Graphs and Network Flows
ISE 411

Lecture 10

Dr. Ted Ralphs
References for Today’s Lecture

- Required reading
  - Sections 21.3

- References
  - AMO Chapter 5
  - CLRS Chapter 25
Shortest Path Algorithms

• Special structure $\Rightarrow$ easy!
  – **Special topology**: Reaching Algorithm
  – **Special cost structure**: Dijkstra’s Algorithm

• General network with negative cycles $\Rightarrow$ much harder!
  – Identify a negative cycle if one exists **OR**
  – Solve the problem if no negative cycle exists
General Label-Correcting Algorithms

Maintain a distance label $d(j)$ for all nodes $j \in N$

- If $d(j)$ is infinite, the algorithm has not found a path joining the source node to node $j$.
- If $d(j)$ is finite, it is the distance from the source node to that node along some path (upper bound).
- No label is permanent until the algorithm terminates.
Optimality Conditions

**Theorem 1. [5.1]** For every node \( j \in N \), let \( d(j) \) denote the length of some directed path from the source node to node \( j \). Then, the numbers \( d(j) \) represent the shortest path distances if and only if they satisfy the following for all \((i, j) \in A\):

\[
d(j) \leq d(i) + c_{ij}.
\]

**Proof:**

\((\Rightarrow)\) If \( d(j) \) represent shortest path distances, they satisfy \( d(j) \leq d(i) + c_{ij}, \ \forall (i, j) \in A \).

\((\Leftarrow)\) If a set of labels satisfies \( d(j) \leq d(i) + c_{ij} \), then they represent shortest path distances.
Distance Labels and Negative Cycles

Claim 1. If the network contains a negative cycle, then no set of distance labels satisfies \( d(j) \leq d(i) + c_{ij} \) for all \((i, j) \in A\).

For each arc, we define the reduced arc length \( c_{ij}^d \) of an arc \((i, j)\) with respect to distance labels \( d(\cdot) \) as \( c_{ij}^d = c_{ij} + d(i) - d(j) \).

To prove this claim, we need the following properties.

Property 1. [5.2] 1. For any directed cycle \( W \), \( \sum_{(i, j) \in W} c_{ij}^d = \sum_{(i, j) \in W} c_{ij} \).

2. For any directed path \( P \) from node \( k \) to node \( l \), \( \sum_{(i, j) \in P} c_{ij}^d = \sum_{(i, j) \in P} c_{ij} + d(k) - d(l) \).

3. If the vector \( d \) represents shortest path distances, then \( c_{ij}^d \geq 0 \) for every arc \((i, j) \in A\).
Generic Label-Correcting Algorithm

Input: A network $G = (N, A)$ and a vector of arc lengths $c \in \mathbb{Z}^A$
Output: $d(i)$ is the length of a shortest path from node $s$ to node $i$ and $\text{pred}(i)$ is the immediate predecessor of $i$ in an associated shortest paths tree.

$d(s) \leftarrow 0$ and $\text{pred}(s) \leftarrow 0$

$d(j) \leftarrow \infty$ for each $j \in N - \{s\}$

while $\exists (i, j) \in A$ such that $d(j) > d(i) + c_{ij}$ do

$d(j) \leftarrow d(i) + c_{ij}$

$\text{pred}(j) \leftarrow i$

end while
Predecessor Graph

- The collection of arcs \((\text{pred}(j), j)\) for every finitely labeled node \(j\) (except source)
- Directed out-tree rooted at the source that spans all nodes with finite distance labels
- Each distance update using the arc \((i, j)\) produces a new predecessor graph
  - delete the arc \((\text{pred}(j), j)\)
  - add the arc \((i, j)\)
- For every arc \((i, j)\) in the predecessor graph \(c^d_{ij} \leq 0\).
- When the algorithm terminates, the predecessor graph is a shortest path tree.
Termination

• Will the algorithm terminate in a finite number of iterations?

• What is the complexity of the algorithm?
Detecting Negative Cycles

- Check whether distance label is less than $-nC$
- Check whether predecessor graph contains a directed cycle