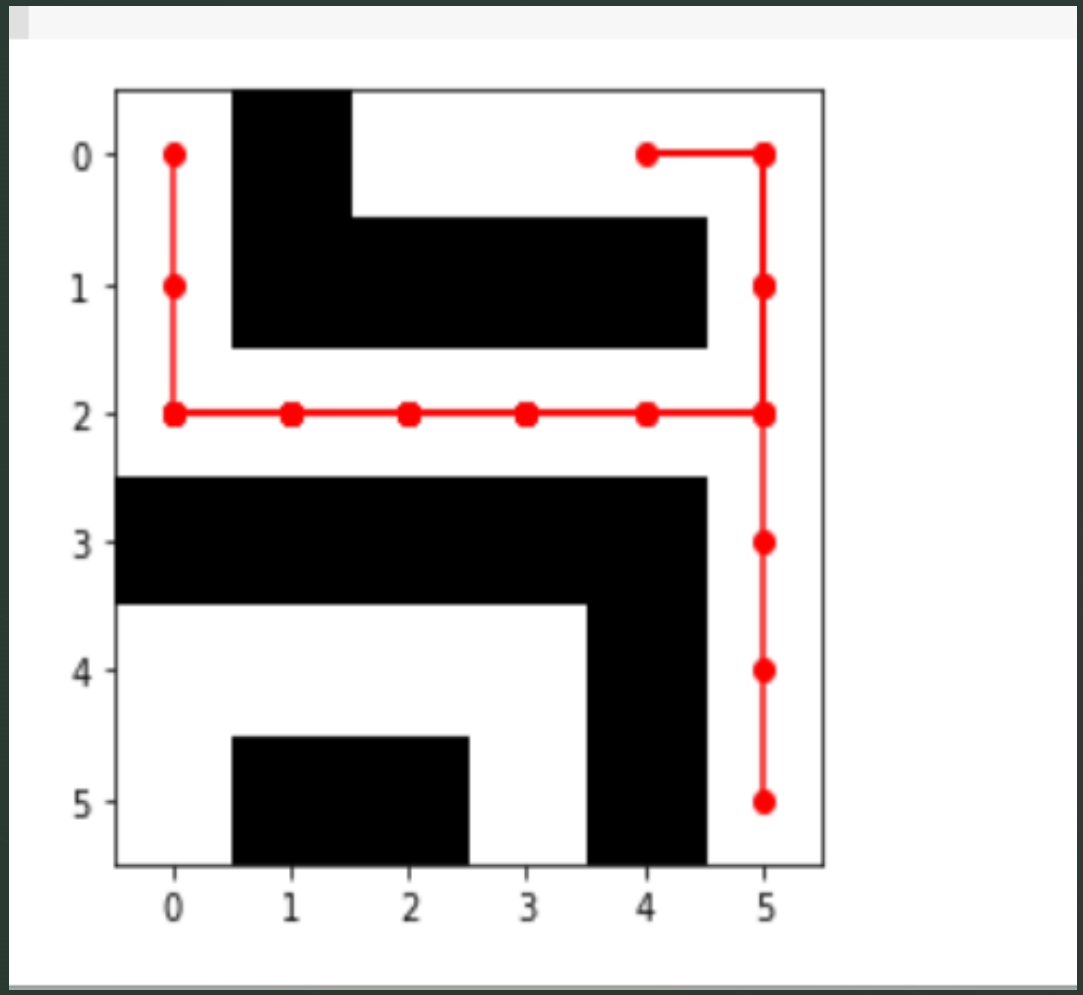
Stochastic Shortest Path

Nanditha R

CB.EN.D*R4CEN22005-FT

Stochastic Shortest Path for Maze

The Stochastic Shortest Path (SSP) algorithm is a method for finding the shortest path between two points in a maze or graph, where the path is selected randomly from the available options at each step. The algorithm works by starting at the start point and randomly selecting a neighbor as the next step, until the goal is reached. If there are no more valid neighbors, the algorithm returns **None** indicating that no path was found.



Program Code

```
    Returns:
    list: The list of valid neighbor cells.
    """
    x, y = cell
    neighbors = [(x-1, y), (x+1, y), (x, y-1), (x, y+1)]
    return [c for c in neighbors if c[0] >= 0 and c[0] < len(maze) and c[1] >= 0 and c[1] < len(maze[0]) and maze[c[0]][c[1]] == 0]

def stochastic_shortest_path(maze, start, goal):
    """Find the stochastic shortest path in a maze.

    Parameters:
    maze (list): The 2D maze representation.
    start (tuple): The starting cell coordinates.
    goal (tuple): The goal cell coordinates.

    Returns:
    list: The list of cells in the path, or None if no path was found.
    """
    path = [start]
    current = start
```

```
# Define the start and goal positions
start = (0, 0)
goal = (5, 5)

# Find the stochastic shortest path
path = stochastic_shortest_path(maze, start, goal)

# Plot the maze and path
if path:
    plt.imshow(maze, cmap='binary')
    path_x = [cell[0] for cell in path]
    path_y = [cell[1] for cell in path]
    plt.plot(path_y, path_x, 'ro-')
    plt.show()
else:
    print("No path found.")
```

- Define the maze as a 2D matrix or grid, where 0 represents a free cell and 1 represents a blocked cell.

Define the start and goal positions within the maze.

Create a function to find the neighbors of a given cell, which are the cells that are adjacent to it in the 4 cardinal directions (north, south, east, west).

Create a function that implements the SSP algorithm. It starts at the start position and selects a random neighbor from the available neighbors, until the goal is reached. If there are no more valid neighbors, the function returns none.

Call the SSP function and store the path that is returned,or display a message indicating that no path was found.

Visualize the maze and the path using a graphical plotting library, such as Matplotlib.

THANK YOU