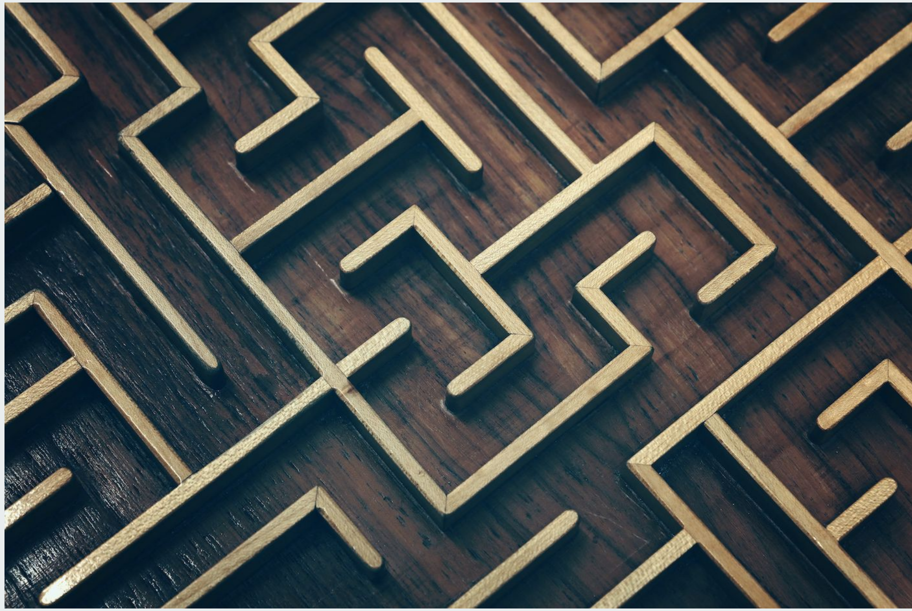
Maze Project

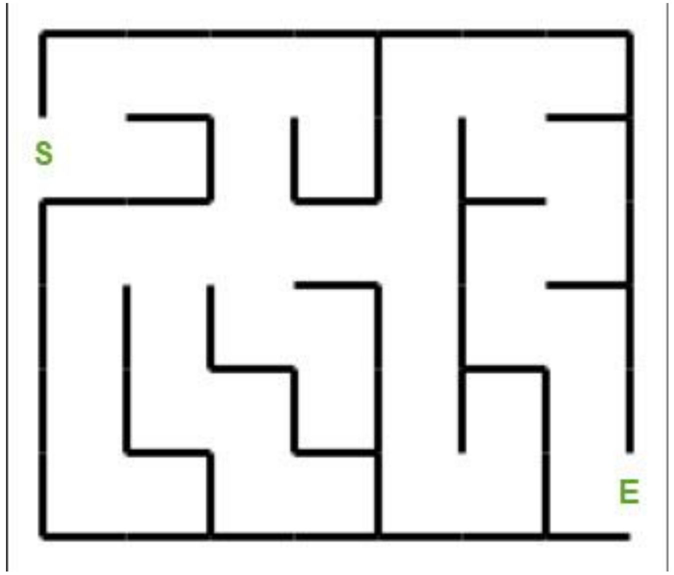
Dan Liu
March 2021



Subjects

Introduction

Challenge: How to find the shortest path of the the following maze:



Shortest Paths

Two ways to solve single-source shortest path problem:

1. Dijkstra's Algorithm

On a directed weighted graph *G = (V, E)*, where all the edges are non-negative.

The time complexity of **Dijkstra's algorithm** dependents on the implementation of extract-minimum. The simplest version stores vertices as array or linked list, and use a linear search through all vertices. The running time is **$O(|E| + |V|^2)$**, $|E|$ is the number of edges and $|V|$ is the number of vertices. This algorithm can be implemented more efficiently using min-heap to implement extract- minimum function.

Two ways to solve single-source shortest path problem:

2. Bellman Ford Algorithm

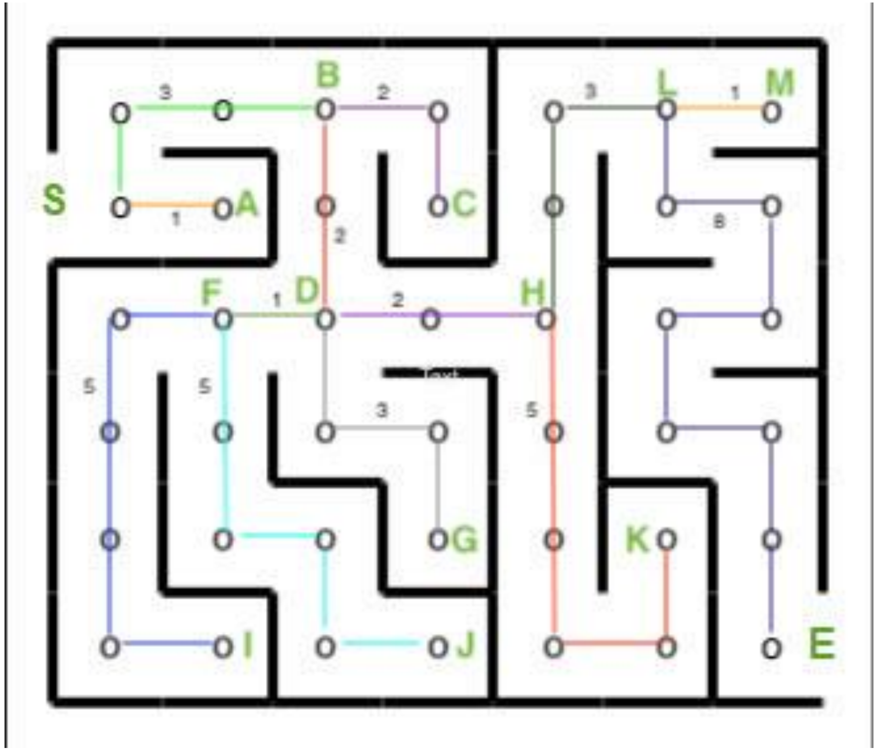
On a directed graph *G = (V, E)*, where the edge weights may be negative.

Bellman-Ford algorithm relaxes all the edges and runs |V - 1| times, the time complexity of this algorithm is **O(|V| . |E|)**. |E| is the number of edges and |V| is the number of vertices.

Implementation

Using Dijkstra’s Algorithm

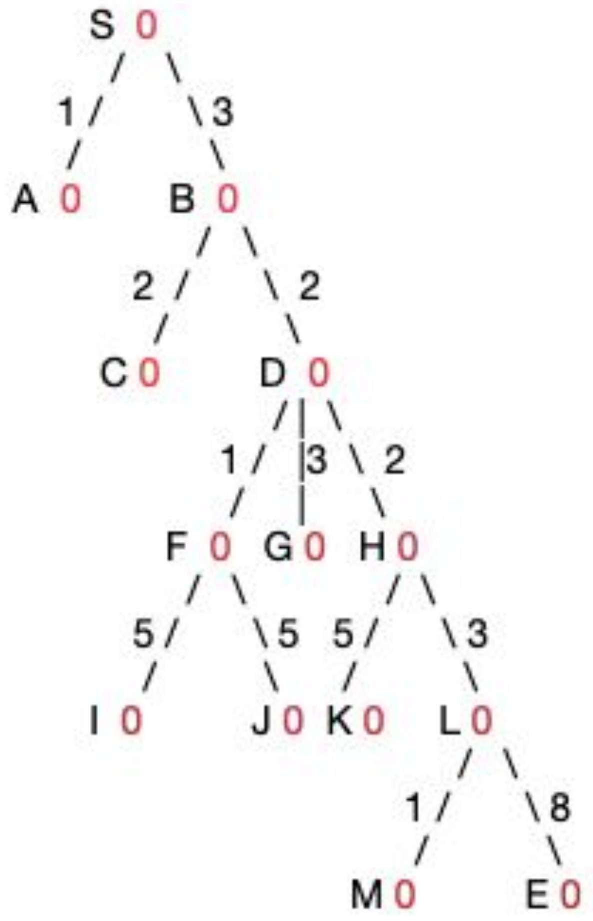
Step 1:
Draw the maze route, mark the number of edges on each route and label nodes with alphabet.



Using Dijkstra's Algorithm

Step 2:

1. Draw the tree representation of the maze;
2. Label each node of the tree sequentially with alphabets;
3. Mark each edge with a number indicating the distance.



Using Dijkstra's Algorithm

Step 3:
Using Dijkstra's Algorithm to find the minimum distance of E from S.

The shortest path from S to E is:
S -> B -> D -> H -> L -> E

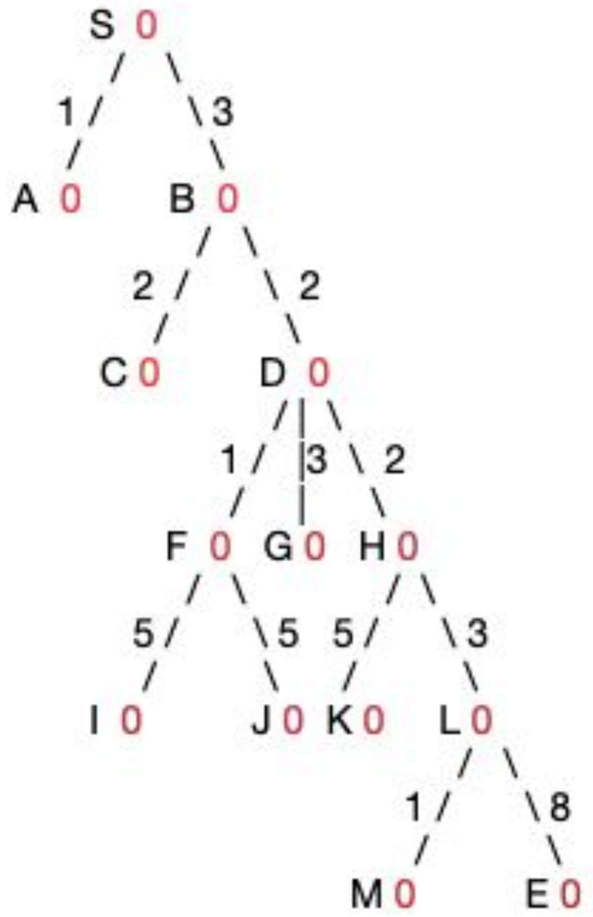
Total distance is 18.

Vertex	Initial	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Step 7	Step 8	Step 9	Step 10	Step 11	Step 12	Step 13
		S	A	B	C	D	F	H	G	L	I	J	M	K
	Next	Next	Next	Next	Next	Next	Next	Next	Next	Next	Next	Next	Next	End
	S	A	B	C	D	F	H	G	L	I	J	M	K	E
S	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A	∞	1	1	1	1	1	1	1	1	1	1	1	1	1
B	∞	3	3	3	3	3	3	3	3	3	3	3	3	3
C	∞	∞	∞	5	5	5	5	5	5	5	5	5	5	5
D	∞	∞	∞	5	5	5	5	5	5	5	5	5	5	5
F	∞	∞	∞	∞	∞	6	6	6	6	6	6	6	6	6
G	∞	∞	∞	∞	∞	8	8	8	8	8	8	8	8	8
H	∞	∞	∞	∞	∞	7	7	7	7	7	7	7	7	7
I	∞	∞	∞	∞	∞	∞	11	11	11	11	11	11	11	11
J	∞	∞	∞	∞	∞	∞	11	11	11	11	11	11	11	11
K	∞	∞	∞	∞	∞	∞	∞	12	12	12	12	12	12	12
L	∞	∞	∞	∞	∞	∞	∞	10	10	10	10	10	10	10
M	∞	∞	∞	∞	∞	∞	∞	∞	∞	11	11	11	11	11
E	∞	∞	∞	∞	∞	∞	∞	∞	∞	18	18	18	18	18

Using Bellman Ford Algorithm

Step 1:

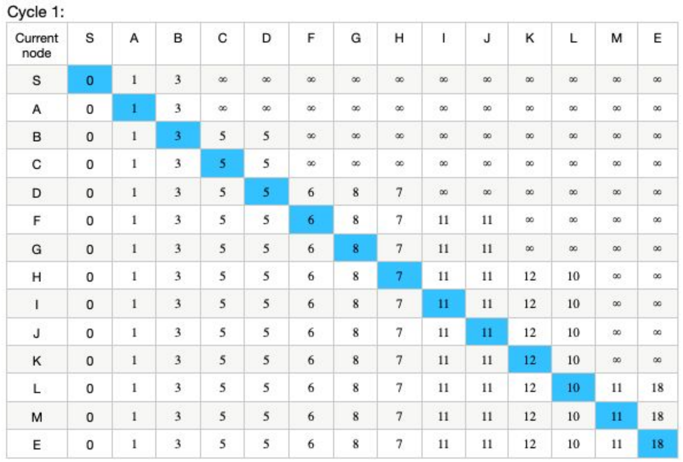
1. Use the same tree representation of the maze;
2. Get started: 14 vertices = 13 iterations



Using Bellman Ford Algorithm

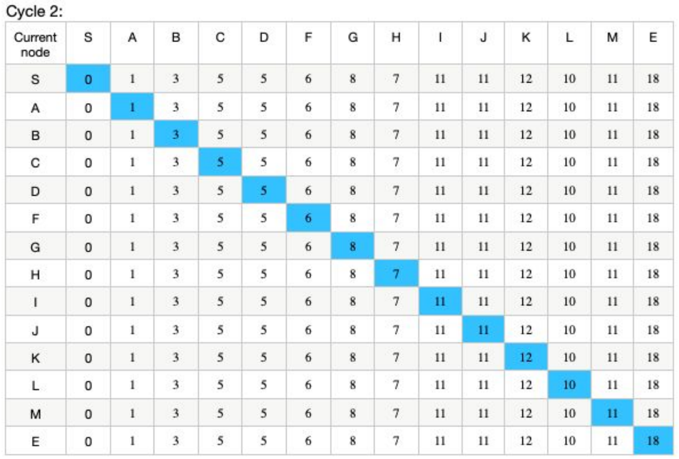
Step 2:

1. Iterate every vertice in the tree in each cycle;
2. The process ends at cycle 2 when none of the vertices is changed.



Cycle 1:

Current node	S	A	B	C	D	F	G	H	I	J	K	L	M	E
S	0	1	3	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
A	0	1	3	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
B	0	1	3	5	5	∞	∞	∞	∞	∞	∞	∞	∞	∞
C	0	1	3	5	5	∞	∞	∞	∞	∞	∞	∞	∞	∞
D	0	1	3	5	5	6	8	7	∞	∞	∞	∞	∞	∞
F	0	1	3	5	5	6	8	7	11	11	∞	∞	∞	∞
G	0	1	3	5	5	6	8	7	11	11	∞	∞	∞	∞
H	0	1	3	5	5	6	8	7	11	11	12	10	∞	∞
I	0	1	3	5	5	6	8	7	11	11	12	10	∞	∞
J	0	1	3	5	5	6	8	7	11	11	12	10	∞	∞
K	0	1	3	5	5	6	8	7	11	11	12	10	∞	∞
L	0	1	3	5	5	6	8	7	11	11	12	10	11	18
M	0	1	3	5	5	6	8	7	11	11	12	10	11	18
E	0	1	3	5	5	6	8	7	11	11	12	10	11	18



Cycle 2:

Current node	S	A	B	C	D	F	G	H	I	J	K	L	M	E
S	0	1	3	5	5	6	8	7	11	11	12	10	11	18
A	0	1	3	5	5	6	8	7	11	11	12	10	11	18
B	0	1	3	5	5	6	8	7	11	11	12	10	11	18
C	0	1	3	5	5	6	8	7	11	11	12	10	11	18
D	0	1	3	5	5	6	8	7	11	11	12	10	11	18
F	0	1	3	5	5	6	8	7	11	11	12	10	11	18
G	0	1	3	5	5	6	8	7	11	11	12	10	11	18
H	0	1	3	5	5	6	8	7	11	11	12	10	11	18
I	0	1	3	5	5	6	8	7	11	11	12	10	11	18
J	0	1	3	5	5	6	8	7	11	11	12	10	11	18
K	0	1	3	5	5	6	8	7	11	11	12	10	11	18
L	0	1	3	5	5	6	8	7	11	11	12	10	11	18
M	0	1	3	5	5	6	8	7	11	11	12	10	11	18
E	0	1	3	5	5	6	8	7	11	11	12	10	11	18

Enhancement Ideas

Bellman Ford Algorithm can be used to find shortest paths when there is no any negative weighted cycle. Besides, it can be applied in some applications since it may have negative edges. For example:

1. Imagine a logistic network where the weight w(i, j) of an edge i, j is the cost from vertex i to vertex j. In the situation of a business transportation with other companies, w(i, j) is a profit, you can interpret the weight as a negative cost.
2. Represent the speed driving from one place to another. The speed above average is positive, while below average is negative.

Conclusion

1. Dijkstra’s Algorithm and Bellman Ford Algorithm both can be used to find shortest paths.
2. The time complexity of Dijkstra’s Algorithm dependents on the implementation of extract-minimum. The running time of simplest version is O(|E| + |V|2).
   The time complexity of Bellman Ford Algorithm is O(|V| . |E|).
3. Bellman Ford Algorithm can have other applications because of its property of negative edge.

References

References

1. https://npu85.npu.edu/~henry/npu/classes/algorithm/graph_alg/slide/maze.html
2. https://npu85.npu.edu/~henry/npu/classes/algorithm/tutorialpoints_daa/slide/shortest_paths.html
3. https://www.cs.uah.edu/~rcoleman/CS221/Graphs/ShortestPath.html
4. https://npu85.npu.edu/~henry/npu/classes/algorithm/tutorialpoints_daa/slide/bf.html