1. **Upper Confidence Bound**
Suppose we are conducting an experiment where we have to perform a measurement. Let us say we device N different ways of making the same measurement and that these take on values $x_i, i = 1...N$ (note here, x_i is random variable representing the value of measurement using method i. If we do multiple measurements using the same method, we may get slightly different values each time). The random variable representing the mean value of the measurement using N-methods is then given by

$$\bar{X} = \frac{x_1 + x_2 + ... + x_N}{N} \tag{1}$$

Show that the standard-deviation of $\bar{X}$ scales as $\frac{1}{\sqrt{N}}$. This is the reason for the $\sqrt{N}$ term in upper-confidence bound which also shows that the mean becomes a better estimate of the actual measurement as N increases.

Solution. The formula for standard deviation of $\bar{X}$ is,

$$\sigma(\bar{X}) = \frac{\sigma(X)}{\sqrt{N}} \tag{2}$$

where,

- $\bar{X}$ is the sample mean
- $\sigma(X)$ is the population standard deviation
- $\sigma(\bar{X})$ is the standard deviation of sample mean
- N is the sample size

$$\begin{aligned}
\Rightarrow Var(\bar{X}) &= Var\left(\frac{x_1 + x_2 + ... + x_N}{N}\right) \\
&= Var\left(\frac{x_1}{N} + \frac{x_2}{N} + ... + \frac{x_N}{N}\right) \\
&= \frac{Var(x_1)}{N^2} + \frac{Var(x_2)}{N^2} + ... + \frac{Var(x_N)}{N^2} \\
&= \frac{1}{N^2}[\sigma^2 + \sigma^2 + ... + \sigma^2] \\
&= \frac{\sigma(X)^2}{N}
\end{aligned}$$

$$\Rightarrow Var(\bar{X}) = \frac{\sigma(X)^2}{N} \tag{3}$$

Now the standard deviation of sample mean $\bar{X}$ is,

$$\begin{aligned}\Rightarrow \sigma(\bar{X}) &= \sqrt{Var(X)}\\ &= \sqrt{\frac{\sigma(X)^2}{N}}\\ &= \frac{\sigma(X)}{\sqrt{N}}\end{aligned}$$

$$\sigma(\bar{X}) = \frac{\sigma(X)}{\sqrt{N}} \tag{4}$$

This shows that increasing the sample size can improve the precision of our estimates of the population mean. ■

2. **Frozen lake random walk**
Consider the Frozen lake example discussed in class with 0 reward for both holes and regular grids, and a reward of 1 for the final prized state on grid 15. Model the agent as a 2-D symmetric random walker on a square grid (assuming the boundaries are far-away) and derive an analytic expression for the probability of the walker to be on a grid-point (3,3) grid points away from the starting position (you may take the starting position to be central grid point). Simulate the frozen lake example and compare the number of iterations it takes for the agent to find the optimal-path with what you can estimate from the random walk model.

Solution. Assuming each grid has equal probability of being visited and $(0, 0)$ to be the starting position as well as the central grid-point and $(3, 3)$ to be the desired grid-point.

Using **lattice path** counting method, each step corresponds to one of the following actions: **up, down, left, right**.

The distance between the two points is 6 grid units. Each step has probability $\frac{1}{4}$ of begin taken, i.e.,

$$P(up) = P(down) = P(left) = P(right) = \frac{1}{4} \tag{5}$$

Using the binomial coefficient formula,

$$^{n}C_k = \frac{n!}{k!(n-k)!} \tag{6}$$

where,

- n: the number of steps
- k: the number of steps in one of the two perpendicular directions

here, $n = 6$ and $k = 3$

$\therefore$ the number of lattice paths $= {}^6C_3 = \dfrac{6!}{3!3!} = 20$

Each lattice path has probability $\dfrac{1}{4^6}$ of occurring. (*because* each step has probability $\dfrac{1}{4}$).

The total probability of the walker being on grid-point (3, 3) is,
$P(3, 3) = 20 \times \dfrac{1}{4^6} = \dfrac{20}{4096} = 0.0048828125.$

So the probability of the walker begin on grid-point $(3, 3)$ is ≈ 0.0049 or 0.49%.

Deriving an analytic expression for the probability distribution of a 2D random walk using Central Limit Theorem.

Let X, and Y be random variables, which are identically distributed, and independent, representing the particles' movements in the x and y directions, the mean is 0, and variance σ^2.

The position of the particle after 'n' time steps can be represented by the random vector (S_X, S_Y),
where,

$$S_X = x_1 + x_2 + ... + x_n \tag{7}$$

$$S_Y = y_1 + y_2 + ... + y_n \tag{8}$$

The mean of the position vector is given by,

$$E[S_X] = E[x_1 + x_2 + ... + x_n] = nE[X] = 0 \tag{9}$$

$$E[S_Y] = E[y_1 + y_2 + ... + y_n] = nE[Y] = 0 \tag{10}$$

The variance of the position vector is given by,

$$Var[S_X] = Var[x_1 + x_2 + ... + x_n] = nVar[X] = n\sigma^2 \tag{11}$$

$$Var[S_Y] = Var[y_1 + y_2 + ... + y_n] = nVar[Y] = n\sigma^2 \tag{12}$$

The covariance between S_X and S_Y is given by,

$$Cov[S_X, S_Y] = E[(x_1 + x_2 + ... + x_n)(y_1 + y_2 + ... + y_n)] - E[S_X]E[S_Y] \tag{13}$$

$\because$ X, and Y are independent with mean 0 and variance σ^2, their product $x_i y_i$ has mean 0 and variance σ^2.

$\therefore$ Central Limit Theorem implies that (S_X, S_Y) converges to a bi-variate normal distribution with mean $(0, 0)$. The subsequent covariance matrix is given by,

$$\Sigma = \begin{bmatrix} Var[S_X] & Cov[S_X, S_Y] \\ Cov[S_X, S_Y] & Var[S_Y] \end{bmatrix} = \begin{bmatrix} n\sigma^2 & 0 \\ 0 & n\sigma^2 \end{bmatrix} \tag{14}$$

The probability density function of a bi-variate normal distribution with mean $(0, 0)$ and covariance matrix Σ is given by,

$$f(x, y) = \frac{1}{2\pi\sqrt{\det \Sigma}} \exp\left(-\frac{1}{2}\left(\frac{x^2}{\Sigma_{XX}} + \frac{y^2}{\Sigma_{YY}} - \frac{2\rho xy}{\Sigma_{XY}}\right)\right) \tag{15}$$

where,

- $\det \Sigma$: determinant of Σ
- ρ: $\dfrac{Cov[S_X, S_Y]}{\sigma_x \sigma_y}$
- $\Sigma_{XX}, \Sigma_{YY}, \Sigma_{XY}$: elements of Σ

On simplification,

- $\Sigma_{XX} = \Sigma_{YY} = n\sigma^2$
- $\Sigma_{XY} = 0$

$$f(x, y) = \frac{1}{2\pi n\sigma^2} \exp\left(-\frac{1}{2}(x^2 + y^2)\right) \tag{16}$$

For the analytical expression, the probability of the walker being at any point in the grid after n steps is given by,

$$P(x, y, n) = \left(\frac{1}{4^n}\right) \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} {}^{(i+j+1)}C_i^{(n-i-1+j)} C_j \tag{17}$$

In general Q-learning may require more iterations to converge than the analytical expression or 2D symmetric random walk, especially for complex environments or large state spaces.
However, Q-learning has the advantage of begin able to handle stochastic environments and learning optimal policies from experience without requiring an explicit model of the environment or knowledge of the transition probabilities.

Comparing the number of steps taken to converge for both symmetric random walk and q-learning using computational simulation,

- Symmetric 2D: 12
- Q-learning: 10

■

3. **Value and Quality Function**
Value Function: In the simplest form of reinforcement learning in a deterministic environment, the take of the agent is to learn a policy, $\pi : S \rightarrow A$ for selecting its next action $a_t \in A$ based on their current observed state $s_t \in S$ i.e., $\pi(s_t) = a_t$. An important question then is: how do we specify which policy an agent should learn? One such approach is for the agent to learn a policy which produces the greatest possible cumulative reward over time. This requirement is mathematically expressed by defining a value function $V_\pi(s_t)$ which is achieved by following some policy π from an arbitrary initial state s_t as follows:

$$V_\pi(s_t) = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + ... = \sum_{k=0}^{\infty} \gamma^k r_{t+k} \tag{18}$$

where, $0 \leq \gamma < 1$ is a constant that determines the relative value of future versus immediate rewards. Here, the sequence of rewards r_{t+k} is generated by beginning at state s_t and repeatedly following the same policy π to select actions $a_t = \pi(s_t)$ (likewise, $a_{t+k} = \pi(s_{t+k})$ and so on). The agents learning task is then: To learn a policy π that maximizes $V_{pi}(S)$ over all states S. Such a policy is called the optimal policy which we denote as

$$\pi^* = \arg\max_\pi V_\pi(s) \tag{19}$$

Thus $V^*(s) = V_{\pi^*}(s)$ is the maximum discounted cumulative reward that an agent can obtain starting from a state s. Note, the value function is only a function of the state space (and not of the action space).

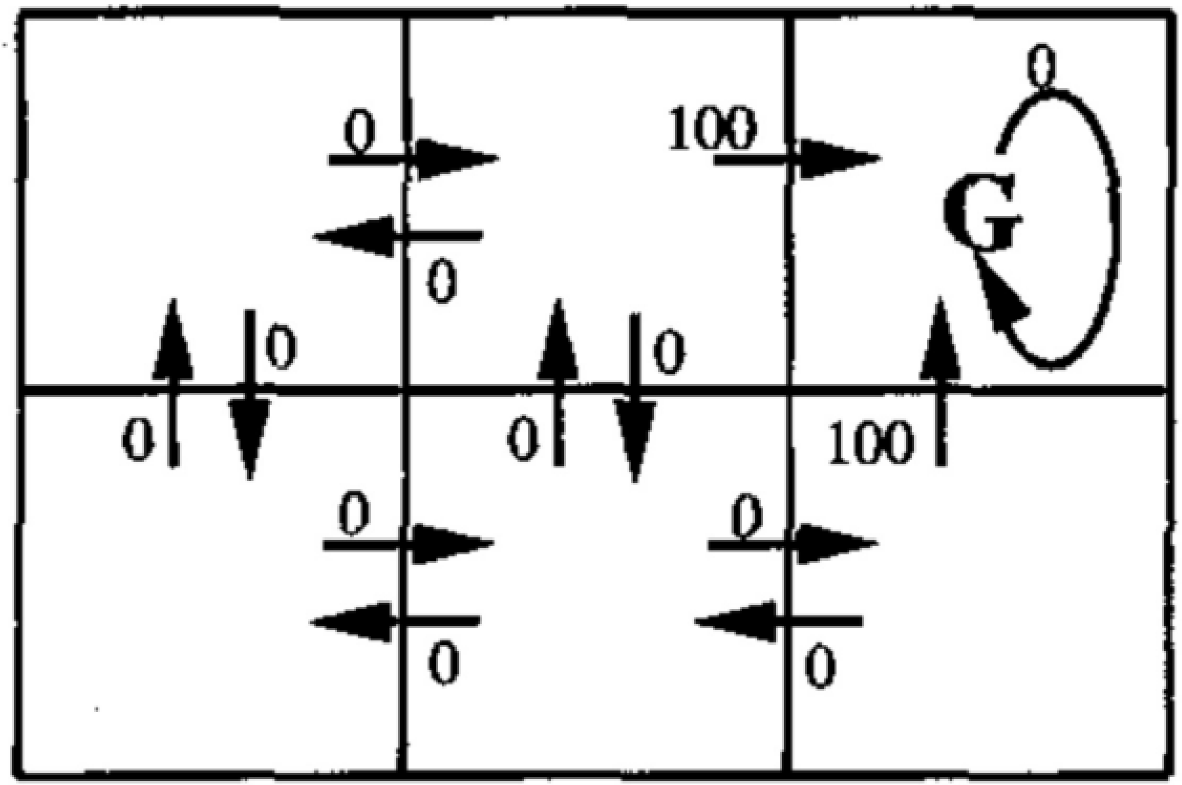

Figure 1: A 3x2 grid where the arrows represent possible action directions. Each grid represents a state and the numbers associated with the arrows represent the immediate reward received upon taking that action. Here, G is a terminal state and if the agent reaches there, the it remains there indefinitely.

Question 3.1: Consider the 3x2 rectangular grid shown in Figure (1). Choose at-least two arbitrary policies and write the corresponding value for each state. Finally, find an optimal policy and write the corresponding value for each state. You may choose the discount factor anywhere between $[0, 1)$.

Solution. Choosing two arbitrary, deterministic policies where the agent always takes the same action in each state.

- Policy 1: Always move right; assuming $\gamma = 0.9$

$$\begin{aligned} V(s_1) &= 0.9(0) + 0.9(0) + 0.9(100) &&= 90 \\ V(s_2) &= 0.9(0) + 0.9(0) + 0.9(0) &&= 0 \\ V(s_3) &= 0.9(0) + 0.9(100) + 0.9(0) &&= 90 \end{aligned}$$

- Policy 2: Alternate between moving right and moving up; assuming $\gamma = 0.9$

$$\begin{aligned} V(s_1) &= 0.9(0) + 0.9(0) + 0.9(100) &&= 90 \\ V(s_2) &= 0.9(0) + 0.9(90) + 0.9(0) &&= 81 \\ V(s_3) &= 0.9(0) + 0.9(0) + 0.9(81) &&= 72.9 \end{aligned}$$

Finding optimal policy and corresponding values for each state using Bellman optimality to find optimal value function $V*(s)$ for each state

$$\begin{aligned}
V^*(s_1) &= \max\left(0 + 0.9V(s_2), 0 + 0.9V(s_1), 100 + 0.9V(s_1)\right) && = 100 \\
V^*(s_2) &= \max\left(0 + 0.9V(s_1), 0 + 0.9V(s_3)\right) && = 90 \\
V^*(s_3) &= \max\left(0 + 0.9V(s_2), 100 + 0.9V(s_3)\right) && = 100
\end{aligned}$$

To find the optimal policy select the action that maximizes the expression in the Bellman optimality equation for each state.
The corresponding values for each state under the optimal policy are:

$$\begin{aligned}
V^*(s_1) &= 100 \\
V^*(s_2) &= 90 \\
V^*(s_3) &= 100
\end{aligned}$$

From the optimal value function, we can see that the optimal policy is to always move towards the terminal state by moving either right or up, depending on the current state. ■

Question 3.2: Consider again the 3x2 rectangular grid shown in Figure (1). Write a sequence of updates for $Q(s, a)$ for each of the state-action pairs in the grid. Show that eventually the optimal state value function $V^*(s) = max_{a'}Q(s, a')$.

Solution. Q-learning update rule:

$$Q(s, a) = Q(s, a) + \alpha[r(s, a) + \gamma \max_{a'} Q(s', a') - Q(s, a)]\alpha = 0.1, \gamma = 0.9 \tag{20}$$

Initializing all Q-values to 0,

$$\begin{aligned}
Q((1, 1), up) &= 0 \\
Q((1, 1), right) &= 0 \\
Q((2, 1), up) &= 0 \\
Q((2, 1), down) &= 0 \\
Q((2, 1), right) &= 0 \\
Q((3, 1), down) &= 0 \\
Q((3, 1), right) &= 0
\end{aligned}$$

After several iterations, the values of $Q(s, a)$ will converge to their optimal values. And since $V^*(s) = \max_{a'}(s, a')$ the optimal state value function will also be obtained by taking the maximum of $Q(s,a)$ for each state.

$$\begin{aligned}
Q(1,1,up) &= 0 + \gamma max(Q(1,2,left) \\
Q(2,1,up)) &= 0 + \gamma max(0,0) = 0 \\
Q(1,1,down) &= 0 + \gamma Q(2,1,down) = 0 + \gamma 0 = 0 \\
Q(1,1,left) &= 0 + \gamma Q(1,1,left) = 0 + \gamma 0 = 0 \\
Q(1,2,up) &= 0 + \gamma max(Q(1,1,right) \\
Q(2,2,left), Q(1,3,up)) &= 0 + \gamma max(0,0,0) = 0 \\
Q(1,2,right) &= 0 + \gamma Q(1,2,right) = 0 + \gamma 0 = 0 \\
Q(1,2,down) &= 0 + \gamma max(Q(1,1,down) \\
Q(2,2,right)) &= 0 + \gamma max(0,0) = 0 \\
Q(2,1,up) &= 0 + \gamma Q(1,1,up) = 0 + \gamma 0 = 0 \\
Q(2,1,left) &= 0 + \gamma Q(2,1,left) = 0 + \gamma 0 = 0 \\
Q(2,1,down) &= 0 + \gamma \max(Q(2,2,left) \\
Q(1,1,down), Q(2,3,up)) &= 0 + \gamma \max(0,0,0) = 0 \\
Q(2,2,up) &= 0 + \gamma max(Q(2,1,right) \\
Q(1,2,left)) &= 0 + \gamma max(0,0) = 0 \\
Q(2,2,right) &= 0 + \gamma \max(Q(2,3,left) \\
Q(1,2,right), Q(2,1,down)) &= 0 + \gamma \max(0,0,0) = 0 \\
Q(2,2,down) &= 0 + \gamma \max(Q(2,1,left) \\
Q(1,2,down)) &= 0 + \gamma \max(0,0) = 0 \\
Q(1,3,up) &= 100 + \gamma Q(1,3,up) = 100 + \gamma 0 = 100 \\
Q(2,3,up) &= 0 + \gamma Q(2,3,up) = 0 + \gamma 0 = 0
\end{aligned}$$

To show that the optimal state value function $V^*(s)$ equals the maximum action-value function over all actions a, using Bellman optimality equation for $V^*(s)$:

$$V^*(s) = \max_a \sum s' P(s'|s,a)[R(s,a,s') + \gamma V^*(s')] \tag{21}$$

where,

- $P(s'|s,a)$: probability of transitioning to state s' from state s after taking action a
- $R(s,a,s')$: reward received for transitioning from state s' to s after taking action a.
- γ : discount factor

Using Bellman optimality equation for $Q^*(s,a)$,

$$Q^*(s,a) = \sum s' P(s'|s,a)[R(s,a,s') + \gamma \max_{a'} Q^*(s',a')] \tag{22}$$

To prove $V^*(s) = Q^*(s,a)$,

$$V^*(s) = \max_a \sum s' P(s'|s,a)[R(s,a,s') + \gamma V^*(s')] \tag{23}$$

Substituting $Q^*(s,a)$ into this,

$$V^*(s) = \max_a \sum s' P(s'|s,a)[R(s,a,s') + \gamma \max_{a'} Q^*(s',a')] \tag{24}$$

$$\therefore V^*(s) = \max_a Q^*(s,a) \tag{25}$$

Hence Proved!

■

4. **Boltzmann distribution policy gradient with Q-values**
(a) Simulate a 2-arm bandit problem using the Boltzmann distribution exploration-exploitation method discussed in class, where you can take the probability to choose action A_i to be given by:

$$Pr(A_i) = \frac{e^{\frac{Q(A_i)}{T}}}{\sum_{k=1}^{2} e^{\frac{Q(A_k)}{T}}} \tag{26}$$

where, $Q(A_i)$ is the quality of action A_i with $i \in 1, 2$. Here, T is a hyper-parameter (analogous to temperature).

Solution. Assuming each action has a certain reward associated with it, which is unknown to us.
We can only sample each action to get a reward, and find the best action with the minimum number of samples.

$$P(A_i) = \frac{e^{\frac{Q(A_i)}{T}}}{e^{\frac{Q(A_1)}{T}} + e^{\frac{Q(A_2)}{T}}} \tag{27}$$

where,

- $Q(A_i)$: is the quality of action A_i
- T : hyper-parameter (analogous to temperature); controls the level of exploration v/s exploitation.

To simulate the 2-arm bandit problem,
Initialize $Q(A_i)$ to 0 and iteratively update it after each sample using:

$$Q(A_i) = Q(A_i) + \alpha(R - Q(A_i)) \tag{28}$$

where,

- α : learning rate

- R : reward from action A_i
- $Q(A_i)$: previous estimate of the quality of action A_i

■

(b) Show the if $T \to \infty$ (very large), the policy gradient method approaches pure exploration, while if $T \to 0$ (very small), the policy gradient methods approaches pure exploitation.

Solution.

$$P(A_i) = \frac{e^{\frac{Q(A_i)}{T}}}{e^{\frac{Q(A_1)}{T}} + e^{\frac{Q(A_2)}{T}}} \tag{29}$$

Analyzing the above equation as $T \to \infty$,

$$\lim_{T\to\infty} P(A_i) = \lim_{T\to\infty} \frac{e^{\frac{Q(A_i)}{T}}}{e^{\frac{Q(A_1)}{T}} + e^{\frac{Q(A_2)}{T}}} = \frac{e^0}{e^0 + e^0} = \frac{1}{2} \tag{30}$$

This means that the probability of selecting each action approaches $\frac{1}{2}$, regardless of the quality of the actions. Therefore the agent chooses actions with almost equal probabilities, without taking into account the expected rewards. This behavior corresponds to pure exploration since the agent is not exploiting the information it has about the actions' expected rewards.
Analyzing the equation for $T \to \infty$,

$$\begin{aligned} \lim_{T\to 0} e^{\frac{Q(A_i)}{T}} &= 0 && if Q(A_i) < Q(A_j), \\ &= \infty && if Q(A_i) > Q(A_j), \\ &= 1 && if Q(A_i) = Q(A_j) \end{aligned}$$

$i, j \in 1, 2$ and $i \neq j$
As T approaches 0, exponential function becomes very sensitive to differences in the Q-values.
Assuming $Q(A_1) > Q(A_2)$,
As $T \to 0$,

$$\begin{aligned} \lim_{T\to 0} P(A_1) &= \lim_{T\to 0} \frac{e^{\frac{Q(A_1)}{T}}}{e^{\frac{Q(A_1)}{T}} + e^{\frac{Q(A_2)}{T}}} \\ &= \frac{1}{1 + e^{\frac{Q(A_2) - Q(A_1)}{T}}} \\ &= 1 \end{aligned}$$

$$\begin{aligned}\lim_{T\to 0} P(A_2) &= \lim_{T\to 0} \frac{e^{\frac{Q(A_2)}{T}}}{e^{\frac{Q(A_1)}{T}} + e^{\frac{Q(A_2)}{T}}}\\ &= \frac{e^{\frac{Q(A_2)-Q(A_1)}{T}}}{1 + e^{\frac{Q(A_2)-Q(A_1)}{T}}}\\ &= 0\end{aligned}$$

$\therefore$ as $T \to 0$, and $Q(A_1) > Q(A_2)$, the probability of selecting action A_1 approaches 1 and A_2 approaches 0. This corresponds to pure exploitation since the agent always chooses the action with highest expected reward.
Similarly, if $Q(A_2) > Q(A_1)$, as $T \to 0$, the probability of A_2 approaches 1, and that of A_1 approaches 0. ■

5. **Cliff-walking simulation**
Write your own version of the Cliff-Walking simulation discussed in class (you may refer to Chapter 6 of the book [1]), using both Qlearning and SARSA. Compare the number of iterations it takes to converge to the final path for both simulations. Use any of the policies, but keep the hyper-parameters the same for both methods.

Solution. The parameters are set as follows:

- $\epsilon = 0.1$
- $\alpha = 0.5$
- $\gamma = 0.9$

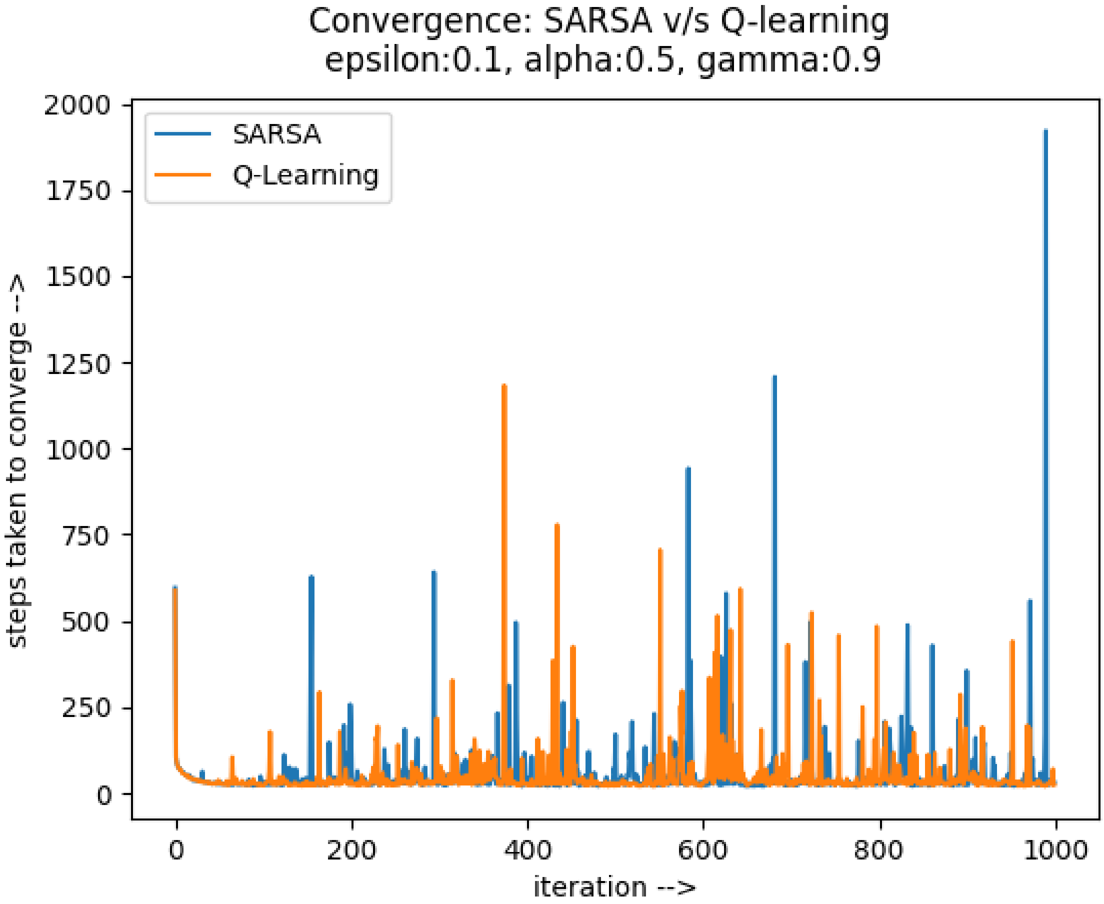

Figure 2: Graph shows the average number of steps taken to converge for both SARSA (on-policy TD control) and Q-learning (off-policy TD control) for 100 trials each consisting of 1000 episodes.
On an average Q-learning is much faster than SARSA. ■