Chapter 4, Graphs:

- Concepts and implementation (last week Monday)
- Traversal (last week Wednesday (plus lab last week Thursday))
- Minimum spanning trees (last week Friday and this past Monday)
- Single-source shortest paths (**this week Wednesday and Friday**)
- Review for test (next week Monday)

Wednesday and Friday:

- (MST loose ends)
- The SSSP problem
- General concepts for SSSP algorithms
- The most unlucky graph for SSSP
- The Bellman-Ford algorithm plus analysis
- Dijkstra's algorithm plus analysis

Minimum Spanning Tree Problem

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

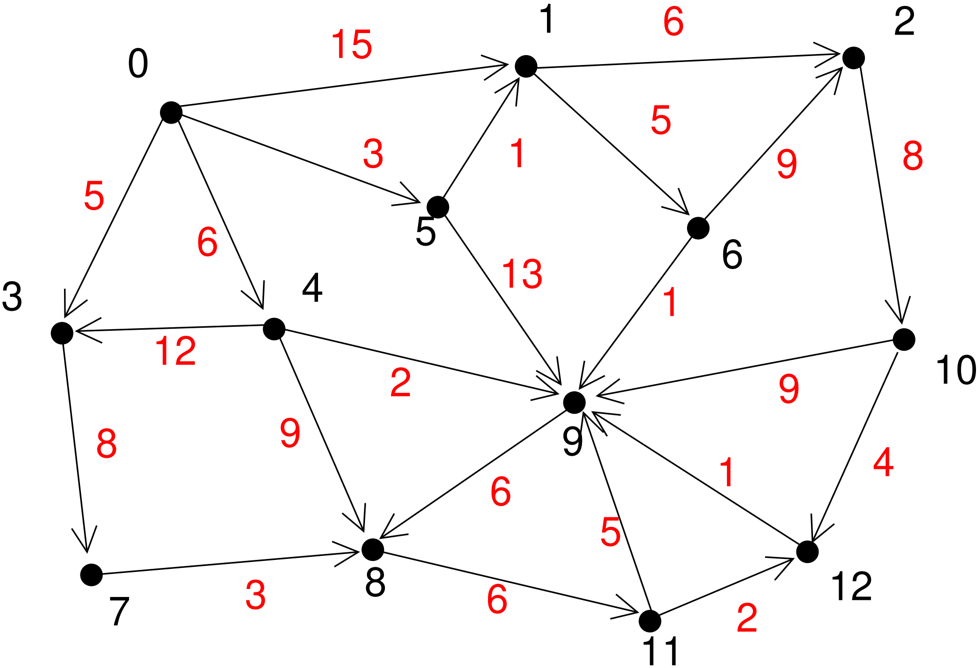
Single-Source Shortest Paths Problem

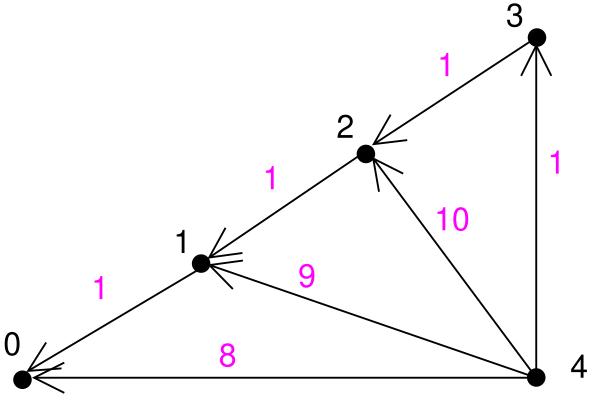
Given a weighted directed graph and a source vertex, find the tree comprising the shortest paths from that source to all other reachable vertices.

- 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
1
12
2
9
8
9
4
1
6
5
3
6
2



3
1
2
1
1
10
1
1
9
0
8
4

Let X be the set of vertices whose distance bounds are correct, that is, $v \in X$ if `distances`[v] is the total weight of the shortest path from s to v. For a single-source shortest path algorithm to be correct, all vertices reachable from s are in set X at termination, and if *all* vertices are reachable, this implies $X = V$. Let Y be the set of vertices that have been removed from the priority queue. Our intent is that $Y \subseteq X$: all vertices have correct distance bound at the time they are removed from the priority queue, though at any point there may also be some correct ones still in the priority queue. We claim

Invariant (Main loop of Dijkstra's algorithm)

Let X and Y be as defined above.

(a) $Y \subseteq X$.

(b) *If v is the vertex in the priority queue with least distance bound, then $v \in X$.*

(c) $|Y|$ *is the number of iterations completed.*

Coming up:

Do **MST** *project (due Wednesday, Oct 9)*
Do **SSSP** *project (due Friday, Oct 18)*

Due **Fri, Oct 11** *(end of day)*
Read Section 4.5
Do Exercises 4.(50, 51, 59)
Take SSSP quiz