15.093 Optimization Methods

Lecture 11: Network Optimization
The Network Simplex Algorithm
Network Optimization

Why do we care?

- Networks and associated optimization problems constitute reoccurring structures in many real-world applications.
- The network structure often leads to additional insight and improved understanding.
- Given integer data, the standard models have integer optimal solutions.
- The network structure also enables us to design more efficient algorithms.
Network Optimization

A Comparison

Sample Instance...

1,772 nodes and 2,880 arcs
## Network Optimization

### A Comparison

#### Running Times

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Running Time (sec)</th>
<th># Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard Simplex</td>
<td>334.59</td>
<td>42759</td>
</tr>
<tr>
<td>Network Simplex</td>
<td>7.37</td>
<td>23306</td>
</tr>
<tr>
<td><strong>Ratio</strong></td>
<td><strong>2.2 %</strong></td>
<td><strong>54 %</strong></td>
</tr>
</tbody>
</table>

Average over 5 random instances with **10,000** nodes and **25,000** arcs each.

SMA-HPC ©2000 MIT
The Simplex Algorithm: A Reminder
The Network Simplex: A Combinatorial View
The Network Simplex: An Animated View
The Network Simplex: An Algebraic View
The Simplex Algorithm

A Reminder

The Problem...

\[
\begin{align*}
\min & \quad c'x \\
\text{s.t.} & \quad Ax = b \\
& \quad x \geq 0
\end{align*}
\]
The Simplex Algorithm

1. Start with basis $B = [A_{B(1)}, \ldots, A_{B(m)}]$ and BFS $x$.

2. Compute $\bar{c}_j = c_j - c'_B B^{-1} A_j$.
   - If $\bar{c}_j \geq 0$; $x$ optimal; stop.
   - Select $j$ such that $\bar{c}_j < 0$.

3. Compute $u = B^{-1} A_j$. $\theta^* = \min_{1 \leq i \leq m, u_i > 0} \frac{x_B(i)}{u_i} = \frac{x_B(\ell)}{u_\ell}$.

4. Form a new basis by replacing $A_{B(\ell)}$ with $A_j$.

5. $y_j = \theta^*$; $y_{B(i)} = x_B(i) - \theta^* u_i$. 
The Network Simplex Algorithm

Determine a least cost shipment of a commodity through a network in order to satisfy demands at certain nodes from available supplies at other nodes. Arcs have costs associated with them.
The Network Simplex Algorithm

- Network $G = (N, A)$.
- Arc costs $c : A \rightarrow \mathbb{Z}$.
- Node balances $b : N \rightarrow \mathbb{Z}$.

\[
\begin{align*}
\text{min} & \quad \sum_{(i,j) \in A} c_{ij} x_{ij} \\
\text{s.t.} & \quad \sum_{j: (i,j) \in A} x_{ij} - \sum_{j: (j,i) \in A} x_{ji} = b_i \quad \text{for all } i \in N \\
& \quad x_{ij} \geq 0 \quad \text{for all } (i, j) \in A
\end{align*}
\]
A tree is a graph that is connected and has no cycles.

A spanning tree of a graph $G$ is a subgraph that is a tree and contains all nodes of $G$.

A flow $x$ forms a tree solution with a spanning tree of the network if every non-tree arc has flow 0.
A *tree* is a graph that is connected and has no cycles.

A *spanning tree* of a graph $G$ is a subgraph that is a tree and contains all nodes of $G$.

A flow $x$ forms a *tree solution* with a spanning tree of the network if every non-tree arc has flow 0.
The Network Simplex Algorithm

Tree Solutions

- A **tree** is a graph that is connected and has no cycles.
- A **spanning tree** of a graph $G$ is a subgraph that is a tree and contains all nodes of $G$.
- A flow $\mathbf{x}$ forms a **tree solution** with a spanning tree of the network if every non-tree arc has flow 0.
What is the flow in arc (4,3)?
What is the flow in arc (5,3)?
What is the flow in arc (3,2)?
What is the flow in arc (2,6)?
What is the flow in arc (7,1)?
What is the flow in arc (1,2)?
Note: there are two different ways of calculating the flow on (1,2), and both ways give a flow of 4. Is this a coincidence?
• Every tree flow has a corresponding tree (and perhaps more than one).
• Given a tree, we obtain a unique tree flow associated with it.
Theorem 1 If the objective function is bounded from below, a min-cost flow problem always has an optimal tree solution.
Theorem 1 \textit{If the objective function is bounded from below, a min-cost flow problem always has an optimal tree solution.}
Theorem 1 If the objective function is bounded from below, a min-cost flow problem always has an optimal tree solution.
Theorem 1 If the objective function is bounded from below, a min-cost flow problem always has an optimal tree solution.
Theorem 1 If the objective function is bounded from below, a min-cost flow problem always has an optimal tree solution.
Theorem 2 A (feasible) tree $T$ is optimal if, for some choice of node potentials $p_i$,

(a) $\overline{c}_{ij} = c_{ij} - p_i + p_j = 0$ for all $(i, j) \in T$,

(b) $\overline{c}_{ij} = c_{ij} - p_i + p_j \geq 0$ for all $(i, j) \in A \setminus T$.

Proof:

- $\min \sum_{(i,j) \in A} c_{ij} x_{ij}$ is equivalent to $\min \sum_{(i,j) \in A} \overline{c}_{ij} x_{ij}$.
- $\min \sum_{(i,j) \in A} \overline{c}_{ij} x_{ij}$ is equivalent to $\min \sum_{(i,j) \in A \setminus T} \overline{c}_{ij} x_{ij}$.
- For any solution $x$, $x_{ij} \geq x^*_{ij}$ for all $(i, j) \in A \setminus T$. 
Here is a spanning tree with arc costs. How can one choose node potentials so that reduced costs of tree arcs are 0?
There is a redundant constraint in the minimum cost flow problem.

One can set $p_1$ arbitrarily. We will let $p_1 = 0$.

What is the node potential for 2?
What is the node potential for 7?
What is the potential for node 3?
What is the potential for node 6?
What is the potential for node 4?
What is the potential for node 5?
These are the node potentials associated with this tree. They do not depend on arc flows, nor on costs of non-tree arcs.
Node potentials
Original costs

The Network Simplex Algorithm

Tree Solutions
Updating the Tree...
Flow on arcs
Reduced costs
The Network Simplex Algorithm

Tree Solutions

Updating the Tree...

Flow on arcs

```
Flow on arcs
```

```
Flow on arcs
```

```
Flow on arcs
```

```
Flow on arcs
```

```
Flow on arcs
```

```
Flow on arcs
```

```
Flow on arcs
```
The Network Simplex Algorithm

Tree Solutions

Updating the Tree...

Diagram of a network with nodes and edges labeled with numbers.
The Network Simplex Algorithm

Tree Solutions

Updating the Tree...
1. Determine an initial feasible tree $T$. Compute flow $x$ and node potentials $p$ associated with $T$.

2. Calculate $\overline{c}_{ij} = c_{ij} - p_i + p_j$ for $(i, j) \notin T$.
   - If $\overline{c} \geq 0$, $x$ optimal; stop.
   - Select $(i, j)$ with $\overline{c}_{ij} < 0$.

3. Add $(i, j)$ to $T$ creating a unique cycle $C$. Send a maximum flow around $C$ while maintaining feasibility. Suppose the exiting arc is $(k, \ell)$.

4. $T := (T \setminus (k, \ell)) \cup (i, j)$. 
Our reasoning has two important and far-reaching implications:

- There always exists an integer optimal flow (if node balances $b_i$ are integer).
- There always exist optimal integer node potentials (if arc costs $c_{ij}$ are integer).
The Network Simplex Algorithm

An Animation
The Network Simplex Algorithm

The Algebraic View

- Bases and trees.
- Dual variables and node potentials.
- Changing bases and updating trees.
- Optimality testing.
The constraint matrix $A$ of the min-cost flow problem is the node-arc incidence matrix of the underlying network.

$$
\begin{array}{cccccccc}
(1, 2) & (2, 6) & (3, 2) & (4, 3) & (4, 5) & (5, 3) & (5, 6) & (6, 7) & (7, 1) \\
1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
2 & -1 & +1 & -1 & 0 & 0 & 0 & 0 & 0 \\
3 & 0 & 0 & +1 & -1 & 0 & -1 & 0 & 0 \\
4 & 0 & 0 & 0 & +1 & +1 & 0 & 0 & 0 \\
5 & 0 & 0 & 0 & 0 & -1 & +1 & +1 & 0 \\
6 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & +1 \\
7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
\end{array}
$$

The rows of $A$ are linearly dependent.
Let $B$ be the submatrix corresponding to the tree

$$
(1, 2) \quad (2, 6) \quad (3, 2) \quad (4, 3) \quad (5, 3) \quad (7, 1)
$$

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>(2)</th>
<th>(3)</th>
<th>(4)</th>
<th>(5)</th>
<th>(7)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>-1</td>
<td>+1</td>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>+1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Let $B$ be the submatrix corresponding to the tree

\[
\begin{align*}
(1, 2) & \quad (2, 6) & \quad (3, 2) & \quad (4, 3) & \quad (5, 3) & \quad (7, 1) \\
4 & \quad 0 & \quad 0 & \quad 0 & \quad +1 & \quad 0 & \quad 0 \\
5 & \quad 0 & \quad 0 & \quad 0 & \quad 0 & \quad +1 & \quad 0 \\
6 & \quad 0 & \quad -1 & \quad 0 & \quad 0 & \quad 0 & \quad 0 \\
7 & \quad 0 & \quad 0 & \quad 0 & \quad 0 & \quad 0 & \quad +1 \\
3 & \quad 0 & \quad 0 & \quad +1 & \quad -1 & \quad -1 & \quad 0 \\
2 & \quad -1 & \quad +1 & \quad -1 & \quad 0 & \quad 0 & \quad 0 \\
1 & \quad +1 & \quad 0 & \quad 0 & \quad 0 & \quad 0 & \quad -1
\end{align*}
\]

Permuting Rows
Let $B$ be the submatrix corresponding to the tree.

\[ \begin{pmatrix}
(4, 3) & (5, 3) & (2, 6) & (7, 1) & (3, 2) & (1, 2) \\
4 & +1 & 0 & 0 & 0 & 0 & 0 \\
5 & 0 & +1 & 0 & 0 & 0 & 0 \\
6 & 0 & 0 & -1 & 0 & 0 & 0 \\
7 & 0 & 0 & 0 & +1 & 0 & 0 \\
3 & -1 & -1 & 0 & 0 & +1 & 0 \\
2 & 0 & 0 & +1 & 0 & -1 & -1 \\
1 & 0 & 0 & 0 & -1 & 0 & +1 \\
\end{pmatrix} \]
Corollary 1

(a) The matrix $A$ has rank $n - 1$.
(b) Every tree solution is a basic solution.
Theorem 3 *Every tree defines a basis and, conversely, every basis defines a tree.*

Suppose the graph defined by a basis contains a cycle $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6$:

\[
\begin{array}{cccccccc}
(1, 2) & (2, 3) & (4, 3) & (5, 4) & (5, 6) & (1, 6) \\
1 & +1 & 0 & 0 & 0 & 0 & +1 \\
2 & -1 & +1 & 0 & 0 & 0 & 0 \\
3 & 0 & -1 & -1 & 0 & 0 & 0 \\
4 & 0 & 0 & +1 & -1 & 0 & 0 \\
5 & 0 & 0 & 0 & +1 & +1 & 0 \\
6 & 0 & 0 & 0 & 0 & -1 & -1 \\
\end{array}
\]
Remember, the simplex algorithm computes the dual variables $p$ as the solution to $p^{'B} = c'_B$.

$$
\begin{pmatrix}
+1 & 0 & 0 & 0 & 0 & 0 \\
0 & +1 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & +1 & 0 & 0 \\
-1 & -1 & 0 & 0 & +1 & 0 \\
0 & 0 & +1 & 0 & -1 & -1
\end{pmatrix}
$$

$$(p_4, p_5, p_6, p_7, p_3, p_2) = (c_{43}, c_{53}, c_{26}, c_{71}, c_{32}, c_{12})$$

Hence, $p_2 = -c_{12}$, $p_3 = c_{32} + p_2$, $p_7 = c_{71}$, ...
Remember, the simplex algorithm computes the reduced costs $\bar{c}$ as 

$$\bar{c}_{ij} = c_{ij} - p_i' A_{ij}.$$ 

<table>
<thead>
<tr>
<th></th>
<th>(1, 2)</th>
<th>(2, 6)</th>
<th>(3, 2)</th>
<th>(4, 3)</th>
<th>(4, 5)</th>
<th>(5, 3)</th>
<th>(5, 6)</th>
<th>(6, 7)</th>
<th>(7, 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>2</td>
<td>-1</td>
<td>+1</td>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>+1</td>
<td>-1</td>
<td>0</td>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+1</td>
<td>+1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>+1</td>
<td>+1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>+1</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>+1</td>
</tr>
</tbody>
</table>

Therefore, 

$$\bar{c}_{ij} = c_{ij} - p_i + p_j.$$
The network simplex algorithm is extremely fast in practice.

Relying on network data structures, rather than matrix algebra, causes the speedups. It leads to simple rules for selecting the entering and exiting variables.

Running time per pivot:
- arcs scanned to identify an entering arc,
- arcs scanned of the basic cycle,
- nodes of the subtree.

A good pivot rule can dramatically reduce running time in practice.