Chalmers University of Technology

Applied Optimisation lp 4 VT08

Network Models

April 11, 2008

Birgit Grohe

Network Models - Examples

Many different problems can be formulated as graph/network models:

- Find the shortest/fastest connection from Chalmers to Lindholmen.
- Connect a number of base stations minimizing the total cost.
- Find the maximum capacity in a given water pipeline network.
- Find a time schedule (start and completion times) for activities in a project.
- ...

Definition and Terminology

A *graph* consists of a set of *nodes*, N, linked by a set of *edges* or *arcs*, A.

For many applications, it is necessary to introduce distances d_{ij} for the arcs.

Arcs can be *directed* or *undirected*. If the arcs are directed then we have a *directed graph*.

A *path* is a sequence of arcs that joins two nodes. A path is a *cycle/loop* if it connects a node with itself. (*Directed paths and cycles, simple path*). In a *connected graph* there exists at least one path between each pair of nodes.

A *tree* is a connected graph without cycles that connects a *subset* of all nodes. A*spanning tree* is a tree that connects *all* nodes.

The Minimum Spanning Tree (MST) Problem

Given a graph/network with nodes, arcs and distances for each arc. Find a subset of all arcs that connects all nodes at minimum total distance.

Taha, figure 6.6

Prim's Algorithm for MST

Idea: Start at an arbitrary node. Among the nodes that are not connected yet, choose the one that can be connected at minimum cost. Stop when all nodes are connected.

The MST problem is a very simple problem and can be solved by *Greedy methods*.

Another greedy MST algorithm is Kruskal's algorithm. Idea: Sort edges by increasing distance. Choose edges that do not form cycles until all nodes are connected.

Taha, example 6.2-1, figure 6.7

The Shortest Path (SP) Problem

Given a network with nodes, arcs and distances on the arcs. Find the shortest path from a source node to a destination node.

Examples for problems that can be formulated as (SP):

- Find the shortest connection from Chalmers to Lindholmen.
- Find most reliable route. Taha, example 6.3-2.
- Find the best strategy for equipment replacement. Ex 6.3-1.
- Solve the three-jug puzzle. Taha, example 6.3-3.
- ...

Example: Equipment Replacement

(Taha, example 6.3-1) RentCar wants to find a replacement strategy for its cars for a 4-year planning period. Each year, a car can be kept or replaced. The replacement cost for each year and period is given in the table below (Taha, page 240).

Each car must be used at least 1 year and at most 3 years.

Taha, figure 6.10

Example: Most Reliable Route

(Taha, example 6.3-2) Mr Q drives to work daily. All roads he can choose for a path to work are patrolled by the police; it is possible to assign a probability of not being stopped by the police. He wants to find the 'shortest' path in the sense that the probability of being stopped is as low as possible.

Taha, figure 6.11 and 6.12

Algorithms for the SP Problem: Dijkstra

Dijkstra's algorithm finds the shortest path between a source node s and node i if all distances on the arcs are non-negative.

Handwritten example on blackboard (get a handout from a teacher)

Dijkstra's Algorithm

Let N be the set of all nodes, s the source node, and u_i the distance from s to i (if there is no direct link from s to i then $u_i = \infty$).

Step 0: $S := \{s\}$ and $\bar{S} := N \setminus \{s\}$. u_i for all i.

Step 1:

(a) If $\bar{S} = \emptyset$, stop. Else find node k with smallest u_k. $S := S \cup \{k\}$ and $\bar{S} := \bar{S} \setminus \{k\}$

(b) For all $j \in \bar{S}$ and $i \in S$ if $u_j > u_i + d_{ij}$ set $u_j := u_i + d_{ij}$ and $pred(j) = i$.

Dijkstra actually finds SP from the source to *all* other nodes!

Floyd's Algorithm

Floyd's algorithm computes shortest paths between each pair of nodes. Negative distances are allowed (no negative cycles).

Idea: Given three nodes i, k, j and their distances, it is shorter to go from i to j passing through k if

$$d_{ik} + d_{kj} < d_{ij}$$

In each iteration $1 \ldots k$, check if d_{ij} can be improved by using the 'shortcut' via k.

Administration of the algorithm: Maintain two matrices per iteration, D_k for the distances and $Pred_k$ to keep track of the predecessors for each node.

Taha, figure 6.19

Floyd's Algorithm

Step 0: Initialize D_0 and S_0.

Step k: $D_k := D_{k-1}$, $S_k := S_{k-1}$.
For each element d_{ij} in D_k, if $d_{ik} + d_{kj} < d_{ij}$ then
$d_{ij} := d_{ik} + d_{kj}$ and $s_{ij} := k$.
$k := k + 1$. If $k > n$ stop, else repeat step k.

Taha, Example 6.2-1

Definition of a Network Problem

A network consist of a set of *nodes*, N, linked by a set of *arcs*, A. A distance d_{ij} is associated with each arc.

Each node in the network may have a supply a_i or demand b_i.
Each arc has a (unknown) amount of flow x_{ij} that restricted by a maximum capacity $C_{ij} \in [0, \infty]$.

A *graph* is a special case of a network.

directed/undirected networks, connected networks

A Network formulation of the SP Problem

Find the shortest path from node s to node t. Let for each arc $x_{ij} = 1$ be the flow on arc (ij), $x_{ij} = 1$ if the arc is used in the SP path and $x_{ij} = 0$ otherwise.

LP-formulation:

$$
\begin{aligned}
\min \quad & \sum_{(ij)\in A} d_{ij} x_{ij} \\
\text{s.t.} \quad & \textstyle\sum_j x_{sj} = 1 \\
& \textstyle\sum_j x_{jt} = 1 \\
& \textstyle\sum_i x_{ik} - \sum_j x_{kj} = 0 \quad k \neq s, t \\
& x_{ij} \geq 0
\end{aligned}
$$

The constraints in the last row are called *flow conservation constraints.*

Maximum Flow Models

Consider a network with pipelines that transports some environment-friendly energy from a number of sources to a number of destinations. Intermediate pump stations need to be passed. Each pipe segment has a maximum capacity $\bar{C}_{ij}$. A pipe can be uni- or bidirectional.

What is the maximum total amount of energy flow through this network?

Taha, figure 6.27

Max Flow – Min Cut

A *cut* is a set of arcs which when deleted interrupt all flow in the network between the source and the sink. The *cut capacity* equals the sum of all capacities on the arcs in the cut.

Network flow theorem: max flow = min cut.

Taha, figure 6.29

Solving the Max Flow Problem

Alternative 1: Enumerate all possible cuts and select smallest. But how do we then find the actual flow on each arc? Also, there can be very many cuts.

Alternative 2: Basic idea of flow algorithm: Find paths through the network and push as much flow as possible on them without violating the capacity constraints. Administration: For each direction of an arc, keep track of the *residuals (remaining capacities)* (c_{ij}, c_{ji}) after some flow has been pushed along the arc.

Max Flow Algorithm (Short version of Taha 6.4.2)

Step 1: **Initialize** residuals $(c_{ij}, c_{ji}) = (\bar{C}_{ij}, \bar{C}_{ji})$, i=source, goto 2.

Step 2: Find S_i, set of nodes reachable from i with $c_{ij} > 0$. If $S_i = \emptyset$ the goto 4, else goto 3.

Step 3: Choose node $k \in S_i$ with maximum c_{ik}. If $k = n$ goto 5. Else $i = k$ and goto 2.

Step 4: Getting stuck. **Backtrack** to previous node and goto 2.

Step 5: **Breakthrough path found.** Calculate max flow along the path and update residuals.

Step 6: **Solution.** Sum up flows on all breakthrough paths. Find flow on each arc by considering the residuals.

Max Flow Algorithm: Example

Taha, example 6.4-2

LP Formulation of the Max Flow Problem

Taha, example 6.4-3

Critical Path Method (CPM)

Given a project with a number of activities with durations d_{ij}.
Given precedence constraints among some of the activities.

Network model: Each activity is one arc with a unique start and end node (and d_{ij}). Introduce arcs with $d_{ij} = 0$ to restore the precedence constraints.

What is the minimum completion time of all activities? Find the critical path (=longest path) in this network. Can be modelled as a SP problem (negate d_{ij}).

Literature

Taha: Operations Research, An Introduction

Kleinberg, Tardos: Algorithm Design

Cormen, Leiserson, Rivest, Stein: Introduction to Algorithms

Brassard, Bratley: Fundamentals of Algorithmics