RBE 550 - Programming Assignment 2

Abizer Patanwala

February 7, 2023

1 Code and Algorithm Explanation

This assignment uses Dijkstra and A* to find a path from the start node to the goal node. Both algorithms give the shortest path from the start position to the goal position

1.1 Working of Dijkstra and A*

With weighted graph G, vertex values V, edge values E, and a notable source vertex s, Dijkstra searches the graph of the goal node just like the breadth-first search. Unlike breadth-first search where neighbors are given equal importance, in Dijkstra since an edge each neighbor has different weights, it chooses the next node to explore by the minimum cost of the nodes which are not in the closed list. The closed list contains the nodes whose minimum cost from the start node has been determined. The cost of a node(cost to come) is the cost from the start node to that node via a path. The A star algorithm also works in a similar fashion except that the cost use is different. In A star the cost of a node is the summation cost to come and the heuristic value associated with that node.

For the assignment, the heuristic chosen for A star is the manhattan distance from a node to the goal node. Dijkstra and the A star algorithm terminate when they find the goal, when the open list becomes empty(suggesting the goal is an obstacle or no path exists), or when the start node itself is an obstacle. For maintaining the node attributes, a 2D list of nodes is created where each node corresponds to a grid cell. Each node's row and column are equal to the row and column of the grid cell. These nodes act as a graph for Dijkstra and A star with cells being vertices and the edges being the direction of travel from one cell to another. Here each edge weight is equal to 1. The Pseudocode of Dijkstra is shown in Figure 1 and that of A start is shown in Figure 2.

```
DIJKSTRA(G, w, s)
 1 INITIALIZE-SINGLE-SOURCE(G, s)
 2 S = Ø
 3 Q = Ø
 4 for each vertex u ∈ G.V
 5     INSERT(Q, u)
 6 while Q ≠ Ø
 7     u = EXTRACT-MIN(Q)
 8     S = S ∪ {u}
 9     for each vertex v in G.Adj[u]
10         RELAX(u, v, w)
11         if the call of RELAX decreased v.d
12             DECREASE-KEY(Q, v, v.d)
```

Figure 1: Pseudocode of Dikstra [1].

Algorithm 24 A^* Algorithm

Input: A graph

Output: A path between start and goal nodes

repeat
  Pick n_{best} from O such that $f(n_{best}) \leq f(n), \forall n \in O$.
  Remove n_{best} from O and add to C.
  If $n_{best} = q_{goal}$, EXIT.
  Expand n_{best}: for all $x \in \text{Star}(n_{best})$ that are not in C.
  if $x \notin O$ **then**
    add x to O.
  else if $g(n_{best}) + c(n_{best}, x) < g(x)$ **then**
    update x's backpointer to point to n_{best}
  end if
until O is empty

Figure 2: Pseudocode of A star [2].

1.2 Similarity and difference between BFS and DFS

Both algorithms are search techniques to find the shortest path between a source node and a goal node. They both work on directed and undirected graphs. The important differences between DFS and BFS are shown below:

- **Path Optimality**: Although A star and Dijkstra both are optimal and complete algorithms, there's a caveat associated with A star. A start only returns optimal solutions if the heuristic chosen is always less than or equal to the actual distance.
- **Speed**: A star performs a lot better in speed as compared to Dijkstra. This is because Dijkstra only looks at the cost to come, whereas The A star uses heuristics to estimate the distance of a node to the goal and then make a decision. Even though the heuristic may not be the actual value, it still provides a direction for the A star to explore nodes towards the goal and hence A star works faster.
- **Application**: If the heuristic chosen is guaranteed to be always less than the actual distance, then A star is always better than Dijkstra since it returns the optimal path in less time. But if the heuristic does not obey the condition, then A star returns the sub-optimal path, in which case either use the Dijkstra or find a better heuristic.

2 Test example, test result, and explanation

The results of Dijkstra and A star run on the map.csv are shown in Figure 3 and Figure 4 respectively. Figure 5 shows the number of steps it took to find the goal location. As expected Dijkstra takes larger steps(64) than A star(51). This is expected since Disktra just selects the minimum node based on the cost to come while A star is more guided by its heuristics and therefore Dijkstra explores more nodes as compared to A star before finding the path.

References

[1] Cormen, Thomas H., et al. Introduction to algorithms. MIT press, 2022.

[2] Choset, Howie, et al. Principles of robot motion: theory, algorithms, and implementations. MIT press, 2005.

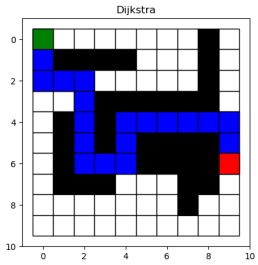

Figure 3: Path from start to goal node found with Dijkstra.

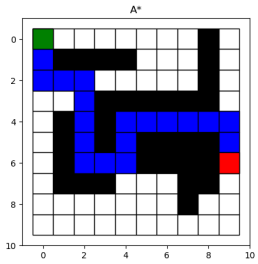

Figure 4: Path from start to goal node found with A star.

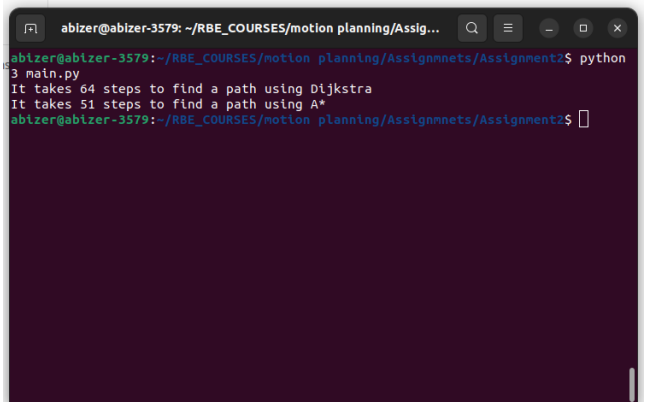

Figure 5: No of nodes traversed(steps) before reaching the goal node by Dijkstra and A star.