**TITLE**

**NAME: - OPTIMAL PATH FINDER USING REINFORCEMENT LEARNING**

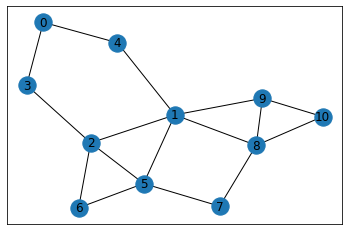
**Version: - 1.0 v**

**Date: -18/02/2020**

**INTRRODUCTION: -**

Reinforcement Learning (RL) is a machine learning technique that deals with the problems of finding the optimum actions that must be done in a given situation in order to maximize rewards.

This learning technique, which is inspired by behavioral psychology, is usually described as follows. An agent in any environment makes certain movements in this environment and gains rewards as a result of these movements. The main aim is to maximize the total reward and learn the optimal policy for the longest range.

****

# I want to find shortest path from 0 to 10. I need to attract walks to edges involving 10, therefore I give these actions high reward. In networkx library, G[node] gives all nodes which form an edge with the node.

# I set all rewards 0 except the actions arriving node 10. These actions are going from 8 to 10 or 9 to 10. Like Rewards, Q-values are initialized in a matrix. To eliminate impossible actions, their Q-values are set -100. For example, in the graph, it is not possible to go directly from 2 to 10, therefore its Q-value set as -100. Possible actions are initialized as 0.

**REQUREMENTS: -**

**Recommended system Requirements:**

1. windows, linux, Mac

2. RAM above 2 GB

3.500MB or Higher hard disk space.

**Recommended Software Requirements:**

1. Install Python
2. Install Jupyter Notebook

**CONFIGURATION: -**

Open the command prompt and use the pip command to install the following libraries.

1. Matplotlib (pip install matplotlib)
2. Network(pip install networkx )
3. Pandas (pip install pandas)
4. Numpy (pip install numpy)

**INSTALLATION: -**

Extract to short paths like C:/user or D:/user (using Winrar or zip)

Open “optimal path finder using reinforcement learning.ipynb” in Jupyter Notebook and run it.

**CODE: -**

import matplotlib.pyplot as plt

import networkx as nx

import numpy as np

import random

%matplotlib inline

def learn(er,lr,discount):

for i in range(50000):

start = np.random.randint(0,11)

next\_node=next\_number(start,er)

updateQ(start,next\_node,lr,discount)

learn(0.5,0.8,0.8)

**VERSION:  
1.0 v**

**CONTACT INFORMATION: -**

[arsalan@rapsitech.com](mailto:arsalan@rapsitech.com)