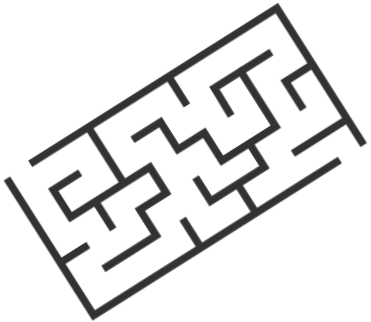
Bellman Ford's Algorithm to find the shortest path of the maze



Shoumya Singh

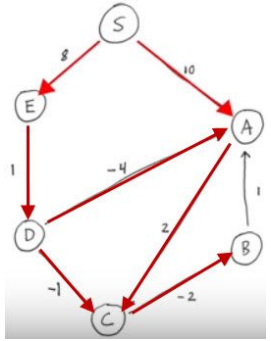


Table of Content

Richard Bellman and Lester Ford Jr.

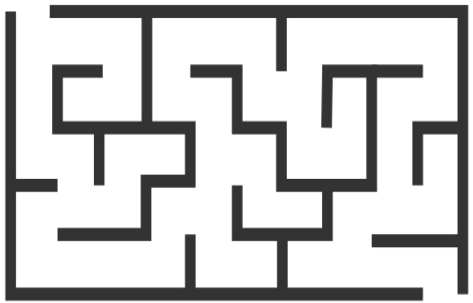
- **Richard Ernest Bellman** was an American applied mathematician, who introduced dynamic programming in 1953, and made important contributions in other fields of mathematics.
- **Lester Randolph Ford Jr.** was an American mathematician specializing in network flow problems.
- In 1956, Ford developed the Bellman–Ford algorithm for finding shortest paths in graphs that have negative weights,two years before Richard Bellman also published the algorithm.
- The algorithm was first proposed by Alfonso Shimbel (1955), but is instead named after Richard Bellman and Lester Ford Jr., who published it in 1958 and 1956, respectively.

Introduction

- This algorithm solves the single source shortest path problem of a directed graph ***G = (V, E)*** in which the edge weights may be negative.

- The Bellman-Ford algorithm is a graph search algorithm that finds the shortest path between a given source vertex and all other vertices in the graph. This algorithm can be used on both weighted and unweighted graphs.

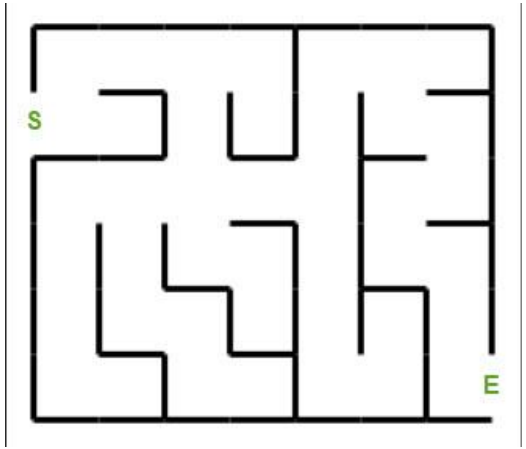
What is a Maze?



- A **maze** is a path or collection of paths, typically from an entrance to a goal.
- A maze can be viewed as a graph, if we consider each intersection in the maze to be a vertex, and we add edges to the graph between adjacent junctures that are not blocked by a wall.
- Our mazes will have random weights specified between any two adjacent intersection. These values can be thought of as the "cost" of traveling from one intersection to an adjacent one. The weights will be used for running Dijkstra's algorithm.

Table of Content

Design (Problem Statement)



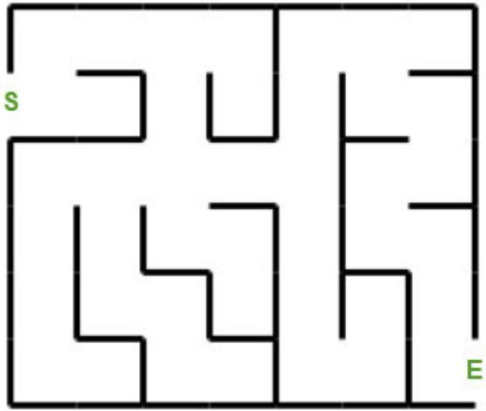
- Use Bellman Ford's Algorithm to find the shortest path of the following maze .
- Convert the maze into a weighted Graph.
- Each node of the tree representation of the maze should be labeled sequentially and each edge should have a number indicating the distance.
- Apply the Bellman Ford's Algorithm to find the shortest path in the weighted graph.

Table of Content

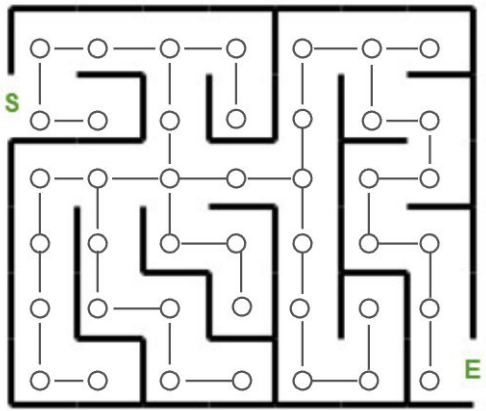
Implementation (Convert Maze to Weighted Graph)

- A maze can be viewed as a graph, if we consider each juncture (intersection) in the maze to be a vertex, and we add edges to the graph between adjacent junctures that are not blocked by a wall.

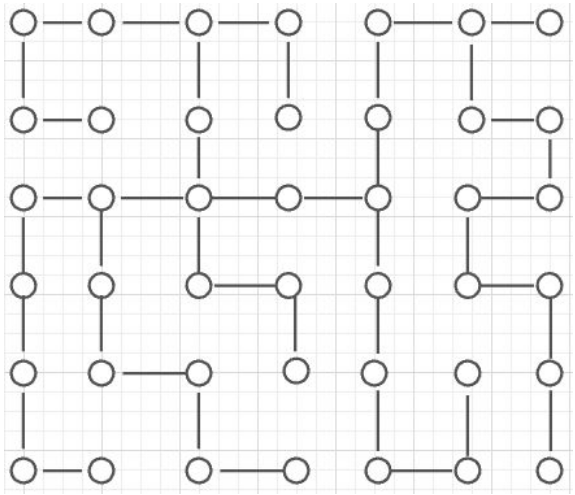
Step 1



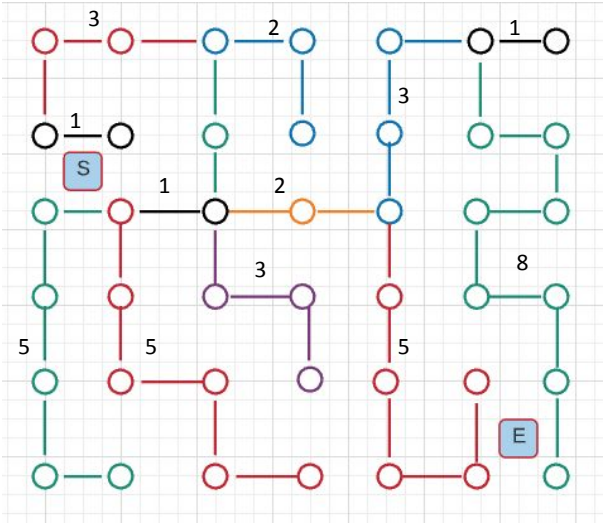
Step 2



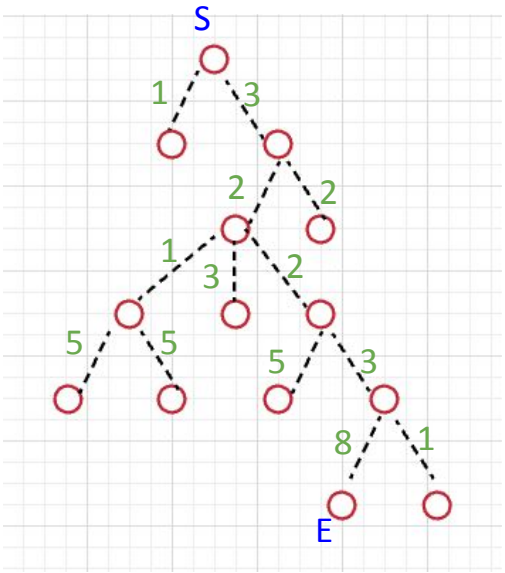
Step 3

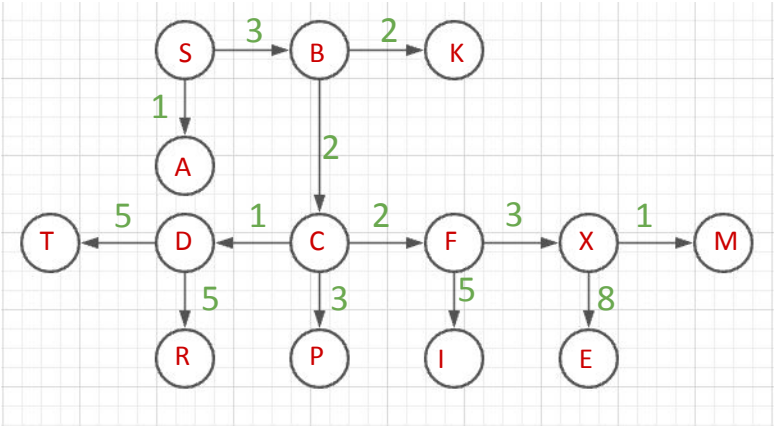


Step 4



Step 5



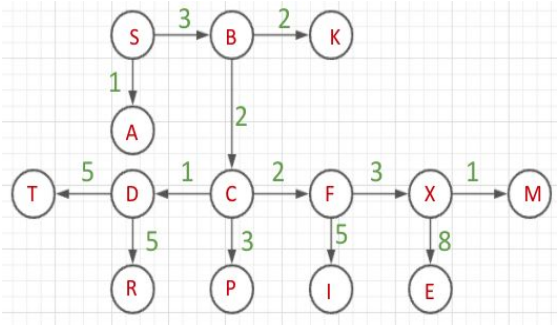


Apply Bellman Ford's Algorithm to Weighted Graph

Step 1: Initialization

Next node to visit => B

S	A	B	C	K	P	D	F	R	T	I	X	M	E
0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞



Step 2: 0+1 < ∞. Change the value of A to 1
0+3 < ∞. Change the value of B to 3

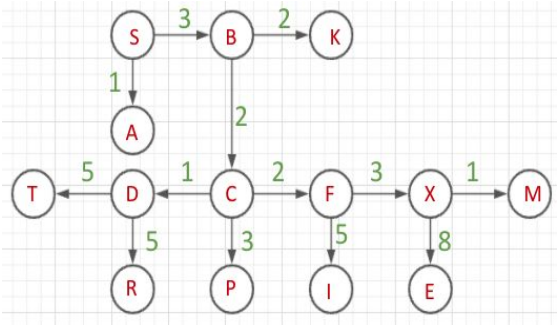
Next node to visit => B

S	A	B	C	K	P	D	F	R	T	I	X	M	E
0	1	3	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞

Step 3: 3+2=5 < ∞. Change the value of k to 5
3+2=5 < ∞. Change the value of C to 5

Next node to visit => C

S	A	B	C	K	P	D	F	R	T	I	X	M	E
0	1	3	5	5	∞	∞	∞	∞	∞	∞	∞	∞	∞



Step 4: 5+1=6 < ∞. Change the value of D to 6
5+2=7 < ∞. Change the value of F to 7
5+3=8 < ∞. Change the value of P to 8

Next node to visit => D

S	A	B	C	K	P	D	F	R	T	I	X	M	E
0	1	3	5	5	8	6	7	∞	∞	∞	∞	∞	∞

Step 5: 5+6=11 < ∞. Change the value of T to 11
5+6=11 < ∞. Change the value of R to 11

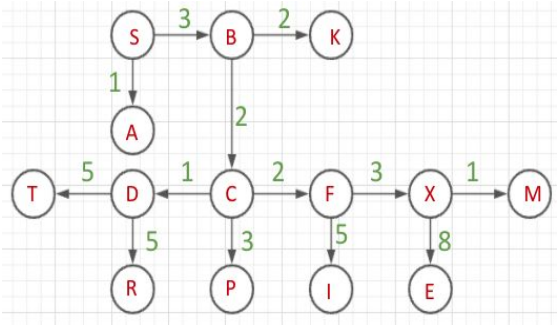
Next node to visit => F

S	A	B	C	K	P	D	F	R	T	I	X	M	E
0	1	3	5	5	8	6	7	11	11	∞	∞	∞	∞

Step 6: 7+5=12 < ∞. Change the value of I to 12
7+3=10 < ∞. Change the value of X to 10

Next node to visit => X

S	A	B	C	K	P	D	F	R	T	I	X	M	E
0	1	3	5	5	8	6	7	11	11	12	10	∞	∞



Step 7: 10+1=11 < ∞. Change the value of M to 11
10+8=18 < ∞. Change the value of E to 18

S	A	B	C	K	P	D	F	R	T	I	X	M	E
0	1	3	5	5	8	6	7	11	11	12	10	11	18

The process ends at cycle one as there are no vertices to change.

Hence, the minimum distance between vertex S and vertex E is 18.

Table of Content

Conclusion

- **Bellman-Ford** algorithm is a single-source shortest path algorithm, so when we have negative edge weight then it can detect negative cycles in a graph.

- The only difference between the two is that **Bellman-Ford** is also capable of handling negative weights whereas **Dijkstra** Algorithm can only handle positives.

- The complexity of **Bellman-Ford** algorithm with respect to time is slower **than** the algorithm of **Dijkstra**.

- The first **for** loop is used for initialization, which runs in ***O(V)*** times. The next **for** loop runs ***|V - 1|*** passes over the edges, which takes ***O(E)*** times.Hence, Bellman-Ford algorithm runs in ***O(V, E)*** time.

- The complexity of this algorithm is fully dependent on the implementation of Extract-Min function. If extract min function is implemented using linear search, the complexity of this algorithm is $O(V^2 + E)$.

Bibliography / References

1. https://en.wikipedia.org/wiki/Bellman%E2%80%93Ford_algorithm
2. https://npu85.npu.edu/~henry/npu/classes/algorithm/tutorialpoints_daa/slide/shortest_paths.html
3. https://www.lucidchart.com/pages/
4. http://www.cs.umd.edu/class/spring2019/cmsc132-020X-040X/Project8/proj8.html
5. https://en.wikipedia.org/wiki/Maze#Solving_mazes
6. https://brilliant.org/wiki/bellman-ford-algorithm/#:~:text=The%20Bellman%2DFord%20algorithm%20is,both%20weighted%20and%20unweighted%20graphs.
7. https://www.geeksforgeeks.org/applications-of-dijkstras-shortest-path-algorithm/