MTM4501-Operations Research

Gökhan Göksu, PhD

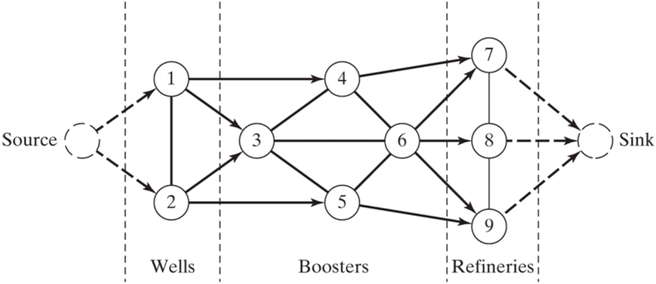
Week5

Course Content

- Definition of OR and Its History
- Decision Theory and Models
- Network Analysis
    - Maximal Flow Algorithm
- Inventory Management Models
- Queue Models

Maximal Flow Algorithm

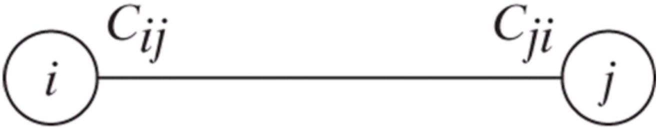
Consider the oil pipeline network that carries crude oil from wells to refineries. Intermediate boosters and pumping stations are installed at appropriate design distances to transport crude oil in the network. Each piece of pipe has a finite maximum rate of oil flow (or capacity). A piece of pipe can be unidirectional or bidirectional, depending on its design. A unidirectional part has a finite capacity in one direction and zero capacity in the opposite direction. A typical pipeline network is given as an example in the figure. How can we determine the maximum capacity of the network between wells and refineries?



Maximal Flow Algorithm

The solution to the proposed problem requires that the network should not be transformed into a single source and single connection network. This requirement is fulfilled by one-way infinite capacity connections as shown in the previous figure.

Given the connection (i, j) where $i < j$, we use the notation $(\overline{C}_{ij}, \overline{C}_{ji})$ to show the output capacities $i \rightarrow j$ and $j \rightarrow i$, respectively. To eliminate the ambiguity, $\overline{C}_{ij}$ is placed on the link close to node i and $\overline{C}_{ji}$ is placed on the link close to node j as shown in the figure below.



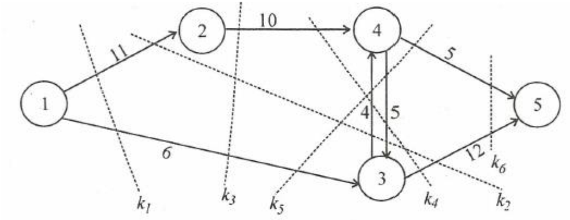
Enumeration of Cuts

A **cut** is defined as a set of arcs which when deleted from network will cause a total disruption of flow between the source and sink nodes. **Cut capacity** is equal to the sum of the capacity of the relevant arcs. Among all possible cuts in the network, the *smallest capacity* cut gives the maximum flow in the network.

In other words, **cut** is the division of the set of nodes N into two subsets S and $\overline{S} = N \setminus S$. Based on this, **cut** can be defined as the set of connections whose endpoints belong to two different subsets S and $\overline{S}$.

Example

By determining the capacities of all cuts, find the maximum flow from the 1^{st} *source node to the* 5^{th} *sink node in the network below.*



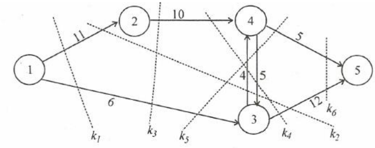
Enumeration of Cuts

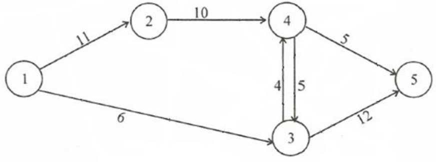
Example

By determining the capacities of all cuts, find the maximum flow from the 1^{st} *source node to the* 5^{th} *sink node in the network below.*

$$k_1 \ : \ C(X,\overline{X}) = 17, \quad k_2 \ : \ C(X,\overline{X}) = 27, \quad k_3 \ : \ C(X,\overline{X}) = 16,$$
$$k_4 \ : \ C(X,\overline{X}) = 26, \quad k_5 \ : \ C(X,\overline{X}) = 16, \quad k_6 \ : \ C(X,\overline{X}) = 17,$$

$$k_3 \quad X = \{1,3\}, \ \overline{X} = \{3,4,5\},$$
$$k_5 \quad X = \{1,2,4\}, \ \overline{X} = \{3,5\}.$$

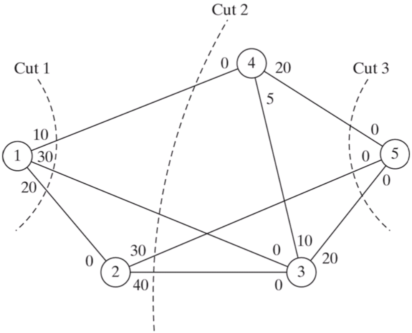




Enumeration of Cuts

Example

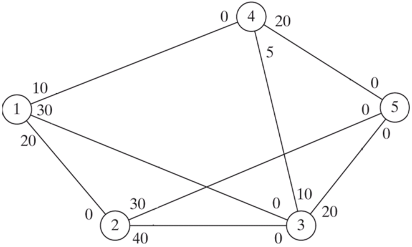
In the network below, two-way capacities are shown on the respective arcs using the previously presented convention. For example, the flow limit for the $(3,4)$ *connection is 10 units from 3 to 4, 5 units from 4 to 3.*



Cut	Associated Arcs	Capacity
1	$(1,2),(1,3),(1,4)$	$20+30+10=60$
2	$(1,3),(1,4),(2,3),(2,5)$	$30+10+40+30=110$
3	$(2,5),(3,5),(4,5)$	$30+20+20=70$

Enumeration of Cuts

All we can get from the three segments is that the maximum flow in the network does not exceed 60 units. We can not tell what the maximum flow will be unless we count **all** segments in the network. Unfortunately, counting all segments is not a simple task. Therefore, the counting procedure is not used to determine the maximum flow in the network.



Exercise

Determine the maximum flow by finding all segments in the network given above.

Maximal Flow Algorithm

The main idea of the maximal flow algorithm is to find the breakthrough paths with net positive flow connecting the source and sink nodes.

Consider the arc (i, j) with initial capacity $(\overline{C}_{ij}, \overline{C}_{ji})$. As portions of these capacities are commited to the flow in the arc, the **residuals** (remaining capacities) of the arc will be updated. The notation (c_{ij}, c_{ji}) is used to represent the remaining capacities. The network with updated remaining capacities is also called **residual capacity**.

The label $[a_j, i]$ is defined for node j, which receives the flow from node i. Here a_j is the flow from node i to node j.

Maximal Flow Algorithm

The steps of the algorithm are given below:

- **Step 1:** Set the capacity equal to the initial capacity for all arcs (i, j), i.e. $(c_{ij}, c_{ji}) = (\overline{C}_{ij}, \overline{C}_{ji})$. Let $a_1 = \infty$ and label source node 1 with $[\infty, -]$. Set $i = 1$ and go to Step 2.
- **Step 2:** Set S_i to be the set of unlabeled nodes j that can be reached directly from node i by arcs with *positive* residuals (i.e. $c_{ij} > 0$ for all $j \in S_i$). If $S_i \neq \emptyset$, go to Step 3, otherwise go to Step 4.
- **Step 3:** Determine $k \in S_i$ such that

  $$c_{ik} = \max_{j \in S_i}\{c_{ij}\}.$$

  Set $a_k = c_{ik}$ and label node k with $[a_k, i]$. If the sink node is labeled (in other words, if $k = n$) and the breakthrough path is found, proceed to Step 5. Otherwise, set $i = k$ and go to Step 2.
- **Step 4 (backtracking):** If $i = 1$, no breakthrough is possible; go to Step 6. Otherwise, label r as the node preceding node i and remove i from the set of nodes adjacent to r. Set $i = r$ and go to the Step 2.

Maximal Flow Algorithm

- **Step 5 (determination of residuals):** Let $N_p = \{1, k_1, k_2, ..., n\}$ define the nodes of the p^{th} breakthrough path from source node 1 to sink node n. Then maximum flow along the path is computed as

  $$f_p = \min\{a_1, a_{k1}, a_{k2}, ..., a_n\}$$

  The residual capacity of each arc along the breakthrough path is decreased by f_p in the flow direction and increased by f_p in the reverse direction. That is, the residual flow is changed from the current (c_{ij}, c_{ji}) to

  - **(a)** $(c_{ij} - f_p, c_{ji} + f_p)$ if the flow is from i to j,
  - **(b)** $(c_{ij} + f_p, c_{ji} - f_p)$ if the flow is from j to i,

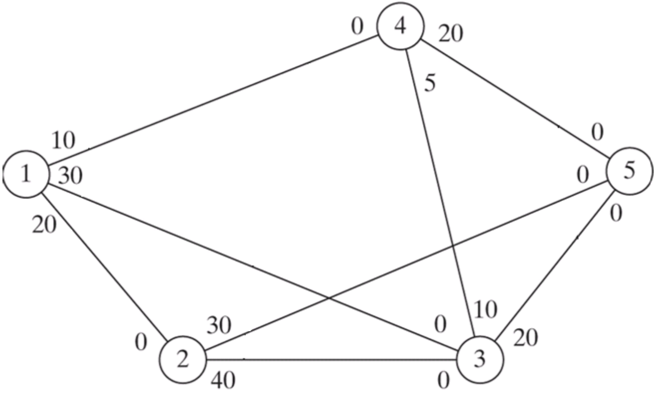
  Reinstate any nodes that were removed in Step 4. Set $i = 1$ and return to Step 2 to attempt a new breakthrough path.

- **Step 6 (solution):**
  - **(a)** Given that m breakthrough paths have been determined, the maximal flow of the network is
    $$F = f_1 + f_2 + ... + f_m.$$
  - **(b)** The *initial* and *final* residuals of arc (i, j) are $(\overline{C}_{ij}, \overline{C}_{ji})$ and (c_{ij}, c_{ji}), respectively. The optimal flow in arc (i, j) is calculated as follows: Let $(\alpha, \beta) = (\overline{C}_{ij} - c_{ij}, \overline{C}_{ji} - c_{ji})$. If $\alpha > 0$, the optimal flow from i to j is α. Otherwise, if $\beta > 0$, the optimal flow from j to i is β. (It is impossible to have both positive α and β.)

Maximal Flow Algorithm

Example

Determine the maximum flow in the following network.



Maximal Flow Algorithm

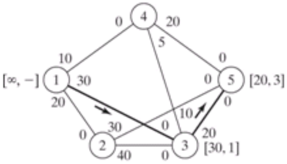
- **Iteration 1:** Set the initial residuals (c_{ij}, c_{ji}) to the initial capacities $(\overline{C}_{ij}, \overline{C}_{ji})$.
  - **Step 1:** Set $a_1 = \infty$ and label node 1 with $[a_1, -]$. Set $i = 1$.
  - **Step 2:** $S_1 = \{2, 3, 4\}$ $(\neq \emptyset)$.
  - **Step 3:** $k = 3$, because $c_{13} = \max\{c_{12}, c_{13}, c_{14}\} = \max\{20, 30.10\} =$30. Set $a_3 = c_{13} = 30$, label node 3 with $[30, 1]$. Set $i = 3$ and repeat step 2.
  - **Step 2:** $S_3 = \{4, 5\}$.
  - **Step 3:** $k = 5$ and $a_5 = c_{35} = \max\{10, 20\} = 20$. Label node 5 with $[20, 3]$. Breakthrough is achieved. Go to step 5.
  - **Step 5:** The breakthrough path consists of labels starting at node 5 and moving backward to node 1:

$$(5) \rightarrow [20, 3] \rightarrow (3) \rightarrow [30, 1] \rightarrow (1).$$

Thus $N_1 = \{1, 3, 5\}$ and $f_1 = \min\{a_1, a_3, a_5\} = \min\{\infty, 30, 20\} = 20$. The residual capacities along N_1 are as follows:

$$(c_{13}, c_{31}) = (30 - 20, 0 + 20) = (10, 20)$$

$$(c_{35}, c_{53}) = (20 - 20, 0 + 20) = (0, 20)$$

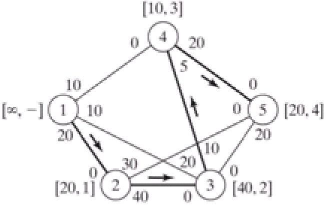


$f_1 = 20$

Maximal Flow Algorithm

- **Iteration 2:**
  - **Step 1:** Set $a_1 = \infty$ and label node 1 with $[a_1, -]$. Set $i = 1$.
  - **Step 2:** $S_1 = \{2, 3, 4\}$.
  - **Step 3:** $k = 2$ and $a_2 = c_{12} = \max\{20, 10, 10\} = 20$. $a_3 = c_{13} = 30$. Set $i = 2$ and repeat step 2.
  - **Step 2:** $S_2 = \{3, 5\}$.
  - **Step 3:** $k = 3$ and $a_3 = c_{23} = 40$. Label node 3 with $[40, 2]$. Set $i = 3$ and repeat step 2.
  - **Step 2:** $S_3 = \{4\}$ (Note that c_{35}, so node 5 is not an element of S_3).
  - **Step 3:** $k = 4$ and $a_4 = c_{34} = 10$. Label node 4 with $[10, 3]$. Set $i = 4$ and repeat step 2.
  - **Step 2:** $S_4 = \{5\}$ (Note that nodes 1 and 3 are already labeled, so these nodes are not included in S_4).
  - **Step 3:** $k = 5$ and $a_5 = c_{45} = 20$. Label node 5 with $[20, 4]$. Breakthrough has been achieved. Go to step 5.
  - **Step 5:** $N_2 = \{1, 2, 3, 4, 5\}$ and $f_2 = \min\{\infty, 20, 40, 10, 20\} = 10$. The residuals along N_2 are:

$$
\begin{aligned}
(c_{12}, c_{21}) &= (20 - 10, 0 + 10) = (10, 10)\\
(c_{23}, c_{32}) &= (40 - 10, 0 + 10) = (30, 10)\\
(c_{34}, c_{43}) &= (10 - 10, 5 + 10) = (0, 15)\\
(c_{45}, c_{54}) &= (20 - 10, 0 + 10) = (10, 10)
\end{aligned}
$$

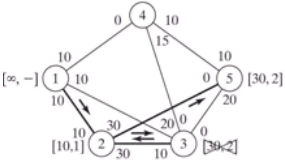


$f_2 = 10$

Maximal Flow Algorithm

- **Iteration 3:**
  - **Step 1:** Set $a_1 = \infty$ and label node 1 with $[a_1, -]$. Set $i = 1$.
  - **Step 2:** $S_1 = \{2, 3, 4\}$.
  - **Step 3:** $k = 2$ and $a_2 = c_{12} = \max\{10, 10, 10\} = 10$ (random selection in case of tie). Label node 2 with $[10, 1]$. Set $i = 2$ and repeat step 2.
  - **Step 2:** $S_3 = \{3, 5\}$.
  - **Step 3:** $k = 3$ and $a_3 = c_{23} = 30$. Label node 3 with $[30, 2]$. Set $i = 3$ and repeat step 2.
  - **Step 2:** $S_3 = \emptyset$ (because $c_{34} = c_{35} = 0$). Go to step 4 to backtrack.
  - **Step 4:** The label $[30, 2]$ at node 3 gives the immediately preceding node $r = 2$. The 3rd node is crossed out to be removed from further consideration in this iteration. Set $i = r = 2$ and repeat step 2.
  - **Step 2:** $S_2 = \{5\}$ (note that node 3 has been removed from the backtracking step).
  - **Step 3:** $k = 5$ and $a_5 = c_{25} = 30$. Label node 5 with $[30, 2]$. Breakthrough has been achieved, go to step 5.
  - **Step 5:** $N_3 = \{1, 2, 5\}$ and $f_3 = \min\{\infty, 10, 30\} = 10$. The residuals along N_3 are:

$$(c_{12}, c_{21}) = (10 - 10, 10 + 10) = (0, 20)$$
$$(c_{25}, c_{52}) = (30 - 10, 0 + 10) = (20, 10)$$



$f_3 = 10$

Maximal Flow Algorithm

- **Iteration 4:** This iteration gives $N_4 = \{1, 3, 2, 5\}$ with $f_4 = 10$ (Exercise!!!)
- **Iteration 5:** This iteration gives $N_5 = \{1, 4, 5\}$ with $f_5 = 10$ (Exercise!!!)
- **Iteration 6:** Since all links except node 1 has zero residuals, there is no possible further breakthroughs.
  - **Step 6:** The maximal flow in the network is

    $$F = f_1 + f_2 + ... + f_5 = 20 + 10 + 10 + 10 + 10 = 60.$$

    The flows on different arcs are calculated by subtracting the last residuals in iteration 6 (i.e. $(c_{ij}, c_{ji})_6$) from the initial capacities $(\overline{C}_{ij}\overline{C}_{ji})$ as given in the table below.

Arc	$(\overline{C}_{ij}, \overline{C}_{ji}) - (c_{ij}, c_{ji})_6$	Flow Amount	Direction
(1,2)	$(20, 0) - (0, 20) = (20, -20)$	20	$1 \to 2$
(1,3)	$(30, 0) - (0, 30) = (30, -30)$	30	$1 \to 3$
(1,4)	$(10, 0) - (0, 10) = (10, -10)$	10	$1 \to 4$
(2,3)	$(40, 0) - (40, 0) = (0, 0)$	0	–
(2,5)	$(30, 0) - (10, 20) = (20, -20)$	20	$2 \to 5$
(3,4)	$(10, 0) - (0, 15) = (10, -10)$	10	$3 \to 4$
(3,5)	$(20, 0) - (0, 20) = (20, -20)$	20	$3 \to 5$
(4,5)	$(20, 0) - (0, 20) = (20, -20)$	20	$4 \to 5$

Linear Programming Formulation of the Maximal Flow Algorithm

Consider the network expressed as $G = (N, A)$. Here N is the set of nodes and A is the set of links.

$$\begin{aligned} x_{ij} &: \text{ amount of flow in arc } (i, j) \\ c_{ij} &: \text{ capacity of arc } (i, j) \\ s &: \text{ source node index} \\ t &: \text{ sink node index} \end{aligned}$$

The linear programming formulation for this network using these notations is as follows:

$$\min\ V = \sum\sum_{j \in N} x_{sj}$$

$$\text{subject to } \sum_{j \in N} x_{ij} - \sum_{k \in N} x_{jk} = \begin{cases} -V, & j = s \\ 0, & j \neq s, t \\ +V, & j = t \end{cases}$$

$$0 \leq x_{ij} \leq c_{ij}, \quad \forall i, j \in N$$

Linear Programming Formulation of the Maximal Flow Algorithm

In the maximal flow model for the network with $s = 1$ and $t = 5$, two different linear programming problems can be considered. These two problems, in which the output from the starting node 1 or the input to the terminal node 5 is maximized, give essentially the same solution and are summarized below.

	x_{12}	x_{13}	x_{14}	x_{23}	x_{25}	x_{34}	x_{35}	x_{43}	x_{45}		
max z_1	1	1	1								
max z_2					1		1		1		
Node 2	1			-1	-1					$=$	0
Node 3		1		1		-1	-1	1		$=$	0
Node 4			1			1		-1	-1	$=$	0
Capacities	20	30	10	40	30	10	20	5	20		

The optimal solution obtained using either objective function will be

$$x_{12} = 20,\ x_{13} = 30,\ x_{14} = 10,\ x_{25} = 20,\ x_{34} = 10,\ x_{35} = 20,\ x_{45} = 20,$$

and the associated maximum flow is

$$z_1 = z_2 = 60.$$