= Q_n + \frac{1}{n}\left(R_n - Q_n\right) \end{aligned}$$

Incremental Implementation

- General form of update rule
  - $NewEstimate \leftarrow OldEstimate + \alpha[Target - OldEstimate]$
  - α: step size
    - This value changes over time in the incremental method
  - $Target - OldEstimate$: error

Simple Bandit Algorithm



```
A simple bandit algorithm

Initialize, for $a = 1$ to $k$:
    $Q(a) \leftarrow 0$
    $N(a) \leftarrow 0$

Loop forever:
    $A \leftarrow \begin{cases} \arg\max_a Q(a) & \text{with probability } 1-\varepsilon \quad \text{(breaking ties randomly)} \\ \text{a random action} & \text{with probability } \varepsilon \end{cases}$
    $R \leftarrow bandit(A)$
    $N(A) \leftarrow N(A) + 1$
    $Q(A) \leftarrow Q(A) + \frac{1}{N(A)}\left[R - Q(A)\right]$
```

Sutton and Barto,
Reinforcement
Learning, 2018

Outline

- ☑ K-armed Bandit Problem
- ☑ Action-value Methods
- ☑ Incremental Implementation
- ➡ □ **Tracking a Nonstationary Problem**
- □ Optimistic Initial Values
- □ UCB Action Selection
- □ Gradient Bandit
- □ Contextual Bandits
- □ Conclusion

Nonstationary Problem

- Nonstationary problem: reward probabilities change over time

- It makes sense to give more weight to recent rewards than to long-past rewards
  - $Q_{n+1} = Q_n + \alpha\,(R_n - Q_n)$
  - Note that α is a constant value

Nonstationary Problem

- Q_{n+1} becomes a weighted average of past rewards and the initial estimate Q_1
- $Q_{n+1} = Q_n + \alpha(R_n - Q_n)$

$$= \alpha R_n + (1-\alpha)Q_n$$

$$= \alpha R_n + (1-\alpha)[\alpha R_{n-1} + (1-\alpha)Q_{n-1}]$$

$$= \alpha R_n + (1-\alpha)\alpha R_{n-1} + (1-\alpha)^2 Q_{n-1}$$

$$= \alpha R_n + (1-\alpha)\alpha R_{n-1} + (1-\alpha)^2 \alpha R_{n-2} + \cdots + (1-\alpha)^{n-1}\alpha R_1 + (1-\alpha)^n Q_1$$

$$= (1-\alpha)^n Q_1 + \sum_{i=1}^{n} \alpha(1-\alpha)^{n-i} R_i$$

- This is called *exponential recency-weighted average*

Step Size

- It is convenient to vary the step-size parameter over time
- $\alpha_n(a)$: step-size parameter after *n*th selection of action a
- $\alpha_n(a) = \frac{1}{n}$ guarantees to converge to the true action values by the law of large numbers
- Conditions required to assure convergence:
  - $\sum_{n=1}^{\infty} \alpha_n(a) = \infty,$ and $\sum_{n=1}^{\infty} \alpha_n^2(a) < \infty$
  - Note that $\alpha_n(a) = \frac{1}{n}$ satisfies both conditions
    - $\sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6}$
  - Setting $\alpha_n(a)$ to a constant does not satisfy the conditions
    - However, this is in fact desirable in a non-stationary environment which is common in RL

Outline

☑ K-armed Bandit Problem

☑ Action-value Methods

☑ Incremental Implementation

☑ Tracking a Nonstationary Problem

➡ □ **Optimistic Initial Values**

□ UCB Action Selection

□ Gradient Bandit

□ Contextual Bandits

□ Conclusion

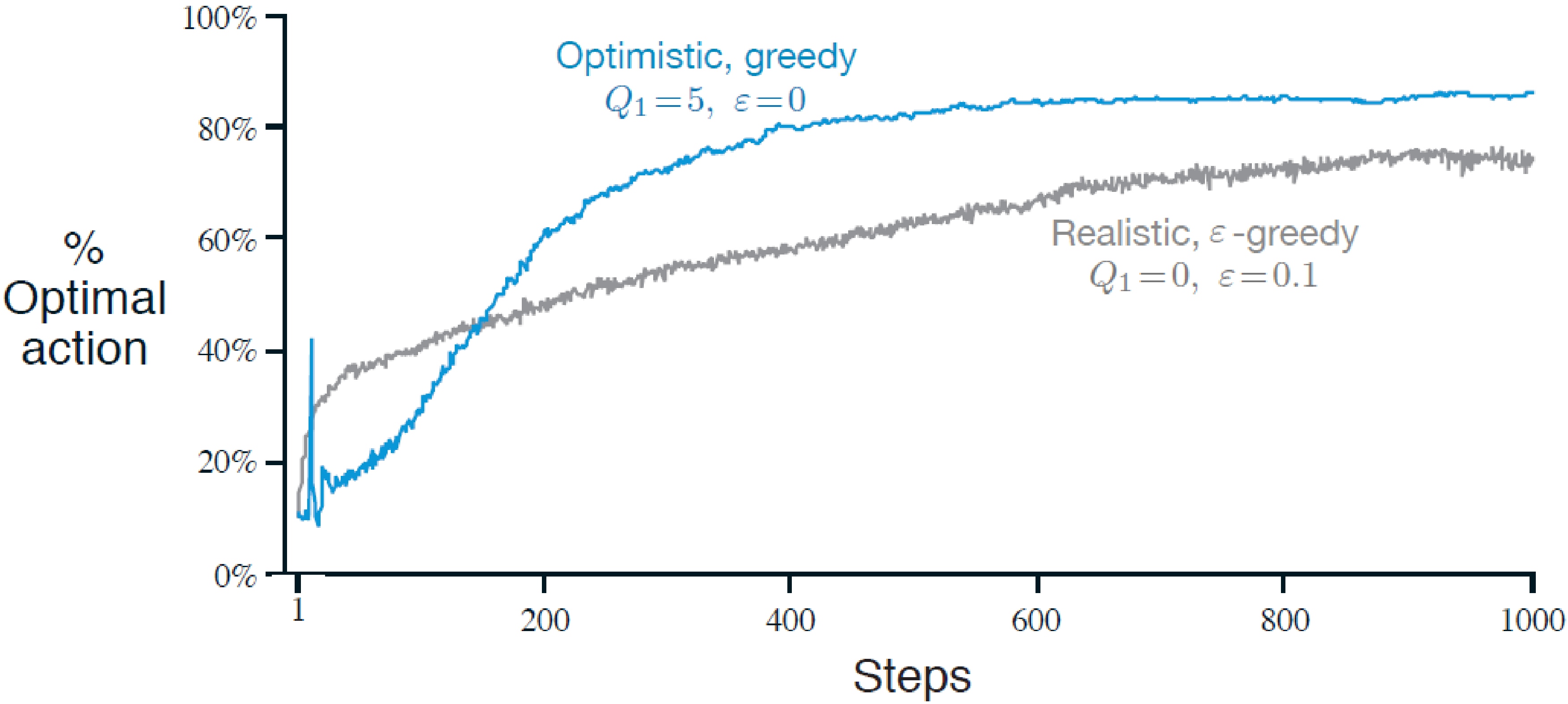
Optimistic Initial Values

- All the methods we have discussed so far depend on the initial action-value estimates, $Q_1(a)$
  - These methods are biased by the estimates
- For the sample-average methods, the bias disappears once all actions have been selected at least once
- For constant step size, the bias is permanent
  - In fact, this can be helpful, if we carefully select the initial estimates; this provides an easy way to supply some prior knowledge about what level of rewards can be expected

Optimistic Initial Values

- Initial action values can also be used to encourage exploration
- 10-armed testbed
  - Suppose we set the initial action values to 5, not 0
  - This value is very optimistic, since $q_*(a) \sim N(0;\ 1)$
  - This optimism encourages action-value methods to explore; all actions are tried several times before the value estimates converge
  - The system does a fair amount of exploration even if greedy actions are selected all the time

Optimistic Initial Values



Sutton and Barto, Reinforcement Learning, 2018

Optimistic Initial Values

- This trick is effective on stationary problems
- But it is not well suited to nonstationary problems, because the exploration is temporary
- If the task changes, creating a renewed need for exploration, this method cannot help; The beginning of time occurs only once, and thus we should not focus on it too much

Outline

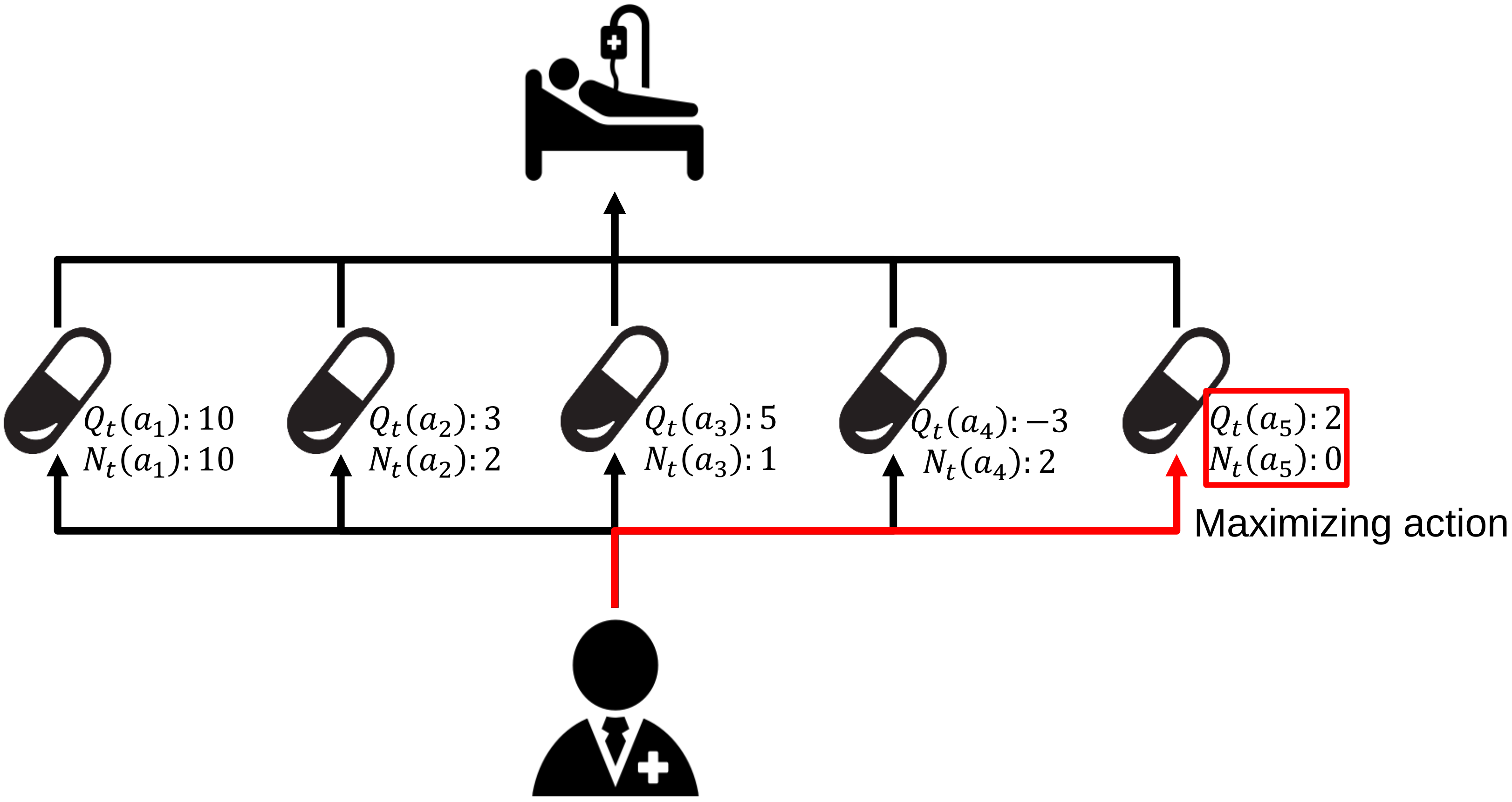
- ☑ K-armed Bandit Problem
- ☑ Action-value Methods
- ☑ Incremental Implementation
- ☑ Tracking a Nonstationary Problem
- ☑ Optimistic Initial Values
- ➡ □ **UCB Action Selection**
- □ Gradient Bandit
- □ Contextual Bandits
- □ Conclusion

UCB Action Selection

- Exploration is needed because of uncertainty about the accuracy of action-value estimates
- ϵ-greedy action selection forces the non-greedy actions to be tried, but indiscriminately, with no preference for those that are uncertain
- It would be better to select among the non-greedy actions according to their potential for actually being optimal, and their uncertainties

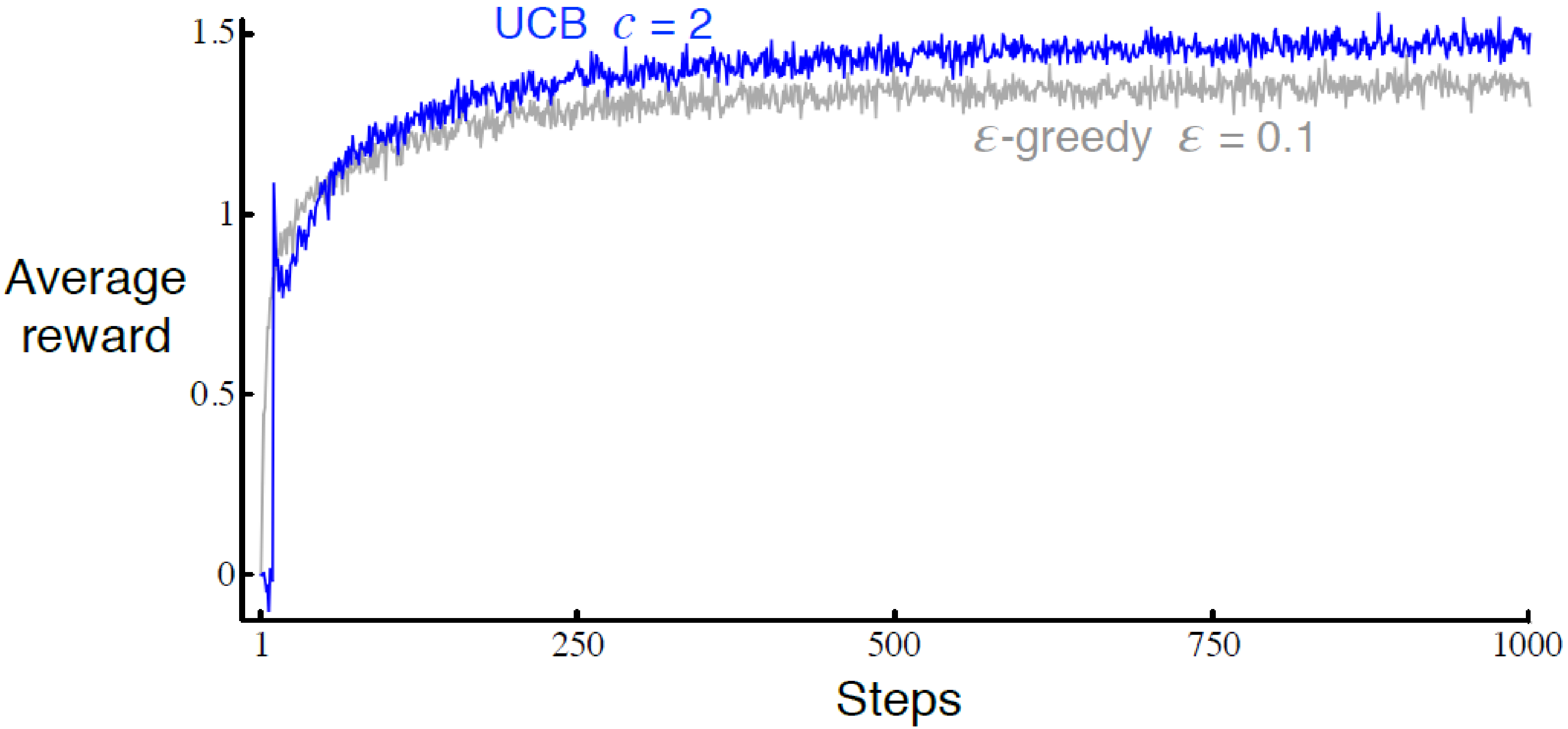
UCB Action Selection

- Upper Confidence Bound (UCB) action selection
  - $A_t = argmax_a[Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}}]$
  - $N_t(a)$: # of times a has been selected prior to time t
  - When $N_t(a) = 0$, a is considered to be a maximizing action
  - Intuition
    - $\sqrt{N_t(a)}$ is a measure of uncertainty or variance in the estimate of a's value
    - Prefer to select an uncertain action, since it may give good reward



UCB Action Selection

UCB Action Selection



Sutton and Barto,
Reinforcement
Learning, 2018

UCB Action Selection

- UCB
  - $A_t = argmax_a[Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}}]$
- UCB does not extend well to more general RL settings
- Difficulty
  - Dealing with nonstationary problems
  - Dealing with large state spaces, particularly when using function approximation

Outline

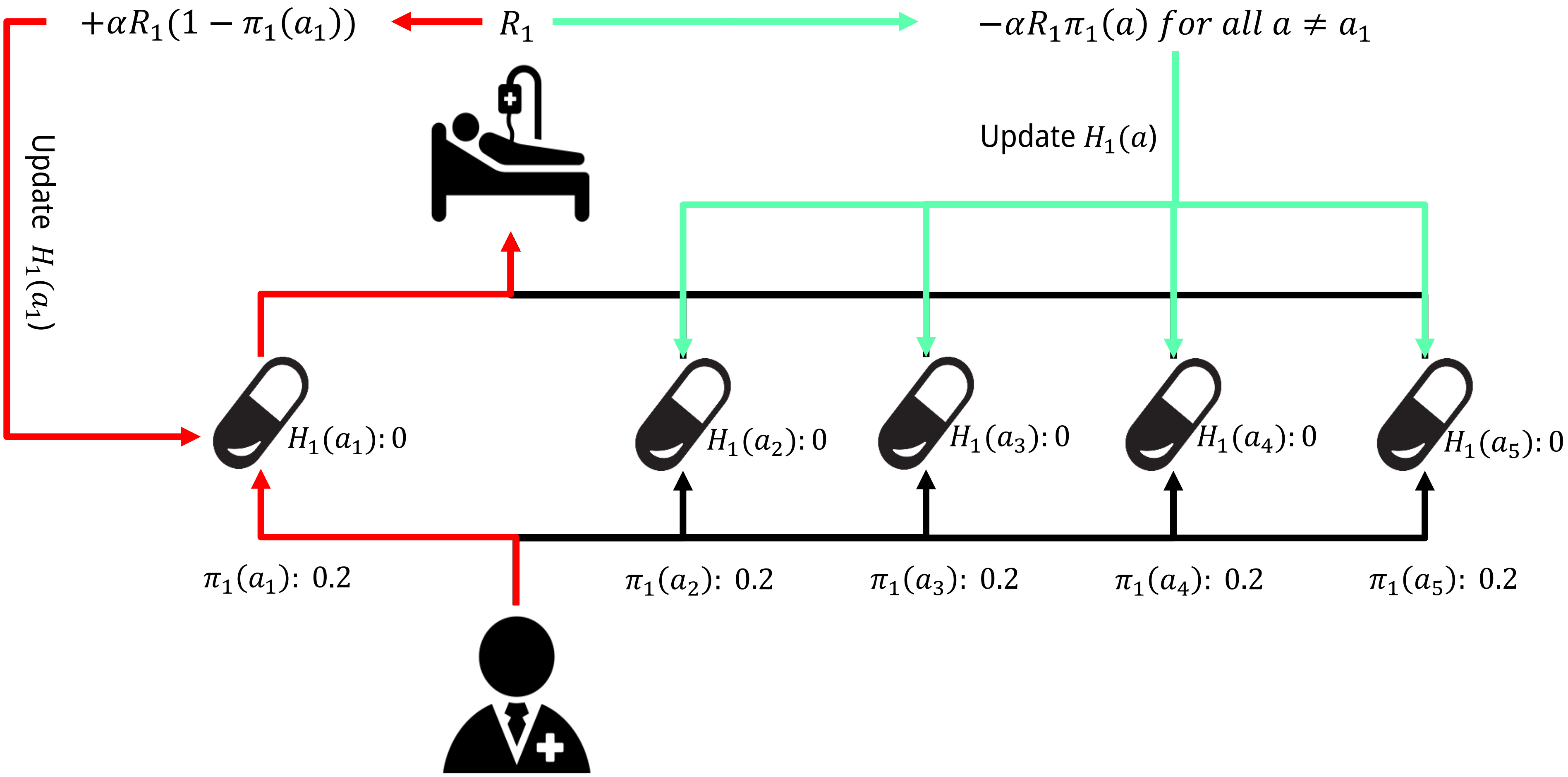
- ☑ K-armed Bandit Problem
- ☑ Action-value Methods
- ☑ Incremental Implementation
- ☑ Tracking a Nonstationary Problem
- ☑ Optimistic Initial Values
- ☑ UCB Action Selection
- ➡ □ **Gradient Bandit**
- □ Contextual Bandits
- □ Conclusion

Gradient Bandit

- Up to this point we considered methods that estimate action values and use them to select actions
- Alternative: consider learning a numerical preference $H_t(a)$ for each action a
- The larger the preference, the more often that action is taken, but the preference has no interpretation in terms of reward. Only the relative preference of one action over another is important
- Select action based on soft-max distribution
  - $P(A_t = a) = \frac{e^{H_t(a)}}{\sum_{b=1}^{k} e^{H_t(b)}} = \pi_t(a)$
  - Initially, all action preferences are the same ($H_1(a) = 0$ for all a)

Learning Gradient Bandit

- After selecting action A_t and receiving reward R_t at time t, update action preference
  - $H_{t+1}(A_t) \leftarrow H_t(A_t) + \alpha(R_t - \bar{R}_t)(1 - \pi_t(A_t))$, and
  - $H_{t+1}(a) \leftarrow H_t(a) - \alpha(R_t - \bar{R}_t)\pi_t(a)$ for all $a \neq A_t$
  - α: step-size parameter
  - $\bar{R}_t$: average reward up to time t



$+\alpha R_1(1-\pi_1(a_1))$

R_1

$-\alpha R_1 \pi_1(a)\ for\ all\ a \neq a_1$

Update $H_1(a_1)$

Update $H_1(a)$

$H_1(a_1)$: 0

$H_1(a_2)$: 0

$H_1(a_3)$: 0

$H_1(a_4)$: 0

$H_1(a_5)$: 0

$\pi_1(a_1)$: 0.2

$\pi_1(a_2)$: 0.2

$\pi_1(a_3)$: 0.2

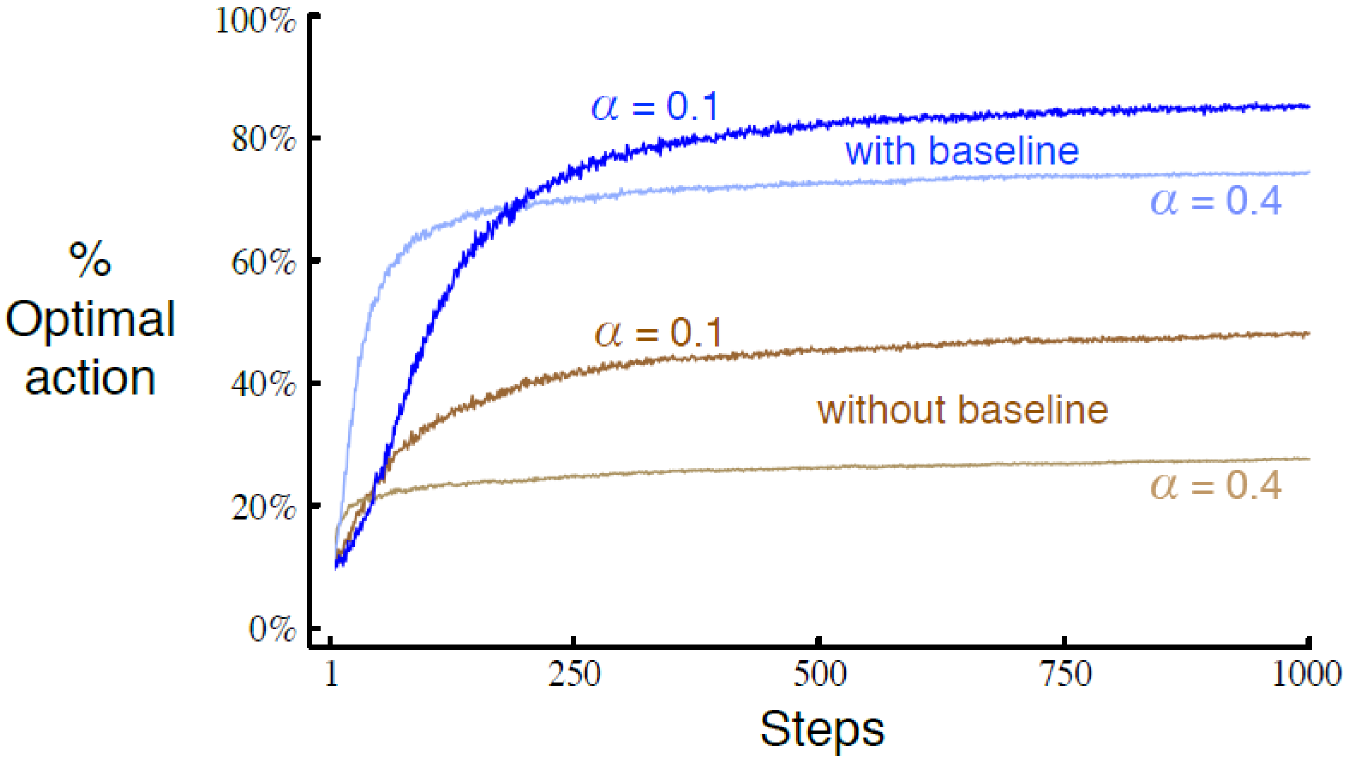
$\pi_1(a_4)$: 0.2

$\pi_1(a_5)$: 0.2

Gradient bandit
at t = 1 (Initial)

Gradient Bandit

- Results on 10-armed testbed, where the true rewards are selected from a Gaussian with mean +4
  - $H_{t+1}(A_t) \leftarrow H_t(A_t) + \alpha(R_t - \bar{R}_t)(1 - \pi_t(A_t))$, and
  - $H_{t+1}(a) \leftarrow H_t(a) - \alpha(R_t - \bar{R}_t)\pi_t(a)$ for all $a \neq A_t$



Sutton and Barto, Reinforcement Learning, 2018

Outline

- ☑ K-armed Bandit Problem
- ☑ Action-value Methods
- ☑ Incremental Implementation
- ☑ Tracking a Nonstationary Problem
- ☑ Optimistic Initial Values
- ☑ UCB Action Selection
- ☑ Gradient Bandit
- ➡ □ **Contextual Bandits**
- □ Conclusion

Contextual Bandits

- We have discussed nonassociative tasks with a single state: no need to associate different actions with different states

- In general RL, there are more than one states, and the goal is to learn a policy: a function from state to action

- Associative search = contextual bandits

Contextual Bandits

- Associative search (contextual bandits)
  - Assume there are several different k-armed bandit tasks; on each step we confront one of these at random
  - One approach: consider it as a nonstationary k-armed bandit task; it will not work well
  - Assume that when a bandit task is selected for us, we are given some clue about its identity (but not its action values)
  - Then, we can learn a policy associating each task with the best action to take

Contextual Bandits

- Associative search tasks are intermediate between the k-armed bandit problem and the full RL problem.
- They are like the full RL since they learn a policy, but each action affects only the intermediate reward
- If actions are allowed to affect the next situation as well as the reward, then we have the full RL problem

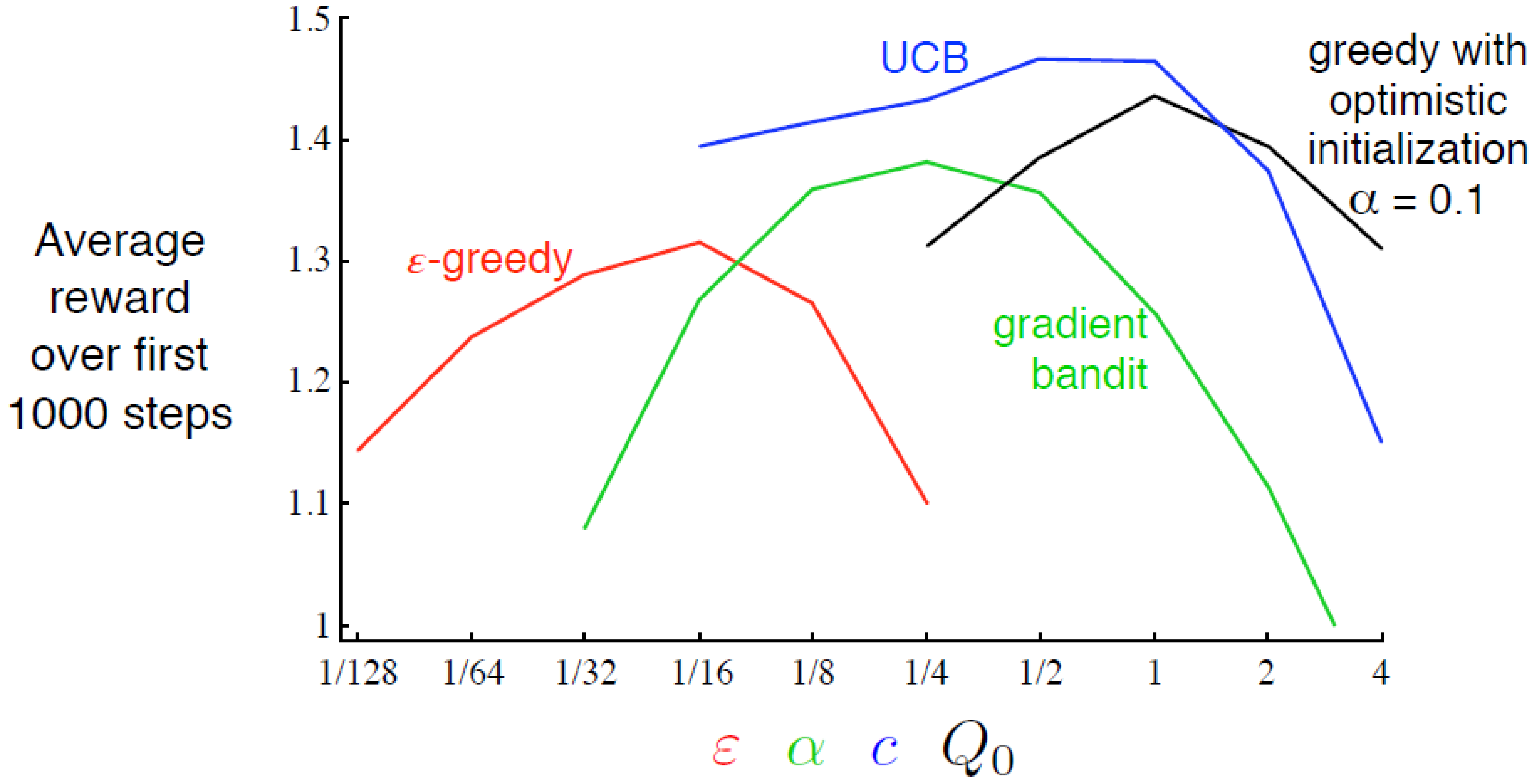
Outline

- ☑ K-armed Bandit Problem
- ☑ Action-value Methods
- ☑ Incremental Implementation
- ☑ Tracking a Nonstationary Problem
- ☑ Optimistic Initial Values
- ☑ UCB Action Selection
- ☑ Gradient Bandit
- ☑ Contextual Bandits
- ➡ □ **Conclusion**

Conclusion

- Ways of balancing exploration and exploitation
  - ϵ–greedy: choose randomly in a small fraction of the time
  - UCB: choose deterministically, but subtly favor actions that have so far received fewer samples
  - Gradient bandit: estimate action preferences, and favor the more preferred actions in a probabilistic manner using a soft-max distribution
  - Optimistic initialization: initializing estimates optimistically causes even greedy methods to explore significantly

Conclusion



Sutton and Barto, Reinforcement Learning, 2018

Exercise

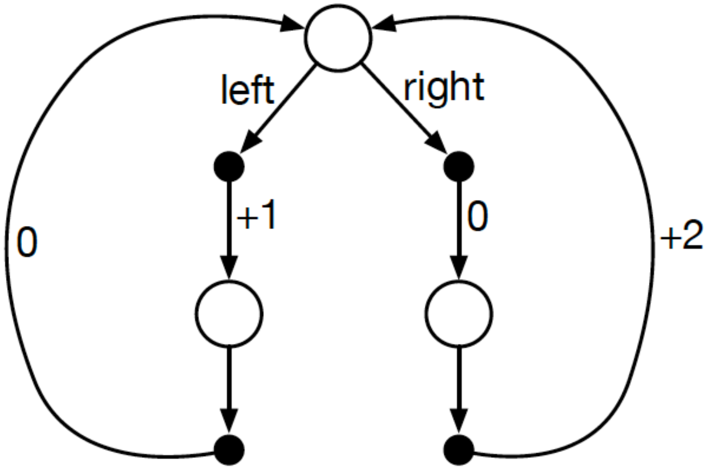
- (Question 1)
- Consider a k-armed bandit problem with k = 4 actions, denoted 1, 2, 3, and 4. Consider applying to this problem a bandit algorithm using ε-greedy action selection, sample-average action-value estimates, and initial estimates of Q_1(a) = 0, for all a. Suppose the initial sequence of actions and rewards is A_1 = 1, R_1 = -1, A_2 = 2, R_2 = 1, A_3 = 2, R_3 = -2, A_4 = 2, R_4 = 2, A_5 = 3, R_5 = 0.
- On which time steps did the random action selection definitely occur? On which time steps could this possibly have occurred?

Exercise

- (Answer)
- The initial sequence of actions and rewards is A_1 = 1, R_1 = -1, A_2 = 2, R_2 = 1, A_3 = 2, R_3 = -2, A_4 = 2, R_4 = 2, A_5 = 3, R_5 = 0
- The action values are as follows
  - Initial action values: 0, 0, 0, 0
  - T_1: -1, 0, 0, 0 (greedy or random)
  - T_2: -1, 1, 0, 0 (greedy)
  - T_3: -1, -0.5, 0, 0 (greedy)
  - T_4: -1, 0.333, 0, 0 (random)
  - T_5: -1, 0.333, 0, 0 (random)

Exercise

- (Question 2)
- Consider the continuing MDP shown to the below. The only decision to be made is that in the top state, where two actions are available, left and right. The numbers show the rewards that are received deterministic policies, π_{left} and π_{right}. What policy is optimal if $\gamma = 0$? If $\gamma = 0.9$? If $\gamma = 0.5$?



Exercise

- (Answer)
- The initial sequence of actions and rewards is A_1 = 1, R_1 = -1, A_2 = 2, R_2 = 1, A_3 = 2, R_3 = -2, A_4 = 2, R_4 = 2, A_5 = 3, R_5 = 0
- The action values are as follows
  - Initial action values: 0, 0, 0, 0
  - T_1: -1, 0, 0, 0 (greedy or random)
  - T_2: -1, 1, 0, 0 (greedy)
  - T_3: -1, -0.5, 0, 0 (greedy)
  - T_4: -1, 0.333, 0, 0 (random)
  - T_5: -1, 0.333, 0, 0 (random)

Questions?