

# Palestine Technical University (Kadoorie) Faculty of Engineering and Technology Computer Systems Engineering

**Path Pathfinding Visualization using A\* and BFS Algorithms**

**By:**

|  |  |
| --- | --- |
| **Raja’ Asad** | **201911051** |

**Project submitted in Artificial Intelligence Course Requirement for the Bachelor Degree**

**Tulkarm, Palestine**

**2023**

I

**Path Finding using A\* and BFS Algorithms:**

**Introduction:**

This program finds the shortest path from a start spot to an end spot, taking into account any barriers you specify. To use the program, choose the start and end spots and any barriers you want.

Then, press 'a' to run the A\* algorithm or 'b' to run the BFS algorithm.

**Features:**

1. If there is no path from the start spot to the end spot, the program will print "NO Path".

2. After running the algorithm, the program will print the total cost of the path.

3. For the A\* algorithm, the program will print the f function for each iteration and each neighbor.

4. To select an algorithm, press 'A' to run the A\* algorithm or 'B' to run the BFS algorithm.

**A\* Algorithm:**

The A\* algorithm is a popular search algorithm that is used in many applications, including pathfinding and robotics. The algorithm searches for the shortest path between a start node and a goal node in a graph or network.

The algorithm uses a heuristic function to estimate the distance between the current node and the goal node. The heuristic function is used to prioritize the nodes that are more likely to lead to the goal node and therefore speed up the search process.

The A\* algorithm works by exploring the graph one node at a time, starting with the start node and looking for the goal node. It uses a priority queue to keep track of the nodes that need to be explored.

At each step, the algorithm selects the node with the lowest cost function f(n) = g(n) + h(n), where g(n) is the cost of the path from the start node to the current node, and h(n) is the heuristic estimate of the cost of the path from the current node to the goal node.

If the selected node is the goal node, the algorithm terminates and returns the path from the start node to the goal node. If the selected node is not the goal node, the algorithm generates the neighboring nodes of the selected node and adds them to the priority queue.

The algorithm continues to explore the nodes in the priority queue until the goal node is found or the priority queue is empty, in which case there is no path between the start node and the goal node.

The A\* algorithm is widely used because it is efficient, optimal, and can handle large graphs. However, it may not always find the optimal solution in certain cases, especially when the heuristic function is not admissible or consistent.

**Breadth-First Search Algorithm:**

# It is a graph traversal algorithm that starts traversing the graph from the root node and explores all the neighboring nodes. It then moves to the next level of nodes and continues to explore until it reaches the end of the graph.

# The algorithm maintains a queue to keep track of the nodes to be visited. It starts by adding the root node to the queue. Then, it removes the first node

# from the queue and visits all its neighbors. If any of the neighbors have not

# been visited before, they are added to the queue. This process continues until

# the queue is empty.