Lab 7 – A* Search Algorithm (Improved Version)

Overview

This lab implements the A* Search Algorithm using a priority queue (heapq). A* combines actual cost g(n) and heuristic h(n) to find the optimal shortest path.

Working

1. Open list is maintained as a priority queue based on f(n)=g(n)+h(n).
2. Closed list stores visited nodes.
3. g(n) values update when a shorter path is found.
4. Parents dictionary reconstructs the final path.
5. The algorithm terminates when the destination node is reached.

Code Explanation

- Graph class stores adjacency list.
- get_neighbors() returns neighboring nodes.
- h() provides heuristic value.
- Priority queue ensures optimal node selection.
- A* algorithm loop:
* Pop node with lowest cost
* Check if destination reached
* Explore neighbors, update costs
* Push updated states back into priority queue

Importance

A* is one of the most widely used informed search strategies. It guarantees optimality when heuristic is admissible.

Conclusion

The implemented A* algorithm successfully finds the optimal path using heuristics efficiently.