FEDERAL STATE AUTONOMOUS EDUCATIONAL INSTITUTION

OF HIGHER EDUCATION

ITMO UNIVERSITY

Report

on the practical task No. 6

“Algorithms on graphs. Path search algorithms on weighted graphs”

Performed by

Pakulev Aleksandr

J4133c

Accepted by

Dr Petr Chunaev

St. Petersburg

2021

Goal:

Implement algorithms for searching way in weighted graphs: Djeicstra, Belman-Ford, A*

Task:

1. Generate a random adjacency matrix for a simple undirected weighted graph of 100 vertices and 500 edges with assigned random positive integer weights (note that the matrix should be symmetric and contain only Os and weights as elements). Use Dijkstra's and Bellman-Ford algorithms to find shortest paths between a random starting vertex and other vertices. Measure the time required to find the paths for each algorithm. Repeat the experiment 10 times for the same starting vertex and calculate the average time required for the paths search of each algorithm. Analyse the results obtained.
2. Generate a 10x10 cell grid with 30 obstacle cells. Choose two random non-obstacle cells and find a shortest path between them using A* algorithm. Repeat the experiment 5 times with different random pairs of cells. Analyse the results obtained.
3. Describe the data structures and design techniques used within the algorithms.

Brief theoretical part:

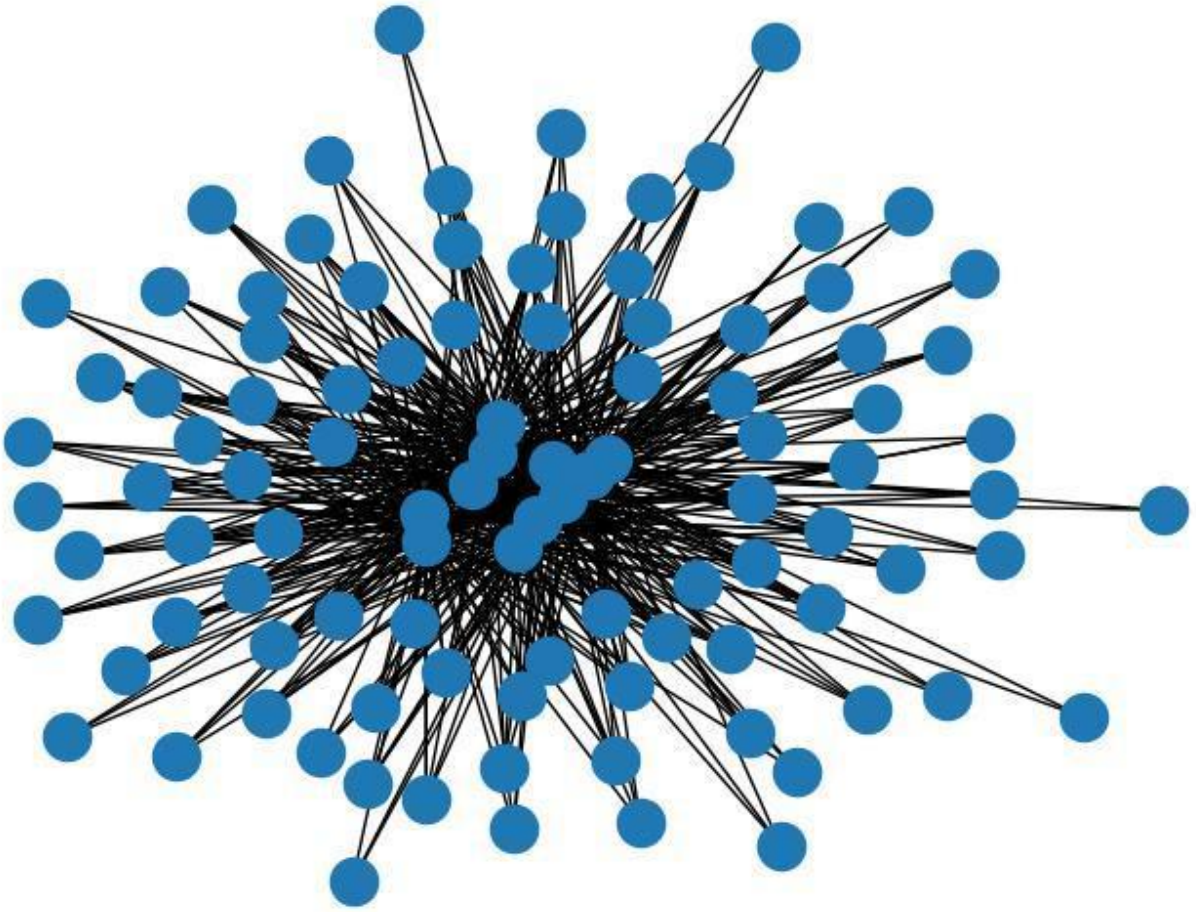
Graph – is a pair (V, E) where V is set of graph vertex and E is set of incoming and outcoming edjes from vertexes V. In this task we consider case when edjes might have weights > 0.

Here a lot of algorithms, which may solve problem of founding the shortest way from one node to all other. At the first lets consider Dijkstra algorithm. This is dynamic programming approach i.e. answer based on previous answers. Additionally we has initial state dist[from] = 0 and infinity for others. And each step we iterate over all nodes, and find node with the least distance, then we try to update weights from founded node one step ago. Time complexity is O(n * n + m), where n is nodes count and m is edges count.

Belman-Ford algorithms the same found the shortes way from one vertex to all others. At the first step we initialize initial node dist[from]=0, and then do n iteration at each iteration we iterate over all edges and try to update dist array. Time complexity O(n * m), where n is nodes count and m is edges count.

A-star algorithm the same as all above allows to find the shortest paths from one vertex to certain another one. At each iteration, A* try to determine how to extend the path basing on the cost of the current path from the initial veretex an estimate of the cost required to extend the path to the target vertex.

Result:



Generated image of graph with 100 vertex and 500 edges.

Then I launch djeicstra and belmand-ford algorithms from node 1 and receive results:

0->91

1->0

2->25

3->78

4->70

5->46

6->74

7->66

8 -> 144

9->54

10 -> 113

11 ->
200
12 ->
201
13 ->
162
14 -> 86

15 ->
176
16 ->
192
17 -> 58

18 -> 32

19 ->
172
20 ->
193
…..

…
.
90 -> 77

91 -> 21

92 -> 56

93 ->
179
94 ->
103
95 ->
158
96 ->
240
97 ->
142
98 -> 70

99 ->
108

Then I measure execution time:

Djeicstra algorithm: nanos 1600000.0

Belman-ford algorithm: nanos 92100000.0

As you can see the first algorithm working faster then the second, because has less time complexity

Then I began to implement A*, for this at the first I generate matrix 10x10 with some blocks, and launch received algorithm.

Start cell = (0, 0)

End cell=(7,6),

Generated map:

[[0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 1. 0. 0. 0. 1.]

[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

[1. 1. 1. 0. 0. 0. 0. 0. 1. 0.]

[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

[0. 0. 0. 0. 1. 1. 1. 0. 0. 0.]

[0. 0. 0. 0. 0. 1. 0. 0. 1. 0.]

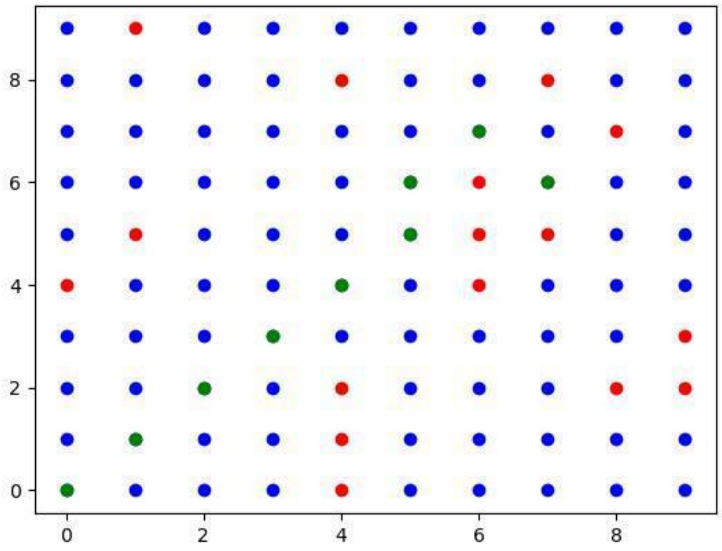
[0. 0. 1. 0. 0. 0. 0. 1. 0. 0.]

[0. 0. 1. 1. 0. 0. 0. 0. 0. 0.]]

Founded way:

[(0, 0), (1, 1), (2, 2), (3, 3), (4, 4), (5, 5), (5, 6), (6, 7), (7, 6)]

And visualize way with map:



Besides I measure execution time in: nanos 1594000.0

Conclusion:

In this task I implemented graph data strucute. Graph has a widely usage in a lot of spheres of life. Additionally I implement warious algorithms on graph for founding the shortest way between start node and end node.