Lesson 3: Shortest-path Problems

Notes

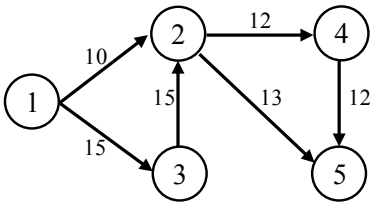
Book acknowledgment:

Goals

- Learn the Shortest Path Problem
- Learn and implement the following algorithms:
  - Dijkstra's Algorithm
  - Bellman-Ford Algorithm

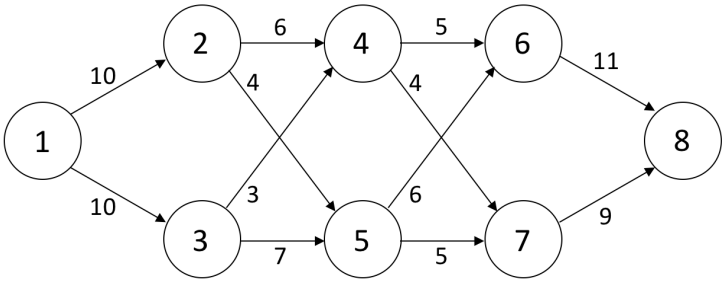
1 Try it on your own

Assume that you are starting at node 1, and the label on each arc is the number of dollars required to use that arc. What's the minimum cost required to move from node 1 to node 5 moving from one node to another using the arcs in the network?



a. Determine a shortest path from 1 to 5.

b. Determine a shortest path from 1 to every other node.



c. Determine a shortest path from 1 to 8.

d. Determine a shortest path from 1 to every other node.

2 Problem Definition

- Given a directed graph $G = (V, A)$, let s be the source (origin) node and t be the terminal (sink) node.
- Let c_{ij} be the cost of traversing arc $(i, j) \in A$.

Shortest path problem: Determine the minimum cumulative cost path from s to t.

How would you go about solving this on your own? (Given what you already know from previous courses.)

3 Dijkstra's Algorithm

- Dijkstra's algorithm is very similar to Prim's Algorithm in that we will keep track of which nodes are and are not included in a shortest path from s to t, so far.
- In doing so, we will actually build a shortest-path tree rooted at the source node s.
- At each step of the algorithm, we will search for a node not yet in the shortest-path tree having a minimum distance from the source.

Written Steps:

1. Initialize set $S = \emptyset$ that contains all the nodes currently in the shortest-path tree. Assign a minimum distance value v_i to each node $i \in V$ in the graph. Initialize $v_i = \infty,\ \forall i \in V$, and set $v_s = 0$. Proceed to Step 4.

2. If $S = V$, then proceed to Step 4. Otherwise, pick a node $u \in V \setminus S$ having a minimum distance value $v_u = \min\{v_i : i \in V \setminus S$, add u to S, and proceed to Step 3.

3. Update the distance value for each adjacent node of u. For all $(u, w) \in A$, if $v_w > v_u + c_{uw}$, then set $v_w = v_u + c_{uw}$ and set the parent of w to be u. Return to Step 4.

4. Terminate with the shortest-path tree S.

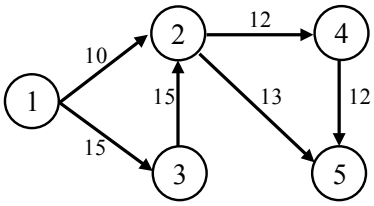
Pseudocode:

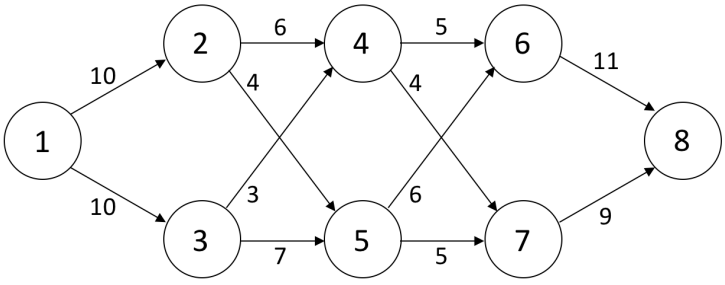
Algorithm 1 Determine a shortest-path tree rooted at source node s

```
Let S = ∅ be the set of shortest-path tree nodes.
Set v_i = ∞, ∀i ∈ V
Set v_s = 0.
while S ⊂ V: do
  Pick a node u ∈ V \ S for which v_u = min{v_i : i ∈ V \ S}.
  Add u to S.
  for (u, w) ∈ A do
    if v_w > v_u + c_uw, then
      Set v_w = v_u + c_uw
      Set parent(w) = u
    end if
  end for
end while
```

Let's practice!

Use Dijkstra's Algorithm to solve the two examples from earlier.

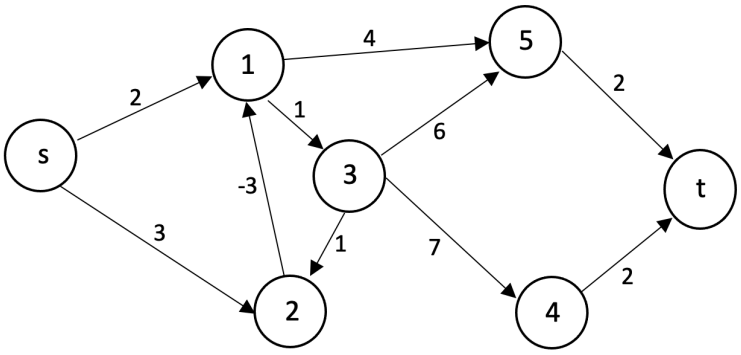




4 What could go wrong?

- The two examples from earlier all have positive arc costs. What could go wrong if some arcs have a negative cost?

Use Dijkstra's Algorithm to determine the shortest path from s to every other node in $V \setminus \{s\}$.



5 Bellman-Ford Algorithm

In the worst-case, the Bellman-Ford Algorithm is slightly slower than Dijkstra's Algorithm. Alternatively, it is more versatile because it is capable of handling graphs having negative arc costs. If a graph/network has a negative-cost cycle, then the Bellman-Ford Algorithm is able to detect and report it.

Like Dijkstra's Algorithm, the Bellman-Ford iteratively updates approximations to the minimum computed distance to each node in the network. Whereas Dijkstra's uses a greedy approach that selects the *closest* node that has not yet been added to the shortest-path tree set S, Bellman-Ford updates all edges at most $|V| - 1$ times.

Written Steps:

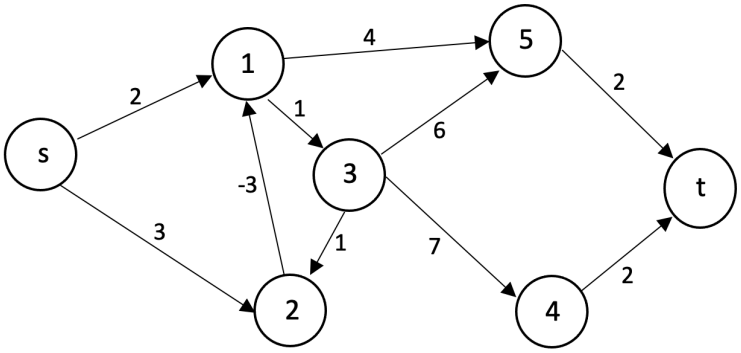
1. Set $k = 0$, $v_s = 0$, and $v_i = \infty, \forall i \in V \setminus \{s\}$
2. If $k < |V|$, then proceed to Step 4. Otherwise, et $k = k + 1$ and proceed to Step 3.
3. For all $(i, j) \in A$, if $\infty > v_j > v_i + c_{ij}$, then set $v_j = v_i + c_{ij}$ and set the parent of j to be i. Return to Step 2.
4. Terminate the Bellman-Ford Algorithm with the shortest-path tree corresponding to the parent vector.

Pseudocode:

Algorithm 2 Determine a shortest-path tree rooted at source node s

```
Set k = 0, v_s = 0, v_i = ∞, ∀i ∈ V \ {s}, and parent(i) = −1, ∀i ∈ V \ {s}.
while k < |V|: do
  for (i, j) ∈ A do
    if ∞ > v_j > v_i + c_ij then
      Set v_j = v_i + c_ij.
      Set parent(j) = i.
    end if
  end for
  Set k = k + 1.
end while
```

Use the Bellman-Ford Algorithm to determine the shortest path from s to every other node in $V \setminus \{s\}$.



6 Running Time Analysis

- What's the worst-case running time for Dijkstra's Algorithm?

- What's the worst-case running time for the Bellman-Ford Algorithm?