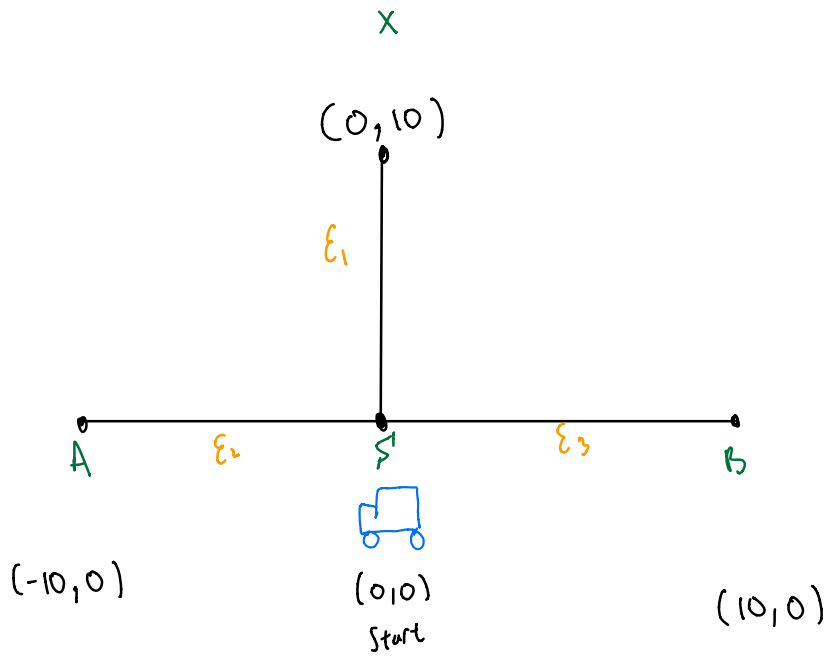
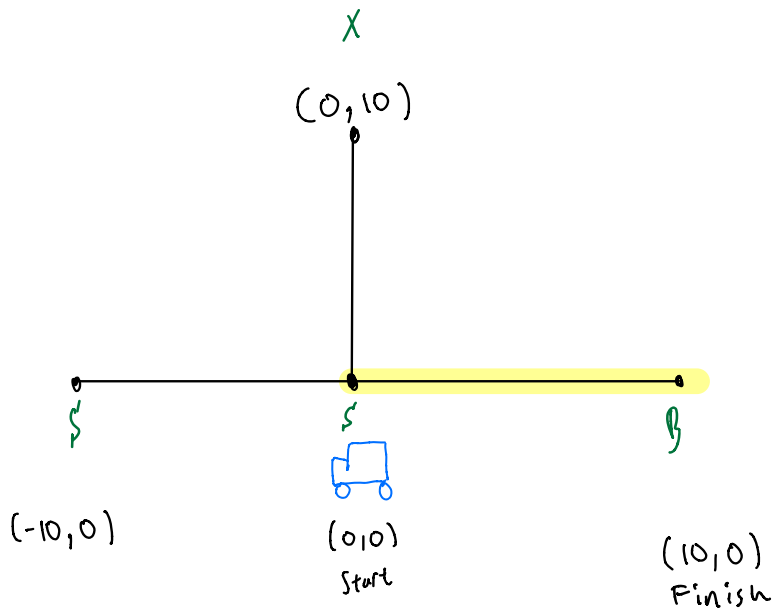

X
(0,10)
ε1
A
ε2
S
ε3
B
(-10,0)
(0,0)
Start
(10,0)

Goal

Is to not make the route manually but to type in Desetinations and have some algorithm create the route for us.

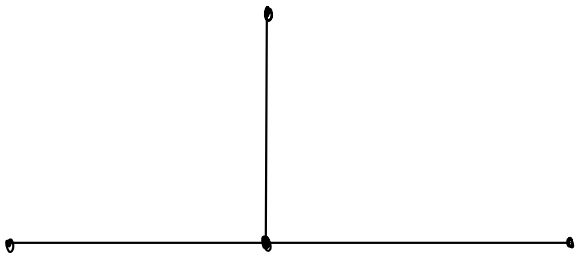
Lets call the Algorithm "travel"
Lets say we want to go from S to B

travel('S', 'B') would be the function call

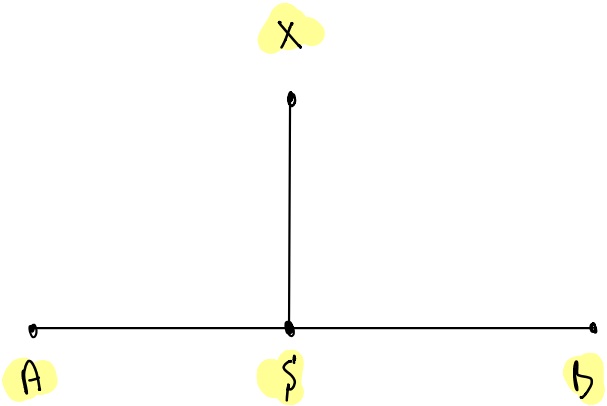


Graph, Vertices, edges

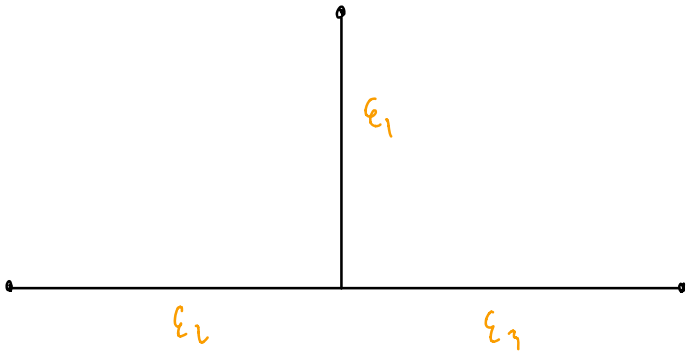
Graph



Vertices



Edges



Test on nodes vs edges

Graph

Node 3 { name = "A", edges = [e1, e2]
position (10, 10), visited (false) }

Edge 4 { (A, B) }

Graph

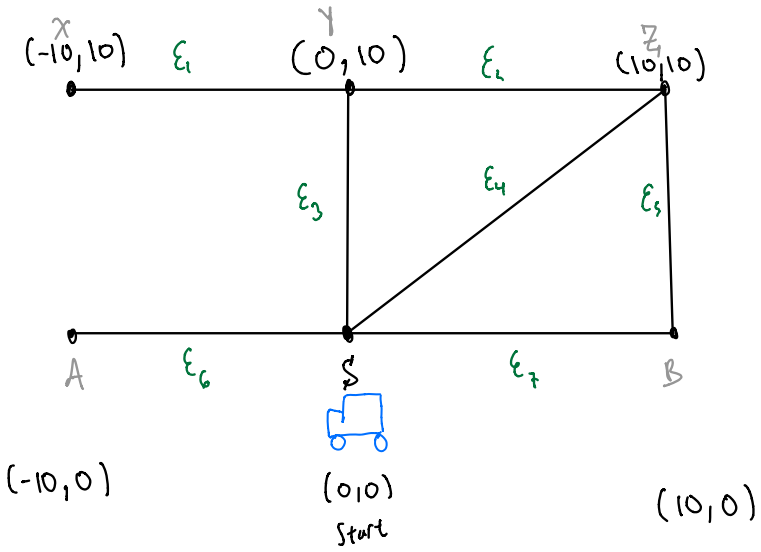
node 1 [edges]
node 2 [edges]
:
node n

edge (node 1, node 2)

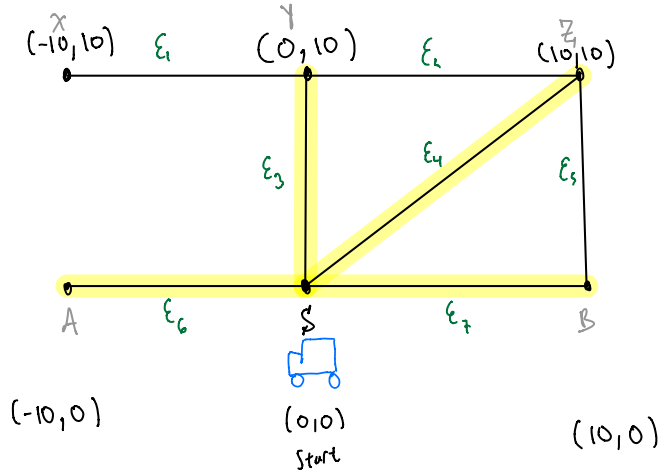
BFS Search

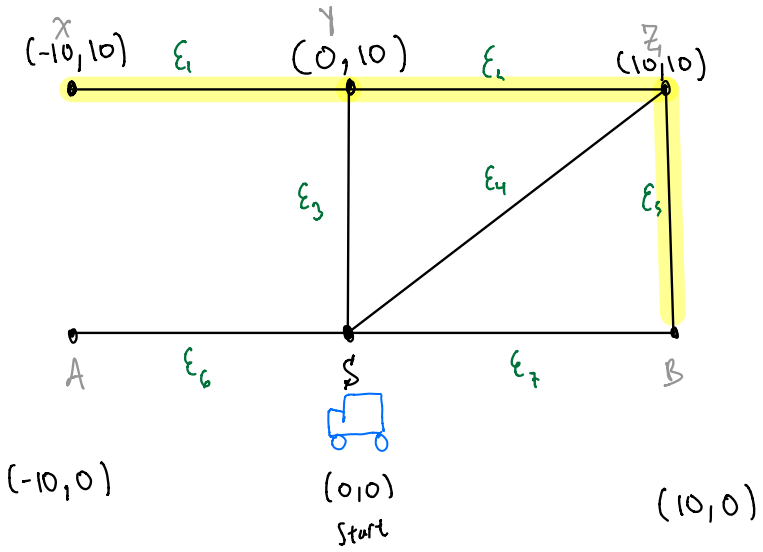
travel (S, X)

Nodes discovered on iteration 1



Nodes discovered on iteration 1





found X on iteration 2

Queue

$$q = [(currPath, currVertex)]$$

```
BFS ShortestPath (startVertex, destinationVertex):

    queue ⟵ empty
    add None & startVertex to queue
    while (q is not empty):
        (currPath, currVertex) = queque.deque()
        mark currVertex as visited
        if currVertex == destinationVertex
            return currPath
        for all outgoingEdges from currVertex to Neighbor:
            if Neighbor has not been visited:
                add (currPath.append(Edge, Vertex), currVertex)
    return "Fail"
```