Some Algorithms for Weighted Graphs

Dijkstra’s algorithm

- Finds the **single-source shortest path** in a weighted graph: given one starting vertex, the algorithm finds the shortest path to all other vertices in the graph
- Developed by Edsger W. Dijkstra in 1959

Dijkstra’s algorithm

1. Consider all vertices “unvisited” initially.
2. Assign each vertex a “distance” value. The distance of the starting vertex is 0; all other vertices have a distance of ∞.
3. Pick the minimum-distance unvisited vertex from the graph. Call this the current vertex, *C*.

Dijkstra’s algorithm

4. For each of *C*’s unvisited neighbors *N*, compute *N*’s “tentative distance” as *C*’s distance plus the weight of the edge connecting *C* and *N*. If the tentative distance is less than *N*’s existing distance, replace the distance and mark *C* as *N*’s parent. If the tentative distance is not less than *N*’s existing distance, do nothing.
5. Mark vertex *C* as visited.
6. Repeat steps 3-5 until all vertices have been visited.

Dijkstra’s algorithm

- Once the algorithm finishes, you can find the shortest path to any vertex by going to that vertex and following its parent vertices back to the starting vertex.

Prim's algorithm

- Finds a **minimum spanning tree** in a weighted graph – a set of edges connecting all vertices that has the lowest possible sum of weights
- Developed by R. C. Prim in 1957

Prim's algorithm

1. Pick any vertex from the graph (can be done randomly).
2. Look for the minimum-weight edge that is incident to the starting vertex. Add that edge to the spanning tree.
3. The spanning tree now contains one edge and two vertices. Look for the minimum-weight edge (to a vertex not yet in the spanning tree) that is incident to either of the vertices. Add that edge to the spanning tree.
4. Repeat until all vertices have been included in the spanning tree.