Minimum Cost Path Between Two Nodes

Livădaru Alexandru-Valentin

Politehnica University of Bucharest
Faculty of Automatic Control and Computer Science
livadarualex@gmail.com
Group: 323CA

Abstract. This homework is dedicated to finding the minimum cost path of a weighted graph between the source node and destination. Three algorithms were chosen to determine the minimum path: Dijkstra, Bellman-Ford and last but not least Floyd-Warshall. Finally, the performance of these algorithms will be analysed.

Keywords: Floyd-Warshall · Bellman-Ford · Dijkstra · weighted graph · shortest path

1 Description

These algorithms consist of graphs. A graph is a nodes and edges formed data structure. The edges are sometimes referred to as arcs (or lines) and the nodes are also called vertices.
Simply put, a graph is an ordered pair G = (V, E) consisting of: V which is a set of points and E a set of lines.
Graphs are used in many real-world problems such as networks. Networks can be viewed as a telephone network or a street network in a city. Moreover, in social networks like Facebook each person is regarded as a node.

2 Solutions

2.1 Floyd-Warshall

First of all, the input graph matrix is equal to the solution matrix. Taking into consideration all the intermediate vertexes we will update the solution matrix. This means that we have to pick all the vertices (one at a time) and calculate the shortest path also using the chosen node(it will play the role of an intermediary) . For example when we select the node "f", keep in mind that we already already selected the nodes {0, 1, ..j-1} as intermediate nodes.
For a source and a destination pair (j, l) we can have the following situations: if the node is not a intermediate one, we keep value(j)(l) at the current value, else value(j)(l) equals to value(j)(f) + value(f)(l).
Considering V as the number of nodes, we can say that the time complexity of this algorithm is $O(V^3)$ (for building the path matrix).

2.2 Bellman-Ford

Mainly, Bellman-Ford algorithm associates the value 0 with the distance to the source and the infinity value to all the remaining nodes. Considering all the edges, if the value of the distance to the destination can be updated to a lower one by taking an edge, then so it does. There are $|V| - 1$ existing paths and as a result we have to make the same number of tests before to ensure that we found the shortest path.
The time complexity of this algorithm is $O(VE)$.

2.3 Dijkstra

At fist, the algorithm sets all the nodes to be unvisited and creates a set of them called "unvisited set". After that, set the first node as current and assign it the value 0 and infinity to the others. Having this node, check all the unvisited notes and figure out "a tentative distance" from the current node. Compare the new "tentative" result to the old one and set the value to the smaller one. When all the unvisited notes are checked, mark the current note as visited and pop it from the "unvisited set". If the smallest distance between the current node and the destination one is infinity or if the destination node has been marked, then program is over. Differently, move to the smallest "tentative distance" node that is unvisited and calculate the "tentative distance" of this node's neighbours again and so on.

3 Evaluation Criteria

First of all, we should check if the input suits the algorithm. For example, we must have a weighted graph as an input in order to apply Floyd-Warshall algorithm and we must have no negative cost edges in order to apply Dijkstra.
We could begin testing by taking some small input examples and test them on the paper. Afterwards, we could implement this examples on the computer and then move on to bigger inputs. In this way we can trace the evolution of the result and compare with the expected output. In special cases we can also use graphics to spot the differences between the algorithms.

References

1. https://medium.com/basecs/a-gentle-introduction-to-graph-theory-77969829ead8
2. https://www.geeksforgeeks.org/graph-data-structure-and-algorithms/
3. https://www.codingame.com/playgrounds/1608/shortest-paths-with-dijkstras-algorithm/dijkstras-algorithm
4. https://brilliant.org/wiki/floyd-warshall-algorithm/
5. https://www.programiz.com/dsa/bellman-ford-algorithm
6. https://www.programiz.com/dsa/bellman-ford-algorithm
7. Thomas H. Cormen, Charles E. Leiserson, Roland L. Rivest, Cliord Stein: Introduction to Algorithms, 3rd edition, The MIT Press, Massachusetts Institute of Techology (2009).
8. https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/