Lecture 11
Network Models in OR

MATH3220 Operations Research and Logistics
Mar. 17, 2015

Pan Li
The Chinese University of Hong Kong

Agenda

1. **Basic Terms on Graph and Network**

2. **Trees, Spanning Trees and MST**

Basic Terms on Graph and Network

Graph

$G = (N, E)$

N = set of nodes (or vertices), $E \subseteq N \times N$ = set of edges.

Notations: $N = \{x_i\}$; $N = \{i\}$.

$E \subseteq \{(x_i, x_j) | x_i \in N, x_j \in N\}$; $E \subseteq \{(i, j) | i \in N, j \in N\}$.

- Elementary chain: sequence of distinct nodes $x_1, x_2, \ldots, x_k$ such that $(x_1, x_2), (x_2, x_3), \ldots, (x_{k-1}, x_k) \in E$.
- Elementary cycle: elementary chain when $x_1 = x_k$.

Directed Graph (or Digraph)

$G = (N, A)$

N = set of nodes (or vertices), $A \subseteq N \times N$ = set of arcs .

Directed edge: referred to as *arc*

(x_i, x_j) or (i, j) then becomes an *ordered* pair.

- Simple path: sequence of distinct nodes $x_1, x_2, \ldots, x_k$ such that $(x_1, x_2), (x_2, x_3), \ldots, (x_{k-1}, x_k) \in A$.
- Simple cycle (or circuit): simple path when $x_1 = x_k$.

Connectedness: every pair of distinct nodes is joined by (or reachable via) an elementary chain (or simple path).

A connected (sub)graph with *no* cycles is called a *tree*.

Network

$G = (N, A)$ directed graph with additional information (or attribution) on nodes and/or arcs.

For example, c_{ij} = capacity or distance of arc (i, j), a_{ij} = cost of arc (i, j); u_i or v_i = node label (e.g. weight or potential) on node i.

Trees

Let $G = (N, E)$ be a simple graph with n nodes. The following statements are equivalent.

1. G is a tree (connected and acyclic).
2. There is a unique path between each pair of nodes in G.
3. G contains $n - 1$ edges and is connected.
4. G contains $n - 1$ edges and is acyclic.
5. G is acyclic and if any two nonadjacent nodes are joined by an edge, the resulting graph has exactly one cycle.

These equivalent definitions lead directly to the following useful properties of a tree.

- If $G = (N, E)$ is a tree and $f \notin E$, then $G' = (N, E \cup \{f\})$ contains exactly one cycle.
- If C is the edge set of the cycle of G' and $e \in C \setminus \{f\}$, $H = (N, E \cup f \setminus \{e\})$ also a tree.

Spanning Trees

- A spanning tree $H = (N, F)$ of a connected graph $G = (N, E)$ is a tree whose set of nodes is N and whose set of edges F is a subset of E.
- Any connected graph indeed has a spanning tree as its subgraph.
- Algorithm for building a spanning tree:

Step 1. Any edge ordering $e_1, e_2, \ldots, e_m$; $F^0 = \phi, i = 1$.

Step 2. If $H = (N, F^{i-1} \cup \{e_i\})$ is acyclic, then $F^i = F^{i-1} \cup \{e_i\}$. Otherwise, $F^i = F^{i-1}$.

Step 3. If $F^i = n - 1$, stop; and (N, F^i) is a spanning tree. Otherwise, $i = i + 1$, and return to Step 2.

Minimum Spanning Tree (MST)

- A spanning tree will correspond to a communication network in which each pair of nodes is connected by exactly one path.
- A minimum spanning tree (MST) is then a communication network of the least possible total distance (or weight) as a whole.
- Algorithms for building a MST:
  1. **Kruskal's Algorithm:**
     (Initially T is empty.)
     Repeat until set T has $n-1$ edges:
     Add to T the shortest edge that does not form a cycle with edges already in T.
  2. **Prim's Algorithm:**
     (Initially T contains of any one edge of shortest length.)
     Repeat until tree T has $n-1$ edges:
     Add to T the shortest edges between a node in T and a node not in T.

Example

Table below shows the distances among the 10 cities that are nicely modelled by a complete (undirected) graph of 10 nodes and 45 edges.

node	2	3	4	5	6	7	8	9	10
1	96	105	50	41	86	46	29	56	70
2		78	49	94	21	64	63	41	37
3			60	84	61	54	86	76	51
4				45	35	20	26	17	18
5					80	36	55	59	64
6						46	50	28	8
7							45	37	30
8								21	45
9									25

Example-con't

node	2	3	4	5	6	7	8	9	10
1	96	105	50	41	86	46	29	56	70
2		78	49	94	21	64	63	41	37
3			60	84	61	54	86	76	51
4				45	35	20	26	17	18
5					80	36	55	59	64
6						46	50	28	8
7							45	37	30
8								21	45
9									25

Both Kruskal's algorithm and Prim's algorithm give the same MST solution. The list of edges chosen is given by

$$\{(1,8),(2,6),(3,10),(4,7),(4,9),(4,10),(5,7),(6,10),(8,9)\}$$

for a total weight of 221.

However, the *orders* of the individual edges chosen are different.

Theorem

Kruskal's algorithm yields an MST.

Proof.

Suppose the algorithm produces the tree $T = (N, F)$ and T is not optimal.

Let $T^* = (N, F^*)$ be an optimal tree with the property that $|F^* \setminus F|$ is minimum over all optimal trees. Note that $F^* \setminus F \neq \phi$ and $F \setminus F^* \neq \phi$. Let f be a smallest-weight edge in $F \setminus F^*$.

Consider the set of edges $F^* \cup \{f\}$, which by the property of a tree, contains a unique cycle. Let C be the edge set of the cycle. Again, since T^* is a tree, there is an edge $f^* \in C \setminus F$ such that the graph $(N, F^* \cup \{f\} \setminus \{f^*\})$ is a tree, sat $\hat{T}$. Moreover, $\hat{T}$ is also an optimal tree, since $w(f) \leq w(f^*)$, where the inequality holds because the algorithm selected f.

Finally, $|\hat{F} \setminus F| = |F^* \setminus F| - 1$, which contradicts the choice of T^*. So, T is optimal. □

Theorem

Prim's algorithm yields an MST.

Proof.

Denote by T_i the tree constructed after i iterations of the algorithm, $i = 1, 2, \ldots, n-1$.
Hence the algorithm produces a spanning tree $T = T_{n-1}$ and suppose T is not optimal. Let $T^* = (N, F^*)$ be an optimal tree that has as many edges in common with T as possible.
As $T \neq T^*$, let $f = (a, b)$ be the first edge chosen by the algorithm (say in its kth iteration, $k \leq n-1$) that is not in T^*. (Thus $f \in T_k \setminus T^*$.) Let P be the path in T^* from a to b; and f^* be an edge of P between a node in T_{k-1} and a node not in T_{k-1} (Thus $f^* \in T^* \setminus T_k$.) Note that edge f also has one end in T_{k-1} and one end not in T_{k-1} (but in T_k). We thus have $w(f) \leq w(f^*)$ because the algorithm has chosen f over f^*.
Now $\hat{T} \equiv (N, F^* \cup \{f\} \setminus \{f^*\})$ obtained from T^* by replacing f^* with f is then an optimal tree and $|\hat{F} \setminus F| = |F^* \setminus F| - 1$, which contradicts the choice of T^*. So, T is optimal. □