**Experiment :- 05**

|  |
| --- |
| **Title: Write a program to implement SARSA algorithm** |

# Objective:

# To understand working of SARSA Algorithm

# To implement SARSA Algorithm

# Books/ Journals/ Websites referred:

* Markov Decision Processes in Artificial Intelligence MDPs, Beyond MDPs and Applications, Edited by Olivier Sigaud, Olivier Buffet, Wiley Publications, 2010
* https://gym.openai.com/docs/
* https://www.geeksforgeeks.org/sarsa-reinforcement-learning/

# Resources Needed:

# Python,

# Numpy,

# Gym (Gym is a toolkit for developing and comparing reinforcement learning algorithms.)

# Theory:

# SARSA algorithm is a slight variation of the popular Q-Learning algorithm. For a learning agent in any Reinforcement Learning algorithm it’s policy can be of two types:-

# 1) On Policy: In this, the learning agent learns the value function according to the current action derived from the policy currently being used.

# 2) Off Policy: In this, the learning agent learns the value function according to the action derived from another policy.

# Q-Learning technique is an Off Policy technique and uses the greedy approach to learn the Q-value. SARSA technique, on the other hand, is an On Policy and uses the action performed by the current policy to learn the Q-value.

# This difference is visible in the difference of the update statements for each technique:-

# 

# Here, the update equation for SARSA depends on the current state, current action, reward obtained, next state and next action. This observation lead to the naming of the learning technique as SARSA stands for State Action Reward State Action which symbolizes the tuple (s, a, r, s’, a’).

# Implementation (Code):

# Kindly find the code and output in E-5\_SARSA.ipynb

# Output Screenshots:

# Conclusion (Students should write in their own words):

# The SARSA (State-Action-Reward-State-Action) algorithm is a reinforcement learning algorithm that aims to learn an optimal policy for an agent in a Markov Decision Process (MDP). It is an on-policy algorithm, which means that it learns the value of the policy it is currently following.

# SARSA works by estimating the Q-values (the expected cumulative reward) of each state-action pair in the MDP. It updates these Q-values using the reward received by the agent and the estimated value of the next state-action pair. The algorithm uses an epsilon-greedy policy to balance exploration and exploitation.

# SARSA has several advantages, such as its ability to handle stochastic environments and to converge to an optimal policy with a high degree of accuracy. It is also relatively simple to implement and can be applied to a wide range of problems.

# However, SARSA also has some limitations. It can be slow to converge in large or complex MDPs, and it may not always find the global optimal policy. It also requires a significant amount of computational resources and can be sensitive to the choice of hyperparameters.

# Overall, SARSA is a useful algorithm for learning optimal policies in reinforcement learning problems, particularly in simpler and smaller MDPs. Its effectiveness can be improved by combining it with other techniques such as function approximation, eligibility traces, or experience replay.

# Applications:

# The SARSA (State-Action-Reward-State-Action) algorithm has several applications in reinforcement learning, including:

# 1) Robotics: SARSA can be used to teach a robot to navigate an environment and perform specific tasks. For example, SARSA can be used to teach a robot to find and pick up objects in a cluttered environment.

# 2) Game AI: SARSA can be used to develop intelligent agents that can play games such as chess, checkers, or Go. By learning from experience, these agents can develop strategies and tactics to beat human players or other agents.

# 3) Control systems: SARSA can be used to optimize control systems, such as those used in power grids or traffic control systems. By learning from feedback and adapting to changing conditions, these systems can be made more efficient and reliable.

# 4) Finance: SARSA can be used to develop trading strategies for financial markets. By learning from historical data and adapting to changing market conditions, these strategies can be optimized to maximize profits and minimize risk.

# 5) Healthcare: SARSA can be used to optimize treatment plans for patients with chronic diseases, such as diabetes or hypertension. By learning from patient data and adapting to individual needs, these plans can be personalized and made more effective.

# Overall, SARSA has a wide range of applications in fields such as robotics, gaming, control systems, finance, and healthcare, and is a useful tool for developing intelligent agents and optimizing complex systems.