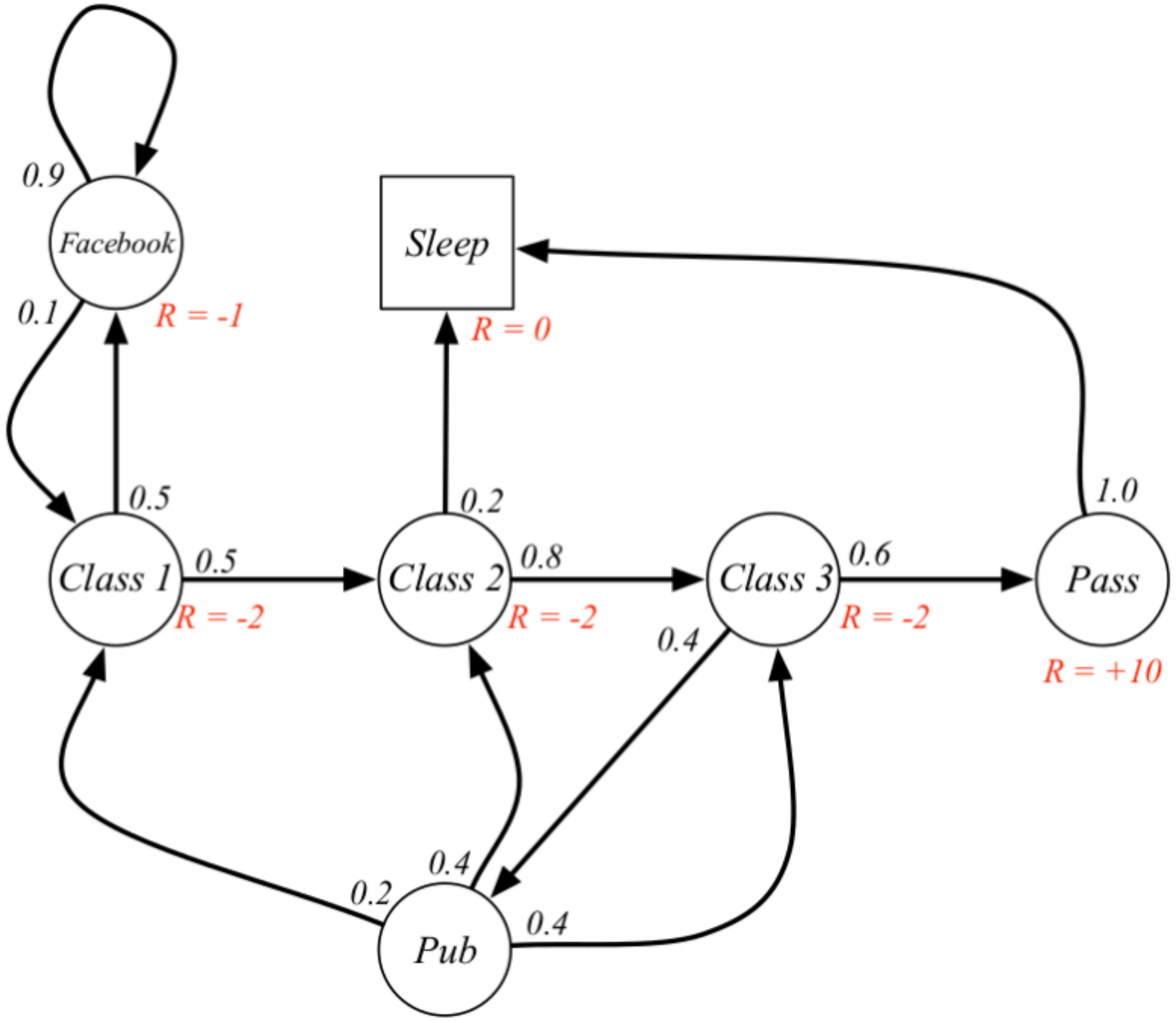
How can you learn what is the best decision?

Readings for today

- Sutton, R. S., & Barto, A. G. (2020). Chapter 1: Introduc- tion. In Reinforcement learning: An introduction (2nd edi- tion). MIT press.
- Sutton, R. S., & Barto, A. G. (2020). Chapter 2: Multi-armed bandits. In Reinforcement learning: An introduction (2nd edi- tion). MIT press.

The state-action problem



What is the best way to strategically ***learn to*** shift from one state to another?

(Ashraf 2021)

The Bellman equation

What is the *optimal* path through potential states that has the highest value?

$$v(s) = \mathbb{E}(G_t \mid S_t = s)$$

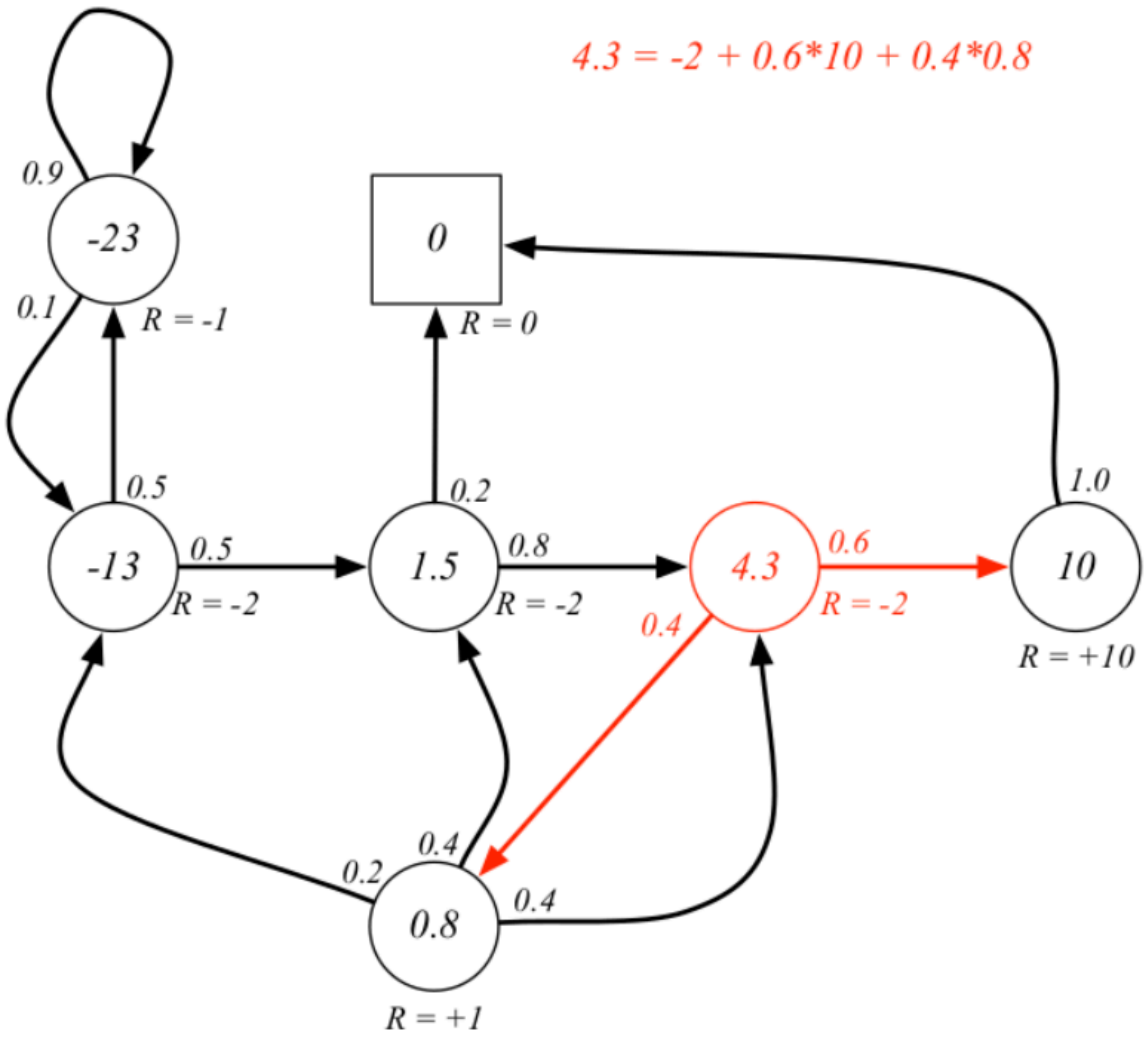
$$G_{t+1} \rightarrow v(S_{t+1})$$

Bellman equation

$$\boxed{v(s) = \mathbb{E}(R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s)}$$

(Ashraf 2021)

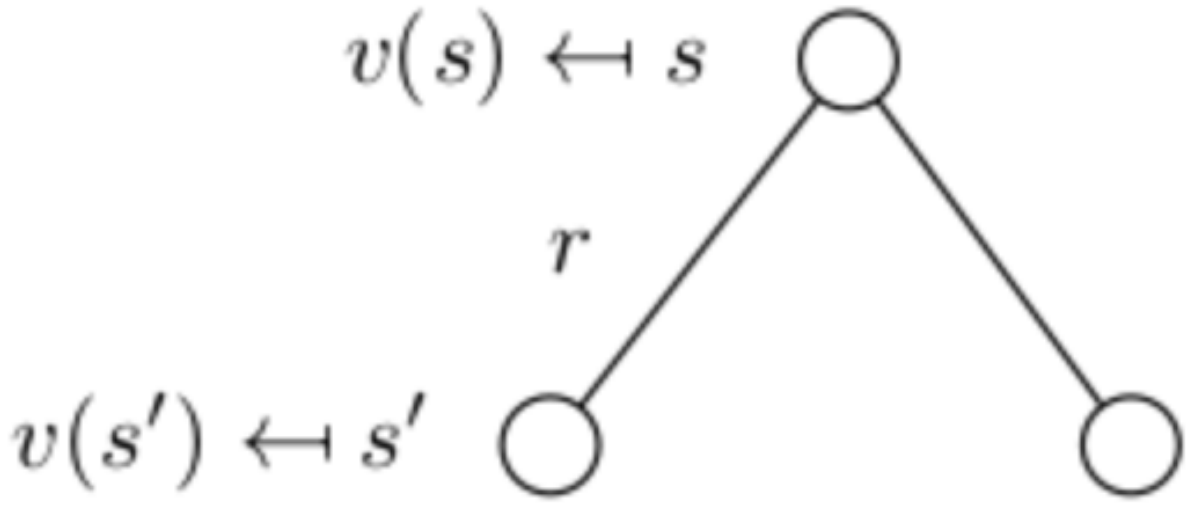
The Bellman equation



The value both depends on the reward <u>and</u> the transition probability .

$$v(s) = \mathbb{E}(R_{t+1} + \gamma v(S_{t+1}) \,|\, S_t = s)$$

$$= \mathbf{R}_S + \gamma \sum_{s' \in S} \mathbf{P}_{ss'} v(s')$$



(Ashraf 2021)

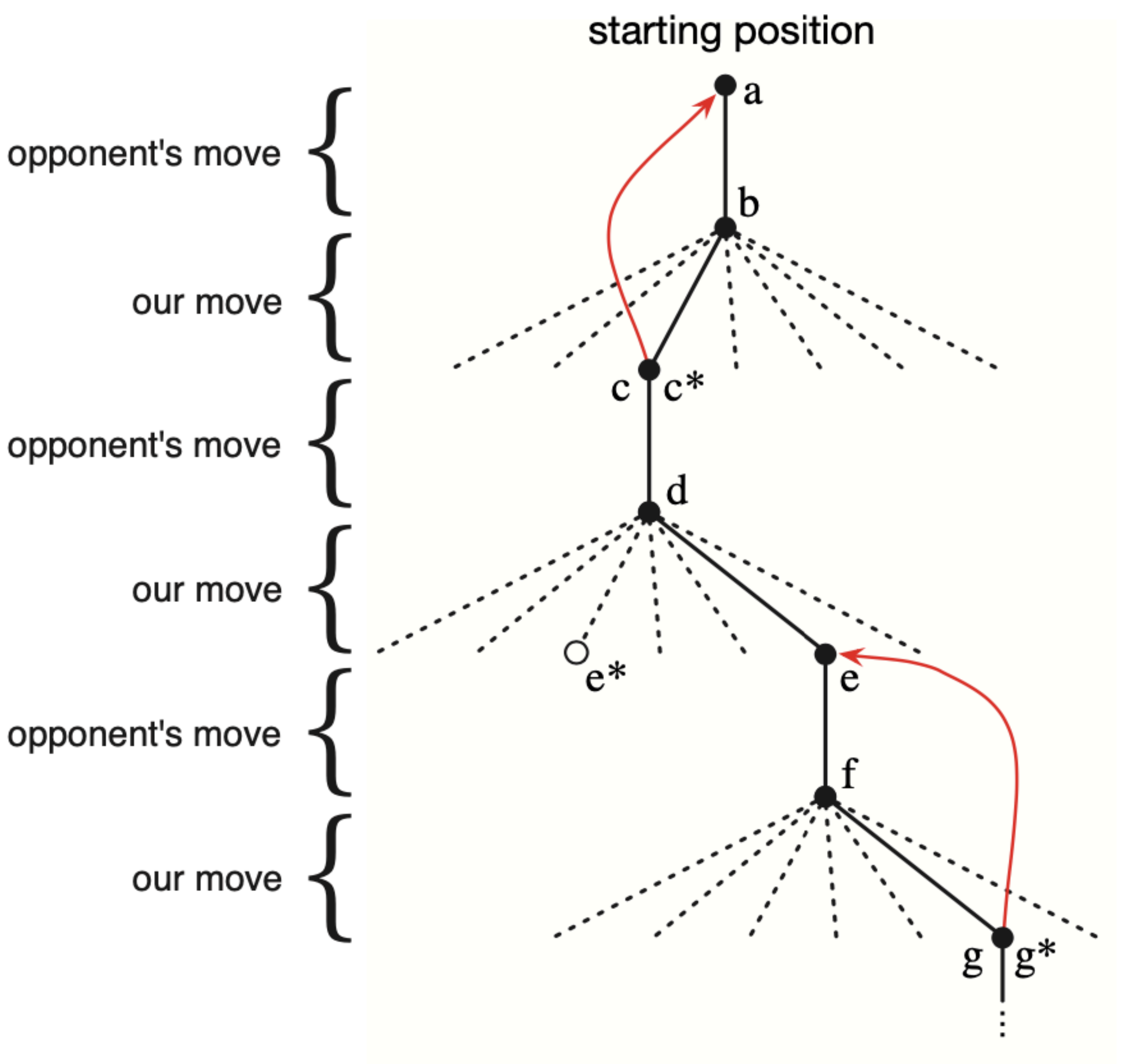
Temporal difference update

How do you update your state value from one move to the next?

$$\underbrace{}_{}V(S_{t+1}) \leftarrow V(S_t) + \alpha[V(S_{t+1}) - V(S_t)]$$

error

learning rate



This works in the forward looking framing of the Bellman equation.

(Sutton & Barto, 2020)

Reinforcement learning

Trial & Error Learning

Optimal Control Theory

Dynamic Programming

Reinforcement Learning

Learning what to do—how to map situations to actions—so as to maximize a numerical reward signal.

(Sutton & Barto, 2020)

The *k*-armed bandit problem

The Bandit task

- There are k many actions (arms), a, that can be made at any time t
- Each action has returns a reward r probabilistically, from a (typically) unchanging probability distribution.
- **Goal**: Find the action (arm) that has the highest expected return $\mathbb{E}[R_t]$

(Sutton & Barto, 2020)

The *k*-armed bandit problem

Action value problem

$$q(a) = \mathbb{E}[R_t \mid A_t = a]$$

$$Q_t(a) = \frac{\sum_{i=1}^{t-1} R_i \mid A_t = a}{\sum_{i=1}^{t-1} A_t = a}$$

(Sutton & Barto, 2020)

The ϵ-greedy method

Action value

$$Q_t(a) = \frac{\sum_{i=1}^{t-1} R_i \,|\, A_t = a}{\sum_{i=1}^{t-1} A_t = a}$$

Best action

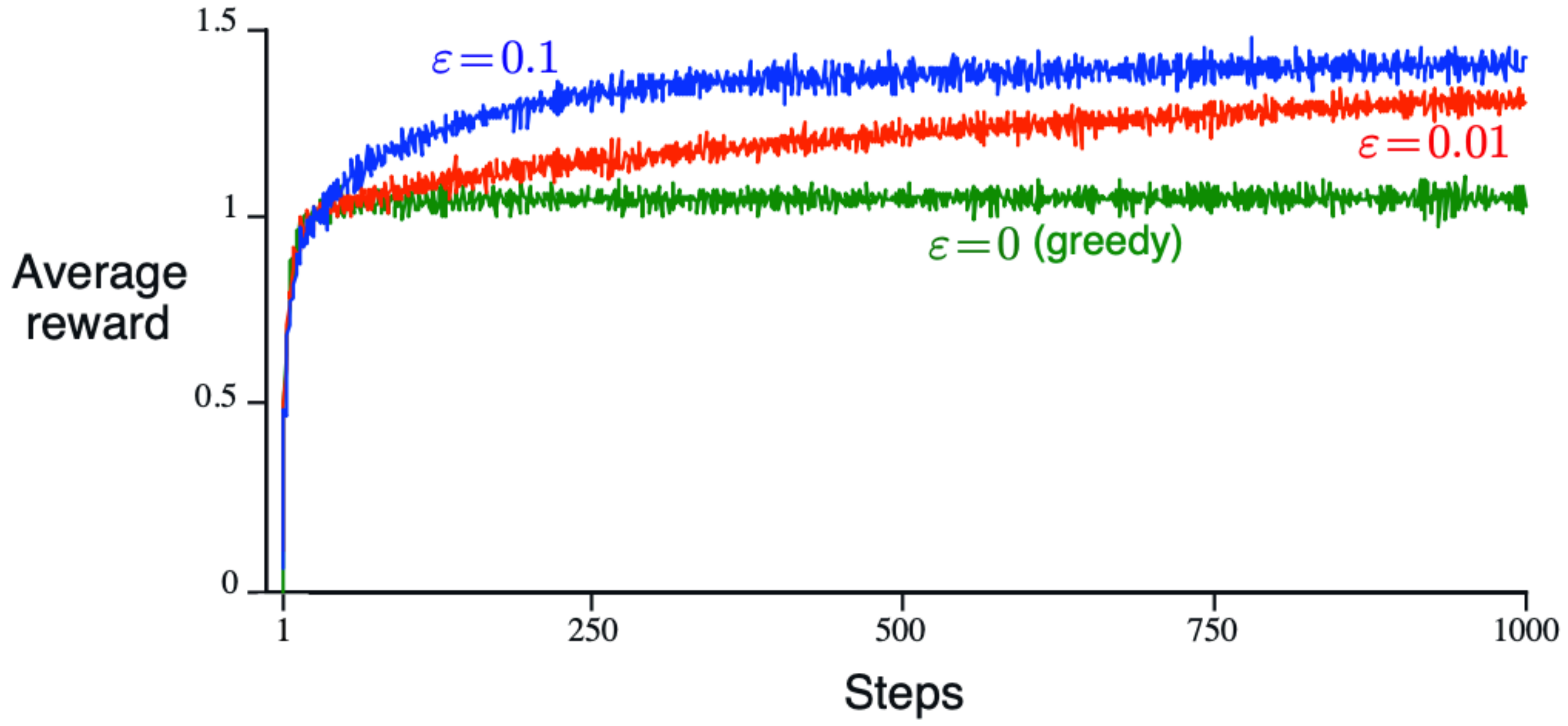
$$A_t = \arg\max_a Q_t(a)$$

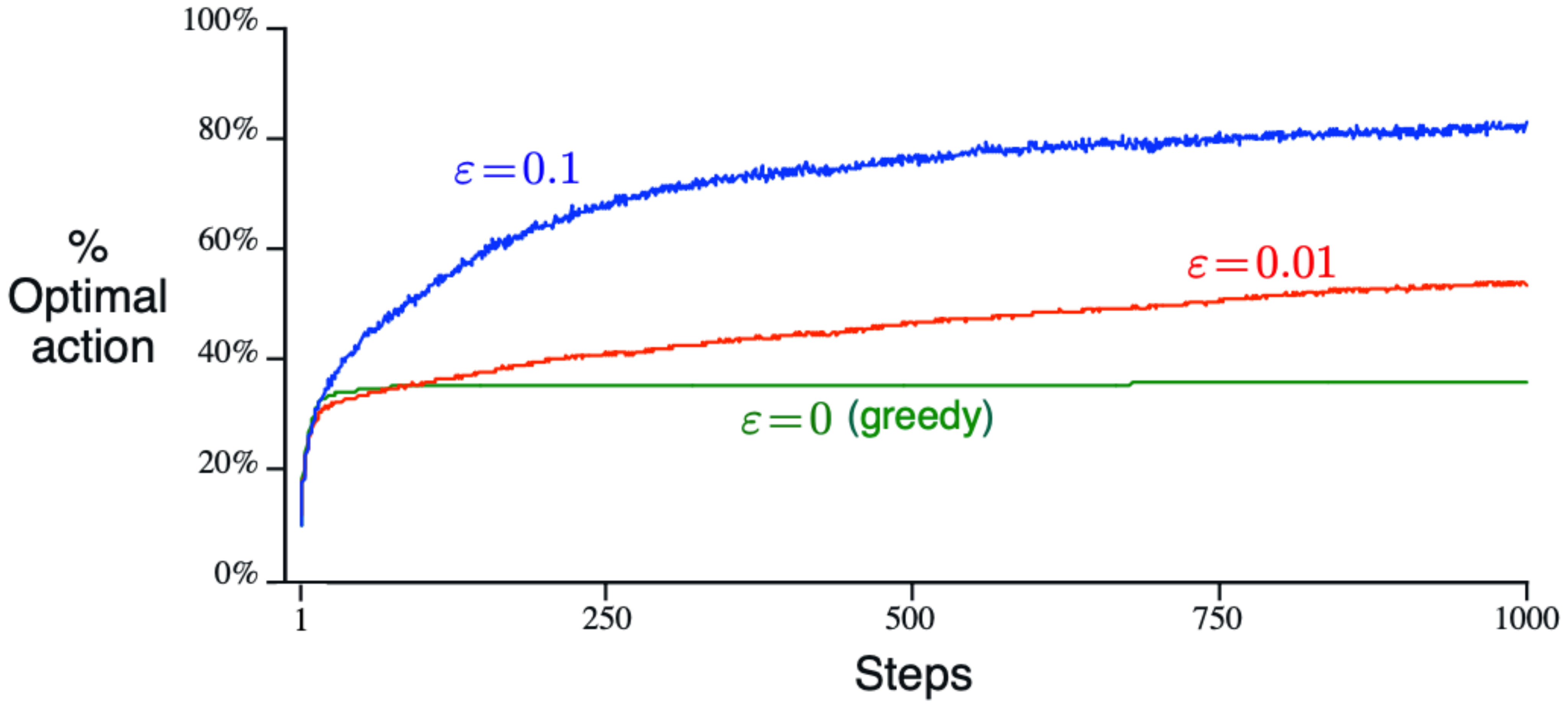
Decision policy

$$\begin{cases} \max Q_t(a), & \text{with probability } 1-\epsilon \\ \text{any } a, & \text{with probability } \epsilon \end{cases}$$

(Sutton & Barto, 2020)

The ϵ-greedy method





Reducing the *greediness* of the algorithm, increases the exploration of the agent, allowing for optimal learning.

A simple bandit algorithm

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

Loop forever:

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

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

(Sutton & Barto, 2020)

Nonstationary problems

Incremental update rule

$$NewEstimate \leftarrow OldEstimate + StepSize \left[Target - OldEstimate \right]$$

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

$$\longrightarrow \quad Q_{n+1} \doteq Q_n + \alpha\Big[R_n - Q_n\Big]$$

(Sutton & Barto, 2020)

The reinforcement learning structure

$$NewEstimate \leftarrow OldEstimate \ + \ StepSize \left[Target - OldEstimate \right]$$

Reward prediction error (RPE)

$$Q_{n+1} \doteq Q_n + \alpha \Big[R_n - Q_n \Big]$$

learning rate

(Sutton & Barto, 2020)

The reinforcement learning structure

$$NewEstimate \leftarrow OldEstimate \ + \ StepSize \left[Target - OldEstimate \right]$$

$$\begin{aligned} Q_{n+1} &= Q_n + \alpha \Big[R_n - Q_n \Big] \\ &= \alpha R_n + (1-\alpha) Q_n \\ &= \alpha R_n + (1-\alpha) \left[\alpha R_{n-1} + (1-\alpha) Q_{n-1} \right] \\ &= \alpha R_n + (1-\alpha) \alpha R_{n-1} + (1-\alpha)^2 Q_{n-1} \\ &= \alpha R_n + (1-\alpha) \alpha R_{n-1} + (1-\alpha)^2 \alpha R_{n-2} + \\ & \qquad \qquad \cdots + (1-\alpha)^{n-1} \alpha R_1 + (1-\alpha)^n Q_1 \\ &= (1-\alpha)^n Q_1 + \sum_{i=1}^{n} \alpha (1-\alpha)^{n-i} R_i. \end{aligned}$$

Assuming $a \in (0,1]$, then Q_t is simply the weighted average of past rewards and the initial estimate Q_1

(Sutton & Barto, 2020)

Food for thought

Believer-Skeptic Time

- The basic RL models that we went over are principled off of decision problems that follow the form of the k-armed bandit task (in fact, the algorithms we discussed were explicitly designed to work in bandit environments). ***Does the bandit task reasonably capture the properties of real life value-based decisions?***