175 Years of Linear Programming

5. Max Flow = Min Cut

Vijay Chandru and M R Rao

Vijay Chandru is a Professor of computer science & automation at the Indian Institute of Science. His research interests are in optimisation, geometry, logic and computer graphics. He confesses to being a linear programming junkie.

M R Rao is Director and Professor of quantitative methods and information systems at the Indian Institute of Management, Bangalore. His research interests include optimisation and risk management in the context of managerial decision making.

The theory of flows in networks began to evolve in the early 1950's.The various linear optimisation questions that could be asked of flows in conserving networks turned out to be neat combinatorial specialisations of linear programming. The simplex method (and its variants) turned out to have very pretty combinatorial interpretations on networks. The algebraic dexterity of linear programming duality led to a unified treatment of many deep theorems in graph theory and combinatorics. In this part, the last of the series on linear programming, we will see glimpses of the theory of network flows through a specific flow optimisation problem – the maximum flow problem.

Maximum $s \to t$ Flow

Networks are perhaps the most familiar and ubiquitous mathematical models to most of us. Transportation networks, electrical power transmission networks, telecommunication networks and pipeline networks (for water, natural gas, etc.) are physical networks that we deal with constantly. The two fundamental optimisation problems on networks are routing problems (shortest paths) and flow problems (maximum flow, minimum cost transshipment, etc.). Linear programming has been used as a framework for designing efficient algorithms for both routing and flow problems on networks. In this part, we will see some of the connections between flow problems on networks and linear programming. Linear programming duality (see Part 1 of this series) will be shown to translate into a neat combinatorial maximin theorem relating maximum flows with minimum capacity cuts.

To study flow problems, we first need to pin down what

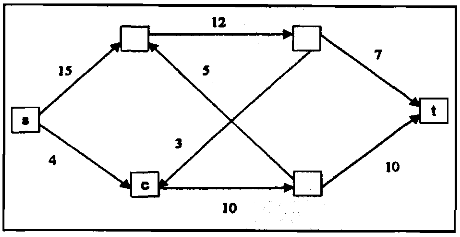


Figure 1. A Flow network.

we mean by a flow network (see *Figure* 1). We are given a network (or a directed graph) $D(\mathcal{N}, \mathcal{A})$ with $\mathcal{N}$ denoting the nodes of the network and $\mathcal{A}$ denoting the arcs or directed links between pairs of nodes. A flow network has, in addition, capacities c_{ij} associated with each arc $(i,j) \in \mathcal{A}$. The capacity denotes the maximum number of units of flow that is permitted to pass through an arc. Unless otherwise specified, we assume that these capacities are non-negative integers. Flow networks can also have costs associated with the flows. Linear cost flows will require a cost per unit flow associated with each arc.

175 Years of Linear Programming
Vijay Chandru and M R Rao

1. The French Connection. Fourier's Algorithm and LP Duality, *Resonance*, Vol.3, No.10, 1998.
2. Pivots in Column Space. The Simplex Method, *Resonance*, Vol.4, No.1, 1999.
3. Pune's Gift. Karmarkar's Projective Scaling Method, *Resonance*, Vol.4, No.5, 1999.
4. Minimax and Cake von Neumann's Minimax Theorem. *Resonance*, Vol.4, No.7, 1999.
5. MaxFlow equals MinCut, The Theory of Network Flows.

A maximum flow problem on a capacitated flow network asks for the value of the maximum flow possible in a network where we assume that all flow starts from the node s and terminates at the node t. The flows are required to stay within the capacity limits and satisfy flow conservation at all nodes other than s and t. A *flow* is a vector $\mathbf{f} = (f_{ij})$ where each f_{ij} is a real number representing the flow on arc (i,j), i.e., the flow from i to j. Now, $\mathbf{f}$ is a *feasible flow* if it meets the conditions:

(i) Flow Bounds $0 \leq fij \leq c_{ij} \ \forall (i,j) \in \mathcal{A}$

(ii) Flow Conservation $\sum_{(i,j)\in\mathcal{A}} f_{ij} - \sum_{(k,i)\in\mathcal{A}} f_{ki} = 0 \ \forall i \in \mathcal{N} \setminus \{s,t\}$.

The *total flow* $F(\mathrm{f})$ from s to t is given by

(iii) Total Flow: $F(\mathrm{f}) = \sum_{(i,t)\in\mathcal{A}} f_{it} - \sum_{(t,j)\in\mathcal{A}} f_{t,j}$.

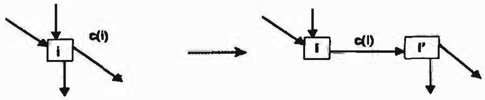


Figure 2 (left). Undirected arc.
Figure 3(right). Node capacities.

Flow conservation (condition (ii) above) implies that

$$F(\mathbf{f}) = \sum_{(s,j)\in\mathcal{A}} f_{s,j} - \sum_{(i,s)\in\mathcal{A}} f_{i,s}$$

i.e., total flow is the same as the net flow out of source s which is the same as the net flow into the sink or terminus t. The maximum flow problem is to find a *feasible flow* $\mathbf{f}$ that maximizes the total flow $F(\mathbf{f})$. In the example flow network shown in *Figure* 1, it is easy to see that the maximum flow from s to t is 14 units (an algorithmic proof will be described in the next section).

Remarks: A couple of reformulation 'tricks' help in handling flow networks that may permit undirected or 'two way' flows in arcs and/or node capacities.

Undirected Flows: Replace each two arc (i, j) with capacity $\tilde{c}_{ij}$ with two arcs (i, j) and (j, i) each having capacity equalling $\tilde{c}_{ij}$ i.e., $c_{ij} = c_{ji} = \tilde{c}_{ij}$ (see *Figure* 2).

Node Capacities: Suppose we have a capacity of $c(i)$ on the total flow through a node i. We handle this by splitting i into i' and i'' as indicated in *Figure* 3. The arc (i', i'') inherits the node capacity c_i and all other capacities remain unchanged.

In addition, we will see that non-zero lower capacities on arcs can also be imposed without seriously affecting the theory and algorithms for maximising flow in a network.

Minimum Capacity $s \to t$ Cuts

Let $S \subseteq \mathcal{N}$ be a subset of nodes such that $s \in S$ and $t \notin S$. $\overline{S}$ is the complement of S ie $\overline{S} = \mathcal{N} \setminus S$. Let $(S; \overline{S})$ denote the set of arcs with tail in S and head in $\overline{S}$. So,

$$\begin{aligned}(S,\overline{S}) &= \{(i,j)\in\mathcal{A} : i\in S, j\in\overline{S}\} \\ (\overline{S},S) &= \{(k,\ell)\in\mathcal{A} : k\in\overline{S}, \ell\in S\}.\end{aligned} \qquad (1)$$

The collection of arcs $(S,\overline{S})\bigcup(\overline{S},S)$ is called an $s\to t$ *cut* of the network. Evidently, $(S,\overline{S})$ are the *forward arcs* and $(\overline{S},S)$ the *reverse arcs* of the cut. Let us first observe that flows across *any* $s\to t$ cut captures the total $s\to t$ flow in every network.

Lemma: For every S such that $s\in S$ and $t\in\overline{S}$ and every conserving flow f we have

$$F(\mathbf{f}) = \sum_{(i,j)\in(S;\overline{S})} f_{ij} - \sum_{(k,\ell)\in(\overline{S};S)} f_{k\ell}.$$

Proof: Recall the flow conservation equations (ii) corresponding to nodes in $\overline{S}\setminus\{t\}$.

$$-\sum_{(i,j)\in\mathcal{A}} f_{ij} + \sum_{(k,i)\in\mathcal{A}} f_{ki} = 0 \text{ for all } i\in\overline{S}\setminus\{t\}.$$

Now, recall the equation (iii) that measures total $s\to t$ flow

$$\sum_{(j,t)\in A} f_{jt} - \sum_{(t,k)\in A} f_{tk} = F(\mathbf{f}).$$

Let us add all these equations and observe that the lemma holds. In the resulting expression on the left hand side, if $(x,y)\in\mathcal{A}$ and both $x,y\in S$ then f_{xy} does not appear at all. If $x,y\in\overline{S}$ then f_{xy} appears once with a $+1$ coefficient (from the y equation) and once with a -1 coefficient (from the x equation) and hence cancels in the sum. If $x\in S$ and $y\in\overline{S}$ then f_{xy} appears once with a $+1$ coefficient from y equation. Finally if $x\in\overline{S}, y\in S$ then f_{xy} appears once with a -1 coefficient from the x equation. This proves the lemma. □

The lemma has a simple physical interpretation. An $s\to t$ cut is simply a dam with a sluice erected across the network to monitor the flow from the s side to the t side. The net flow through the sluice must represent the total flow from s to t, and it really does not matter where we erect the dam as long as s and t fall on opposing sides.

Definition: We define the *capacity* of the $s \to t$ cut defined by S (with $s \in S$ and $t \notin S$) as

$$c(S) = \sum_{(i,j)\in(S,\overline{S})} c_{i,j}.$$

Notice that the capacity of the cut is the sum of the capacities of the forward arcs of the cut. From the lemma above, we know that total $s \to t$ flow equals the flow across any cut. The flow across any cut can be no larger than the sum of the capacities of the forward arcs. This leads to a simple weak duality relation between $s \to t$ flows and $s \to t$ cuts.

Weak Duality Lemma: For any feasible $s \to t$ flow $\mathbf{f}$ and any $s \to t$ cut defined by S separating s from t we have $F(\mathbf{f}) \leq c(S)$.

Box 1. The Ford–Fulkerson Algorithm

(0) Assign some initial flow to the arcs; an assignment $f_{ij} = 0$ forall $(i,j) \in \mathcal{A}$ works when $c_{ij} \geq 0$ and lower capacities are 0.

(1) Mark s labeled and all other nodes unlabeled.

(2) Search for a node v that can be labeled by either a forward or backward labeling. If none exists stop (we have the maximum flow). If we succeed in labeling t we go to step 3, else we repeat step 2.

(3) Starting from t and using the labels, backtrack through the augmenting path. Let us denote the path as the sequence of arcs $\{a(1), a(2), \cdots a(k)\}$.
The flow augmentation possible is $\Delta = \min_{1\leq i\leq k} \Delta(a(i))$.
If $a(j)$ is *forward* in this path we set $f_{a(j)} \leftarrow f_{a(j)} + \Delta$.
If $a(j)$ is reverse (*backward*) set $f_{a(j)} \leftarrow f_{a(j)} - \Delta$.
Return to Step (1).

A consequence of the weak duality lemma is that if we can exhibit a feasible flow $\mathbf{f}^*$ and a valid cut defined by S^* such that $F(\mathbf{f}^*) = c(S^*)$ then $\mathbf{f}^*$ is maximum $s \rightarrow t$ flow vector and S^* is an $s \rightarrow t$ cut of minimum capacity. Indeed, we shall always succeed in showing that maximum flow equals minimum cut by a constructive technique known as the augmenting path method of Ford and Fulkerson.

The Augmenting Path Method

Given a feasible flow vector $\mathbf{f}$, an *augmenting path* is a simple path from s to t which is not necessarily directed, but can be used to increase flow from s to t. Hence if arc (i, j) is forward in this path (i.e., (i, j) points in the direction $s \rightarrow t$) then f_{ij} must satisfy $0 \leq f_{ij} < c_{ij}$ and if (i, j) is reverse then $0 < f_{ij} \leq c_{ij}$. For example, the path $s \rightarrow c \rightarrow d \rightarrow a \rightarrow b \rightarrow t$ (see *Figure* 4) is augmenting with respect to the feasible flow $\mathbf{f} = 0$ in the flow network of *Figure* 1.

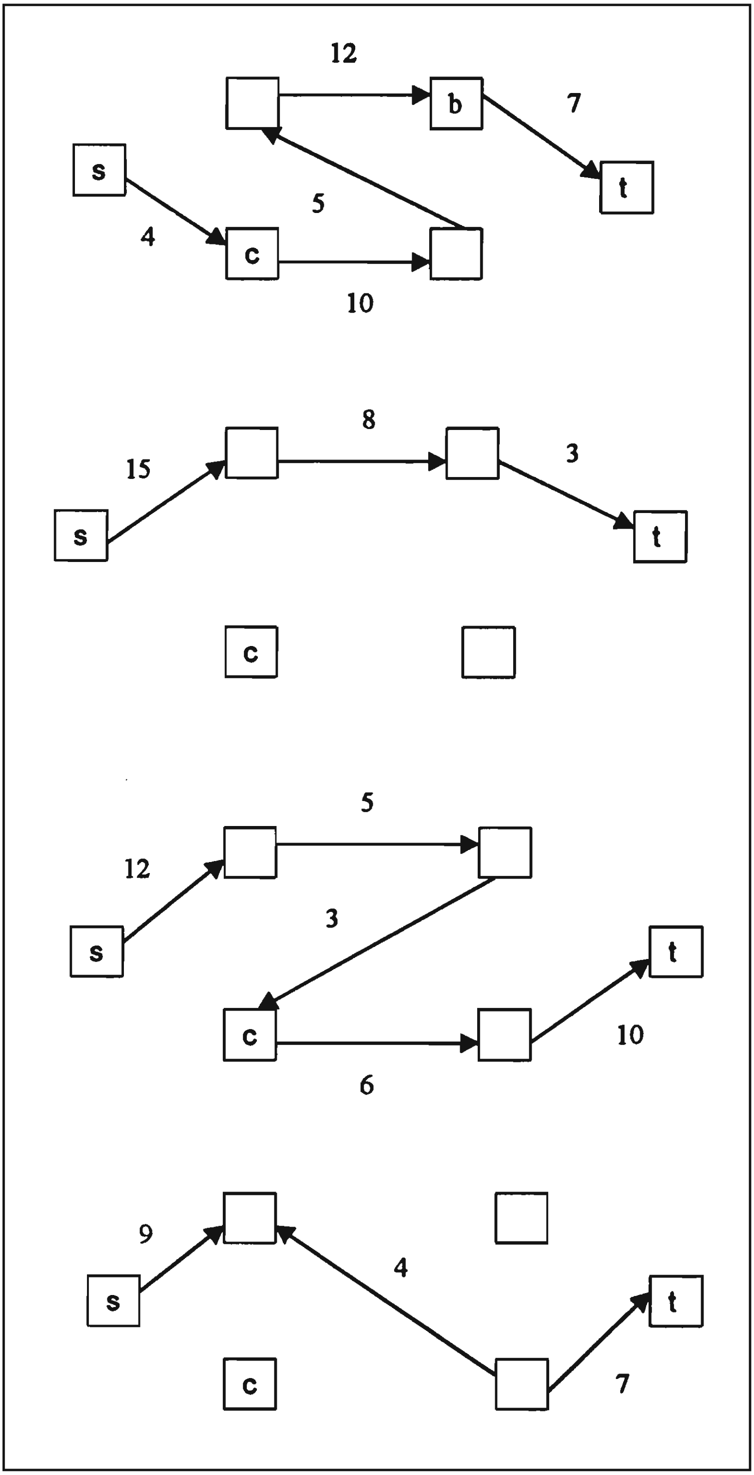
A simple labeling method may be used to detect augmenting paths. Initially s is labeled and then we label every node v connected from s by an augmenting path. If we succeed in labeling t we have an augmenting path to increase (augment) flow and the procedure is repeated. If we are unable to label t from s we will have the maximum flow.

A *forward labeling* of node v by arc (u, v) is applicable if u is labeled and v is not, and in addition $f_{u,v} < c_{u,v}$. The label that v gets is 'uv'. Define $\Delta(u, v) = c_{u,v} - f_{u,v}$.

A *backward labeling* of node k by the arc (k, ℓ) is allowed if ℓ is labeled and k is not, and in addition, $f_{k,\ell} > 0$. The label applied to k is 'k, ℓ' and $\Delta(k, \ell) = f_{k,\ell}$.

Let us run the algorithm on the example flow network of *Figure* 1. One possible outcome is depicted in *Figure* 4. We get four augmenting paths and are able to augment the flow from 0 to 4 to 7 to 10 to 14. Note that the last augmenting path $s \rightarrow a \leftarrow d \rightarrow t$ includes a reverse arc (i.e., in the resulting flow augmentation we back down the flow on the reverse arc (d, a) by 4 units)

Figure 4. Augmenting paths.



Proposition: The Ford–Fulkerson method always terminates with the maximum flow from s to t.

Proof: It is evident that when the algorithm terminates, it does so with a maximal flow (since no further flow augmentations are possible). That the flow is maximum is argued as follows. Let S^* denote the set of nodes labeled from s, in the final pass of the algorithm. Note that the algorithm stops in step (2) only if we are unsuccessful at labeling t and hence $t \in \overline{S}^*$ Every arc (u, v) with $v \in S^*$ and $v \in \overline{S}^*$ must be flow saturated ie $f^*_{u,v} = c_{u,v}$, otherwise v would be forward labeled from u. Also, each arc (k, ℓ) with $k \in \overline{S}^*, \ell \in S^*$ must have zero flow $f^*_{k\ell} = 0$ since otherwise k could be backward labeled from ℓ. Thus we have

$$\sum_{(u,v)\in(S^*,\overline{S}^*)} f^*_{u,v} - \sum_{(k,\ell)\in(\overline{S}^*,S^*)} f^*_{k,\ell} = c(S^*)$$

From the first lemma we know that the LHS is $F(f^*)$. Thus we have exhibited a total flow equalling the capacity of a cut. By the weak duality lemma we have found both a maximum $s \to t$ flow and a minimum capacity $s \to t$ cut.

It remains to show that the algorithm indeed terminates. This follows from the integrality of the capacities. Since we start with integer flows (**f**=0) and each flow augmentation is a positive integer ($\Delta \geq 1$), it follows that the algorithm terminates after a finite number of iterations and that the resulting maximum flow is an all integer vector. □

Note that rational capacities are no different from integer capacities since we can always scale down the units of flow by an appropriate integer (lcm of the denominators) and revert to integer capacities. If the capacities are irrational some pathologies can occur. In fact the Ford–Fulkerson algorithm, as described above, can converge to a sub-optimal solution. However, if we are a bit careful about choosing augmenting paths, it can be shown that the algorithm will always converge to the maximum flow.

We have proved, through constructive means (an algorithm), the following remarkable theorem.

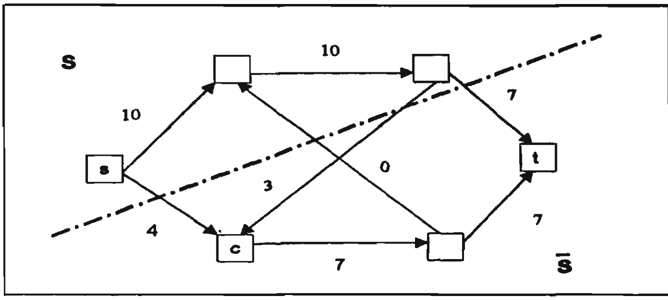
Max Flow = Min Cut (MFMC) Theorem: In any flow network, the maximum $s \to t$ flow equals the minimum capacity of an $s \to t$ cut. Further, if the flow network has integer capacities, the maximum flow is achievable by an all integer flow vector.

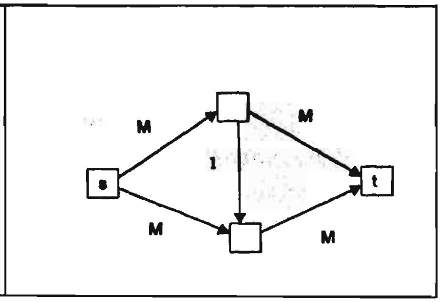
The all integer maximum flow as well as the corresponding minimum cut for the flow network of *Figure* 1 are shown in *Figure* 5.

Remarks: If capacities (c) are assumed to be integers and z^* denotes the maximum flow value $z^* = F(\mathbf{f}^*)$, then the Ford–Fulkerson algorithm has computational complexity bounded by $O(z^*|A|)$. The number of augmentations is bounded by z^* since $\Delta \geq 1$ at each iteration and the augmenting path construction requires at most $2|A|$ arc inspections. This complexity bound is disappointing in the sense that the number of steps is tied directly to the *magnitude* of the capacities. Hence it is not even a polynomial bound (a polynomial-time algorithm can depend on z^* only in a poly-log manner ie $\log^k(z^*)$ for some fixed $k \geq 0$). Ideally we would like the algorithm to have its time – complexity depending only on the size of the graph (ie $|\mathcal{N}|, |\mathcal{A}|$) in a polynomial manner. Such algorithms are called *strongly polynomial* or *genuinely polynomial.*

That the Ford–Fulkerson algorithm actually may take $O(z^*)$ augmentations is illustrated by the flow network in *Figure* 6. The algorithm takes 2M augmentations if the two augmenting paths, $s \to 1 \to 2 \to t$ and $s \to 2 \leftarrow 1 \to t$, are alternated.Notice that the algorithm would have terminated

Figure 5 (left). Maximum Flow = Minimum Cut.
Figure 6 (right). Exponential convergence.





in just two iterations if we had used augmentating paths having the minimum number of arcs i.e., $s \to 1 \to t$ and $s \to 2 \to t$. Indeed, Edmonds and Karp [1] proved that if in the Ford–Fulkerson labeling method, the augmenting paths are chosen to be of minimum cardinality, then the complexity of the algorithm is strongly polynomial.

An enormous amount of effort has been put into improving the asymptotic efficiency of algorithms for the maximum flow problem. The books by Tarjan and Ahuja and others [2,3] are good sources for studying these developments.

Dealing with Lower Capacities on Arcs: In some applications of network flow models, we need to be able to impose lower capacities on arcs. In this case a feasible flow $\mathbf{f}$ has to satisfy

$$b_{i,j} \le f_{i,j} \le c_{i,j} \quad \forall (i,j) \in \mathcal{A}.$$

The entire development described above can be easily modified to deal with this case by noting that *residual capacity* when an arc is *reverse* to the flow direction should now be taken to be $(f_{i,j} - b_{i,j})$ instead of $f_{i,j}$. The only issue that needs resolution is that of finding an initial feasible flow ($\mathbf{f} = 0$ does not work any more).

Following linear programming paradigms we can set up a 'Phase I' max flow problem for which an initial feasible flow is evident. After solving the Phase I problem to optimality we will have an initial feasible flow for the 'Phase II' problem (the original problem) or a message that none exists. Underlying this construction is a 'feasibility' version of the MFMC theorem (a Farkas lemma analogue for network flows) – the Hoffman circulation theorem. We will skip the details of the constructions, but will present the formal statement of the circulation theorem.

Once we have a feasible flow assignment to start, the Ford Fulkerson algorithm can be applied to obtain both a maximum $s \to t$ flow and a minimum capacity $s \to t$ cut. The capacity of a cut $(S, \overline{S})$, in the presence of lower capacities, is redefined to be

$$c(S) = \sum_{(i,j)\in(S,\overline{S})} c_{i,j} - \sum_{(j,i)\in(\overline{S},S)} b_{j,i}.$$

The circulation theorem is stated for flow networks which have no distinguished source or sink nodes. There is an easy formulation trick to convert $s \to t$ networks into circulation networks. We need to introduce an extra arc, the circulant or return arc (t, s) with capacities $(0, \infty)$. Let $\tilde{\mathcal{A}}$ denote $\mathcal{A} \cup \{(t, s)\}$. Adding such an arc allows us to write flow conservation equations at s and t as well.

Let us define $\mathbf{f}$ to be a *feasible circulation* in a capacitated flow network $\mathcal{D}(\mathcal{N}, \tilde{\mathcal{A}})$ if

(i) $\sum_{(u,v)\in\tilde{\mathcal{A}}} f_{u,v} - \sum_{(v,w)\in\tilde{\mathcal{A}}} f_{v,w} = 0$ for all $v \in \mathcal{N}$

(ii) $b_{u,v} \le f_{u,v} \le c_{uv}$ for all $(u, v) \in \tilde{\mathcal{A}}$.

The necessary and sufficient conditions for the existence of a feasible circulation are given by the theorem.

The Circulation Theorem: Assume that $c_{u,v} \ge b_{u,v}$ for all $(u, v) \in \tilde{\mathcal{A}}$, then there exists a feasible circulation for the network $\mathcal{D}(\mathcal{N}, \tilde{\mathcal{A}})$ with capacities b, c if and only if for each $S \subset \mathcal{N}$ we have

$$\sum_{(u,v)\in(S,\overline{S})} c_{u,v} - \sum_{(k,\ell)\in(\overline{S},S)} b_{k,\ell} \ge 0.$$

Various optimisation models can be imposed on feasible circulations by specifying the appropriate linear objective function to be maximised or minimised. To maximise $s \to t$ flow, we simply ask for the maximisation of $f_{t,s}$, the flow on the circulant arc. The celebrated 'out-of-kilter' algorithm of Ford and Fulkerson was designed to work for an arbitrary linear optimisation of feasible circulations.

Linear Programming Interpretations

In this section, we will give linear programming interpretations of the MFMC theorem. We will do this through two completely different formulations – the so-called arc and path formulations. Let us begin with the arc formulation.

Arc Formulation: Now that we are familiar with the circulation form of network flows,the maximum-flow problem can be expressed as the linear programme:

$$(P) \qquad \max f_{t,s} \tag{2}$$
$$\text{s.t } \tilde{N}\mathbf{f} = 0 \tag{3}$$
$$0 \le f_{i,j} \le c_{i,j} \ \forall (i,j) \in \mathcal{A}$$

where $\tilde{N}$ is the node-arc incidence matrix of the circulant network $\mathcal{D}(\mathcal{N}, \tilde{\mathcal{A}})$ with $\tilde{\mathcal{A}} = \mathcal{A} \cup \{(t,s)\}$. The node-arc incidence matrix has rows indexed by the nodes and columns by the arcs. Each column has a $+1$ in the row corresponding to the tail node of the arc and a -1 corresponding to the head node. The rest of the entries are 0. The reader should now check that the matrix equation $\tilde{N}\mathbf{f} = 0$ captures the flow conservation equations of feasible circulations. Thus (P) is exactly the circulation form of the maximum $s \to t$ flow problem.

The linear programming dual of (P) is given by:

$$(D) \qquad \min \sum_{(i,j)\in\mathcal{A}} c_{ij}\gamma_{i,j} \tag{4}$$
$$\text{s.t. } \pi_i - \pi_j + \gamma_{i,j} \ge 0 \ \forall (i,j) \in \mathcal{A} \tag{5}$$
$$\pi_t - \pi_s = 1 \tag{6}$$
$$\gamma_{i,j} \ge 0 \text{ for all } (i,j) \in \mathcal{A}.$$

To see that (D) is a formulation of the minimum capacity $s \to t$ cut problem is a little tricky. However, the following pointers may help. Let $(S, \overline{S})$ denote any $s \to t$ cut with $s \in S$ and $t \in \overline{S}$. Now set π_i to 0 for all $i \in S$ and to 1 for all $i \in \overline{S}$. Set $\gamma_{i,j}$ to 1 if (i,j) is a forward arc in the cut $(S, \overline{S})$ and 0 otherwise. The reader should check now that these values are feasible in (D) and that the objective value is the capacity of the cut $(S, \overline{S})$.

The integrality of optimal solutions (actually all extreme point solutions) to the linear programmes (P) and (D) is a happy consequence of the total unimodularity of the node-arc matrix $\tilde{N}$. A matrix is said to be totally unimodular if every square submatrix has determinant 0 or +1 or −1.

Path Formulation: Flows in networks satisfy a decomposition property known as the conformal decomposition of flows. This property says that if we are given a positive integer value v of total $s \to t$ flow in a network that is achievable, then we can find v directed paths (possibly with repetitions) from $s \to t$ to route flow on and further, these flows will collectively be feasible (that is, they will satisfy the capacity constraints). This is to be distinguished from decomposing flows into augmenting paths, since augmenting paths need not be directed paths from s to t.

In our example flow network (see *Figure* 5) we found that 14 units of flow could be achieved from s to t. Notice that 4 copies of $s \to c \to d \to t$, 7 copies of $s \to a \to b \to t$ and 3 copies of $s \to a \to b \to c \to d \to t$ would be one conformal decomposition.

Let $\mathcal{P}$ denote the collection of directed $s \to t$ paths in the network $\mathcal{D}(\mathcal{N}\mathcal{A})$ and let P denote the path-arc incidence matrix. So P_{ij} would be 1 if arc $a(j)$ is in path p_i and 0 otherwise. The path packing formulation of maximum flow from s to t is

Path Packing Problem (PPP):

$$(PPP) \qquad F = \max \sum_{i:p_i \in \mathcal{P}} y_i \tag{7}$$

$$\text{s.t.} \sum_{i:p_i \in \mathcal{P}} y_i \le c_{a(j)} \text{ for all } a(j) \in \mathcal{A} \tag{8}$$

$$y_i \ge 0 \text{ for all } i : p_i \in \mathcal{P}$$

whose linear programming dual defines the covering problem.

Path Covering Problem (PCP):

$$(PCP) \qquad C = \min \sum_{j:a(j)\in\mathcal{A}} c_{a(j)} x_j \tag{9}$$

$$\text{s.t.} \quad \sum_{j:a(j)\in\mathcal{A}} P_{ij} x_j \geq 1 \forall i : p_i \in \mathcal{P} \tag{10}$$

$$x_j \geq 0 \ \forall j : a(j) \in \mathcal{A}.$$

We have already argued that (PPP) is a formulation of the max $s \to t$ flow problem. It remains to show that (PCP) is a formulation of the minimum capacity $s \to t$ cut problem. This is easy to see since any $s \to t$ cut would necessarily block all $s \to t$ directed paths. Hence, if we take x in (PCP) to be the incidence vector of all forward arcs of any $s \to t$ cut of the network, it would be feasible in (PCP).

Combinatorial Applications

The MFMC theorem has many interesting applications in combinatorics. The applications include problems in graph matching, systems of distinct representatives, partially ordered sets and graph connectivity. In this section we will study only some of these results that can be derived from the MFMC theorem. The complete details of these derivations can be found in the book by Ford and Fulkerson [4].

Bipartite Matching: Consider a *bipartite* network $G_B((S,T);A)$ with $A \subseteq S \times T$ and $|S| = m, |T| = n$. A *matching* $M \subseteq A$ is a set of node disjoint arcs. An *arc cover* $C \subseteq S \cup T$ is a set of nodes that covers all the arcs in $\mathcal{A}$. Equivalently, an arc cover is a set of nodes whose removal (along with all incident arcs) disconnects S from T.

Kónig–Egervary Theorem: $\max\{|M| : M \text{ is a matching}\} = \min\{|C| : C \text{ is an arc cover}\}$

Proof (sketch): Consider the max flow $(s \to t)$ in the network shown in *Figure* 7. Let f^* denote the max flow and $(X^*, \overline{X}^*)$ the min-cut for this problem. Then it is easily verified that

(i) $D = (S \cap \overline{X}^*) \cup (T \cap X^*)$ is an arc cover.

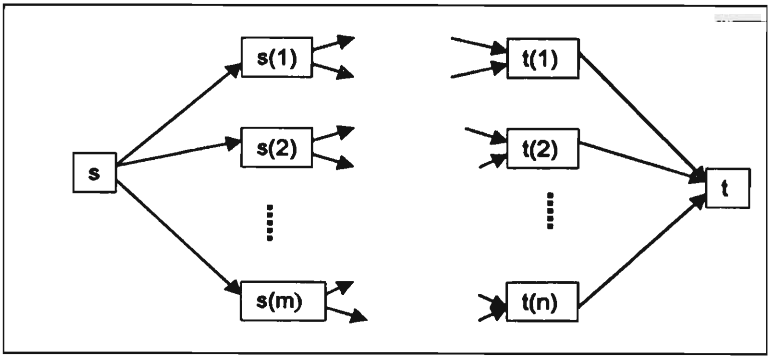


Figure 7. Konig–Egervary theorem.

(ii) $|C^*| = c(X^*) = F(f^*)$ by the MFMC theorem.

(iii) $|M^*| = |C^*|$ where $M^* = \{(i,j) \in \mathcal{A} : f^*_{ij} = 1\}$.

And the theorem follows since by weak duality for any matching M and arc cover C we have $|M| \leq |C|$. □

Graph Connectivity Menger's Theorem (Vertex Form): Given an undirected graph $G(V, E)$ and vertices $s, t \in V$ The maximum number of vertex disjoint $(s - t)$ paths equals the minimum number of vertices whose removal (with incident edges) destroys all $(s - t)$ paths.

Proof (sketch): This is really the MFMC theorem specialized to the case where all vertices in $V \setminus \{s, t\}$ have a capacity of 1. The construction needed to push the proof is

(i) Replace each undirected edge with two directed arcs as in *Figure* 2.

(ii) Use the node-splitting (see *Figure* 3) for handling node capacities.

Steps (i) and (ii) yield a capacitated network and we can apply the max-flow min-cut theorem to this network to obtain the theorem. □

Note that the *vertex connectivity* number of a graph $G(V, E)$ is defined by the pair of vertices which have the smallest number of vertex disjoint paths between them. Also note that Menger's theorem can also be stated for directed graphs. Just replace 'vertex' with 'node', 'edge' with 'arc' and '$(s - t)$path' with '$(s \to t)$ directed path'.

Menger's Theorem (Edge-Form): Given a graph $G(V, E)$ and vertices $s, t \in V$ the maximum number of edge-disjoint paths between s and t equals the minimum number of edges whose removal destroys all paths between s and t.

This is precisely the undirected form of the MFMC theorem with all edges having a flow capacity equalling 1. Note again that a directed version of this theorem can be stated.

System of Distinct Representatives

Given a finite ground set $E = \{e_1, e_2, \cdot\cdot\cdot, e_m\}$, and a family of subsets $\mathcal{S} = \{S_1, S_2, \cdots, S_n\}$ where $S_j \in E \; \forall \; j$.

A system of distinct representatives (SDR) of $\mathcal{S}$ is given by $\{e_{j_1} \cdots e_{j_k}\}$ *distinct* elements of E such that $e_{j_t} \in S_{i_t}$ for $t = 1, 2 \cdots, n$.

Hall's Theorem: There exists an SDR for $\mathcal{S}$ if and only if $|\cup_{S_j \in J} S_j| \geq |J| \; \forall J \subseteq \mathcal{S}$.

Consider the network in *Figure* 8. Note that the arcs in the intermediate part of the network are given by $A = \{(i, j) \in E \times \mathcal{S} : e_i \in S_j\}$. The capacity of the arcs in A are all set to $+\infty$ and to $+1$ for all other arcs. Now, if we apply the MFMC theorem to this network we obtain Hall's theorem.

Dilworth Chain Decomposition: Given a partially ordered set N consisting of n elements with relation $\succ$, a chain is a set of $k > 1$ elements $\{j_1, j_2, \cdots j_k\}$ where $j_1 \succ j_2 \succ \cdots \succ j_k$. A decomposition of N is a partition of N into chains. Two distinct elements i and j of N are unrelated if neither $i \succ j$ nor $j \succ i$. The Dilworth chain decomposition theorem states that the minimum number of chains required

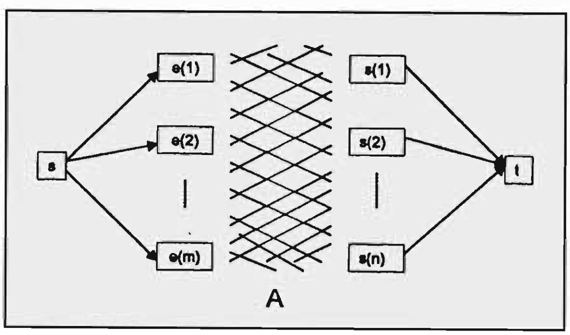


Figure 8. Hall's theorem.

in a decomposition of N is equal to the maximum number of mutually unrelated elements of N

In order to prove this theorem, define $G((S,T);\mathcal{A})$ to be the bipartite graph with $S = \{x_1, x_2, \cdot \quad, x_n\}$; $T = \{y_1, y_2, \cdot \quad, y_n\}$ and $\mathcal{A} = [(x_i, y_j) : i \succ j]$. Using some of the properties of graph G defined above, Dilworths chain decomposition theorem can be proved by applying the MFMC theorem or the Kónig–Egervary theorem.

Extensions

Four extensions of the maximum flow problem are briefly discussed.

1. All Pairs Max Flow Problem: Given an undirected network with arc capacities, it is required to find the maximum flow between each pair of nodes i and j. Obviously, this can be done by solving a max flow problem for each possible source-sink pair of nodes, i.e., solving a max flow problem for $s = 1, 2, \cdot \quad, n$; $t = s + 1, \quad, n$. But Gomory and Hu (see the book by Ahuja and others [3]) have shown that the $\frac{n(n-1)}{2}$ possible maximum flows can take on at most $(n-1)$ numerically different values and this can be determined by solving only $(n-1)$ maximum flow problems.

2. Minimum Cost Flow Problem: The minimum cost flow problem requires that the demand at some of the nodes be satisfied by supply available at some other nodes. The arcs have specified lower and upper capacities as well as specified cost per unit flow that can vary for each arc. The objective is to minimise the total cost of satisfying the demands. By adding an additional source node, an additional sink node and appropriate arcs with capacities and cost, it is straightforward to convert the problem to a minimum cost circulation problem which is essentially the feasible circulation problem mentioned earlier with the objective of minimizing the total cost of flow in the network. The out-of-kilter algorithm of Ford and Fulkerson can be used to solve this problem.

3. Generalised Network Flow Problem: The gener-

Suggested Reading

[1] J Edmonds and R M Karp, Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems, *Journal of the ACM*, 19, No. 2, 248–264, 1972.

[2] R R Tarjan, *Data Structures and Network Algorithms*, CBMS-NSF Lecture Series, SIAM Press, 1983.

[3] R K Ahuja, T L Magnanti and J B Orlin, *Network Flows*, Prentice Hall, 1993.

[4] L R Jr. Ford and D R Fulkerson, *Flows in Networks*, Princeton University Press, 1962.

alised network flow problem is one in which f_{ij} units of flow starting from node i are multiplied by r_{ij} and $r_{ij}f_{ij}$ units arrive at node j. With given arc capacities and flow conservation at all nodes other than the specified source and sink nodes, the objective may be to maximise the flow out of the specified source node. Note that $r_{ij} = 1$ for all (ij) in the usual max flow problem.

4. Multi-Commodity Flow Problem: The multi-commodity flow problem is one in which $r \geq 2$ commodities flow in the same network. In the maximum multi-commodity flow problem, the source sink pairs, s^k, t^k $k = 1, 2, \cdots, r$ are specified for each commodity. An arc (i, j) may have a capacity c^k_{ij} with respect to commodity k i.e. $f^k_{ij} \leq c^k_{ij}$ where f^k_{ij} is the amount of commodity k on arc (i, j). In addition, the sum of the flows of different commodities on arc (i, j) cannot exceed the total capacity c_{ij} where $c_{ij} < \sum_{k=1}^{r} c^k_{ij}$ i.e. $\sum_{k=1}^{r} f^k_{ij} \leq c_{ij}$. For each commodity k, the flow conservation must hold at each node of the network, other than at s^k and t^k. The objective is to maximise the total net flow of the commodities out of the source nodes s^k. That is,

$$\max \sum_{k=1}^{r} \{ \sum_{j:(s^k,j)\in\mathcal{A}} f_{s^k,j} - \sum_{i:(i,s^k)\in\mathcal{A}} f_{i,s^k} \}.$$

Address for Correspondence
Vijay Chandru
Computer Science and Automation, IISc
Bangalore 560 012, India.
Email:chandru@csa.iisc.ernet.in
M R Rao
Director
Indian Institute of Management
Bannerghatta Road
Bangalore 560 076, India.

Resonance, Vol.4, No.7, 1999

Page 26, 2nd paragraph: 'In 1910 when Le.Grand argued that there was no single ruling paradigm in earth sciences, Wegener first thought about the concept of continental drift.'

should read as *'Le Grand has argued that there was no single ruling paradigm in earth sciences in 1910 when Wegener first thought about the concept of continental drift.'*

Resonance, Vol.4, No.9, 1999

Information and Announcment Section: *Vaibhav Vaish, Lucknow has won a silver medal in the Mathematics International Olympiads, 1999.*

The errors are deeply regretted.