Network Optimization Models IV
Maximum Flow Problems

Class 23

April 12, 2023

Handouts

- Notes on Assignment 7
- Assignment 8
- Green Mountain Power Example

Exam 2: Monday Evening, April 24

Network Optimization Problems

Minimal Spanning Tree

Shortest Path

Maximum Flow

An Equipment Replacement Problem

Middlebury College's LIS is developing a replacement policy for its computer servers for a four-year period. At the start of the first year, Middlebury will purchase a server. At the start of each subsequent year, it will decide to keep the server or to replace it. The server will be in service for at least one year but no more than 3 years. This table shows the replacement cost as a function of the period when it is purchased and the years kept in operation:

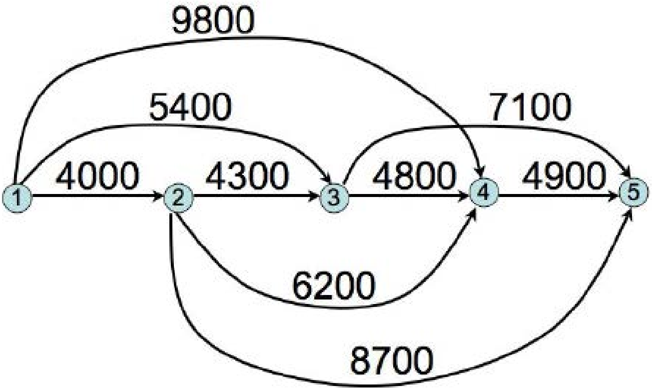
	Years	in	Operation
Start of Year	1	2	3
1	4000	5400	9800
2	4300	6200	8700
3	4800	7100	
4	4900		

Problem: Determine the best decision that minimizes the total cost incurred over the 4 year period.

All Possible Decisions

Decision	Cost				Total Cost
By a server in years 1,2,3,4	4000	4300	4800	4900	18,000
By a server in years 1,2,3	4000	4300	7100	x	15,400
By a server in years 1,2,4	4000	6200	x	4900	15,100
By a server in years 1,3,4	5400	x	4800	4900	15,100
By a server in years 1,2	4000	8700	x	x	12,700
By a server in years 1,3	5400	x	7100	x	12,500
By a server in years 1,4	9800	x	x	4900	14,700

Let's Formulate Problem as Shortest Path Problem



The best decision corresponds to the shortest path from node 1 to node 5, which is $1 \to 3 \to 5$ with the cost of $5400 + 7100 =$ 12,500: Purchase a new server in years 1 and 3.

Maximum Flow Problem


Paddington
Edgware Road
Baker Street
Great Portland Street
Euston
King's Cross St. Pancras
Angel
Marylebone
Warren Street
Euston Square
Farringdon
Old Street
Regent's Park
Barbican
Liverpool Street
Bayswater
Russell Square
Moorgate
Notting Hill Gate
Lancaster Gate
Bond Street
Oxford Circus
Goodge Street
Chancery Lane
Holborn
Tottenham Court Road
St. Paul's
Queensway
Marble Arch
Bank
Aldgate
Covent Garden
High Street Kensington
Hyde Park Corner
Green Park
Cannon Street
Leicester Square
Mansion House
Monument
Tower Hill
Knightsbridge
Piccadilly Circus
Charing Cross
Gloucester Road
Sloane Square
St. James's Park
Blackfriars
Temple
Earl's Court
South Kensington
Victoria
Westminster
Embankment
London Bridge
Waterloo
River Thames


Oxygen, nitrogen, hydrogen and carbon monoxide pipelines
NORTHERN EUROPE networks
NETHERLANDS
BELGIUM
FRANCE
Rozenburg
Bergen-op-Zoom
Antwerpen
Terneuzen
Dunkerque
Mons
Feluy
Charleroi
Liège
Waziers
AIR LIQUIDE

GREEN
MOUNTAIN
POWER

Green Mountain Power (GMP) is a large electrical power generating company.

Study of future requirements for electricity in the region it serves:

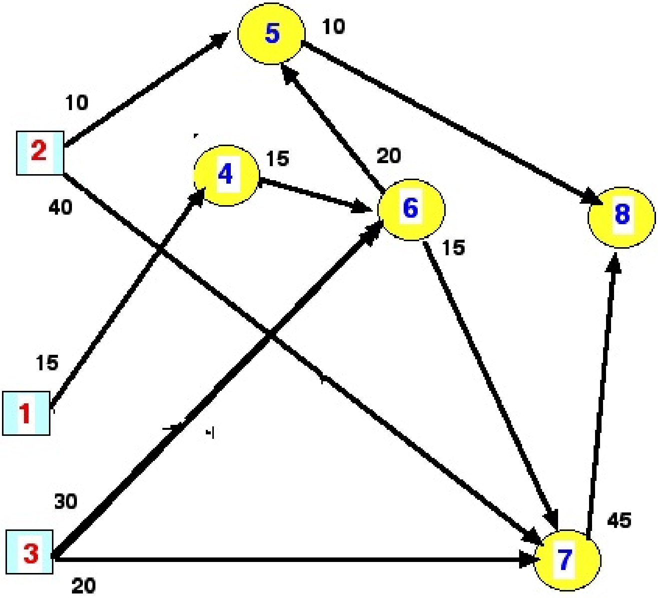
Are existing facilities (transmission lines, relay stations, etc.) adequate for transmitting the larger quantities of electricity required to accommodate increased future demands?

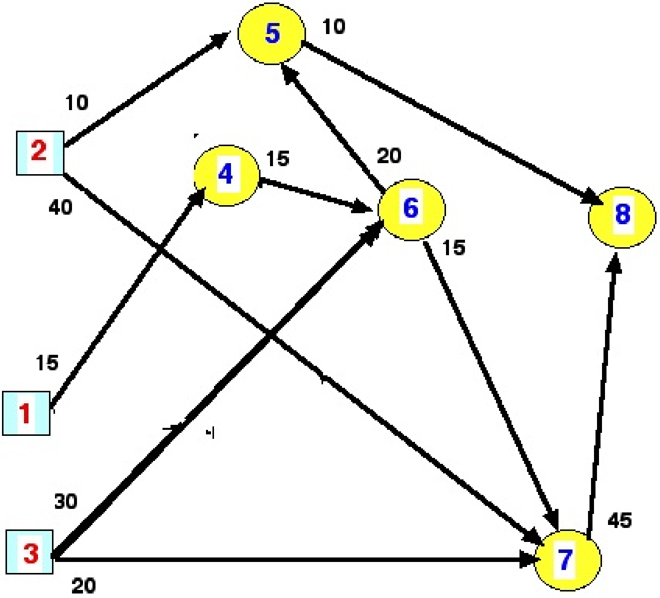
Suppose GMP has three generating stations, labeled $\boxed{1}$ $\boxed{,2}$ $\boxed{3}$, with respective capacities of 15, 10, and 40 megawatts; there are 5 cities, labeled $\boxed{4}$, $\boxed{5}$,$\boxed{6}$, $\boxed{7}$, and $\boxed{8}$, which form nodes on a graph.

There are transmission lines both between the generating stations and some cities and between some pairs of cities.

Each transmission line has a known **capacity** c_{ij}.

Major Question: What is the maximum number of megawatts that can be sent to each city from all three power stations?

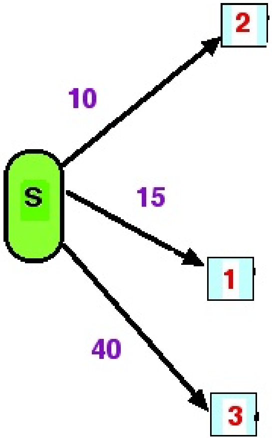

5
10
10
2
4
15
20
6
8
40
15
15
1
30
3
7
45
20

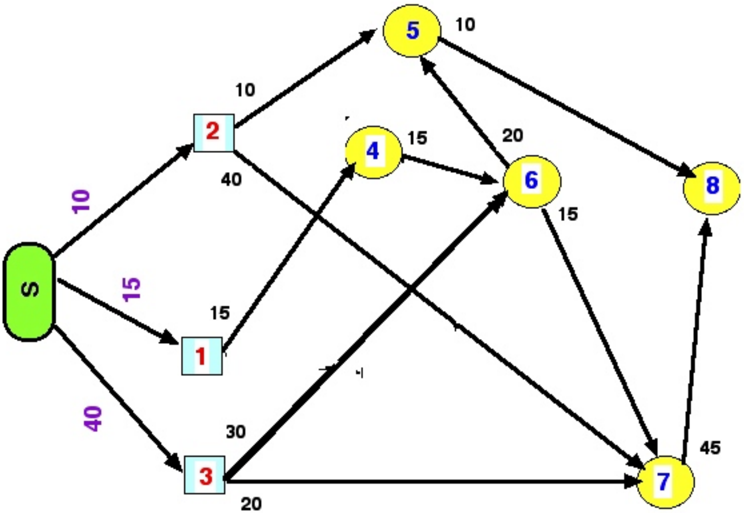

5
10
10
2
4
15
20
40
6
8
15
15
1
30
3
20
7
45

We have a "Multiple Source" Situation

It's convenient to convert to a network with a single **source** and a single **sink**

Add a new node, called a **Super Source S** and put a link from **S** to each original source and make the capacity of that link the number of megawatts that can be generated.

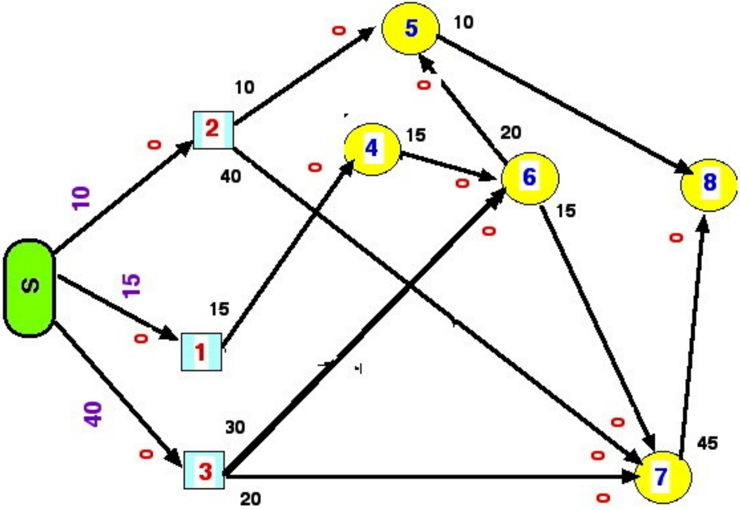



S
10
15
40
2
1
3
10
40
15
30
20
5
4
6
8
7
10
15
20
15
45

For this problem, we actually have 5 different maximum flow problems, one to each city.

We'll focus on node 8 as the destination.

Getting started:
Set all $x_{ij} = 0$.
Mark original capacities near start of each link
Let $c_{ji} = 0$ if there is no capacity in direction from h to i.



S
1
2
3
4
5
6
7
8
10
15
40
0
10
40
15
30
20
15
10
20
15
45

Original Capacities

S1 : 15	S2 : 10	S3: 40
14 : 15		
25 : 10	27:40	
37: 20	46 :15	58: 10
65: 20	67: 15	
78: 45		

Begin at the **source** and select any path to the destination which has positive capacity **c** where **capacity of a path = minimum of capacities of links on that path.**

Example: $\boxed{S} \xrightarrow{40} \boxed{3} \xrightarrow{20} \boxed{7} \xrightarrow{45} \boxed{8}$ has capacity $\mathbf{c} = 20$.

Augment Flows

$$x_{S3} \to x_{S3} + 20 = 0 + 20 = 20$$
$$x_{37} \to x_{37} + 20 = 0 + 20 = 20$$
$$x_{78} \to x_{78} + 20 = 0 + 20 = 20$$

Augment Flows And Update Capacities

$$x_{S3} \to x_{S3} + 20 = 0 + 20 = 20$$
$$x_{37} \to x_{37} + 20 = 0 + 20 = 20$$
$$x_{78} \to x_{78} + 20 = 0 + 20 = 20$$

Modify Capacities of Arcs

$c_{ij} \to c_{ij} - \mathbf{c}$ for all arcs (i, j) on path

$c_{ji} \to c_{ji} + \mathbf{c}$ for all arcs (i, j) on path

This yields a **residual network** with **residual capacities** on arcs.

Then select new path from **S** to $\boxed{8}$, called **Augmenting Path** and continue as before.

Augmenting Paths for GMP Problem

Iteration 1: S → 3 → 7 → 8

Iteration 2: S → 2 → 5 → 8

Iteration 3: S → 1 → 4 → 6 → 7 → 8

Iteration 4: S → 3 → 6 → 5 → 2 → 7 → 8

Capacities

Iteration 1: 20

Iteration 2: 10

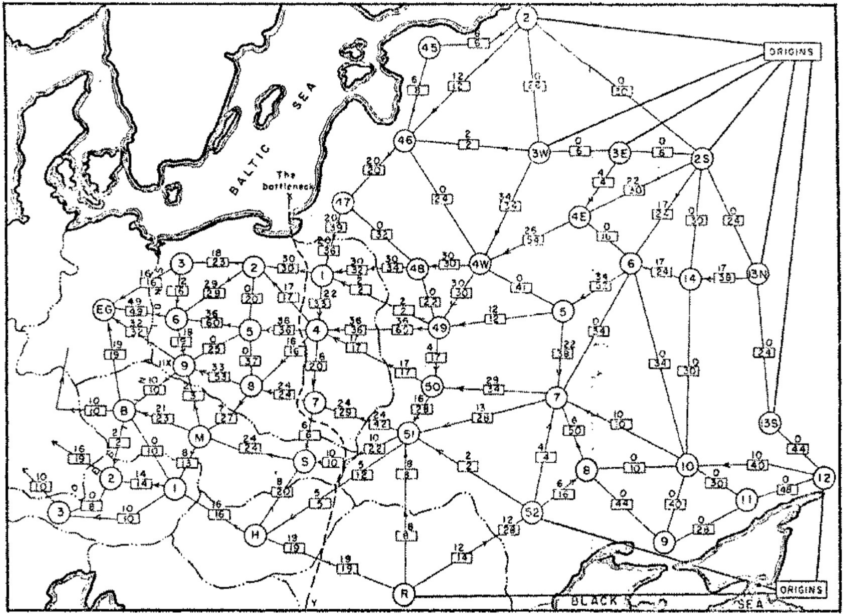
Iteration 3: 15

Iteration 4: 10

Final Flow Assignments

$$
\begin{array}{llll}
x_{S3} = 30 & x_{14} = 15 & x_{25} = 0 & x_{36} = 10 \\
x_{S2} = 10 & & x_{27} = 10 & x_{37} = 20 \\
x_{S1} = 30 & & & \\
& & & \\
x_{46} = 15 & x_{58} = 10 & x_{65} = 10 & x_{78} = 45 \\
& & x_{67} = 15 &
\end{array}
$$

Maximum Flow is 55 megawatts


BALTIC SEA
The Bottleneck
ORIGINS
ORIGINS
BLACK SEA

Schematic diagram of the railway network of the Western Soviet Union and Eastern European countries, with a maximum flow of value 163,000 tons from Russia to Eastern Europe, and a cut of capacity 163,000 tons indicated as "The bottleneck".

T.E. Harris, F.S. Ross, *Fundamentals of a Method for Evaluating Rail Net Capacities*, Research Memorandum RM-1573, The RAND Corporation, Santa Monica, California, 1955.

The Maximum Flow Problem

1) All flow through a directed and connected network originates at one node, called the **source** and terminates at one other node, called the **sink**.

2) All the remaining nodes are *transshipment nodes*.

3) Flow through an arc is allowed only in the direction indicated by the arrowhead, where the maximum amount of flow is given by the *capacity* of that arc. At the **source**, all arcs point away from the node. At the **sink**, all arcs point into the node.

4) The objective is to maximize the total amount of flow from the **source** to the **sink**. This amount is measured in either of two equivalent ways, namely, either the amount leaving the source or the amount entering the sink.

The Ford-Fulkerson Algorithm

After some flows have been assigned to the arcs, the residual network shows the remaining arc capacities (called residual capacities) for assigning additional flows.

Change each directed arc to an undirected arc. The arc capacity in the original direction remains the same and the arc capacity in the opposite direction is zero, so the constraints on flows are unchanged.

Subsequently, whenever some amount of flow is assigned to an arc, that amount is subtracted from the residual capacity in the same direction and added to the residual capacity in the opposite direction.

An **augmenting path** is a directed path from the source to the sink in the residual network such that every arc on this path has strictly positive residual capacity. The minimum of these residual capacities is called the residual capacity of the augmenting path because it represents the amount of flow that can feasibly be added to the entire path. Therefore, each augmenting path provides an opportunity to further augment the now through the original network.

The augmenting path algorithm repeatedly selects some augmenting path and adds a flow equal to its residual capacity to that path in the original network. This process continues until there are no more augmenting paths, so the flow from the source to the sink cannot be increased further. **The key to ensuring that the final solution necessarily is optimal is the fact that augmenting paths can cancel some previously assigned flows in the original network**, so an indiscriminate selection of paths for assigning flows cannot prevent the use of a better combination of flow assignments.

The Augmenting Path Algorithm for the Maximum Flow Problem

1) Identify an augmenting path by finding some directed path from the source to the sink in the residual network such that every arc on this path has strictly positive residual capacity. (If no augmenting path exists, the net flows already assigned constitute an optimal flow pattern.)

2) Identify the residual capacity c* of this augmenting path by finding the minimum of the residual capacities of the arcs on this path. **Increase** the flow in this path by c*.

3) **Decrease by c*** the residual capacity of each arc on this augmenting path. Increase by c* the residual capacity of each arc in the opposite direction on this path. Return to step 1.

- adapted from Hillier and Lieberman

Finding An Augmented Path

1) Begin by determining all nodes that can reached from the source with a single arc with positive residual capacity.

2) For each of these nodes, determine all new nodes that can be reached with a single arc with positive residual capacity.

3) Repeat Step 2

This procedure yields a tree of all nodes that can be reached from the source.

The Max-Flow Min-Cut Theorem

A **cut** is a set of directed arcs containing at least one arc from every directed path from the source to the sink.

The **cut value** of a cut in the sum of the arc capacities of the arcs in the cut.

The Max-Flow Min-Cut Theorem: For any network with a single source and single sink, the maximum feasible flow from source to sink equals the minimum cut value for all cuts of the network.

Delbert Ray Fulkerson
Born: August 14, 1924
Died: January 10, 1976
Biography

Lester Randolph Ford, Jr.
Born: September 23, 1927
Died: February 26, 2017)
Biography

Important Works by Ford and Fulkerson

"Maximal Flow Through a Network," *Canadian Journal of Mathematics*, 8:399 - 404, 1956.

"A Simple Algorithm for Finding Maximal Network Flows," *Canadian Journal of Mathematics*, 9: 210 - 218, 1957.

Flows in Networks, Princeton University Press, 1962.
New Edition in 2010 with a new foreword by Robert G. Bland and James B. Orlin.

Complexity

By adding the flow augmenting path to the flow already established in the graph, the maximum flow will be reached when no more flow augmenting paths can be found.
There is no certainty that this situation will ever be reached. The best that can be guaranteed is that the answer will be correct if the algorithm terminates. If the algorithm runs forever, the flow might not even converge towards the maximum flow.
However, this situation only occurs with irrational flow values.
When the capacities are integers, the runtime of Ford–Fulkerson is bounded by $O(Ef)$ where E is the number of edges in the graph and f is the maximum flow in the graph.
This is because each augmenting path can be found in $O(E)$ time and increases the flow by an integer amount of at least 1 with the upper bound f

A variation of the Ford – Fulkerson algorithm with guaranteed termination and a runtime independent of the maximum flow value is the Edmonds – Karp algorithm, which runs in $O(VE^2)$

Yefim Dinitz
Ben Gurion University

Jack Edmonds
Waterloo

Richard Karp
Berkeley

Dinic, E. A. (1970). "Algorithm for solution of a problem of maximum flow in a network with power estimation". *Soviet Math. Doklady.* 11: 1277 –1280.

Edmonds, Jack; Karp, Richard M. (1972). "Theoretical improvements in algorithmic efficiency for network flow problems". *Journal of the ACM*. Association for Computing Machinery. 19 (2): 248 – 264.