Minimum Spanning Tree

- A MST is a tree consisting of a minimum number of edges required to connect all vertices.
- A MST gives the shortest path between vertices.
- A DFS creates a MST.

Topological Sort

- An ordering of the vertices in a directed graph so that if there is a edge from A to B, A appears before B.
- The algorithm:
  1. Find a vertex that has no successor
  2. Deleted this vertex from the graph and insert its label at the beginning of the list
  3. Repeat 1 and 2 until graph is empty

WMST Algorithm

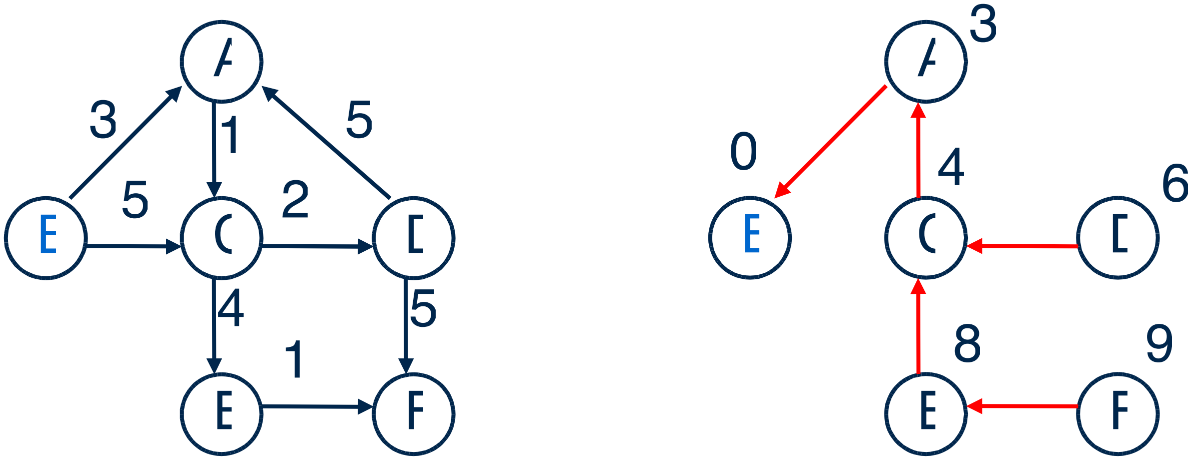
- Algorithm:
  - Start with a vertex, put it in the WMST.
  - Find all the edges from the newest vertex to other vertices that aren’t in the WMST. Put these edges in a priority queue based on minimum of weight.
  - Edges between vertices already in the WMST are removed
  - Pick the edge with the lowest weight,, add this edge and its destination to the WMST.
- Time Complexity??

Shortest-path

- Suppose we want to find the shortest path from node X to node Y
- It turns out that, in order to do this, we need to find the shortest path from X to all other nodes
  - Why?
  - If we don't know the shortest path from X to Z, we might overlook a shorter path from X to Y that contains Z
- Dijkstra's Algorithm finds the shortest path from a given node to all other reachable nodes

Dijkstra’s algorithm I

- Dijkstra’s algorithm builds up a *tree:* there is a path from each node back to the starting node



- Edge values in the graph are weights
- Node values in the tree are *total* weights

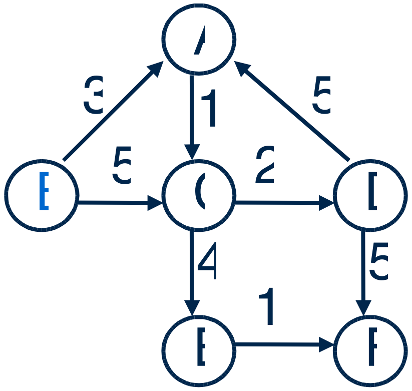
Dijkstra’s algorithm II

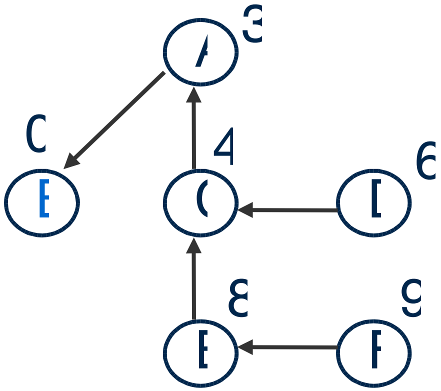
- For each vertex v, Dijkstra’s algorithm keeps track of three pieces of information:
    - A boolean telling whether we *know* the shortest path to that node (initially true only for the starting node)
    - The length of the shortest path to that node known so far (0 for the starting node)
    - The predecessor of that node along the shortest known path (unknown for all nodes)

Dijkstra’s algorithm III

- Dijkstra’s algorithm proceeds in phases—at each step:
  - From the vertices for which we don’t know the shortest path, pick a vertex v with the smallest distance known so far
  - Set v’s “known” field to true
  - For each vertex w adjacent to v, test whether its distance so far is greater than v’s distance plus the distance from v to w; if so, set w’s distance to the new distance and w’s predecessor to v

Dijkstra’s algorithm III





node	init’ly	1	2	3	4	5	6
A	inf	3B	+3B	+3B	+3B	+3B	+3B
B	0-	+0-	+0-	+0-	+0-	+0-	+0-
C	inf	5B	4A	+4A	+4A	+4A	+4A
D	inf	inf	inf	6C	+6C	+6C	+6C
E	inf	inf	inf	8C	8C	+8C	+8C
F	inf	inf	inf	inf	11D	9E	+9E