**** **Bansilal Ramnath Agarwal Charitable Trust’s**

**Vishwakarma Institute of Information Technology, Pune-48**

**(An Autonomous Institute affiliated to Savitribai Phule Pune University)**

**Department of Computer Science and Engineering (Artificial Intelligence)**

**LAB SUBMISSION**

**Artificial Intelligence**

**CAUA31201**

*Submitted by:*

*Avdhoot Nilesh Pimparkar,*

*PRN: 22210178,*

*Roll Number: 381054*

*Third Year*

*Semester I Academic Year 2024-25*

**Assignment: 3**

Aim:

* 1. To understand the A\* algorithm and its principles, including how it combines features of both Dijkstra's algorithm and greedy best-first search.
  2. To learn how to represent a problem in a way that is suitable for applying the A\* algorithm, including defining states, costs and heuristics.
  3. To explore different heuristic functions and their impact on the efficiency and effectiveness of the A\* search process.
  4. To implement the A\* algorithm in a specific application, demonstrating its use in pathfinding or optimization scenarios.
  5. To evaluate the performance of the A\* algorithm compared to other search algorithms, analyzing factors like speed, accuracy and resource consumption.

Theory:

1. **Overview of the A\* Algorithm:**

The A\* algorithm is a popular and efficient pathfinding and graph traversal algorithm used in computer science and artificial intelligence. It is widely employed in various applications, including robotics, video games, and geographic information systems (GIS), for finding the shortest path from a start node to a goal node.

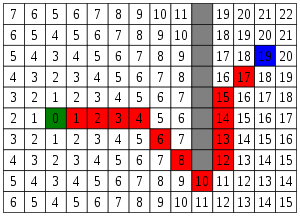


Figure 1. A\* Algorithm Pathfinding

1. **Key Components:**

The A\* algorithm relies on three main components:

* **g(n):** The cost of the path from the start node to the current node n. This value is cumulative and represents the actual cost incurred so far.
* **h(n):** The heuristic estimate of the cost from the current node n to the goal node. This is an informed guess and should ideally be admissible, meaning it never overestimates the actual cost.
* **f(n):** The total estimated cost of the cheapest solution through node n, calculated as:

This function helps prioritize which nodes to explore.

1. **Heuristics:**

Heuristics are critical in guiding the A\* algorithm efficiently towards the goal. A well-chosen heuristic can significantly reduce the number of nodes that need to be evaluated. Common heuristics include:

* **Euclidean Distance:** For applications in a 2D space, it calculates the straight-line distance between the current node and the goal.
* **Manhattan Distance:** Useful for grid-based maps, it calculates the sum of the absolute differences of the coordinates.
* **Chebyshev Distance:** Appropriate for diagonal movements, it considers the maximum of the absolute differences of the coordinates.

1. **Algorithm Steps:**

The A\* algorithm follows these steps:

1. Initialize the open list (nodes to be evaluated) with the start node and the closed list (evaluated nodes) as empty.
2. While the open list is not empty:

* Select the node n from the open list with the lowest F(n)F(n)F(n) value.
* If n is the goal node, reconstruct the path and terminate.
* Move n to the closed list.
* For each neighbour of n:
* If the neighbour is in the closed list, skip it.
* Calculate G(n), H(n) and F(n).
* If the neighbour is not in the open list, add it. If it is, check if the new path is better (lower G(n)).

1. Repeat until the goal is found or the open list is empty (no solution).
2. **Applications:**

The A\* algorithm is widely used in various applications:

* **Pathfinding in Video Games:** Navigating characters through complex maps.
* **Robotics:** Planning efficient routes for robots in dynamic environments.
* **GPS Navigation:** Finding the shortest or fastest routes between locations.

Conclusion:

The A\* algorithm is a smart and efficient way to find the shortest path in various applications, like games and navigation systems. It works by combining the cost of getting to a certain point with a guess of how much it will cost to reach the goal. This helps the algorithm decide which paths to explore first. The better the guess (or heuristic), the faster and more efficient the algorithm will be. Overall, A\* is a key tool for solving complex problems related to finding the best route.