Minimum Spanning Tree Problem

Given a weighted, undirected graph, find the tree with least-total weight that connects all the vertices, if one exists.

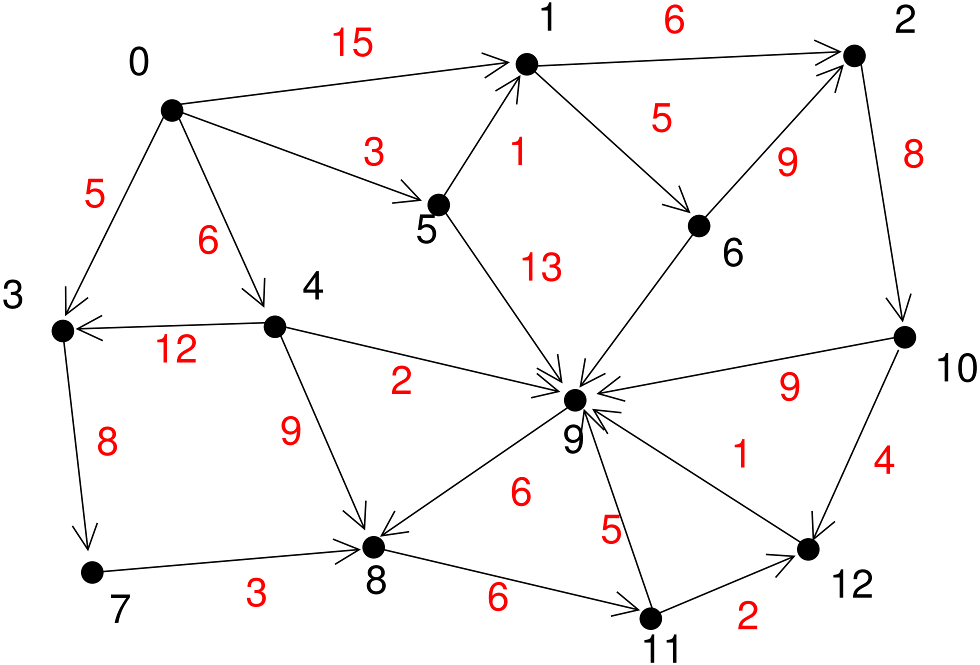
Single-Source Shortest Paths Problem

Given a weighted directed graph and a source vertex, find the tree comprising the shortest paths from that source to all other reachable vertices.

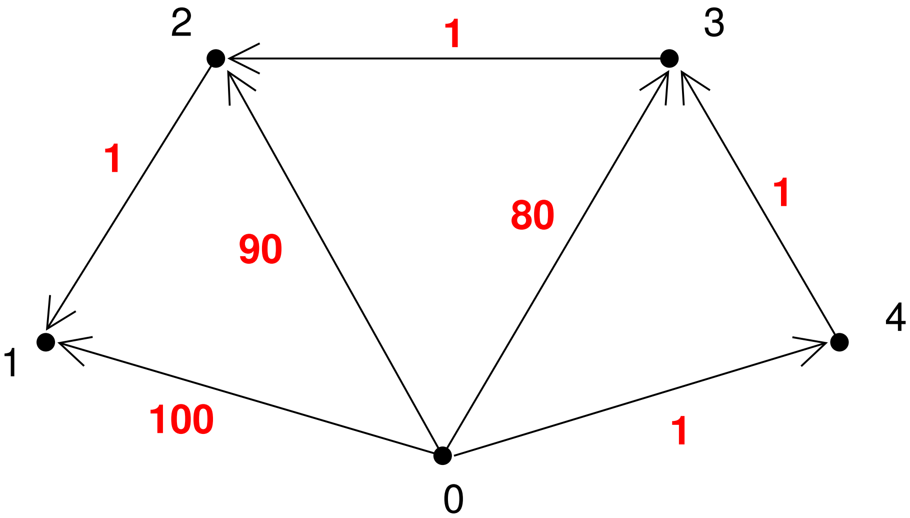
- Both are defined for weighted graphs
- Both produce trees as a result
- Both minmize by weight
- For each we have two algorithms

Input is only a graph
Problem usually is described on an undirected graph
Goal is to minimize total weight
There is no clear winner between the algorithms

Input is a graph and a starting point
Problem usually is described on a directed graph
Goal is to minimize weight on each path
One algorithm is clearly more efficient



0
1
2
3
4
5
6
7
8
9
10
11
12
15
6
3
1
5
9
8
5
6
13
12
2
9
8
9
6
1
4
5
3
6
2



2
1
3
1
90
80
1
1
4
100
1
0

Bellman-Ford Algorithm (SSSP)

Initialize all vertices to have infinite bound and no parent

$s.\texttt{distBound} = 0$

Repeat $|V| - 1$ times

For each $u \in V$

For each $v \in \texttt{adjacents}(u)$

If $v.\texttt{distBound} > u.\texttt{distBound} + (u, v).w$

$v.\texttt{distBound} = u.\texttt{distBound} + (u, v).w$

$v.\texttt{parent} = u$

Dijkstra's Algorithm (SSSP)

Initialize all vertices to have infinite bound and no parent

$s.\texttt{distBound} = 0$

Make a priority queue pq with all vertices

While pq is not empty

    $u = pq.\texttt{extractMax}()$

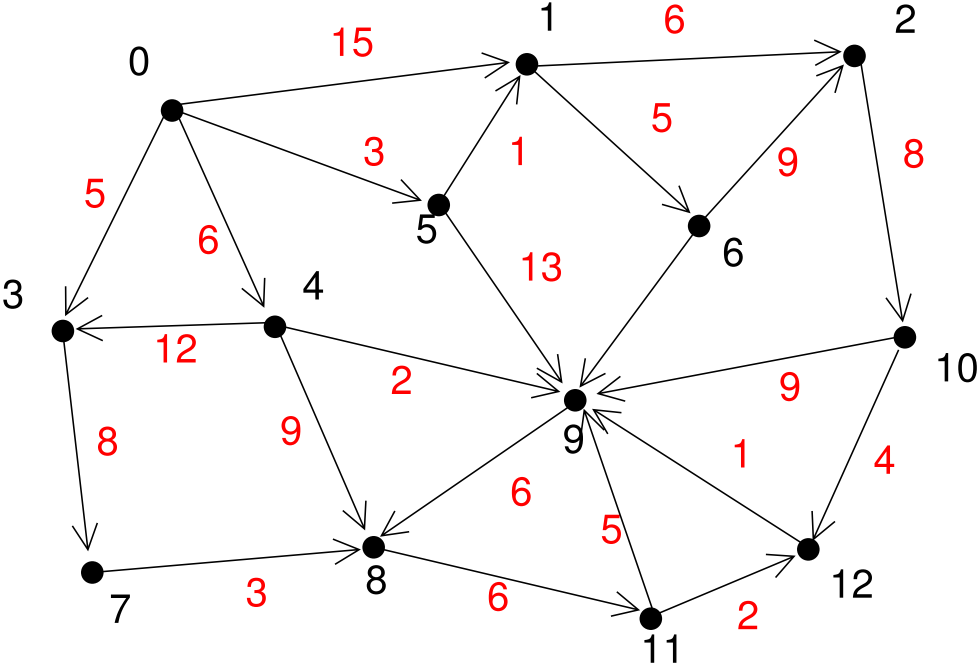
        For each $v \in \texttt{adjacents}(u)$

            If $v.\texttt{distBound} > u.\texttt{distBound} + (u, v).w$

                $v.\texttt{distBound} = u.\texttt{distBound} + (u, v).w$

                $v.\texttt{parent} = u$

                $pq.\texttt{increaseKey}(v)$


0
1
2
3
4
5
6
7
8
9
10
11
12
15
6
3
1
5
9
8
5
6
13
12
2
9
9
8
6
1
4
5
3
6
2