Chapter 5

Flows in Networks

So they roughed him in a bag, tied it to a heavy millstone, and decided to throw him in the deepest waters of the Danube. ”Where is the most water?” Some said it must be uphill ‘cause it always flows from there, others said it ought be down in the valley ‘cause it gathers there.

Păcălă, ROMANIAN FOLK TALE

The combinatorial problems and algorithms herein described can be viewed as part of the larger problem of linear programing: that of maximizing a linear function over a set of points in the n-dimensional space that satisfy some (finite but usually large number of) linear constraints. An algorithm commonly used to perform this task is the so-called simplex algorithm. Our interest is in a subclass of such problems that can be solved by combinatorial and graph theoretical means. The algorithms that accompany the problems in this subclass are known to be very efficient in terms of computing time. We do not dwell much on questions of efficiency, however, but focus instead on the underlying combinatorial and graph theoretical techniques.

The well-known result of Birkhoff and von Neumann that identifies the extreme points of the set of doubly stochastic matrices is our starting point. We present two proofs: one based on graph theoretical arguments, the other within the context of linear programming involving unimodular matrices. Matching and marriage problems are the subject of the second section, with results of König and P. Hall. Minty's arc coloring lemmas are in Section 3. The next two sections contain the max-flow min-cut theorems along with some of their corollaries: Menger's theorem on connectivity, a result on lattices by Dilworth, and yet another by P. Hall. The "out of kilter" algorithm is the subject of Section 6. We end the chapter with a discussion of matroids and the greedy algorithm.

1 EXTREMAL POINTS OF CONVEX POLYHEDRA

5.1

A set in the usual Euclidean n-dimensional space is called convex if together with two distinct points of the set the whole line segment joining the two points lies within that set. (With x and y two given points, a point on the line segment joining them can be written as $ax + (1 - \alpha)y$, with $0 \leq \alpha \leq 1$.) By a *convex polyhedron* we understand the set of all points x that satisfy a finite number of linear inequalities, that is, x satisfies a system of inequalities of the form

$$Ax \leq b,$$

where A is a (rectangular) matrix of scalars and b is a vector of scalars. [We remark, in passing, that any system of linear inequalities (and equalities) can be written in the form $Ax \leq b$, by multiplying inequalities of the form $\geq$ by -1, and replacing $=$ by both $\geq$ and $\leq$.]

A convex polyhedron is said to be *bounded* if it can be placed within a (possibly large) sphere.

The unit cube is in many ways a typical example of a bounded convex polyhedron. It is defined by the system of inequalities given below:

$$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -1 \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \leq \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}.$$

A point of a convex polyhedron is called *extreme* if any segment of positive length centered at that point contains points not belonging to the polyhedron. (Geometrically the extreme points are simply the corners of the polyhedron.)

When one wants to maximize a *linear* function over a set of points that form a bounded convex polyhedron, it is easy (and helpful) to observe that the maximum is reached at an extreme point. This follows immediately upon observing two things: that a linear function is convex and that a local maximum is necessarily a global one. Within this context (of linear programming) it becomes of importance to identify the extreme points of bounded convex polyhedra.

In small dimensions it is not hard to "see" the extreme points of a bounded convex polyhedron. Not so in higher dimensions, when the polyhedron might be described by thousands of linear inequalities (possibly with redundant repetitions), each involving hundreds of variables, maybe. The case of doubly stochastic matrices is a classical example; let us examine it next.

5.2

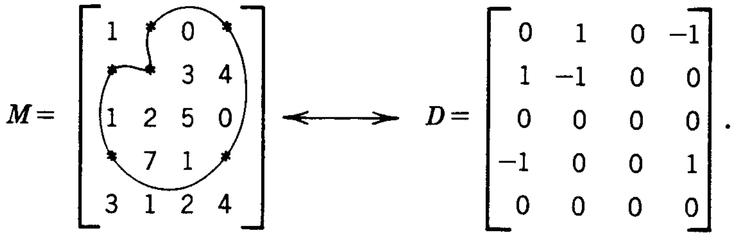
A $n \times n$ matrix (x_{ij}) is called *doubly stochastic* if $x_{ij} \geq 0$ and $\sum_i x_{ij} = \sum_j x_{ij} = 1$. The set of all doubly stochastic matrices, K, is a bounded convex polyhedron in R^{n^2}; R denotes the set of real numbers. [Its dimension is in fact $(n-1)^2$.] The Birkhoff-von Neumann theorem asserts that the extreme points of K are permutation matrices. It follows as an obvious corollary of the following result:

The Birkhoff-von Neumann Theorem. *Let $p_1, \ldots, p_m$; $q_1, \ldots, q_n$ be natural numbers and let C be the convex set of all $m \times n$ matrices (x_{ij}) such that*

$$x_{ij} \geq 0, \quad \sum_i x_{ij} = q_j, \quad \sum_j x_{ij} = p_i.$$

Then the extreme points of C are matrices in C with integer entries.

Proof. We show that any matrix in C that contains noninteger entries is not an extreme point. Let M be such a matrix. Form a (bipartite) graph whose vertices are the rows of M and the columns of M; an edge joins a "row" and a "column" if the corresponding entry in M is not an integer. Evidently each vertex has degree at least two (a row with one noninteger entry must have at least one other noninteger entry). Decompose the graph into its connected components. No component can be a tree since every finite tree has at least two "leaves" (i.e., vertices of degree 1). (To see this, start at any vertex of a finite tree and "walk along it" placing an arrow on each edge traversed and never reversing direction. One cannot return to the starting vertex since a tree has no loops. Eventually one can go no further, i.e., reaches a vertex of degree 1. To get a second leaf start at the first leaf, retrace all steps in reverse direction, and continue to another leaf.) Hence we get a closed path in our original graph. This corresponds to a "closed path" in the matrix M as shown:



Let D be the $m \times n$ matrix as shown, that is, let D have entries 0, 1, -1. The nonzero entries occur only at the vertices of the "closed path" in M and alternative values 1, -1 are assigned as we traverse the path. So D has zero row and column sums. Let $M_1 = M + \varepsilon D$, $M_2 = M - \varepsilon D$ for $\varepsilon > 0$. The row and column sums of M_1 and M_2 satisfy the conditions for matrices in C. For ε small enough (less than the least distance from noninteger entries in M to integers), M_1 and M_2 have nonnegative entries (hence they are in C). Moreover $M = \frac{1}{2}(M_1 + M_2)$ with $M_1, M_2 \in C$, $M_1 \neq M_2$. Therefore M is not an extreme point. This ends our proof.

In the above theorem not all integer-entry matrices are extreme points. Take $m = n = 2$, $p_i = q_j = 2$, and note that

$$\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} + \frac{1}{2}\begin{bmatrix} 0 & 2 \\ 2 & 0 \end{bmatrix}.$$

5.3

Out of the general class of problems dealing with the maximization of a linear function $c^t x$ subject to constraints $Ax \leq b$, the subclass of combinatorial interest is that in which the matrix A (or its transpose) has the following properties: all its entries are -1, 1, or 0, with at most two nonzero entries in a column, and if two nonzero entries occur in a column then they must add up to 0.

We aim to give a proof of the Birkhoff-von Neumann theorem via linear algebra. A general lemma, stated below, plays a central role:

Lemma 5.1 (On Extreme Points of Convex Polyhedra). *Let K be a convex polyhedron in R^n determined by a system $Ax \leq b$ of more than n inequalities. Then any extreme point of K is the unique solution of a subsystem of n equations of the larger* (and possibly inconsistent) *system $Ax = b$.*

Proof. Let e be (the vector in R^n corresponding to) an extreme point of K. Translating by e we may assume, without loss of generality, that $e = 0$. (Note that such a translation, say $y = x - e$, leaves the coefficient matrix A unchanged, i.e., the system $Ax \leq b$ becomes simply $Ay \leq b + Ae$.) With the understanding that $e = 0$, the vector b has nonnegative components (since the extreme point 0 is in K, and thus satisfies $0 = A0 = Ae \leq b$).

Suppose that $b_i = 0$ for the first m equations and that $b_i > 0$ for the remaining equations (of $Ax = b$). We claim that the first m equations produce a system of rank n, with 0 ($= e$) as its unique solution. For if this is not true, then there exists a nonzero solution v to the $m \times n$ system. Since all other $b_i > 0$ we can select $\varepsilon > 0$ such that $tv \in K$ for all $-\varepsilon \leq t \leq \varepsilon$ and this contradicts the assertion that 0 is an extreme point. This proves the lemma.

[Geometrically all that Lemma 5.1 says is that an extreme point (i.e., a corner) of K is the intersection of precisely n "faces" of K (if $K \subseteq R^n$).]

The next lemma gives insight into the algebraic structure of the matrix attached to a system of "combinatorial" constraints:

Lemma 5.2. *Suppose that a $n \times n$ matrix M satisfies the following conditions:*

(i) The entries are -1, 1, or 0.

(ii) Any column has at most two nonzero entries.

(iii) If a column has two nonzero entries, then they add up to zero.

Then the determinant of M is -1, 1, or 0.

Proof. If M has a column with all entries 0, we are done (the determinant is 0). If M has a column with precisely one nonzero entry in it, we expand the determinant by that column and reduce the problem to a $(n-1) \times (n-1)$ matrix with the same properties as M, so we can apply induction. The only other possibility is that all column sums are zero and then the determinant of M is zero (since M^t has $\mathbf{1}$, the vector with all entries 1, in the kernel). This ends our proof.

We conclude Section 5.3 with another proof of the theorem by Birkhoff and von Neumann, stated in Section 5.2. The proof goes as follows.

Note that the convex set C can be described by $-x_{ij} \leq 0$, $\sum_i x_{ij} \leq q_j$, $\sum_i -x_{ij} \leq -q_j$, $\sum_j x_{ij} \leq p_i$, $\sum_j -x_{ij} \leq -p_i$. These inequalities describe a convex polyhedron in R^{mn} (as in Lemma 5.1) and each extreme point is obtained by solving a *nonsingular* system $Mx = b$, which is a *subsystem* of the larger (inconsistent) system:

$$\left.\begin{array}{rcl} \sum\limits_i x_{ij} & = & q_j \\ \sum\limits_j x_{ij} & = & p_i \\ x_{ij} & = & 0. \end{array}\right\} \qquad (5.1)$$

We solve $Mx = b$ by Cramer's rule. All the determinants in question are clearly integers. So it will be sufficient to show that the determinants in the denominators are ± 1, that is, the determinant of M is + 1 or -1.

Denote by A the $(m+n+mn) \times mn$ matrix of the (inconsistent) system (5.1). Note that

$$A = \begin{bmatrix} B \\ \cdots \\ I \end{bmatrix}$$

with I the $mn \times mn$ identity matrix [corresponding to the mn equations $x_{ij} = 0$ in (5.1)], and B a matrix that satisfies the conditions of Lemma 5.2. The $mn \times mn$ coefficient matrix M equals

$$M = \begin{bmatrix} B^* \\ \cdots \\ I^* \end{bmatrix},$$

with B^* consisting of some rows of B and I^* consisting of some rows of the identity matrix I. Expand the determinant of M by rows of I^* until left with a minor of B^*. The determinant of M (which is not 0 since M is nonsingular) equals (up to sign) the determinant of this minor of B^*. The determinant of the minor is therefore nonzero, and from Lemma 5.2 we conclude that it equals in fact + 1 or -1. An extreme point of C has therefore *integral* entries.

REMARK. The coefficient matrix A attached to system (5.1) is in general of the form

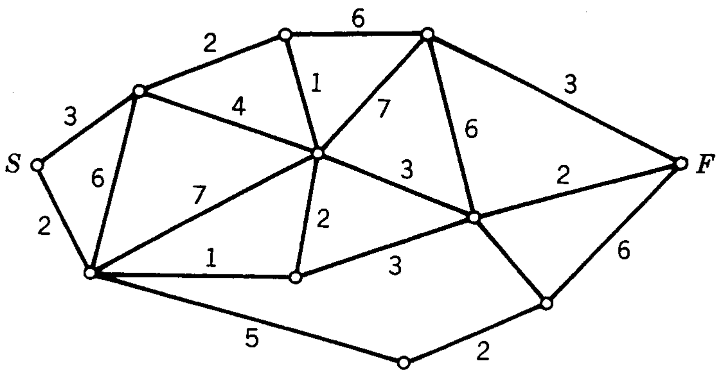
$$A = \left[\begin{array}{cccccc} -1 & -1 & -1 & & & \\ & & & -1 & -1 & -1 \\ \hline 1 & & & 1 & & \\ & 1 & & & 1 & \\ & & 1 & & & 1 \\ \hline 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{array}\right] \begin{array}{l} \\ m \leftrightarrow \sum_j -x_{ij} = -p_i \\ \\ n \leftrightarrow \sum_i x_{ij} = q_j \\ \\ \\ \\ \\ mn \leftrightarrow x_{ij} = 0. \\ \\ \\ \end{array}$$

In this small example the variables are ($m = 2$, $n = 3$):

$$\begin{array}{ccc} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \end{array}.$$

5.4 The Shortest Route Algorithm

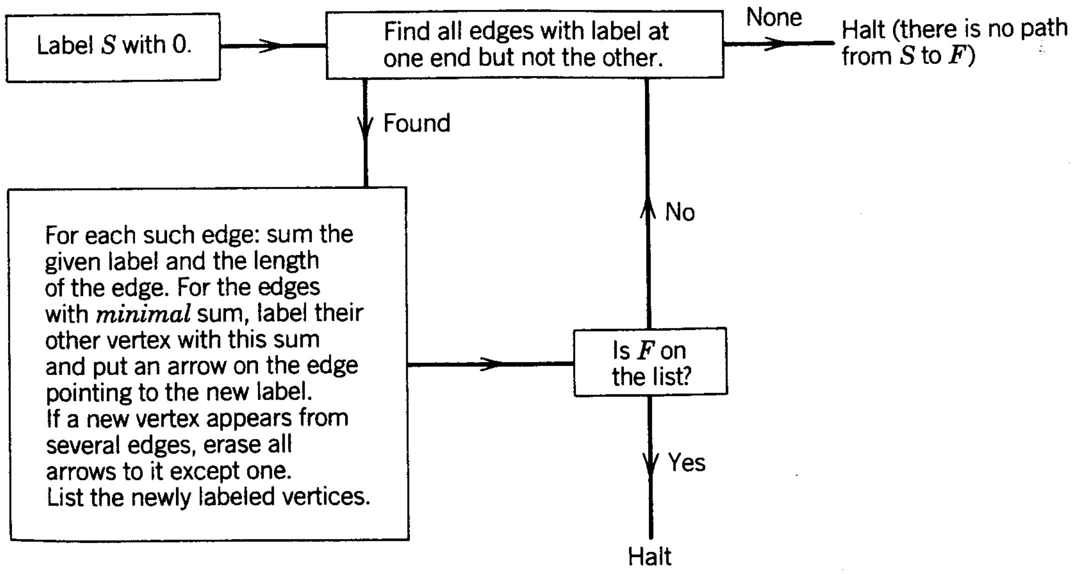
Consider a graph as below:



We wish to find the shortest route from S (start) *to* F (finish), *where the numbers on the edges represent distances.* In particular, if all the distances are 1 we get the path with the minimal number of edges. (It is hopelessly inefficient to measure the length of every path from S to F; if we have n vertices the worst possible case requires much more than $(n-2)!$ comparisons.)

The dual problem to this is that of maximizing the distance between S and F subject to the linear constraints that the distances between points do not exceed specified constants along the edges. These constraints can be written as $-d_{ij} \leq x_i - x_j \leq d_{ij}$, where x_i is the coordinate attached to vertex i (upon projecting the graph on a straight line) and d_{ij} is the specified distance between vertices i and j. When the constraints are written in the form $Ax \leq b$, where b is the vector of d_{ij}'s, the matrix A has -1, 1, or 0 as entries, at most two nonzero entries per column, and if two nonzero entries occur in a column they sum up to zero.

The algorithm that solves the shortest route problem is as follows:


Label *S* with 0.
Find all edges with label at one end but not the other.
None
Halt (there is no path from *S* to *F*)
Found
For each such edge: sum the given label and the length of the edge. For the edges with *minimal* sum, label their other vertex with this sum and put an arrow on the edge pointing to the new label. If a new vertex appears from several edges, erase all arrows to it except one. List the newly labeled vertices.
No
Is *F* on the list?
Yes
Halt

The edges with arrows form a tree. To identify the shortest route, start at F and follow unambiguously back to S against the direction of the arrows.

Looking at this algorithm one can see that we run through the loop less than $n\binom{n}{2}$ times, where n is the number of vertices [and $\binom{n}{2}$ is an upper bound on the number of edges to be checked at each step]. This algorithm is therefore of the order of n^3 (much more efficient than $(n-2)!$, which exceeds 2^n).

[*Aside: An impractical practical solution.* Make a string model with each edge the appropriate length, hold S with left hand and F with right hand, pull apart until string is tight-the shortest route is now visible! (”Oh what a tangled web we weave... .”) One should appreciate the dynamic interplay between the initial (shortest route) problem and the dual (longest distance) problem which this string model vividly brings forth. It is easy to visualize the shortest route algorithm in terms of the string model: lay the string model flat on the table and slowly lift vertex S straight up. The shortest route becomes visible!]

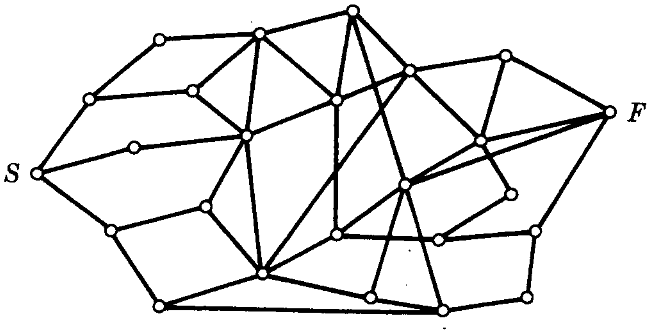
EXERCISES

1. Write the doubly stochastic matrix $n^{-1}J$ (where J is the $n \times n$ matrix with all entries 1) as a convex combination of a *minimal* number of permutation matrices. Do the same for

$$\frac{1}{12}\begin{pmatrix} 5 & 1 & 6 \\ 0 & 8 & 4 \\ 7 & 3 & 2 \end{pmatrix}.$$

2. Prove that every $n \times n$ doubly stochastic matrix is a convex combination of at most $n^2 - 2n + 2$ permutation matrices. Is this best possible for every n?

3. Work through the shortest path algorithm for the graph



(Distances on all edges are 1.)

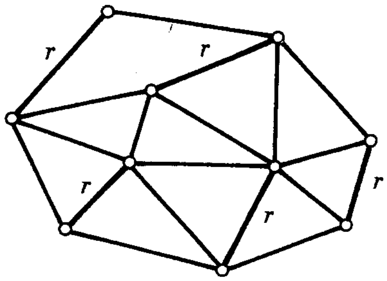
2 MATCHING AND MARRIAGE PROBLEMS

The reader should be informed at the very outset that the contents of this section are more pleasant than the title may suggest. Consequences of the theory herein presented are numerous and its applications reach far beyond the narrow scope suggested by the title. On this latter point still we vigorously inform that these results, although magnanimously designed for the enhancement of happiness at large, remain of limited value on an individual basis.

Having said this, and consequently being left with the mathematically inclined only, we describe two qualitatively different types of applications. The first rests on the algorithm for extracting a *maximal* number of matchings between objects of one type (girls, say) and objects of another (boys) given a matrix of "likings" between the two groups. Matching people with jobs, given a matrix of their qualifications regarding these jobs, is another typical example. Applications of another kind involve the so-called marriage lemma. The proof of the existence of a Haar measure on a compact group is one such consequence, and the proof of the existence of finite approximations to measure preserving transformations is another. Those of the former kind are applied assignment problems while the latter are theoretical.

5.5

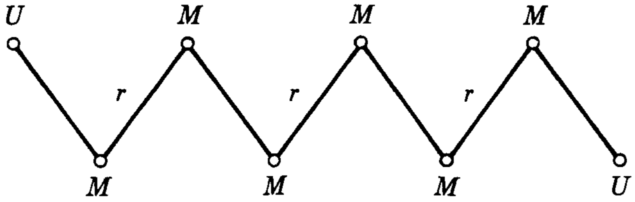
To start with, consider the problem of assigning roommates in a dormitory (a sorority). Form a graph in which the vertices are the girls and edges are drawn when there is a liking between the girls. A matching is a coloring in red (indicated by r on the drawings below) of some of the edges so that for any vertex there is at most one colored edge adjacent (i.e., no bigamy is allowed).



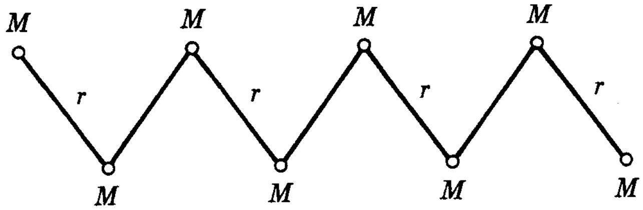
In the diagram above, the red matching is perfect (the assignment problem is completely solved). A perfect matching may not always exist, but we may seek *maximal* matchings, that is, matchings with a largest possible number of edges. Given any matching we then label the vertices U or M to denote unmatched or matched.

Berge observed the following:

∗ *If a matching is not maximal, then the labeled graph contains a path of odd length of the form*

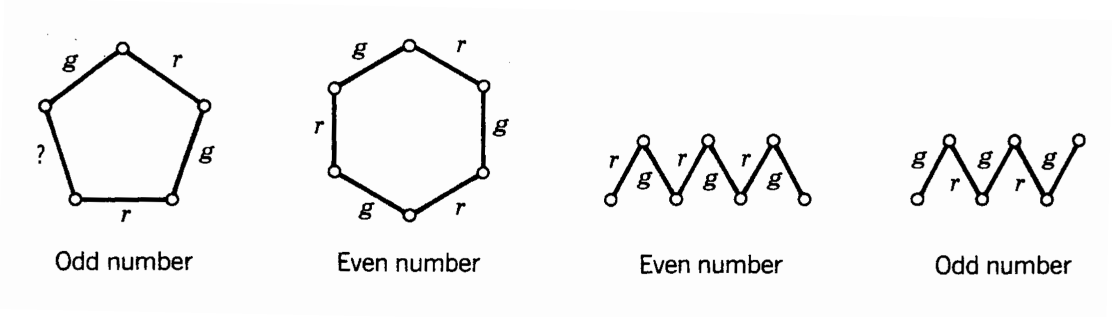


(i.e., a zigzag). (Note that upon locating such a zigzag we can increase by one the number of matchings by relabeling



.)

Proof. Suppose that the given red matching is not maximal. Then there is a green matching (say) with a greater number of edges. From the given graph delete all uncolored edges and all doubly colored edges. Decompose the remainder into its connected components and focus attention on the components that contain at least one edge. No vertex can have degree three or more - else that vertex would be a "bigamist" for red or green. Therefore every vertex has degree one or two. There exists, and we choose, a component with more green than red vertices. It must be one of four possible types:



Odd number Even number Even number Odd number

The first case leads to bigamy while the next two cases have an equal number of red and green edges. Hence we must have an odd zigzag with one more green edge than red. It is now enough to show that the ends of the odd zigzag are unmatched in the red matching. But if an end was matched by a red edge, that edge was deleted (or it would still be there!) and so that edge was both red and green-but then we have green bigamy. This ends our proof.

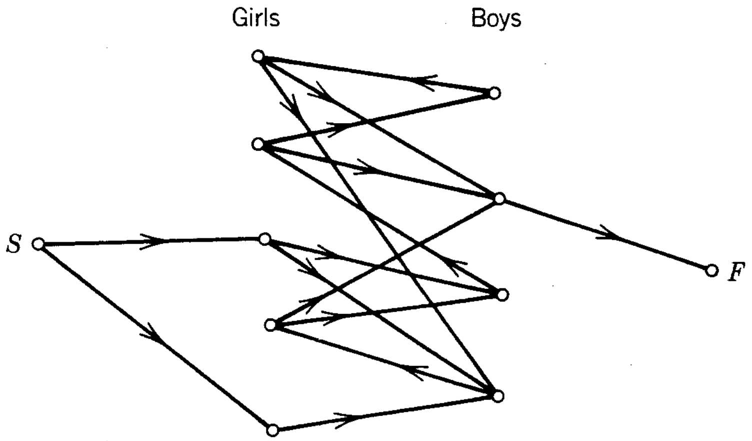
5.6 The Algorithm for Extracting a Maximal Matching

Consider now the marriage problem (i.e., finding a maximal matching) in a *bipartite* graph. In this situation a matching is called a *marriage.* The following method is due to Munkres

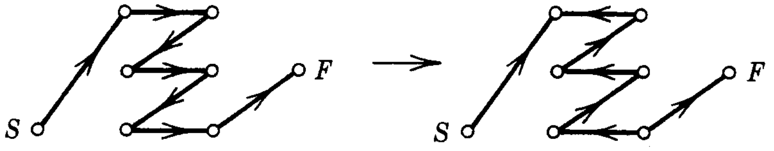
[4].

Line up the girls in the left column and the boys in the right column (as below) and place an edge between a girl and a boy if there is a liking between them (else place no edge). Marriages are allowed only between liking couples, of course. Marry off as many boys and girls as possible by some means (marry off one couple, say).

Adjoin extra vertices S (start) and F (finish).



Connect S to the unmarried girls (and direct these edges left to right). Connect the unmarried boys to F (and direct left to right). Direct liking but not married couples left to right. Direct existing marriages *right to left.* (Every edge is now directed.) Use the shortest route algorithm described in Section 5.4 (and assign distance 1 on all edges) to find any (in particular the shortest) path from S to F. If there is one, it is a zigzag,



and we can increase the number of marriages by one, as shown. Continue in this fashion until a new path becomes unavailable. The process has now ceased and we ask: Is the resulting matching maximal? Yes. This can be seen by reconsidering Berge's observation (see Section 5.5) in the present context. (Indeed, the existence of an odd zigzag with appropriate directions on edges and S and F adjoined provides a directed path from S to F, contradicting the fact that the process cannot be continued.)

5.7

We now examine more carefully the maximal situation to obtain necessary and sufficient conditions to marry all the girls. At this point it is convenient to introduce matrix notation.

Given m girls and n boys form the (0,1) matrix M in which a 0 in the (i, j)th entry means girl i likes boy j and 1 denotes dislike. [Note, conversely, that any $m \times n$ (0,1) matrix M gives a like-dislike graph for m girls and n boys.] We denote marriages by starring some of the 0's. To avoid bigamy we must not have two starred zeros in any row or in any column.

A *set of stars* on the 0 entries is called *admissible* if no two starred zeros occur in the same row or the same column of M. A *set of lines* (that is, rows and/or columns of M) is called *admissible* if their deletion removes all the 0's in M.

Since in particular we have to cross out all starred 0's by an admissible set of lines we conclude that

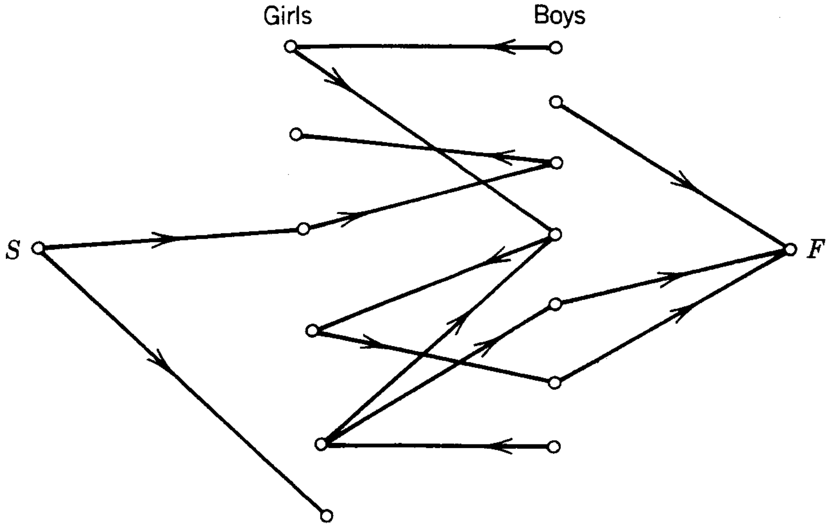
$$|admissible\ \ stars| \leq |admissible\ \ lines|,$$

$$for\ \ arbitrary\ \ admissible\ \ sets.$$

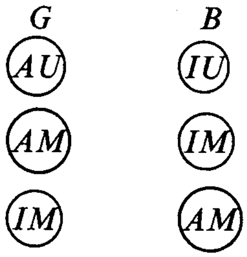
The main result is due to König:

König's Theorem. *There exists an admissible set of stars and an admissible set of lines of the same cardinality; this admissible set of stars is in fact of maximal cardinality and this admissible set of lines is in fact of minimal cardinality.*

This result surfaces upon a careful examination of what, goes on when there is *no path* from S to F. Assume therefore that a number of marriages were made (by the maximal matching algorithm described in Section 5.6 or otherwise) and that we are now at the final stage with *no path* left from S to F.



Partition the girls into four classes by married (M) or unmarried (U); accessible by a directed path from S (A), or inaccessible by such a path (I). Do *exactly* the same for boys. By the construction of the graph all unmarried girls are accessible and since there is no path from S to F no unmarried boy is accessible. The partition is thus as follows:



We now want to examine the possible interconnections between these classes by likes (left to right arrows) and marriages (right to left arrows). It is convenient to interpret this in terms of the zeros and the starred zeros in the matrix M. There are nine cases that (upon suitable permutations of rows and columns) are summarized below:

			B	
		IU	*IM*	*AM*
G	*AU*	☒	☒	
	AM	☒	☒	
	IM			

Possible locations for the *unstarred* zeros

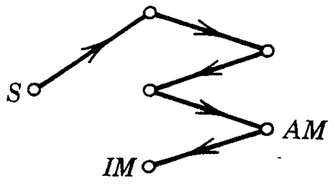
			B	
		IU	*IM*	*AM*
G	*AU*	☒	☒	☒
	AM	☒	☒	
	IM	☒		☒

Possible locations for the *starred* zeros

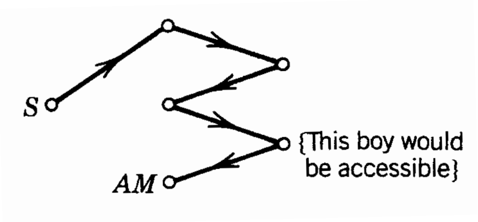
(5.2)

The squares that are crossed out contain no zeros at all (starred or unstarred).

Let us explain in detail the situations summarized in (5.2): For the *unstarred* case there can be no direct link (i.e., no edge) from an accessible girl to an unaccessible boy (else the boy would be accessible). The *starred* zeros represent marriages and so unmarried boys are excluded (i.e., column 1) and, likewise, unmarried girls are excluded (i.e., row 1). There can be no marriage between an IM girl and an AM boy (else the girl would be accessible)- see below:



Also, there can be no marriage between an AM girl and an IM boy - for the last link to the AM girl must be the marriage and thus the boy to whom she is married is accessible (see below):



We therefore conclude that the marriages occur only between two inaccessibles or two accessibles [as the summary (5.2) of the possible positions for the starred zeros indicates].

In the matrix M we therefore have exactly one starred 0 in each column under AM (and it occurs in the AM row region), and we have exactly one starred 0 in each row across from IM (and it occurs in the IM column region). Delete these rows and columns (i.e., these lines) and we delete all the zeros in M (starred and unstarred). The number of lines equals precisely the number of starred zeros. (And this set of lines is admissible, as we just saw.) This proves König's theorem.

We actually proved that when a maximal matching is reached we can extract from it an admissible set of stars and an admissible set of lines of the same cardinality.

5.8

By the list of "eligibles" for a girl we mean the subset of boys that she likes. Each girl g has her list of "eligibles" $L(g)$.

The Marriage Lemma (P. Hall). *All the girls can be married if and only if* $|\cup_{g\in T}L(g)| \geq |T|$*, for all subsets* T *of the set of girls.*

Proof. The proof follows easily from König's theorem. Necessity is obvious (for if the joint lists of t girls contain less than t boys, then there is no way to marry all of these t

girls). Sufficiency is proved as follows: Suppose that we cannot marry all the girls (which are m in total, say). Set up the matrix M of girls versus boys with a 0 in (i, j)th entry if boy j is on the list of eligibles of girl i. By our supposition the number of starred 0's in M is strictly less than m. By König's theorem there exists an admissible set of less than m lines. Without loss assume that the admissible lines are given by the first r rows and the first c columns of M. So $r + c < m$. Let T be the set of girls in the remaining $m - r$ rows; $|T| = m - r$. The total number of eligible boys for these girls is $\leq c < m - r = |T|$. This ends the proof.

There exist fairly short inductive proofs of the marriage lemma. We selected this approach because of its close interplay with the algorithm for actually extracting a maximal matching.

5.9

By the list of eligibles for a boy we understand the set of girls that he likes. A simple sufficient condition to marry all the girls is described below:

$*$ *If each girl has at least k eligible boys, and if each boy has at most k eligible girls, then all the girls can be married.*

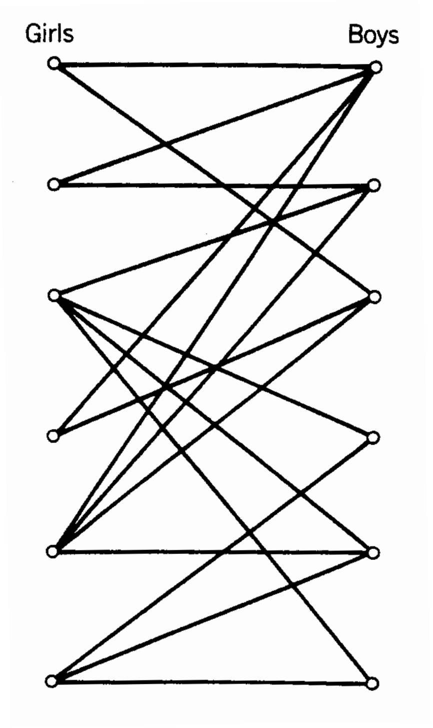
Indeed, let T be a set of girls. Then

$$\begin{aligned} k|T| &\leq |\text{the set of edges emanating from } T| \\ &= |\text{the set of edges terminating in } \bigcup_{g \in T} L(g)| \\ &\leq |\bigcup_{g \in T} L(g)|k. \end{aligned}$$

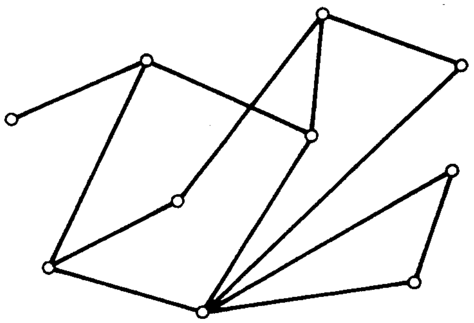
Hence $|\cup_{g \in T} L(g)| \geq |T|$ and the marriage lemma now guarantees that all the girls can be married.

EXERCISES

1. With the likings indicated below, can all the girls be married?



2. With the likings drawn below, can we match up the ten girls in pairs of roommates?



3. Workers A, B, C, D, E are qualified for jobs 1, 2, 3, 4, 5, 6, 7 as shown in the diagram below:

	1	2	3	4	5	6	7		1	2	3	4	5	6	7
A	—	√	√	√	√	√	—	A	√	—	—	√	—	—	√
B	√	—	—	—	—	√	—	B	—	—	√	√	—	√	—
C	√	—	—	—	—	√	—	C	√	—	√	√	—	—	—
D	√	—	√	—	—	—	—	D	√	—	—	√	—	—	—
E	√	—	√	—	—	√	—	E	—	√	—	—	√	—	√

In only one of the two situations all workers can be employed. Which one?

4. An employment agency has nine job openings and many candidates for employment, each qualified for three of the jobs. The agency selects a group of candidates that includes three people qualified for each job. Can all nine jobs be filled with this group of candidates?

5. Twelve students are asked (in some fixed order) to choose their three favorite classes from a long list. As they express preferences, each class is removed from the list as soon as three students have chosen it. Show that each student can be matched with one of his or her favorite classes.

6. Prove that any square matrix with 0 or 1 as entries, and row and column sums equal

to k can be written as a sum of k permutation matrices. Try this for

$$\begin{matrix}
0 & 1 & 1 & 1 & 0 & 1 & 0 \\
0 & 0 & 1 & 1 & 1 & 0 & 1 \\
1 & 0 & 0 & 1 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 & 1 & 1 \\
1 & 0 & 1 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 1 & 0 & 0 & 1 \\
1 & 1 & 1 & 0 & 1 & 0 & 0
\end{matrix}$$

3 THE ARC COLORING LEMMAS

5.10

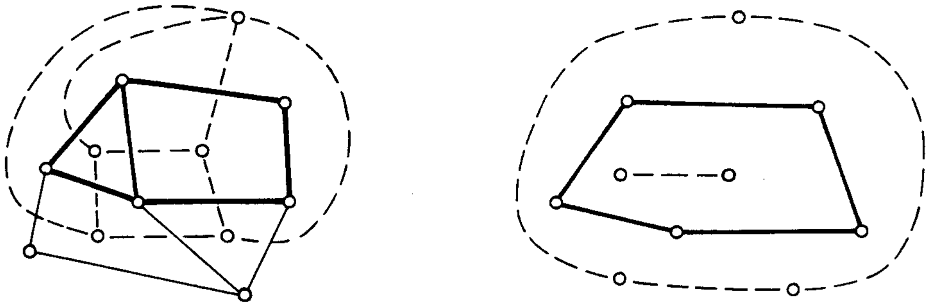
Consider a finite graph G in which each edge is colored green or red. We distinguish two vertices and ask whether there exists a path with green edges between them and, if not, we want to know what *the alternative required feature* of the graph is. It is more convenient to adjoin an extra yellow edge joining the two distinguished vertices. We then ask whether there exists a green closed path in G that includes the yellow edge. It is easy to see that there is a closed path if and only if there is a simple closed path (i.e., a path in which no vertex is passed more than once). We call a simple closed path a *cycle*.

The appropriate alternative object arises naturally in the setting of *planar graphs* (i.e., graphs that can be drawn in the plane, or on a sphere, with no edges intersecting). For such a graph G we may form a *dual graph* G^* – each ”country” in G becomes a vertex in

G^* and we have an edge between two "countries" if they have a common boundary. (Note that the unbounded complement of G is regarded as a country – one can, equivalently, consider graphs on the surface of a sphere where all countries are bounded.)

Example. G = graph with edges of the form ○———○

G^* = graph with edges of the form ○- - - -○

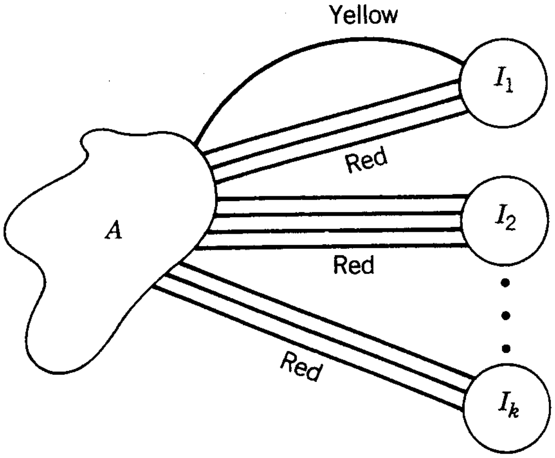


The red cycle in G may be regarded as a cutting out of some edges in G^* that *separates* G^* into two connected components (as shown in the figure at right). Any cycle in G has such effect upon G^*. The cut set of edges in G^* is called a cocycle. From this example one can see that for planar graphs there is a bijective correspondence between the cycles of G and the cocycles of the dual graph G^*.

We may define the concept of a cocycle in *any* graph G (planar or otherwise) as follows: A *separating set* for G is a collection of edges whose deletion increases the number of connected components of G; it is minimal (by inclusion) if no smaller collection of edges serves to increase the number of connected components. A *cocycle* is a minimal separating set.

Minty's Arc Coloring Lemma. *Let G be a colored graph with one yellow edge and the other edges red or green. Then one, and only one, of the following always happens: Either* *there exists a cycle with all edges green except one yellow edge, or there exists a cocycle with all edges red except one yellow edge.*

Proof. Let S be one of the distinguished vertices (i.e., one endpoint of the yellow edge). Mark all the vertices of G that are accessible from S by green paths. This partitions the vertices into accessible A and inaccessible I. Evidently there is no green edge between A and I. If there is no cycle of the required form, then the yellow edge is between A and I. Delete (temporarily) the yellow edge and all red edges between A and I, and decompose the remaining graph into its connected components A and $I_1, I_2, \ldots, I_k$ (say). All vertices in A remain connected to S and so (upon putting back all the edges we deleted) the picture looks like this:



Without loss of generality the yellow edge joins A to I_1. The edges joining A to I_1 provide the required cocycle. This ends our proof.

A useful extension, also due to Minty, is the following:

Lemma 5.3. *Color the edges of a graph green, red, and yellow* (one color per edge). *Place an arrow on each yellow edge and identify one of the yellow edges as the "distinguished "* *yellow edge. Then precisely one of the two possibilities occurs*:

(a) *There exists a cycle containing the "distinguished" yellow edge with all its edges green or yellow oriented in the direction of the "distinguished " yellow edge.*

(b) *There exists a cocycle containing the "distinguished" yellow edge with all its edges red or yellow oriented in the direction of the "distinguished " yellow edge.*

Proof. Label by S and F the endpoints of the "distinguished" yellow edge so that the arrow points from F to S. Partition the vertices into those accessible from S by a path of green edges and yellow edges with arrows pointing away from S, and those inaccessible by such a path. Denote these two disjoint subsets of vertices by A (accessibles) and I (inaccessibles).

If the situation described in (a) does not occur the edges running between A and I are necessarily red and yellow (pointing from I to A). Since F is in I, the edges between A and the connected component of F in I (call it I_1) form the required cocycle.

4 FLOWS AND CUTS

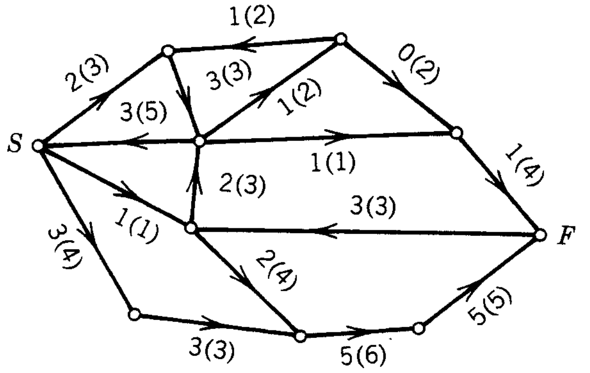
5.11

By a *directed graph* we mean a connected graph with no loops and with one arrow placed on each edge. A *network* is a directed graph in which there are two distinguished vertices, a source S and a sink F, with each edge being assigned a nonnegative real number called its *capacity.* The vertices are labeled $1, 2, \ldots, m$ and the capacity of edge $\{i, j\}$ is denoted by $c(i, j)$. [If there is more than one edge between i and j we may use $c_k(i, j)$ to denote the capacity of the kth edge between them. This is seldom necessary, however.]

One can think of the capacity of an edge as the maximal amount of liquid (or goods of some kind) possible to transport through that edge in the direction of the arrow. Examples of networks may be highway systems, railroad networks, electrical devices, and so on.

A *network flow* is a function assigning a nonnegative number $f(i,j)$ to each edge $\{i,j\}$ (the flow through that edge) so that $f(i,j) \leq c(i,j)$ and $\sum_i f(i,j) = \sum_k f(j,k)$, for each vertex j different from S and F. [This latter condition informs us that all the liquid that enters a vertex (not S or F) must necessarily exit it. In electrical network theory this is known as Kirchhoff's law.] The *value* of a network flow is the sum of the flows on edges emanating from S minus the sum of the flows on edges pointing into S (or, equivalently, the sum of the flows on edges pointing into F minus the flow on edges emanating from F). A *maximal flow* is a network flow of maximal value.

Below we illustrate a network flow with value 3. The capacities are written in parentheses:



(5.3)

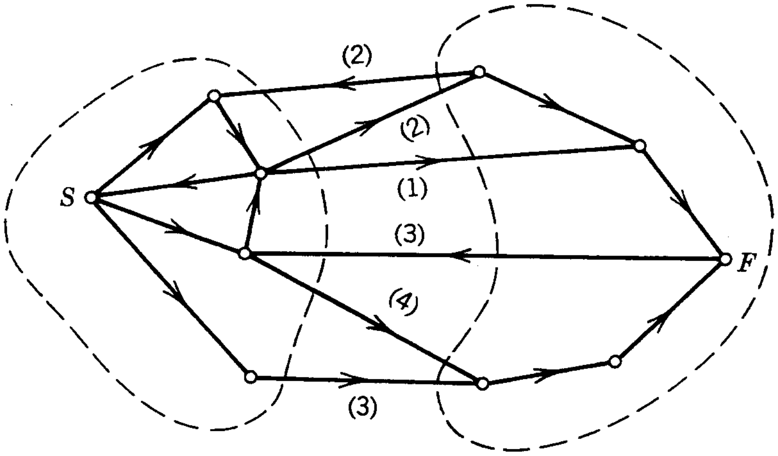
Is this flow maximal? If not, try to find a maximal flow.

The main problem in the theory of network flows is to *determine a maximal flow in a given network.* (The solution is surprising because an obvious necessary condition on capacities turns out to be also sufficient, as we shall see shortly.)

5.12

A *cut* in a network is a subset of edges whose removal disconnects S from F. (By disconnecting S from F we mean that there is no undirected path left between S and F.) When we remove the edges of a cut from a network S and F lie in different connected components of the resulting graph. Let $\overline{S}$ be the connected component of S and $\overline{F}$ that of F ($\overline{S}$ and $\overline{F}$ depend on the choice of the cut, of course). The *capacity of a cut* equals the sum of capacities of those edges of the cut that point from $\overline{S}$ to $\overline{F}$.

In the figure below we display a cut in the network (5.3) (consisting of six edges in total) with capacity $2 + 1 + 4 + 3 = 10$:



(5.4)

5.13

If P is an undirected path between S and some vertex i of the network, we call an edge of P *forward* if its direction along P is from S to i, and *backward* if its direction along P

is from i to S. Given a network flow f we call a path P between S and i *unsaturated* if $f(k,l) < c(k,l)$ for each forward edge $\{k,l\}$ of P, and $f(k,l) > 0$ for each backward edge $\{k,l\}$ of P (the strict inequalities are important in this definition). An unsaturated path between S and F is called *flow augmenting.*

If we can find a flow augmenting path P we can strictly increase the value of the network flow f (thus the terminology). Indeed, look at the list of numbers $c(i,j) - f(i,j)$ for the forward edges $\{i,j\}$ of the path P, and at the list of numbers $f(i,j)$ for the backward edges. All these numbers are strictly positive and finitely many in number. Pick the smallest number from the union of both lists; call it d. Define a new flow g as follows:

$$g(i,j) = \begin{cases} f(i,j) + d, & \text{if } \{i,j\} \text{ is a forward edge of } P \\ f(i,j) - d, & \text{if } \{i,j\} \text{ is a backward edge of } P \\ f(i,j), & \text{if } \{i,j\} \text{ is not an edge of } P. \end{cases}$$

It is easy to check that g is a network flow. Moreover, since the path P starts at S, the value of the flow g exceeds the value of the initial flow f by d (a strictly positive quantity). From this we conclude that if a network flow f admits a flow augmenting path, then f is not a maximal flow. We prefer to summarize this as follows: If f is a maximal flow in a network, then there are no flow augmenting paths in that network.

$$\text{\textit{If f is a maximal flow in a network, then there are no flow augmenting paths in that network.}} \tag{5.5}$$

5.14

Let f be any network flow and C be any cut. The cut C (by its definition) separates S from F. Therefore, the only way that the liquid that emanates from S will reach F is if it passes through the edges of the cut C that point from $\overline{S}$ to $\overline{F}$ (the connected components of S and F that the cut C induces). It thus follows that

The value of any network flow cannot exceed the capacity of any cut. (5.6)

We wish to investigate maximal flows, and the main result in this regard is:

The Max-Flow Min-Cut Theorem (Ford and Fulkerson). *The value of a maximal flow equals the capacity of a minimal cut.*

By a *minimal cut* we understand a cut of minimal capacity.

Proof. From (5.6) we conclude that a maximal flow will have value at most equal to the capacity of a minimal cut. Conversely, assume that f is a maximal flow. Then (5.5) informs us that there is no flow augmenting path in the network. Partition the vertices into two classes: those that can be reached from S by an *unsaturated path*, and those that cannot. Vertices S and F are in different connected components, by assumption; denote their respective components by $\tilde{S}$ and $\tilde{F}$. The edges of the network running between $\tilde{S}$ and $\tilde{F}$ obviously form a cut; call it C. This is the cut we want: Its capacity equals that of f. Indeed, an edge $\{i, j\}$ pointing from $\tilde{S}$ to $\tilde{F}$ has $f(i, j) = c(i, j)$ (or else j would be in $\tilde{S}$). An edge $\{i, j\}$ pointing from $\tilde{F}$ to $\tilde{S}$ has $f(i, j) = 0$ (or else, again, j would be in $\tilde{S}$). The value of f is, therefore, at least equal to $\sum c(i, j)$ (the capacity of C), where the sum runs over all edges of C pointing from $\tilde{S}$ to $\tilde{F}$. We thus conclude that f has value *equal* to the capacity of C. (Note that the arc coloring lemmas of Section 3 are implicit in this argument.) This ends the proof.

5.15

There is a so-called vertex version of the max-flow min-cut theorem. To understand it we have to work with a species of networks in which all edges are assigned *infinite* capacity and, in addition, each vertex has a finite capacity. A *flow* in this network is a function f assigning a nonnegative number to each edge (the flow through that edge) such that the sum of the flows through the edges that enter a vertex does not exceed the capacity of that vertex, and the Kirchhoff law (i.e., flow into a vertex = flow out) is satisfied at all vertices other than S and F. The *value* of a flow, and the notion of a *maximal flow*, are defined as before.

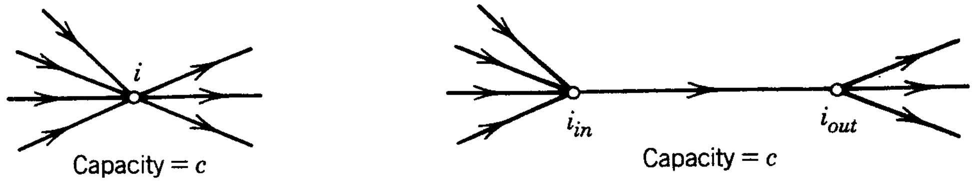
A *vertex cut* is a subset of vertices whose removal disconnects S from F. (By removing a vertex we understand the removal of that vertex *and* all edges coming into or going out of that vertex.) The *capacity of a vertex cut* equals the sum of the capacities of the vertices it contains. By a *minimal vertex cut* we understand a vertex cut of minimal capacity.

The vertex version of the max-flow min-cut theorem states the following:

$*$ *In a network with vertex capacities the value of a maximal flow equals the capacity of a minimal vertex cut.*

Proof. The trick in proving this result is to convert the (species of) network with vertex capacities into a network of the more ordinary kind (i.e., a network with edge capacities only).

Do this by "elongating" vertex i into an edge (whose endpoints we label by i_{in} and i_{out}). All edges of the original network that entered vertex i now enter i_{in} and all edges that left vertex i now leave i_{out}. Place an arrow pointing from i_{in} to i_{out} on the (new) edge $\{i_{in}, i_{out}\}$ and define the capacity of this new edge to be the original capacity of vertex i. The process is graphically illustrated below:



Upon performing this service to each vertex we obtain an ordinary network. In this network the (value of the) maximal flow equals the (capacity of the) minimal cut. Obviously no edges of infinite capacity are in a minimal cut, so the minimal cut corresponds to a vertex cut in the original network. This ends our proof.

5.16

The general problem of finding a maximal flow in a network can be viewed as a problem in linear programming. To see this, attach a variable to each edge: Specifically, write x_{ij} for an edge $\{i, j\}$ with arrow oriented from i to j. Introduce an additional edge between S and F (color it yellow to distinguish it from the rest) and place an arrow on it that points from F to S. Give it infinite capacity and attach to it the variable x_{fs}.

Suppose there are n edges (including the yellow edge) in the network so amended. A flow in the original network corresponds now to a vector with n components with variables x_{kl}'s as entries such that the Kirchhoff law is satisfied at each vertex (*including* S and F).

The task of finding a maximal flow becomes that of maximizing x_{fs}, subject to the linear constraints:

$$\begin{aligned} x_{ij} &\leq c(i,j), \\ -x_{ij} &\leq 0, \end{aligned}$$

and

$$\sum_i x_{ik} = \sum_i x_{ki} \qquad \text{(the Kirchhoff law)}$$

for *all* vertices k of the network.

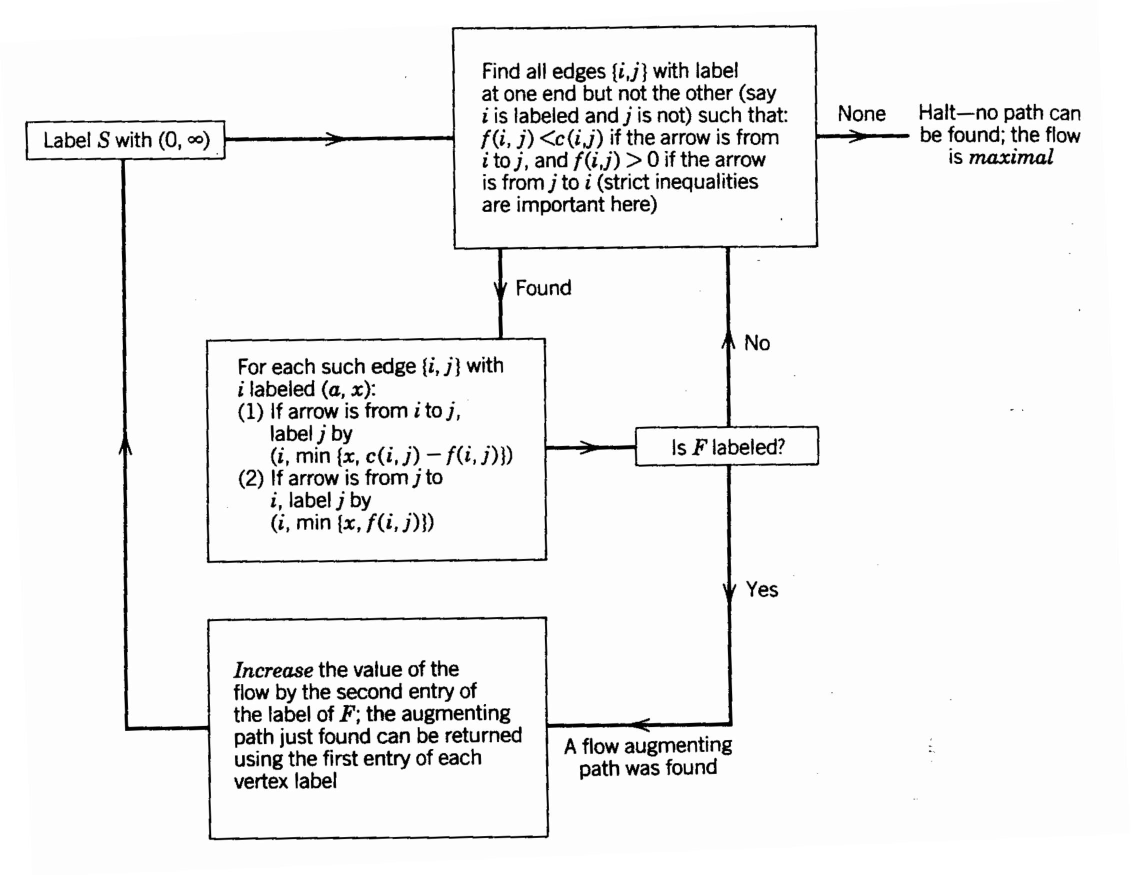
5.17 The Algorithm for Extracting a Maximal Flow

The algorithm described here is based on the observation that

$$\textit{If there is no flow augmenting path, then the flow is maximal.} \tag{5.7}$$

This is the converse of statement (5.5) and is proved as follows: Suppose that no flow augmenting path exists. Partition the vertices into those that can be reached from S by an unsaturated path and those that cannot. Vertices S and F are in different classes ($\tilde{S}$ and $\tilde{F}$, say) of this partition. The flow through any edge pointing from $\tilde{S}$ to $\tilde{F}$ equals the capacity of that edge (by the construction of the partitions). Similarly, flows through edges pointing from $\tilde{S}$ to $\tilde{F}$ are zero. The edges between $\tilde{S}$ and $\tilde{F}$ form a cut whose capacity equals that of the flow. By (5.6) the flow is maximal.

The algorithm we give is for networks with integral capacities (of edges) and in which integral flows only are allowed through the edges. Observation (5.7) will allow us to conclude that when no augmenting path can be found we have reached the maximal flow.


Label S with $(0, \infty)$
Find all edges $\{i,j\}$ with label at one end but not the other (say i is labeled and j is not) such that: $f(i,j) < c(i,j)$ if the arrow is from i to j, and $f(i,j) > 0$ if the arrow is from j to i (strict inequalities are important here)
None
Halt—no path can be found; the flow is *maximal*
Found
No
For each such edge $\{i,j\}$ with i labeled (a,x):
(1) If arrow is from i to j, label j by $(i, \min\{x, c(i,j) - f(i,j)\})$
(2) If arrow is from j to i, label j by $(i, \min\{x, f(i,j)\})$
Is F labeled?
Yes
Increase the value of the flow by the second entry of the label of F; the augmenting path just found can be returned using the first entry of each vertex label
A flow augmenting path was found

5 RELATED RESULTS

The two max-flow min-cut theorems proved in Section 4 can be used to derive several well-known results on partially ordered sets and on connectivity of graphs. Best known, perhaps, are Menger's theorems on connectivity of graphs obtained in 1927, several decades before the max-flow min-cut results.

5.18

Two paths in a graph are called *edge-disjoint* if they have no common edges (but may have common vertices). By two *vertex-disjoint* paths between two nonadjacent vertices we understand two paths with no vertices in common other than the endpoints. We have the following:

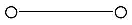
Menger's Results.

(i) *The maximal number of edge-disjoint paths between two nonadjacent vertices equals the minimal number of edges whose removal disconnects the two vertices.*

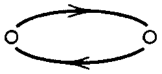
(ii) *The maximal number of vertex-disjoint paths between two nonadjacent vertices equals the minimal number of vertices whose removal disconnects the two initial vertices.*

(By the removal of a vertex we understand, as before, the removal of that vertex *and* all edges adjacent to it.)

Proof of part (i). Produce a network N out of the given graph G by replacing each edge

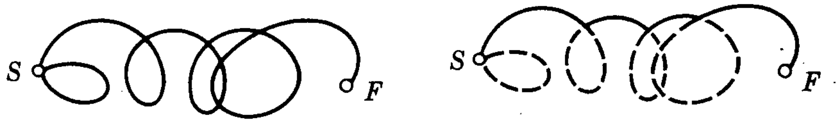


by two edges oriented opposite to each other



Denote the two nonadjacent vertices by S and F; give each edge of the network capacity 1. Through each edge of N we only allow flows of 0 or 1.

Observe that the value of a maximal flow in N equals the maximal number of edge-disjoint paths in G. [To see this, start at S and walk along the edges with flow 1, never against the direction of an arrow and never retracing. We may find ourselves visiting a vertex several times (and this includes S itself) but if the flow out of S is positive we ultimately reach F. Extract a path out of this walk as indicated below:



On all edges of this path change the flow from 1 to 0. What results is *another flow* with value precisely 1 less. Repeat the process. The resulting paths will be edge-disjoint because at each stage we only walk along edges with nonzero flow. By this process we extract as many edge-disjoint paths in G as the value of the maximal flow in N.] By the max-flow min-cut theorem there is a minimal cut in N whose capacity equals the maximal number of edge-disjoint paths in G. But its capacity is simply the minimal number of edges of G that disconnects S from F.

Proof of part (ii). To the graph G attach a network N as in part (i) but allow infinite capacity on edges; give each vertex capacity 1. N is a network with vertex capacities. By "elongating" each vertex into an edge with capacity 1 (as in Section 5.17) obtain a new network with only edge capacities of 1 or ∞. The same arguments as in part (i) lead to the stated result. Observe that he resulting cut corresponds to a vertex cut and that the paths we obtain are vertex-disjoint. This ends the proof.

5.19

It is easy to derive König's theorem (proved "from scratch" in Section 5.7) from the vertex-disjoint version of Menger's theorem. Let us explain how this is done.

Line up the m girls in a column at left and the n boys in a column at right. Place an (undirected) edge between girl i and boy j if they like each other; else place no edge. Join all the girls to a new vertex S at left and join all boys to a new vertex F at right. Call the resulting graph (of $m + n + 2$ vertices) G.

Attach also a $m \times n$ matrix M of likings (girls versus boys) with (i, j)th entry 0 if girl i likes boy j, and 1 otherwise.

By a *line* we mean a row or a column of M. In the graph G a line corresponds to a vertex (i.e., a girl or a boy). We indicate a marriage between girl i and boy j by starring the 0 in position (i, j) of M. In the graph G a marriage between girl i and boy j should be viewed as a path (of *three edges*: Si, ij, and jF) between S and F (with the "middle" edge ij indicating the marriage).

We seek a maximal number of marriages. Since no bigamy is allowed we call a set of starred 0's *admissible* if no two are in the same row or column. A maximal number of marriages corresponds to an admissible set of starred 0's of maximal cardinality.

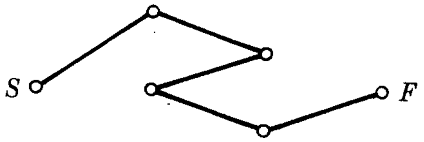
A set of lines is called *admissible* if their deletion removes all the 0's in M. In the graph G an admissible set of lines corresponds to a set of vertices whose removal disconnects S from F (i.e., it leaves no edges between the boys and the girls).

König's theorem is as follows:

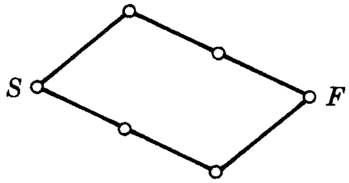
König's Theorem. *The cardinality of a maximal admissible set of starred 0's equals the cardinality of a minimal set of admissible lines.*

By Menger's theorem, part (ii) (see Section 5.18), we conclude that the minimal number of admissible lines equals the maximal number of vertex-disjoint paths between S and

F. To conclude the proof observe that if the number of vertex-disjoint paths is maximal, then all the paths must necessarily have length 3 (i.e., they are marriages); (this is clear, since any path of the form



can be replaced by the two paths below:



contradicting maximality). We proved that *the maximal number of possible marriages equals the minimal number of vertices whose deletion disconnect S from F.* Or, in matrix language, König's theorem (as stated).

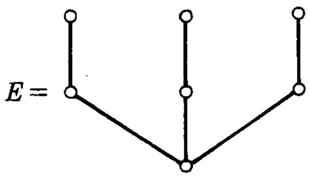
5.20 A Result by Dilworth

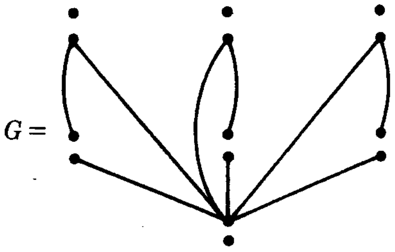
König's theorem implies, in turn, a result on the maximal number of disjoint chains in partially ordered sets, first proved by Dilworth. A *partially ordered set* is a pair $(E, \leq)$ consisting of a set of nodes E and a relation $\leq$ of partial order. By definition $\leq$ satisfies: $x \leq x$; $x \leq y$ and $y \leq x$ imply $x = y$; $x \leq y$ and $y \leq z$ imply $x \leq z$. A *chain* is a sequence of nodes $x_1, x_2, \ldots, x_n$ that satisfies $x_1 \leq x_2 \leq \cdots \leq x_n$. Two chains are said to be *disjoint* if they have no common nodes. We say that a set of chains *fills E* if they form a partition for the nodes of E. By a set of *mutually incomparable nodes* we understand a set of nodes no two of which are comparable (in the partial order $\leq$). The result we prove is:

Dilworth's Result. *The minimum number of disjoint chains that fills a partially ordered set equals the maximum number of mutually incomparable nodes.*

It is obvious that we need a separate chain for each of the incomparable nodes (so there are at least as many disjoint chains as there are incomparable nodes).

We establish the other inequality by a "marriage" argument. Represent the partially ordered set E by the usual (upward) Hasse diagram [i.e., place an edge between nodes x and y if $x \leq y$ and there is no z (other than x and y) to satisfy $x \leq z \leq y$]. Now replace each node of E by a pair of vertices $\begin{matrix}\circ \\ \\ \circ\end{matrix}$, the one on top a "girl" and the one on the bottom a "boy." Fill in the "likes" by the partial order $\leq$ from girls to boys (including the transitive links) to obtain a graph G, as shown below:





Note that *no edge* is placed between the girl and boy that replace a node of E. [*Aside*: Observe the "princesses" at the very top of G. Apparently none of the boys are quite good enough for them... .]

Given a chain $x_1 \leq x_2 \leq \cdots \leq x_n$ (with n distinct nodes) we produce $n-1$ marriages as follows: marry girl in x_1 to boy in x_2, girl in x_2 to boy in x_3, ... , girl in x_{n-1} to boy in x_n. It therefore follows that for a set of chains that *fills* E we have

$$number\ \ of\ \ marriages + number\ \ of\ \ chains = |E|.$$

This last relation now gives:

$$\begin{aligned}
&minimal\ \ number\ \ of\ \ such\ \ chains \\
&\qquad = |E| - maximal\ \ number\ \ of\ \ marriages \\
&\qquad = \{by\ \ Konig's\ \ theorem\} \\
&\qquad = |E| - minimal\ \ number\ \ of\ \ lines\ \ to\ \ delete\ \ all\ \ the \\
&\qquad\quad\ 0's\ \ in\ \ the\ \ girls\ \ versus\ \ boys\ \ matrix \\
&\qquad \leq maximal\ \ number\ \ of\ \ incomparable\ \ nodes.
\end{aligned}$$

To see the last inequality note that a maximal set of incomparable nodes produces a (square) block of 1's in the girls versus boys matrix. (One should simultaneously view this matrix as a node versus node matrix with 1's on the main diagonal.) Without loss the matrix looks like this:

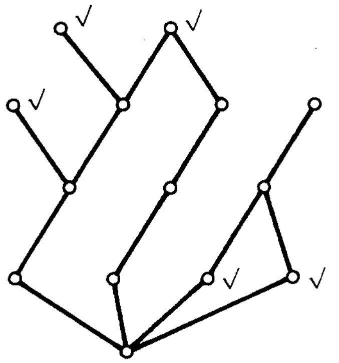
$$\begin{bmatrix}
1 & 1 & \cdots & 1 & | & | & | & & | \\
1 & 1 & \cdots & 1 & | & | & | & & | \\
 & & \vdots & & | & | & | & & | \\
1 & 1 & \cdots & 1 & | & | & | & & | \\
- & - & - & - & - & | & | & & | \\
- & - & - & - & - & - & | & & | \\
- & - & - & - & - & - & - & & | \\
 & & & & & & & \ddots & | \\
- & - & - & - & - & - & - & - & -
\end{bmatrix}$$

Each of the broken bordered "angles", that is,

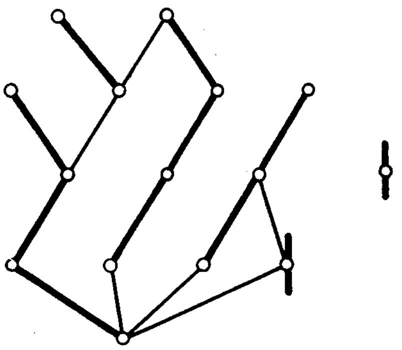
$$\begin{array}{cccc}
 & & & | \\
 & & & | \\
 & & & | \\
- & - & - & -
\end{array}$$

must contain a 0 (else we could get a larger set of incomparable nodes). We need a separate line for each such bordered "angle." The proof is now complete.

Example.



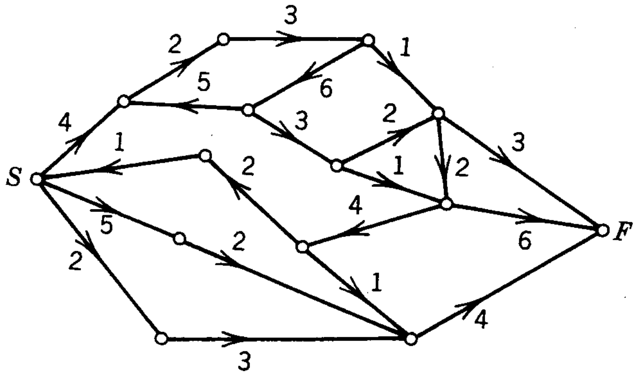
A maximal number of six mutually incomparable nodes indicated by "√"



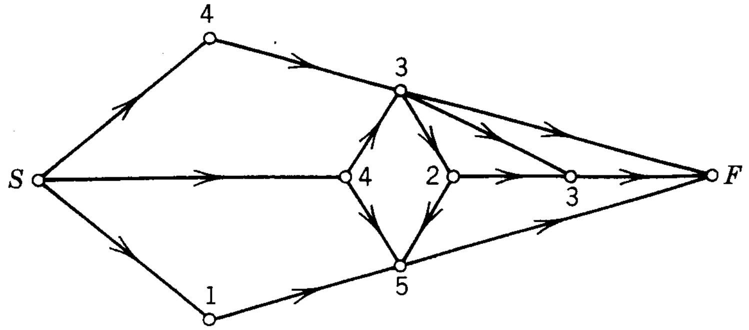
A minimal number of six disjoint chains that fill the partially ordered set

EXERCISES

1. Consider the following network (with capacities on edges as indicated): Find a maximal flow.

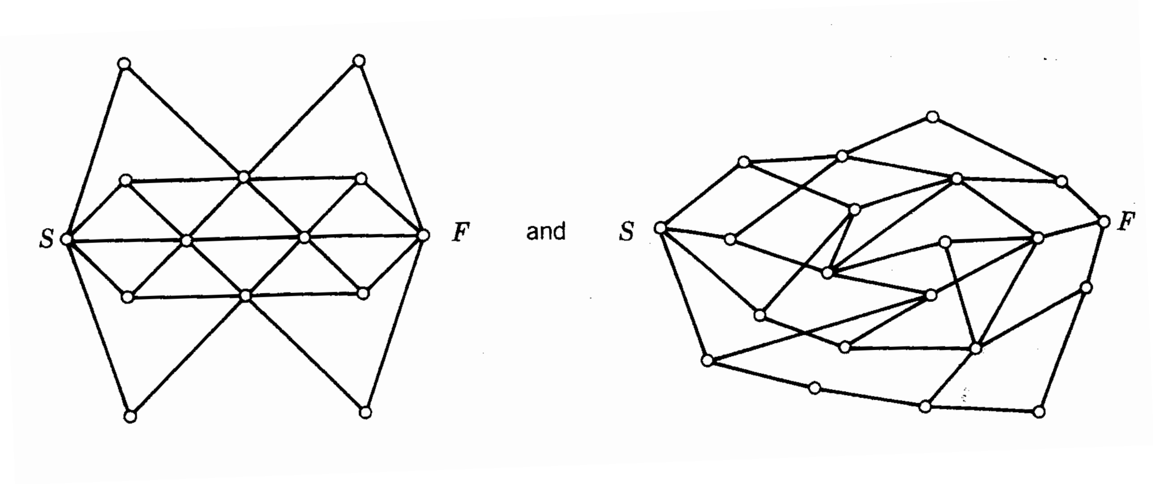


2. For the network with vertex capacities displayed below draw the corresponding

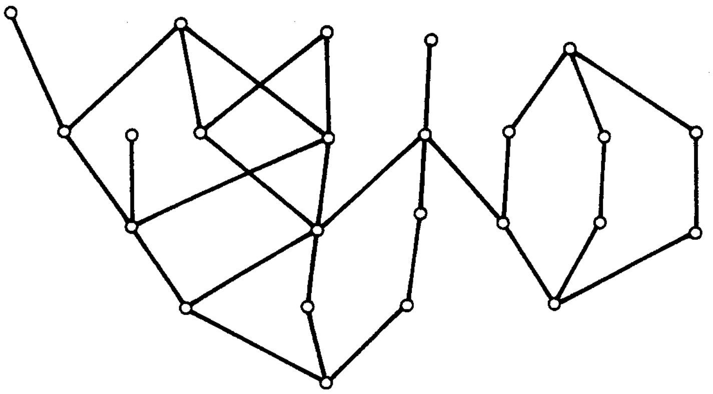


network (with edge capacities only). Find the maximal flow and the corresponding vertex cut.

3. Verify Menger's theorem for the graphs



4. What is the minimal number of disjoint chains that fill the following partially ordered set?



5. Let V be a vector subspace of R^n, and let $I_1, I_2, \ldots, I_n$, be nonempty intervals of the real line. Then there exists a vector $(x_1, x_2, \ldots, x_n) \in V$ such that $x_i \in I_i$, for all $1 \leq i \leq n$, if and only if for each vector of minimal support $(y_1, y_2, \ldots, y_n) \in V^{\perp}$ (the orthogonal complement of V) we have $0 \in \sum_{i=1}^{n} y_i I_i$. (By the sum $\sum_{i=1}^{n} y_i I_i$ we

mean the set of all inner products $\sum_{i=1}^{n} y_i z_i$ where $z_i \in I_i$, for all $1 \leq i \leq n$.) Prove this.

6. Derive the max-flow min-cut theorem as a consequence of the result in Exercise 5.

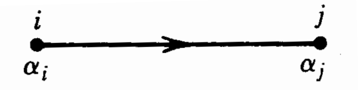
6 THE "OUT OF KILTER" METHOD

To display a certain duality we need to elevate some of the concepts introduced so far to a higher level of algebraic abstraction. Both flows and cuts are viewed as vectors in R^n, where n is the number of edges in the network (including the distinguished "yellow" edge between S and F - see Section 5.15). In such a context we can clearly see the "duality" that exists between flows and cuts.

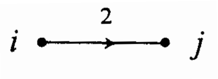
5.21 The Vector Space Formulation

Let N be a network with edges $E_1, \ldots, E_n$ and vertices $1, 2, \ldots, m$. The edge E_n is the distinguished "extra" edge between S (source) and F (sink). Let $K' \leq R^n$ be the set of possible flows; thus K' is the set of vectors $(x_1, \ldots, x_n)$ such that the directed sum at each vertex of the network is zero. (Note that we take $+x_i$ or $-x_i$ in the sum at j according to whether the arrow on E_i leaves j or enters j.) Evidently K' is a subspace. (In terms of electrical networks the conservation laws at the vertices are the Kirchhoff laws for current.)

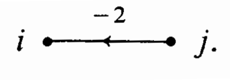
Let $K'' \leq R^n$ be the set of vectors $(y_1, \ldots, y_n)$ such that the directed sum around any cycle in the network is 0 (equivalently around any closed path). Evidently K'' is a subspace. We easily construct $y \in K''$ as follows: Assign arbitrary reals $\alpha_1, \ldots, \alpha_m$ to the vertices $1, 2, \ldots, m$. To each edge



assign $\alpha_j - \alpha_i$ (i.e., the directed difference). This produces a vector in K'' since any directed sum around a cycle becomes a "telescoping sum" and is therefore 0. We allow the possibility of changing the direction of an arrow on an edge if we also change the sign of both the flow and the directed difference through it; that is



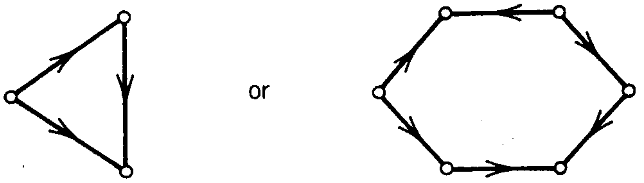
is the same as



Observe that:

∗ *Each $y \in K''$ arises from a labeling of the vertices* (unique up to a constant).

Proof. Select any vertex as base point and label it 0. Label all adjacent vertices to satisfy the directed difference condition for the corresponding y_i. Continue from these vertices to their adjacent vertices and so on, until all are labeled. There are two possible difficulties. We may move to a vertex already labeled or we may have two (or more) edges leading to a new vertex. In either case we get an associated closed path in N, for example,



The possible labels *agree*, since the directed sum around the path is zero.

The subspaces K' and K'' are orthogonal complements. Indeed, we have the following proposition:

Proposition 5.1. $R^n = K' \oplus K''$ (orthogonal direct sum).

Proof. Let $x \in K'$, $y \in K''$. Then

$$\begin{aligned}
\sum_{k=1}^{n} x_k y_k &= \sum_{k=1}^{n} x_k(\alpha_{j(k)} - \alpha_{i(k)}) \\
&= \sum_{i=1}^{m} \alpha_i \left(\sum_{\text{at vertex } i} \pm x_{k(i)} \right) \\
&= 0 \,(\text{the directed sum at } i \text{ being ”correctly” directed}).
\end{aligned}$$

(This shows that K' and K'' are orthogonal subspaces.)

We now show that they are orthogonal complements in R^n. Choose a spanning tree T of N. Since N has m vertices, T has $t = m - 1$ edges. We produce t linearly independent vectors in K'' (one for each edge of T) as follows: Pick an edge of T. Remove it to disconnect T. In one of the two resulting connected components label all vertices 0, and in the other label them all 1, and thence construct a vector in K''. All entries in this vector that correspond to edges of T are zero, except for the removed edge (which takes value -1 or 1). We thus obtain t vectors in K''. They are linearly independent; indeed, the vector that takes nonzero value (i.e., value -1 or 1) on an edge of T cannot be a linear combination of the others because they all take value 0 on that edge. Hence dim $K'' \geq t$.

On the other hand, with each edge not in T we produce a vector in K'. Each such edge produces a cycle with some edges of T. Appropriately assign -1 or 1 to the edges of the cycle and 0 to the other edges, to produce a flow (i.e., a vector in K'). We thus produce $n - t$ linearly independent vectors in K' and conclude that dim $K' \geq n - t$.

Hence $n = (n-t) + t \leq \dim K' + \dim K''$ and therefore $R^n = K' \oplus K''$. This ends our proof.

5.22

The maximal flow problem now takes the following form: We are given a network N with distinguished edge E_n. For each of the other edges E_i we have a (nonempty) interval of capacities I_i, $1 \leq i \leq n-1$.

We wish to find the maximal value of x_n subject to $x \in K'$ and $x_i \in I_i$ $(1 \leq i \leq n-1)$. We may change to a more symmetrical formulation by assigning an arbitrary interval I_n to E_n and asking when there is a *solution* to the problem $x \in K'$, $x_i \in I_i$, $(1 \leq i \leq n)$. For example, we may take I_n to consist of a single point and seek the maximal flow through E_n by gradually moving this point to the right (until a solution ceases to exist). [In our initial formulation (see Section 5.10) we took $I_i = [-c_i, c_i]$ with c_i the capacity of edge i, $l \leq i \leq n-1$.]

A *cocycle* in N is a set of edges whose removal disconnects N (i.e., it separates N into two connected components). As we shall see, some vectors of K'' correspond to cocycles in N. By the support of a vector we understand the set of coordinates with nonzero entries. A vector has minimal support if its support is minimal with respect to inclusion. As is easy to see, the vectors of minimal support *in a subspace* are unique up to scalar multiples (and consequently finitely many in number, up to such multiples).

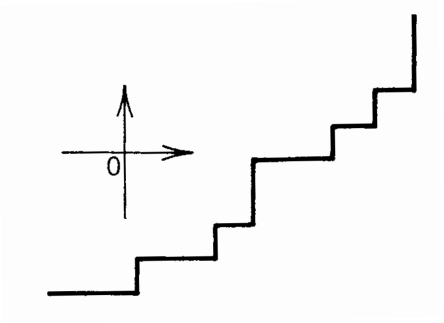
It turns out that we have a bijective correspondence between cocycles and vectors of minimal support with entries 0, ± 1 in K''. For any cocycle we can produce $y \in K''$ by labeling vertices in one connected component 0, and 1 in the other. Conversely, the (minimal) support of $y \in K''$ corresponds to edges with different end labels. Pick one such edge with labels $0 < 1$. Partition the vertices into those labeled $\leq \frac{1}{2}$ (say A) and those labeled $> \frac{1}{2}$ (say B). The edges running between A and B form the cocycle we want; note that the minimality of the support implies that *all* vertices in A are necessarily labeled 0 and those in B are labeled 1. [A parallel discussion goes for the cycles of N and the vectors with minimal support in K'.] We summarize:

$*$ *There is a bijective correspondence between the cycles (respectively, cocycles) of N and the vectors with entries 0, ± 1 of minimal support in K' (respectively K'').*

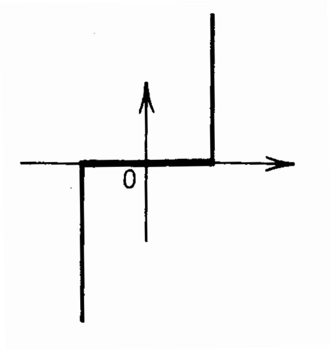
To translate the max-flow min-cut theorem in this language we first attach to each edge an interval I_i $(1 \leq i \leq n-1)$ of the form $[-c_i, c_i]$. A cut is a cocycle that omits E_n, the distinguished "yellow" edge. Evidently the maximal flow does not exceed the minimal sum of cut set capacities. We actually achieve equality by "lining up" all the arrows on a cut set and taking $I_n = \{-s\}$, where $s = \sum_i c_i$, with c_i the capacities of the cut set that maximizes s.

5.23 Problem Data for the "Out of Kilter" Algorithm

We are given a network with m vertices $1, 2, \ldots, m$ and n edges $E_1, E_2, \ldots, E_n$ (and we *do not* distinguish any of the edges at this stage). For each edge we have an increasing function (defined on some interval of the real line) *with vertical lines allowed.* By an increasing function we actually mean a discretized "straw" graph as follows:



[This generalizes the case described in Section 7.22 in which the graphs look like this:



– that is, we have (in essence) n nonempty intervals $I_1, I_2, \ldots, I_n$, one for each edge. It helps to parallelly interpret what we do next in this important special case.]

Let C_i be the "straw" graph attached to edge i. Annex to this background the subspaces K' and K'' of flows and "cocycles."

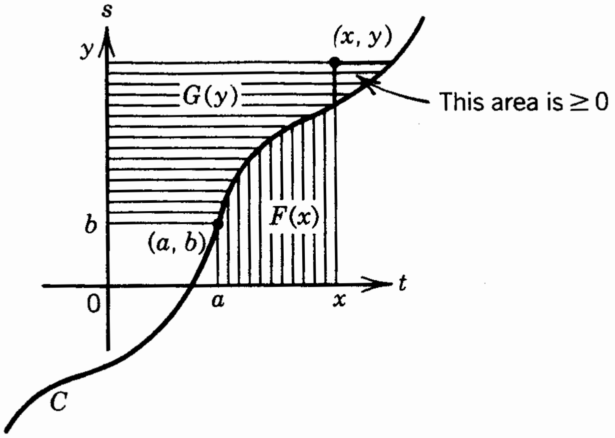
PROBLEM. *Find $x \in K'$ and $y \in K''$ such that $(x_i, y_i) \in C_i$, $i = 1, 2, \ldots, n$ (if any such x and y exist).*

The problem (as stated) treats K' and K'' on equal footing. This formulation permits us to treat questions on flows and cuts simultaneously and in duality. To find vectors x and y as in the problem, we initially start with a pair of vectors (in K' and K'', respectively) that do not quite satisfy $(x_i, y_i) \in C_i$, for all i. We can always move "a step closer," however, by being able to find *either* a cycle in K' or a *cocycle* in K''; this crucial fact is precisely Minty's arc coloring lemma working at its full potential (and it is in this context that that lemma was discovered).

REMARK. Assume that the curves C_i correspond to *intervals.* The task of finding a maximal flow x_n through E_n is implied by our *problem.* For if we have a way of finding solutions to $x \in K'$, $y \in K''$, and $(x_i, y_i) \in C_i$, we can in particular seek a solution in which the coordinate x_n in x is maximal (and discard y completely). Conversely, a solution $(x_1, \ldots, x_n) = x \in K'$ with $x_i \in I_i$, (or C_i, equivalently) and x_n maximal allows us to always take $(x_1, \ldots, x_n) = x \in K'$, $0 \in K''$, and then $(x_i, 0) \in C_i$, for all i.

5.24 A Connection to Young's Inequality

The *problem*, as stated in Section 5.23, is directly linked to two separate optimization problems. To understand the connection we need first recall Young's inequality for increasing functions. This inequality is so "graphical" that we can both state and prove it by looking at the following graphical display:



(5.8)

Here the point (a, b) is on the curve C and

$$F(x) = \int_a^x C(t)dt \text{ and } G(y) = \int_b^y C^{-1}(s)ds$$

are the antiderivatives of the increasing functions C and C^{-1} (the inverse function of C), respectively.

The inequality we mentioned is as follows:

Young's Inequality. $F(x) + G(y) + ab - xy \geq 0$ *with equality if and only if* $(x, y) \in C$.

[The statement is apparent from display (5.8) since $F(x) + G(y) + ab - xy$ is the area marked "this area is ≥ 0."]

Suppose now that *there exist* $x \in K'$ and $y \in K''$ such that $(x_i, y_i) \in C_i$ for *given* increasing curves C_i, $1 \leq i \leq n$. Select points (a_i, b_i) on the curves C_i, define antiderivatives F_i and G_i, and consider the function

$$H(x, y) = \sum_{i=1}^{n} F_i(x_i) + G_i(y_i) + a_i b_i - x_i y_i.$$

By Young's inequality $H(x, y) \geq 0$, with equality if and only if $(x_i, y_i) \in C_i$, for all i. Denote by $\overline{H}$ the restriction of H to $K' \times K''$. Since K' and K'' are orthogonal subspaces for $(x, y) \in K' \times K''$ we have $\sum_{i=1}^{n} x_i y_i = 0$ and hence $\overline{H}(x, y) = \sum_{i=1}^{n} F_i(x_i) + G_i(y_i) + a_i b_i$. Observe now that $\overline{H}(x, y) \geq 0$, and it equals 0 if and only if $(x, y) \in K' \times K''$ is such that $(x_i, y_i) \in C_i$, for all i. Moreover, since the variables are separate in $\overline{H}(x, y)$ we can conclude that $\overline{H}(x, y)$ attains the minimal value of 0 if and only if x in K' minimizes $\sum_{i=1}^{n} F_i(x_i)$ and y in K'' minimizes $\sum_{i=1}^{n} G_i(y_i)$. In conclusion we state the following:

$*$ *Let* n *increasing curves* C_i *with antiderivatives* F_i *and* G_i *be given. There exist vectors* $x \in K'$ *and* $y \in K''$ *such that* $(x_i, y_i) \in C_i$, *for all* i, *if and only if there exists* x *in* K'

that minimizes $\sum_{i=1}^{n} F_i(x_i)$ *and there exists* y *in* K'' *that minimizes* $\sum_{i=1}^{n} G_i(y_i)$.

A solution to our *problem* can thus be found by solving two *separate* minimization problems.

To each increasing curve C_i we attach a (nonempty) interval I_i on the horizontal axis, namely its *projection* on that axis, and similarly a (nonempty) interval J_i on the vertical axis. An obvious *necessary* condition for our *problem* to have a solution is that vectors $x \in K'$ and $y \in K''$ exist, such that $x_i \in I_i$ and $y_i \in J_i$, for all i. As it turns out this necessary condition is actually sufficient and this is explained in a constructive manner by the "out of kilter" algorithm.

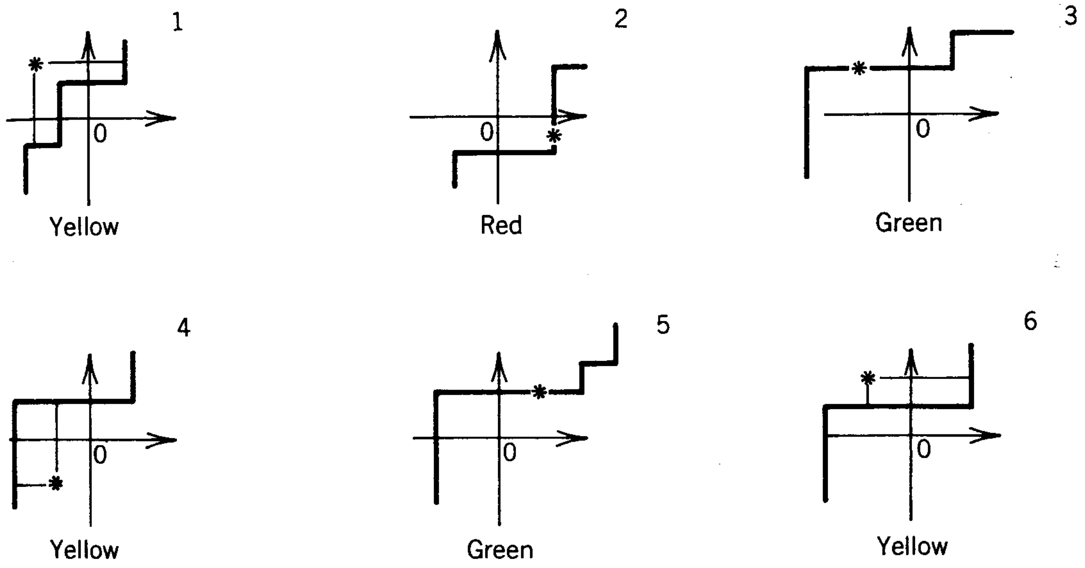
5.25 The "Out of Kilter" Algorithm

Let a network N be given with an increasing "straw" curve attached to each one of its n edges. We adopt the following fundamental *convention.* We may change the direction of an arrow on an edge provided that we also do three other things along with that: Change the *sign of the flow* through that edge, change the *sign of the directed difference* through that edge, and also *rotate by* 180° the increasing curve attached to that edge. (This change corresponds to a simple change of variable – an actual change of sign, in fact.)

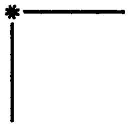
Denote the edges of N by $E_1, \ldots, E_n$ and the associated increasing "straw" curves by $C_1, \ldots, C_n$. Also let $I_1, \ldots, I_n$ and $J_1, \ldots, J_n$ be the intervals obtained by projecting the curves on the horizontal and vertical axes, respectively. *Assume* there exist "starting points" $x \in K'$ and $y \in K''$ such that $x_i \in I_i$ and $y_i \in J_i$, for all $1 \leq i \leq n$. If in fact $(x_i, y_i) \in C_i$ for all i, *we are done.* A solution to our *problem* (see Section 5.23) was found.

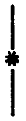
If not, there exists at least one edge (edge i, say) for which (x_i, y_i) is not on the curve C_i (we say that edge i is out of kilter, hence the terminology). Focus attention on edge i: Color it yellow and think of it as being the distinguished yellow edge. The essence of what we do is two-fold: We bring edge i in kilter *and* keep in kilter all edges that were already in kilter. [The rigorous definition of an edge (say, edge j) being in *kilter* is the existence of x in K' and y in K'' such that $(x_j, y_j) \in C_j$.]

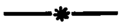
The general step is graphically illustrated below:



We have $n = 6$ edges, and the six "straw" curves C_i (trails of honey) are drawn in somewhat thicker lines. The asterisk (a *bear* in Minty's colorful orations) on edge i is the point (x_i, y_i), with $x = (x_1, \ldots, x_6) \in K'$ and $y = (y_1, \ldots, y_6) \in K''$. Edges 2, 3, and 5 are in kilter (and they will continue to remain so). *Color* the edges of the network as follows: If the local picture (indicated in the lighter lines for edges not in kilter) is:

color the edge yellow 

color the edge red 

color the edge green. 

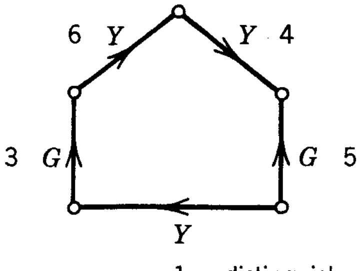
(If we have then change the arrow on that edge; accordingly change the sign of the flow and the directed difference through it and *rotate* the graph attached to that edge by 180°. Now the local picture becomes and we color the edge *yellow.* Edge 4 above offers this opportunity.) Select an edge that is out of kilter (it will be a yellow edge) and think of it as the "distinguished" yellow edge; edge 1 will be our choice.

Each edge of the network is now colored yellow, red, or green (with edge 1 being distinguished yellow). Minty's arc coloring lemma, see Lemma 5.3, *assures* the existence of *precisely one* of the following two possibilities:

(a) A *cycle* containing the distinguished yellow edge with all its edges green or yellow oriented in the direction of the distinguished yellow edge.

(b) A *cocycle* containing the distinguished yellow edge with all its edges red or yellow oriented in the direction of the distinguished yellow edge.

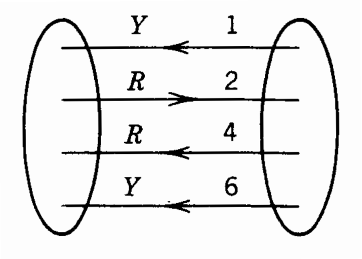
If we find a cycle as in (a), say



1 = distinguished yellow

we move the asterisks to the *right* on the graphs attached to edges 1, 3, 6, and 4, but to the *left* on the graph attached to edge 5 (because its arrow is against the direction of the distinguished yellow edge – equivalently we could first change the direction of the arrow on edge 5, thus *rotating* the graph of edge 5 by 180°, and then move *all* the asterisks to the right). Note that upon performing these changes to the graphs of the edges that occur in the cycle all edges that were in kilter *are still in kilter*, and that the asterisk on the graph of the distinguished yellow edge (edge 1 in our case) is definitely *one step closer* to its thicker curve (curve C_1 in our case).

If we find a cocycle as in (b), say the one drawn below (for the sake of conversation),



we move the asterisks *down* on the graphs attached to edges l, 4, and 6, but up on the graph attached to edge 2 (because its arrow is against the direction of our distinguished yellow edge – equivalently we could first change the direction of the arrow on edge 2, thus *rotating* the graph of edge 2 by 180°, and then move *all* the asterisks down). Note that upon performing these changes to the graphs of the edges that occur in the cocycle all edges that were in kilter *are still in kilter*, and that the asterisk on the graph of the distinguished yellow edge (edge 1 in our case) is *definitely one step closer* to its thicker curve (curve C_1 in our case).

The motions of the asterisks thus induced generate new vectors $(x_1, \ldots, x_6) = x \in K'$ and $(y_1, \ldots, y_6) = y \in K''$ such that for the distinguished yellow edge (edge 1 in our case) we have the asterisk (x_1, y_1) *one step closer* to C_1 than before (and such that all edges that were in kilter before remain in kilter).

Repeat this process until edge 1 (the distinguished yellow edge) is in kilter; then, if there are other edges out of kilter (edge 2, say), call edge 2 the distinguished yellow edge and bring *it* in kilter, until all edges are in kilter. We thus found $x \in K'$ and $y \in K''$ such that $(x_i, y_i) \in C_i$, $1 \leq i \leq n$ $(= 6)$, in our example. This ends the description of the algorithm.

A Concluding Remark

All that we went through was done under the assumption that a solution to the "projected problem" exists, that is, that there exist $x \in K'$ and y in K'' such that $x_i \in I_i$ and $y_i \in J_i$, for all i.

On occasion such a solution may be apparent from the general context of the problem and we can use it as our starting point for the "out of kilter" algorithm. If not, start with $0 \in K'$ and $0 \in K''$ and apply the "out of kilter" algorithm to the same network but with curves C_i replaced by their projections I_i; stop when a cocycle appears (then there is no solution), else obtain a point x in K' with $x_i \in I_i$, $1 \leq i \leq n$. Do the same for the J_i's and stop when a cycle appears. If no cycle occurs obtain a point y in K'' with $y_i \in J_i$, for all i. The vectors x and y are starting points for the "out of kilter" algorithm.

7 MATROIDS AND THE GREEDY ALGORITHM

Notions such as linear independence, trees, or matchings in a graph can be treated in a unified axiomatic way through what has become known as matroid theory. There exists also an algorithmic characterization of matroids: It states (roughly speaking) that a combinatorial optimization problem should be solved using a ”greedy” algorithm if and only if a matroid is lurking somewhere in the background.

5.26 The Definition of a Matroid

A *matroid* (P, S) consists of a set P of points with a set S of distinguished subsets of P called *independent sets* satisfying the two axioms below:

$$\textit{If } I \textit{ is in } S, \textit{ then all subsets of } I \textit{ are in } S. \textit{ The empty set is in } S. \tag{5.9}$$

$$\textit{If } I \textit{ and } J \textit{ are subsets of } S \textit{ with } n \textit{ and } n+1 \textit{ points, respectively, then there exists a point } e \textit{ in } J \textit{ but not in } I \textit{ such that } I \cup \{e\} \textit{ is in } S. \tag{5.10}$$

These axioms are motivated by three examples that we now bring to attention.

Example 1 (linear Independence). Let V be a finite-dimensional vector space over an arbitrary field, finite or infinite. Take the vectors in V as points. The set S of distinguished subsets consists of all linearly independent sets of vectors.

We take as our premise that the empty set belongs to S. Using freely the definitions and results of linear algebra, axiom (5.9) is immediately verified (it simply states that any subset of a linearly independent set of vectors is linearly independent). Axiom (5.10) is a little more difficult to verify directly. But it is a well-known result called the exchange principle. [Using familiar concepts in linear algebra, if I and J are as in the axiom, and if each vector in J is a linear combination of vectors in I, then J is part of the subspace generated by I. This is not possible, since I generates a subspace of dimension n, and J consists of $n+1$ independent vectors. Axiom (5.10) is therefore satisfied.]

Example 2 (Cycle-Free Subsets of a Graph). P is the set of edges of a graph. A subset of edges is distinguished (and thus belongs to S) if it contains no cycles. One may verify that the pair (P, S) with P and S so defined is a matroid. Verifying axiom (5.10) can be a little tricky but not exceedingly so.

Example 3 (The Matching Matroid). Let the set of points consist of the vertices of a graph. The distinguished subsets are those subsets of points for which a matching covering all the vertices of the subset exists. We refer to Section 5.5 for the definition of a matching. This defines a matroid. Details of verification are omitted.

5.27

We now explore some consequences of our definition. Let (P, S) be a matroid. Fix a subset A of P (independent or not). Set A contains independent subsets, since the empty set is independent. We assert that *all independent subsets of A maximal with respect to inclusion have the same cardinality.* [Indeed, let I and J be two maximal subsets of A. Assume that $|I| < |J|$. Select a subset of J with $|I| + 1$ points; call it K. Axiom (5.9) informs us that K, being a subset of J, is independent. Axiom (5.10) assures the existence of a point e in K but not in I such that $I \cup \{e\}$ is independent. The set $I \cup \{e\}$ contains I strictly, thus contradicting the assumed maximality of I. We thus conclude that $|I| < |J|$ cannot occur, and neither can $|J| < |I|$ for completely analogous reasons. Thus $|I| = |J|$, as stated.] This observation allows us to define the *rank* of A as the cardinality of any of its maximal independent subsets. We write $r(A)$ for the rank of A.

Some properties of the notion of rank are easy to perceive. It is clear, for instance, that $r(\emptyset) = 0$. Furthermore, *for a subset A and a point e not in A we have either $r(A \cup \{e\}) = r(A)$ or $r(A \cup \{e\}) = r(A) + 1$.* [To see this, let T be a maximal independent set of $A \cup \{e\}$. If $T \subseteq A$, then clearly $r(A \cup \{e\}) = |T| = r(A)$. If not, then necessarily $T = N \cup \{e\}$, with N a maximal independent set of A. In this latter case $r(A \cup \{e\}) = |T| = |N| + 1 = r(A) + 1$.]

Observe yet another property of the rank. *If e_1 and e_2 are points not in A and if $r(A \cup \{e_1\}) = r(A \cup \{e_2\}) = r(A)$, then $r(A \cup \{e_1\} \cup \{e_2\}) = r(A)$.* [This is easy to establish by observing that neither e_1 nor e_2 can be part of a maximal independent set of $A \cup \{e_1\} \cup \{e_2\}$ (otherwise the rank of either $A \cup \{e_1\}$ or $A \cup \{e_2\}$ will exceed the rank of A, contrary to our assumptions). The maximal independent sets of A and $A \cup \{e_1\} \cup \{e_2\}$ are therefore the same, and so then are their ranks.]

It turns out that a function r defined on the subsets of a finite set and possessing the properties of the rank mentioned above is in fact the rank function of a matroid with independent sets described as follows: $S = \{I : r(I) = |I|\}$. Apart from this, many other characterizations of a matroid are known.

To readers interested in more work with axioms we suggest proving the following result due to Nash and Williams:

$*$ *If* (P_1, S_1) *is a matroid and* f *is a function from* P_1 *to* P_2*, then* $(P_2, f(S_1))$ *is also a matroid. (By* $f(S_1)$ *we understand* $\{f(T) : T \in S_1\}$*.)*

5.28 The Result of Rado and Edmonds

Let (P, S) be a matroid. To each point e of P we assign a nonnegative weight $w(e)$. The *weight* of a finite subset of P equals (by definition) the sum of weights of its points. *Our objective is to find an independent set of maximum weight.*

We take interest in a special ordering induced by the weight function w on the independent sets: the *lexicographic* ordering. It is this ordering that usually suggests itself at the outset in many maximization problems, for it accommodates well the impulses of a "greedy" mind. We can describe it as follows:

Let $I = \{a_1, a_2, \ldots, a_n\}$ and $J = \{b_1, b_2, \ldots, b_m\}$ be two independent sets with points listed in the order of weight $w(a_1) \geq w(a_2) \geq \cdots \geq w(a_n)$ and $w(b_1) \geq w(b_2) \geq \cdots w(b_m)$. We say that I is lexicographically greater than or equal to J if there exists an index k such that $w(a_i) = w(b_i)$ for $i = 1, 2, \ldots, k-1$ and $w(a_k) > w(b_k)$; or else $n \geq m$ and $w(a_i) = w(b_i)$ for $i = 1, 2, \ldots, m$. Any two independent sets are comparable (but it may happen that I is lexicographically greater than or equal to J and J is lexicographically greater than or equal to I without I and J being actually equal). An independent set that is not lexicographically less than any other set is said to be lexicographically maximum.

The greedy algorithm is based on the following result:

The Rado-Edmonds Theorem.

(a) *If (P, S) is a matroid, then:*

For any nonnegative weighing of the points in P, a lexicographically maximum set in S has maximum weight. (5.11)

(b) *If (P, S) is a finite structure of points and subsets satisfying* (5.11) *and axiom* (5.9), *then (P, S) is a matroid.*

Proof. We prove part (a) first. Let $I = \{a_1, a_2, \ldots, a_n\}$ be a lexicographically maximum set, where $w(a_1) \geq w(a_2) \geq \cdots \geq w(a_n)$. Let $J = \{b_1, b_2, \ldots, b_m\}$ be any independent set, where $w(b_1) \geq w(b_2) \geq \cdots \geq w(b_m)$. We assert that $w(a_i) \geq w(b_i)$, for all i. [Indeed, if $w(a_k) < w(b_k)$ for some k, then focus attention on the sets $\{a_1, a_2, \ldots, a_{k-1}\}$ and $\{b_1, b_2, \ldots, b_k\}$. By axiom (5.10) there exists a point in the latter set (b_j, say) such that $\{a_1, a_2, \ldots, a_{k-1}, b_j\}$ is an independent set. But this set is then lexicographically greater than I, a contradiction.] Since $w(a_i) \geq w(b_i)$ for all i, we conclude that I is an independent set of maximum weight.

Part (b) is proved next. Assume that (P, S) is a finite structure of points and subsets satisfying (5.11) and axiom (5.9). We also assume that (P, S) is not a matroid and derive a contradiction.

First, we assert that there exists a subset A of P and two subsets I and J in S both maximal in A with respect to inclusion such that $|I| < |J|$. [For if not $|I| = |J|$ for all maximal subsets I and J of any subset A of P. Then let Q and R be in S, of cardinalities n and $n + 1$, respectively. Let $A = Q \cup R$. Since $|Q| = n$ and $|R| = n + 1$, Q cannot be maximal in A. Hence there must exist a point e in $R - Q$ such that $Q \cup \{e\}$ is in S. But this is what axiom (5.10) states. It thus follows that (P, S) is a matroid, contrary to one of our assumptions.]

With subset A, and subsets I and J thus identified, assign weights as follows: Each point in I has weight $1 + \varepsilon$ (for ε an as yet unspecified positive number), each point in $J - I$ has weight 1, and all of the remaining points in P have weight 0. If ε is chosen positive and suitably small the set I is contained in a lexicographically maximum set whose weight is less than that of J. This violates assumption (5.11). The finite structure (P, S) must therefore be a matroid. This ends our proof.

In complete analogy to what we just accomplished one may characterize a matroid in terms of lexicographically minimum sets and sets of minimum weight. All definitions and arguments parallel the ones used above.

5.29 The Greedy Algorithm

Based on the result of Rado and Edmonds we can now describe the greedy algorithm.

We have before us a matroid (P, S) and a weight function w defined on the set of points P. Our objective is to find an independent set of maximum weight.

Observe that we may assume without loss of generality that each point is an independent set of cardinality 1. [For if some point is not, then (by axiom (5.9)) this point cannot occur in any of the independent sets. Therefore we might as well omit it altogether from the set P of the initial points.]

The Algorithm

Select a point e_1 of maximal weight. Let $S_1 = \{e_1\} \in S$. Then choose a point e_2 not in S_1 such that $S_1 \cup \{e_2\}$ is an independent set of maximal weight. Let $S_2 = S_1 \cup \{e_2\}$. Choose a point e_3 not in S_2 such that $S_2 \cup \{e_3\}$ is an independent set of maximal weight. Proceed... .

After a finite number of steps (r steps, say) a lexicographically maximal independent set S_r will be found. By the result of Rado and Edmonds in Section 5.28, S_r is in fact an independent set of maximal weight. End of algorithm.

The problem of locating an independent set of minimum weight is treated in a completely analogous way.

The algorithm is clearly a "greedy" one for it maximizes weight at each step of the selection process. But if the underlying structure is a matroid this greedy procedure is in fact the desired one.

5.30

Generally a "greedy" form of selection does not produce maximum weight. In the simplest of situations one can see the severe drawbacks of such an attitude:

Being the matchmaker that I am, Larry offers me \$10 for a date with Mary and \$8 for a date with Sherry; Harry gives me \$8 for a date with Mary and \$1 for a date with Sherry. If I am greedy in the sense that I wish to maximize weight (income in this case) at each and every step, I will first take the \$10 from Larry and pair him off with Mary; then I must pair up Harry with Sherry for \$1, for a total income of \$11. (I could have cashed in \$16 though by having Larry date Sherry and Harry date Mary.) Following a "greedy" procedure results in a loss of \$5.

One would not expect the greedy algorithm to be of much use in most optimization problems, and rightly so. The maximum matching algorithm (of Section 5.6) will in all likelihood be easier to adapt to a general optimization problem. But the Rado-Edmonds theorem tells us precisely which optimization problems can be solved by a greedy algorithm: those to which a matroid can be attached in a natural way. Let us examine two such problems.

PROBLEM 1. Given a finite set of vectors in a finite-dimensional vector space find a set of linearly independent vectors whose sum of Euclidean lengths is maximum.

There is a matroid (P, S) that awaits notice: P is the set of vectors available and S the linearly independent subsets of P. The solution to this problem can be obtained by a greedy algorithm: Select the longest vector, then the next longest linearly independent of the first, the third longest linearly independent of the first two, and so on. The resulting linearly independent set will be of maximal total length (for this is what the Rado-Edmonds theorem asserts).

PROBLEM 2. Given a connected graph with weighted edges, find a spanning tree of minimum weight (i.e., a spanning tree with minimum sum of weights on its edges).

A matroid is quietly waiting to be noticed here as well. It is (P, S), with P the set of edges of the graph and S the cycle-free subsets of edges. The greedy algorithm tells us how to find such a spanning tree. First select the lightest edge, then the next lightest that forms no cycle with the first, then the third lightest that forms no cycles with the first two, and so on. This greedy process leads in the end to a spanning tree of minimum weight. Surprised? No, because of the matroid lurking in the background and because the Rado-Edmonds theorem is at work.

EXERCISES

1. Let (P, S) be a matroid. The *span* of a subset A of P is the maximal set containing A and having the same rank as A (by maximal we mean maximal with respect to inclusion). Show that the span of a subset is unique.

2. Take as points the elements of the set $\{1, 2, 3, 4, 5, 6, 7\}$. Let the maximal independent sets consist of all subsets of three points except $\{1, 2, 4\}$, $\{2, 3, 5\}$, $\{3, 4, 6\}$, $\{4, 5, 7\}$, $\{5, 6, 1\}$, $\{6, 7, 2\}$, and $\{7, 1, 3\}$. Is this finite structure a matroid?

3. A maximal independent set of a matroid is called a *basis*. Let $\overline{B}$ be the set of bases of a matroid. Then:

   (a) $\overline{B} \neq \emptyset$, and no set in $\overline{B}$ contains another properly.

   (b) If B_1 and B_2 are in $\overline{B}$ and e_1 is a point in B_1, then there exists a point e_2 in B_2 such that $(B - \{e_1\}) \cup \{e_2\}$ is also in $\overline{B}$.

   Show that, conversely, if $(P, \overline{B})$ is a finite structure satisfying (a) and (b) above, then (P, S) is a matroid, where

$$S = \{I : I \subseteq B, \text{ for some } B \text{ in } \overline{B}\}.$$

4. Let (P, S_1) and (P, S_2) be two matroids on the same set of points. By $sp_i(A)$ we denote the span of set A in the matroid (P, S_i), $i = 1, 2$ (for a definition of span read Exercise 1). Let I and J be subsets each belonging to both S_1 and S_2. Prove that there exists K, a subset belonging to both S_1 and S_2, such that $K \subseteq I \cup J$, $sp_1(I) \subseteq sp_1(K)$ and $sp_2(J) \subseteq sp_2(K)$.

5. Let (P_1, S_1) and (P_2, S_2) be matroids. Define P and S as follows: $P = P_1 \cup P_2$, and $S = \{I : I = I_1 \cup I_2, \ I_1 \in S_1, \ \text{and} \ I_2 \in S_2\}$. Show that (P, S) is a matroid.

NOTES

The material presented in this chapter is now available in several books. For more complete information we refer the reader to [1], [2], and [3]. The point of view and much of the details took shape in my mind during the numerous informal discussions with George Minty, a specialist on network flows. Most of the material comes from a course in combinatorial theory that we taught jointly in the spring of 1983. With sadness I wish to mention that Professor Minty has recently passed away; he will be fondly remembered.

Unimodular matrices go back a long way, and Lemma 5.2 can be traced to the work of Poincaré. The maximum matching algorithm in bipartite graphs was apparently first given by Munkres [4]. Section 2 contains the classical results of König and P. Hall on matchings in bipartite graphs. All books on the subject include them. Short inductive proofs can be given but we preferred the narrative, more "constructive" approach. The arc coloring lemmas are Minty's, they being the cornerstone of his "out of kilter" method

[5] described in Section 6. In Section 4 the main result is the well-known max-flow min-cut theorem; it was discovered by Ford and Fulkerson (see [6]). Matroids were invented by Whitney and several of the characterizations we mentioned are due to him. The greedy algorithm was so dubbed by Edmonds; it appears in [7].

REFERENCES

1. E. Lawler, *Combinatorial Optimization: Networks and Matroids*, Holt, Rinehart, and Winston, New York, 1976.

2. R. T. Rockafeller, *Network Flows and Monotropic Optimization*, Wiley, New York, 1984.

3. R. C. Bose and B. Manvel, *Introduction to Combinatorial Theory*, Wiley, New York, 1984.

4. J. Munkres, Algorithms for the assignment and transportation problems, *J. Soc. Indust. Appl. Math.*, **5**, 32-38 (1957).

5. G. J. Minty, Monotone networks, *Proc. Royal Soc. London, Ser. A*, **257**, 194-212 (1960).

6. L. R. Ford, Jr. and D. R. Fulkerson, Maximal flow through a network, *Canad. J. Math.*, **8**, 399-404 (1956).

7. J. Edmonds, Matroids and the greedy algorithm, *Math. Programming*, **1**, 127-136 (1971).