Part I

Paths and Flows

Paths belong to the most basic and important objects in combinatorial optimization. First of all, paths are of direct practical use, to make connections and to search. One can imagine that even in very primitive societies, finding short paths and searching (for instance, for food) is essential. 'Short' need not be just in terms of geometric distance, but might observe factors like differences in height, crosscurrent, and wind. In modern societies, searching is also an important issue in several kinds of networks, like in communication webs and data structures. Several other nonspatial problems (like the knapsack problem, dynamic programming) can be modelled as a shortest path problem. Also planning tools like PERT and CPM are based on shortest paths.

Next to that, paths form an important tool in solving other combinatorial optimization problems. In a large part of combinatorial algorithms, finding an appropriate path is the main issue in a subroutine or in the iterations. Several combinatorial optimization problems can be solved by iteratively finding a shortest path.

Disjoint paths were first investigated in a topological setting by Menger starting in the 1920s, leading to Menger's theorem, a min-max relation equating the maximum number of disjoint $s-t$ paths and the minimum size of an $s-t$ cut. The theorem is fundamental to graph theory, and provides an important tool to handle the connectivity of graphs.

In a different environment, the notion of *flow* in a graph came up, namely at RAND in the 1950s, motivated by a study of the capacity of the Soviet and East European railway system. It inspired Ford and Fulkerson to develop a maximum flow algorithm based on augmenting paths and to prove the max-flow min-cut theorem. As flows can be considered as linear combinations of incidence vectors of paths, there is a close connection between disjoint paths and flow problems, and it turned out that the max-flow min-cut theorem and Menger's theorem can be derived from each other.

Minimum-cost flows can be considered as the common generalization of shortest paths and disjoint paths/flows. Related are minimum-cost circulations and transshipments. This connects the topic to the origins of linear programming in the 1940s, when Koopmans designed pivot-like procedures for minimum-cost transshipment in order to plan protected ship convoys during World War II.

Actually, linear programming and polyhedral methods apply very favourably to path and flow problems, by the total unimodularity of the underlying constraint matrices. They lead to fast, strongly polynomial-time algorithms for such problems, while also fast direct, combinatorial algorithms have been found.

Chapters:

Chapter 6

Shortest paths: unit lengths

The first three chapters of this part are devoted to the shortest path problem. A number of simple but fundamental methods have been developed for it.

The division into the three chapters is by increasing generality of the length function. In the present chapter we take unit lengths; that is, each edge or arc has length 1. Equivalently, we search for paths with a minimum number of edges or arcs. We also consider 'zero length', equivalently, searching for *any* path.

Next, in Chapter 7, we consider nonnegative lengths, where Dijkstra's method applies. Finally, in Chapter 8, we go over to arbitrary lengths. If we put no further constraints, the shortest path problem is NP-complete (in fact, even if all lengths are -1). But if there are no negative-length directed circuits, the problem is polynomial-time solvable, by the Bellman-Ford method.

The methods and results in this chapter generally apply to directed and undirected graphs alike; however, in case of an undirected graph with length function such that each circuit has nonnegative length, the problem is polynomial-time, but the method is much more involved. It can be solved in polynomial time with nonbipartite matching methods, and for this we refer to Section 29.2.

In this chapter, graphs can be assumed to be simple.

6.1. Shortest paths with unit lengths

Let $D = (V, A)$ be a digraph. In this chapter, the length of any path in D is the number of its arcs. For $s, t \in V$, the *distance* from s to t is the minimum length of any $s - t$ path. If no $s - t$ path exists, we set the distance from s to t equal to ∞.

There is an easy min-max relation, due to Robacker [1956b], characterizing the minimum length of an $s - t$ path. Recall that a subset C of A is an $s - t$ *cut* if $C = \delta^{\text{out}}(U)$ for some subset U of V satisfying $s \in U$ and $t \notin U$.[1]

[1] $\delta^{\text{out}}(U)$ and $\delta^{\text{in}}(U)$ denote the sets of arcs leaving and entering U, respectively.

Theorem 6.1. *Let $D = (V, A)$ be a digraph and let $s, t \in V$. Then the minimum length of an $s-t$ path is equal to the maximum number of disjoint $s-t$ cuts.*

Proof. Trivially, the minimum is at least the maximum, since each $s-t$ path intersects each $s-t$ cut in at least one arc. That the minimum is equal to the maximum follows by considering the $s-t$ cuts $\delta^{\text{out}}(U_i)$ for $i = 1, \ldots, d$, where d is the distance from s to t and where U_i is the set of vertices of distance less than i from s. ∎

(This is the proof of Robacker [1956b].)

Dantzig [1957] observed the following. Let $D = (V, A)$ be a digraph and let $s \in V$. A rooted tree $T = (V', A')$ rooted at s is called a *shortest paths tree* (*rooted at* s) if V' is the set of vertices in D reachable from s and $A' \subseteq A$, such that for each $t \in V'$, the $s-t$ path in T is a shortest $s-t$ path in D.

Theorem 6.2. *Let $D = (V, A)$ be a digraph and let $s \in V$. Then there exists a shortest paths tree rooted at s.*

Proof. Let V' be the set of vertices reachable in D from s. Choose, for each $t \in V' \setminus \{s\}$, an arc a_t that is the last arc of some shortest $s-t$ path in D. Then $A' := \{a_t \mid t \in V' \setminus \{s\}\}$ gives the required rooted tree. ∎

The above trivially applies also to *undirected* graphs.

6.2. Shortest paths with unit lengths algorithmically: breadth-first search

The following algorithm of Berge [1958b] and Moore [1959], essentially *breadth-first search*, determines the distance from s to t. Let V_i denote the set of vertices of D at distance i from s. Then $V_0 = \{s\}$, and for each i:

(6.1) V_{i+1} is equal to the set of vertices $v \in V \setminus (V_0 \cup V_1 \cup \cdots \cup V_i)$ for which $(u, v) \in A$ for some $u \in V_i$.

This gives us directly an algorithm for determining the sets V_i: we set $V_0 := \{s\}$ and next we determine with rule (6.1) the sets $V_1, V_2, \ldots$ successively, until $V_{i+1} = \emptyset$.

In fact, it gives a linear-time algorithm, and so:

Theorem 6.3. *Given a digraph $D = (V, A)$ and $s, t \in V$, a unit-length shortest $s-t$ path can be found in linear time.*

Proof. Directly from the description. ∎

In fact, the algorithm finds the distance from s to all vertices reachable from s. Moreover, it gives the shortest paths, even the shortest paths tree:

Theorem 6.4. *Given a digraph $D = (V, A)$ and $s \in V$, a shortest path tree rooted at s can be found in linear time.*

Proof. Use the algorithm described above. ∎

6.3. Depth-first search

In certain cases it is more useful to scan a graph not by breadth-first search as in Section 6.2, but by *depth-first search* (a variant of which goes back to Tarry [1895]).

Let $D = (V, A)$ be a digraph. Define the operation of *scanning* a vertex v recursively by:

(6.2) For each arc $a = (v, w) \in \delta^{\text{out}}(v)$: delete all arcs entering w and scan w.

Then depth-first search from a vertex s amounts to scanning s. If each vertex of D is reachable from s, then all arcs a chosen in (6.2) form a rooted tree with root s. This tree is called a *depth-first search tree*.

This can be applied to find a path and to sort and order the vertices of a digraph $D = (V, A)$. We say that vertices $v_1, \ldots, v_n$ are in *topological order* if $i < j$ for all i, j with $(v_i, v_j) \in A$. So a subset of V can be topologically ordered only if it induces no directed circuit.

To grasp the case where directed circuits occur, we say that vertices $v_1, \ldots, v_n$ are in *pre-topological order* if for all i, j, if v_j is reachable from v_i and $j < i$, then v_i is reachable from v_j. So if D is acyclic, any pre-topological order is topological.

We can interpret a pre-topological order as a linear extension of the partial order $\prec$ defined on V by:

(6.3) $v \prec w \iff w$ is reachable from v, but v is not reachable from w.

Thus a pre-topological ordering is one satisfying: $v_i \prec v_j \Rightarrow i < j$.

The following was shown by Knuth [1968] and Tarjan [1974d] (cf. Kahn [1962]):

Theorem 6.5. *Given a digraph $D = (V, A)$ and $s \in V$, the vertices reachable from s can be ordered pre-topologically in time $O(m')$, where m' is the number of arcs reachable from s.*

Proof. Scan s. Then recursively all vertices reachable from s will be scanned, and the order in which we *finish* scanning them is the opposite of a pre-

topological order: for vertices v, w reachable from s, if D has a $v - w$ path but no $w - v$ path, then scanning w is finished before scanning v. ∎

This implies:

Corollary 6.5a. *Given a digraph $D = (V, A)$, the vertices of D can be ordered pre-topologically in linear time.*

Proof. Add a new vertex s and arcs (s, v) for each $v \in V$. Applying Theorem 6.5 gives the required pre-topological ordering. ∎

For acyclic digraphs, it gives a topological order:

Corollary 6.5b. *Given an acyclic digraph $D = (V, A)$, the vertices of D can be ordered topologically in linear time.*

Proof. Directly from Corollary 6.5a, as a pre-topological order of the vertices of an acyclic graph is topological. ∎

(The existence of a topological order for an acyclic digraph is implicit in the work of Szpilrajn [1930].)

One can also use a pre-topological order to identify the strong components of a digraph in linear time (Karzanov [1970], Tarjan [1972]). We give a method essentially due to S.R. Kosaraju (cf. Aho, Hopcroft, and Ullman [1983]) and Sharir [1981].

Theorem 6.6. *Given a digraph $D = (V, A)$, the strong components of D can be found in linear time.*

Proof. First order the vertices of D pre-topologically as $v_1, \ldots, v_n$. Next let V_1 be the set of vertices reachable to v_1. Then V_1 is the strong component containing v_1: each v_j in V_1 is reachable from v_1, by the definition of pre-topological order.

By Theorem 6.5, the set V_1 can be found in time $O(|A_1|)$, where A_1 is the set of arcs with head in V_1. Delete from D and from $v_1, \ldots, v_n$ all vertices in V_1 and all arcs in A_1, yielding the subgraph D' and the ordered vertices $v'_1, \ldots, v'_{n'}$. This is a pre-topological order for D', for suppose that $i < j$ and that v'_i is reachable from v'_j in D' while v'_j is not reachable in D' from v'_i. Then v'_j is also not reachable in D from v'_i, since otherwise V_1 would be reachable in D from v'_i, and hence $v'_i \in V_1$, a contradiction.

So recursion gives all strong components, in linear time. ∎

As a consequence one has for undirected graphs (Shirey [1969]):

Corollary 6.6a. *Given a graph $G = (V, E)$, the components of G can be found in linear time.*

Proof. Directly from Theorem 6.6. ▮

If we apply depth-first search to a connected undirected graph $G = (V, E)$, starting from a vertex s, then the depth-first search tree T has the property that

(6.4) for each edge $e = uv$ of G, u is on the $s - v$ path in T or v is on the $s - u$ path in T.

So the ends of each edge e of G are connected by a directed path in T.

6.4. Finding an Eulerian orientation

An orientation $D = (V, A)$ of an undirected graph $G = (V, E)$ is called an *Eulerian orientation* if

(6.5) $$\deg_A^{\text{in}}(v) = \deg_A^{\text{out}}(v)$$

for each $v \in V$. As is well-known, an undirected graph $G = (V, E)$ has an Eulerian orientation if and only if each $v \in V$ has even degree in G. (We do not require connectivity.)

An Eulerian orientation can be found in linear time:

Theorem 6.7. *Given an undirected graph $G = (V, E)$ with all degrees even, an Eulerian orientation of G can be found in $O(m)$ time.*

Proof. We assume that we have a list of vertices, and, with each vertex v, a list of edges incident with v.

Consider the first nonisolated vertex in the list, v say. Starting at v, we make a walk such that no edge is traversed more than once. We make this walk as long as possible. Since all degrees are even, we terminate at v.

We orient all edges traversed in the direction as traversed, delete them from G, find the next nonisolated vertex, and iterate the algorithm. We stop if all vertices are isolated. Then all edges are oriented as required. ▮

6.5. Further results and notes

6.5a. All-pairs shortest paths in undirected graphs

Theorem 6.3 directly gives that determining the distances in a digraph $D = (V, A)$ between all pairs of vertices can be done in time $O(nm)$; similarly for undirected graphs.

Seidel [1992,1995] gave the following faster method for dense undirected graphs $G = (V, E)$, assuming without loss of generality that G is connected (by Corollary 6.6a).

For any undirected graph $G = (V, E)$, let $\mathrm{dist}_G(v, w)$ denote the distance between v and w in G (with unit-length edges). Moreover, let G^2 denote the graph with vertex set V, where two vertices v, w are adjacent in G^2 if and only if $\mathrm{dist}_G(v, w) \leq 2$.

Let $M(n)$ denote the time needed to determine the product $A \cdot B$ of two $n \times n$ matrices A and B, each with entries in $\{0, \ldots, n\}$.

It is not difficult to see that the following problem can be solved in time $O(M(n))$:

(6.6) given: an undirected graph $G = (V, E)$;
find: the undirected graph G^2.

Moreover, also the following problem can be solved in time $O(M(n))$:

(6.7) given: an undirected graph $G = (V, E)$ and the function dist_{G^2};
find: the function dist_G.

Lemma 6.8α. *Problem* (6.7) *can be solved in time* $O(M(n))$.

Proof. Let A be the adjacency matrix of G and let T be the $V \times V$ matrix with $T_{v,w} = \mathrm{dist}_{G^2}(v, w)$ for all $v, w \in V$. Note that $\mathrm{dist}_{G^2} = \lceil \mathrm{dist}_G/2 \rceil$. Let $B := T \cdot A$. So for all $u, w \in V$:

$$B_{u,w} = \sum_{v \in N(w)} \mathrm{dist}_{G^2}(u, v). \tag{6.8}$$

Now if $\mathrm{dist}_G(u, w) = 2\lceil \mathrm{dist}_G(u, w)/2 \rceil$, then $\lceil \mathrm{dist}_G(u, v)/2 \rceil \geq \lceil \mathrm{dist}_G(u, w)/2 \rceil$ for each neighbour v of w, and hence $B_{u,w} \geq \deg(v) \lceil \mathrm{dist}_G(u, w)/2 \rceil$. On the other hand, if $\mathrm{dist}_G(u, w) = 2\lceil \mathrm{dist}_G(u, w)/2 \rceil - 1$, then $\lceil \mathrm{dist}_G(u, v)/2 \rceil \leq \lceil \mathrm{dist}_G(u, w)/2 \rceil$ for each neighbour v of w, with strict inequality for at least one neighbour v of w (namely any neighbour of w on a shortest $u - w$ path). So we have $B_{u,w} < \deg(v) \lceil \mathrm{dist}_G(u, w)/2 \rceil$. We conclude that having G, $\mathrm{dist}_{G^2} = \lceil \mathrm{dist}_G/2 \rceil$, and B, we can derive dist_G. As we can calculate B in time $O(M(n))$, we have the lemma. ∎

The all-pairs shortest paths algorithm now is described recursively as follows:

(6.9) If G is a complete graph, the distance between any two distinct vertices is 1. If G is not complete, determine G^2 and from this (recursively) dist_{G^2}. Next determine dist_G.

Theorem 6.8. *Given an undirected graph G on n vertices, the function* dist_G *can be determined in time* $O(M(n) \log n)$. *Here $M(n)$ denotes the time needed to multiply two $n \times n$ matrices each with entries in* $\{0, \ldots, n\}$.

Proof. Determining G^2 from G and determining dist_G from G and dist_{G^2} can be done in time $O(M(n))$. Since the depth of the recursion is $O(\log n)$, the algorithm has running time $O(M(n) \log n)$. ∎

The results on fast matrix multiplication of Coppersmith and Winograd [1987, 1990] give $M(n) = o(n^{2.376})$ (extending earlier work of Strassen [1969]).

Seidel [1992,1995] showed in fact that also shortest paths can be found in this way. More precisely, for all $u, w \in V$ with $u \neq w$, a neighbour v of w can be found such that $\mathrm{dist}_G(u, v) = \mathrm{dist}_G(u, w) - 1$, in time $O(M(n) \log n + n^2 \log^2 n)$. Having this, one can find, for any $u, w \in V$, a shortest $u - w$ path in time $O(\mathrm{dist}_G(u, w))$.

6.5b. Complexity survey

Complexity survey for all-pairs shortest paths with unit lengths (∗ indicates an asymptotically best bound in the table):

	$O(nm)$	Berge [1958b], Moore [1959]
	$O(n^{\frac{3+\omega}{2}} \log^3 n)$	Alon, Galil, and Margalit [1991,1997]
∗	$O(nm \log_n(n^2/m))$	Feder and Motwani [1991,1995]
∗	$O(n^\omega \log n)$	*undirected* Seidel [1992,1995]

Here ω is any real such that any two $n \times n$ matrices can be multiplied by $O(n^\omega)$ arithmetic operations (e.g. $\omega = 2.376$).

Alon, Galil, and Margalit [1991,1997] extended their method to digraphs with arc lengths in $\{-1, 0, 1\}$.

Related work was done by Fredman [1976], Yuval [1976], Romani [1980], Aingworth, Chekuri, and Motwani [1996], Zwick [1998,1999a,2002], Aingworth, Chekuri, Indyk, and Motwani [1999], and Shoshan and Zwick [1999].

6.5c. Ear-decomposition of strongly connected digraphs

Let $D = (V, A)$ be a directed graph. An *ear* of D is a directed path or circuit P in D such that all internal vertices of P have indegree and outdegree equal to 1 in D. The path may consist of a single arc — so any arc of D is an ear. If I is the set of internal vertices of an ear P, we say that D arises from $D - I$ by *adding ear* P. An *ear-decomposition* of D is a series of digraphs $D_0, D_1, \ldots, D_k$, where $D_0 = K_1$, $D_k = D$, and D_i arises from D_{i-1} by adding an ear ($i = 1, \ldots, k$).

Digraphs having an ear-decomposition are be characterized by:

Theorem 6.9. *A digraph $D = (V, A)$ is strongly connected if and only if D has an ear-decomposition.*

Proof. Sufficiency of the condition is easy, since adding an ear to a strongly connected graph maintains strong connectivity.

To see necessity, let $D = (V, A)$ be strongly connected. Let $D' = (V', A')$ be a subgraph of D which has an ear-decomposition and with $|V'| + |A'|$ as large as possible. (Such a subgraph exists, as any single vertex has an ear-decomposition.)

Then $D' = D$, for otherwise there exists an arc $a \in A \setminus A'$ with tail in V'. Then a is contained in a directed circuit C (as D is strongly connected). This circuit C contains a subpath (or circuit) P such that P can be added as an ear to D'. This contradicts the maximality of $|V'| + |A'|$. ∎

A related decomposition of strongly connected digraphs was described by Knuth [1974]. Related work was done by Grötschel [1979].

6.5d. Transitive closure

Complexity survey for finding the transitive closure of a directed graph ($*$ indicates an asymptotically best bound in the table):

	$O(n^3)$	Warshall [1962]
$*$	$O(nm)$	Purdom [1970], cf. Coffy [1973] (also Ebert [1981])
	$O(n^3/\log n)$	Arlazarov, Dinits, Kronrod, and Faradzhev [1970]
	$\widetilde{O}(n^\omega)$	Furman [1970], Munro [1971]
	$O(n^\omega \log n \log\log n \log\log\log n)$	Aho, Hopcroft, and Ullman [1974]
$*$	$O(n^\omega \log n \log\log\log n \log\log\log\log n)$	Adleman, Booth, Preparata, and Ruzzo [1978]

Again, ω is any real such that any two $n \times n$ matrices can be multiplied by $O(n^\omega)$ arithmetic operations (e.g. $\omega = 2.376$). Moreover, $f = \widetilde{O}(g)$ if $f = O(g \log^k g)$ for some k.

For more on finding the transitive closure we refer to Fischer and Meyer [1971], Munro [1971], O'Neil and O'Neil [1973], Dzikiewicz [1975], Sysło and Dzikiewicz [1975], Warren [1975], Eve and Kurki-Suonio [1977], Adleman, Booth, Preparata, and Ruzzo [1978], Schnorr [1978a], Schmitz [1983], Ioannidis and Ramakrishnan [1988], Jakobsson [1991], Ullman and Yannakakis [1991], and Cohen [1994a,1997].

Aho, Garey, and Ullman [1972] showed that finding a minimal directed graph having the same transitive closure as a given directed graph, has the same time complexity as finding the transitive closure.

6.5e. Further notes

For the decomposition of graphs into 3-connected graphs, see Cunningham and Edmonds [1980]. Karzanov [1970] and Tarjan [1974b] gave linear-time algorithms (based on a search method) to find the bridges of an undirected graph.

Theorem 6.1 implies the result of Moore and Shannon [1956] that if $D = (V, A)$ is a digraph, $s, t \in V$, and l is the minimum length of an $s - t$ path and w is the minimum size of an $s - t$ cut, then $|A| \geq lw$ (the *length-width inequality*).

Finding a shortest (directed) circuit in a (directed) graph can be reduced to finding a shortest path. More efficient algorithms were given by Itai and Rodeh [1978].

Barnes and Ruzzo [1991,1997] gave a polynomial-time algorithm to test if there exists an $s - t$ path in an undirected graph, using sublinear space only. This was extended to directed graphs by Barnes, Buss, Ruzzo, and Schieber [1992,1998]. Related work was done by Savitch [1970], Cook and Rackoff [1980], Beame, Borodin, Raghavan, Ruzzo, and Tompa [1990,1996], Nisan [1992,1994], Nisan, Szemeredi, and Wigderson [1992], Broder, Karlin, Raghavan, and Upfal [1994], and Armoni, Ta-Shma, Wigderson, and Zhou [1997,2000].

Karp and Tarjan [1980a,1980b] gave algorithms for finding the connected components of an undirected graph, and the strong components of a directed graph, in $O(n)$ expected time. More on finding strong components can be found in Gabow [2000a].

Books discussing algorithmic problems on paths with unit lengths (reachability, closure, etc.) include Even [1973], Aho, Hopcroft, and Ullman [1974,1983], Christofides [1975], Cormen, Leiserson, and Rivest [1990], Lengauer [1990], Jungnickel [1999], and Mehlhorn and Näher [1999]. Berge [1958b] gave an early survey on shortest paths.

Chapter 7

Shortest paths: nonnegative lengths

In this chapter we consider the shortest path problem in graphs where each arc has a nonnegative length, and describe Dijkstra's algorithm, together with a number of speedups based on heaps.
In this chapter, graphs can be assumed to be simple. If not mentioned explicitly, *length* is taken with respect to a given function l.

7.1. Shortest paths with nonnegative lengths

The methods and results discussed in Chapter 6 for unit-length arcs can be generalized to the case where arcs have a not necessarily unit length. For any 'length' function $l : A \to \mathbb{R}$ and any path $P = (v_0, a_1, v_1, \ldots, a_m, v_m)$, the *length* $l(P)$ of P is defined by:

$$l(P) := \sum_{i=1}^{m} l(a). \tag{7.1}$$

The *distance* from s to t (with respect to l), denoted by $\mathrm{dist}_l(s,t)$, is equal to the minimum length of any $s-t$ path. If no $s-t$ path exists, $\mathrm{dist}_l(s,t)$ is set to $+\infty$.

A weighted version of Theorem 6.1 is as follows, again due to Robacker [1956b] (sometimes called the 'max-potential min-work theorem' (Duffin [1962])):

Theorem 7.1. *Let $D = (V, A)$ be a digraph, let $s, t \in V$, and let $l : A \to \mathbb{Z}_+$. Then the minimum length of an $s-t$ path is equal to the maximum size k of a family of $s-t$ cuts $C_1, \ldots, C_k$ such that each arc a is in at most $l(a)$ of the cuts C_i.*

Proof. Again, the minimum is not smaller than the maximum, since if P is any $s-t$ path and $C_1, \ldots, C_k$ is any collection as described in the theorem, then

$$(7.2)\qquad l(P) = \sum_{a\in AP} l(a) \geq \sum_{a\in AP} (\text{number of } i \text{ with } a \in C_i)$$
$$= \sum_{i=1}^{k} |C_i \cap AP| \geq \sum_{i=1}^{k} 1 = k.$$

To see equality, let d be the distance from s to t and let U_i be the set of vertices at distance less than i from s, for $i = 1, \ldots, d$. Taking $C_i := \delta^{\text{out}}(U_i)$, we obtain a collection $C_1, \ldots, C_d$ as required. ∎

A rooted tree $T = (V', A')$, with root s, is called a *shortest paths tree* for a length function $l : A \to \mathbb{R}_+$, if V' is the set of vertices reachable from s and $A' \subseteq A$ such that for each $t \in V'$, the $s - t$ path in T is a shortest $s - t$ path in D. Again, Dantzig [1957] showed:

Theorem 7.2. *Let $D = (V, A)$ be a digraph, let $s \in V$, and let $l : A \to \mathbb{R}_+$. Then there exists a shortest paths tree for l, with root s.*

Proof. Let V' be the set of vertices reachable from s. Let A' be an inclusionwise minimal set containing for each $t \in V'$ a shortest $s - t$ path of D. Suppose that some vertex v is entered by two arcs in A'. Then at least one of these arcs can be deleted, contradicting the minimality of A'. One similarly sees that no arc in A' enters s. ∎

7.2. Dijkstra's method

Dijkstra [1959] gave an $O(n^2)$ algorithm to find a shortest $s - t$ path for nonnegative length functions — in fact, the output is a shortest paths tree with root s. We describe Dijkstra's method (the idea of this method was also described by Leyzorek, Gray, Johnson, Ladew, Meaker, Petry, and Seitz [1957]).

We keep a subset U of V and a function $d : V \to \mathbb{R}_+$ (the *tentative distance*). Start with $U := V$ and set $d(s) := 0$ and $d(v) = \infty$ if $v \neq s$. Next apply the following iteratively:

(7.3) Find $u \in U$ minimizing $d(u)$ over $u \in U$. For each $a = (u, v) \in A$ for which $d(v) > d(u) + l(a)$, reset $d(v) := d(u) + l(a)$. Reset $U := U \setminus \{u\}$.

We stop if $d(u) = \infty$ for all $u \in U$. The final function d gives the distance from s. Moreover, if we store for each $v \neq s$ the last arc $a = (u, v)$ for which we have reset $d(v) := d(u) + l(a)$, we obtain a shortest path tree with root s.

Clearly, the number of iterations is at most $|V|$, while each iteration takes $O(n)$ time. So the algorithm has running time $O(n^2)$. Thus:

Theorem 7.3. *Given a digraph $D = (V, A)$, $s \in V$, and a length function $l : A \to \mathbb{Q}_+$, a shortest paths tree with root s can be found in time $O(n^2)$.*

Proof. We show correctness of the algorithm. Let $\mathrm{dist}(v)$ denote the distance from s to v, for any vertex v. Trivially, $d(v) \geq \mathrm{dist}(v)$ for all v, throughout the iterations. We prove that throughout the iterations, $d(v) = \mathrm{dist}(v)$ for each $v \in V \setminus U$. At the start of the algorithm this is trivial (as $U = V$).

Consider any iteration (7.3). It suffices to show that $d(u) = \mathrm{dist}(u)$ for the chosen $u \in U$. Suppose $d(u) > \mathrm{dist}(u)$. Let $s = v_0, v_1, \ldots, v_k = u$ be a shortest $s - u$ path. Let i be the smallest index with $v_i \in U$.

Then $d(v_i) = \mathrm{dist}(v_i)$. Indeed, if $i = 0$, then $d(v_i) = d(s) = 0 = \mathrm{dist}(s) = \mathrm{dist}(v_i)$. If $i > 0$, then (as $v_{i-1} \in V \setminus U$):

$$d(v_i) \leq d(v_{i-1}) + l(v_{i-1}, v_i) = \mathrm{dist}(v_{i-1}) + l(v_{i-1}, v_i) = \mathrm{dist}(v_i). \tag{7.4}$$

This implies $d(v_i) \leq \mathrm{dist}(v_i) \leq \mathrm{dist}(u) < d(u)$, contradicting the choice of u. ∎

7.3. Speeding up Dijkstra's algorithm with k-heaps

If $|A|$ is asymptotically smaller than $|V|^2$, one may expect faster methods than $O(n^2)$. Such a method based on 'heaps' (introduced by Williams [1964] and Floyd [1964]), was given by Murchland [1967b] and sharpened by Johnson [1972], Johnson [1973b,1977a] and Tarjan [1983] (see Section 8.6g).

In Dijkstra's algorithm, we spend (in total) $O(m)$ time on updating the values $d(u)$, and $O(n^2)$ time on finding a $u \in U$ minimizing $d(u)$. As $m \leq n^2$, a decrease in the running time bound requires a speedup in finding a u minimizing $d(u)$.

A way of doing this is based on storing U in some order such that a $u \in U$ minimizing $d(u)$ can be found quickly and such that it does not take too much time to restore the order if we delete a u minimizing $d(u)$ or if we decrease some $d(u)$.

This can be done by using 'heaps', two forms of which we consider: k-heaps (in this section) and Fibonacci heaps (in the next section).

A *k-heap* is an ordering $u_0, \ldots, u_n$ of the elements of U such that for all i, j, if $ki < j \leq k(i+1)$, then $d(u_i) \leq d(u_j)$.

This is a convenient way of defining (and displaying) the heap, but it is helpful to imagine the heap as a rooted tree on U: its arcs are the pairs (u_i, u_j) with $ki < j \leq k(i+1)$. So u_i has outdegree k if $k(i+1) \leq n$. The root of this rooted tree is u_0.

If one has a k-heap, one easily finds a u minimizing $d(u)$: it is the root u_0. The following two theorems are basic for estimating the time needed for updating the k-heap if we change U or values of $d(u)$. To *swap* u_i and u_j means exchanging the positions of u_i and u_j in the order (that is, resetting $u_j := u_i$ and $u_i :=$ the old u_j).

Theorem 7.4. *If u_0 is deleted, the k-heap can be restored in time $O(k \log_k n)$.*

Proof. Reset $u_0 := u_n$ and $n := n-1$. Let $i = 0$. While there is a j with $ki < j \leq ki + k$, $j \leq n-1$, and $d(u_j) < d(u_i)$, choose such a j with smallest $d(u_j)$, swap u_i and u_j, and reset $i := j$.

The final k-heap is as required. ∎

The operation described is called *sift-down*. The following theorem describes the operation *sift-up*.

Theorem 7.5. *If $d(u_i)$ is decreased, the k-heap can be restored in time $O(\log_k n)$.*

Proof. While $i > 0$ and $d(u_j) > d(u_i)$ for $j := \lfloor \frac{i-1}{k} \rfloor$, swap u_i and u_j, and reset $i := j$. The final k-heap is as required. ∎

In Dijkstra's algorithm, we delete at most $|V|$ times a u minimizing $d(u)$ and we decrease at most $|A|$ times any $d(u)$. So using a k-heap, the algorithm can be done in time $O(nk \log_k n + m \log_k n)$. This implies:

Theorem 7.6. *Given a digraph $D = (V, A)$, $s \in V$, and a length function $l : A \to \mathbb{Q}_+$, a shortest paths tree with root s can be found in time $O(m \log_k n)$, where $k := \max\{2, \frac{m}{n}\}$.*

Proof. See above. ∎

This implies that if for some class of digraphs $D = (V, A)$ one has $|A| \geq |V|^{1+\varepsilon}$ for some fixed $\varepsilon > 0$, then there is a linear-time shortest path algorithm for these graphs.

7.4. Speeding up Dijkstra's algorithm with Fibonacci heaps

Using a more sophisticated heap, the 'Fibonacci heap', Dijkstra's algorithm can be speeded up to $O(m + n \log n)$, as was shown by Fredman and Tarjan [1984,1987].

A *Fibonacci forest* is a rooted forest (U, A), such that for each $v \in U$ the children of v can be ordered in such a way that the ith child has at least $i-2$ children. (If $(u, v) \in A$, v is called a *child* of u, and u the *parent* of v.)

Lemma 7.7α. *In a Fibonacci forest (U, A), each vertex has outdegree at most $2 \log_2 |U|$.*

Proof. We show:

(7.5) if u has outdegree at least k, then at least $\sqrt{2}^k$ vertices are reachable from u.

This implies the lemma, since $\sqrt{2}^k \leq |U|$ is equivalent to $k \leq 2\log_2 |U|$.

In proving (7.5), we may assume that u is a root. We prove (7.5) by induction on k, the case $k = 0$ being trivial. If $k \geq 1$, let v be the highest ordered child of u. So v has outdegree at least $k-2$. Then by induction, at least $\sqrt{2}^{k-2}$ vertices are reachable from v. Next delete arc (u, v). We keep a Fibonacci forest, in which u has outdegree at least $k-1$. By induction, at least $\sqrt{2}^{k-1}$ vertices are reachable from u in the new forest. Hence at least

$$\sqrt{2}^{k-2} + \sqrt{2}^{k-1} \geq \sqrt{2}^{k} \tag{7.6}$$

vertices are reachable from u in the original forest. ∎

(The recursion (7.6) shows that the Fibonacci numbers give the best bound, justifying the name Fibonacci forest. The weaker bound given, however, is sufficient for our purposes.)

A *Fibonacci heap* consists of a rooted forest $F = (U, A)$ and functions $d : U \to \mathbb{R}$ and $\phi : U \to \{0, 1\}$, such that:

(7.7)
- (i) if $(u, v) \in A$, then $d(u) \leq d(v)$;
- (ii) for each $u \in U$, the children of u can be ordered such that the ith child v satisfies $\deg^{\text{out}}(v) + \phi(v) \geq i - 1$;
- (iii) if u and v are distinct roots of F, then $\deg^{\text{out}}(u) \neq \deg^{\text{out}}(v)$.

Condition (7.7)(ii) implies that F is a Fibonacci forest. So, by Lemma 7.7α, condition (7.7)(iii) implies that F has at most $1 + 2\log_2 |U|$ roots.

The Fibonacci heap will be specified by the following data structure, where $t := \lfloor 2\log_2 |U| \rfloor$:

(7.8)
- (i) for each $u \in U$, a doubly linked list of the children of u (in any order);
- (ii) the function parent $: U \to U$, where parent(u) is the parent of u if it has one, and parent$(u) = u$ otherwise;
- (iii) the functions $\deg^{\text{out}} : U \to \mathbb{Z}_+$, $\phi : U \to \{0, 1\}$, and $d : U \to \mathbb{R}$;
- (iv) a function $b : \{0, \ldots, t\} \to U$ with $b(\deg^{\text{out}}(u)) = u$ for each root u.

Theorem 7.7. *When inserting p times a new vertex, finding and deleting n times a root u minimizing $d(u)$, and decreasing m times the value of $d(u)$, the structure can be restored in time $O(m + p + n \log p)$.*

Proof. Inserting a new vertex v, with value $d(v)$, can be done by setting $\phi(v) := 0$ and by applying:

(7.9) *plant*(v):
Let $r := b(\deg^{\text{out}}(v))$.
If r is a root with $r \neq v$, then:

$$\begin{cases} \text{if } d(r) \le d(v), \text{ add arc } (r,v) \text{ to } A \text{ and plant}(r);\\ \text{if } d(r) > d(v), \text{ add arc } (v,r) \text{ to } A \text{ and plant}(v);\end{cases}$$
else define $b(\deg^{\text{out}}(v)) := v$.

Throughout we update the lists of children and the functions parent, $\deg^{\text{out}}$, ϕ, and b.

A root u minimizing $d(u)$ can be found in time $O(\log p)$, by scanning $d(b(i))$ for $i = 0, \dots, t$ where $b(i)$ is a root.

The root u can be deleted as follows. Let $v_1, \dots, v_k$ be the children of u. First delete u and all arcs leaving u from the forest. This maintains conditions (7.7)(i) and (ii). Next, condition (7.7)(iii) can be restored by applying plant(v) for each $v = v_1, \dots, v_k$.

If we decrease the value of $d(u)$ for some $u \in U$ we do the following:

(7.10) Determine the longest directed path P in F ending at u such that each internal vertex v of P satisfies $\phi(v) = 1$. Reset $\phi(v) := 1 - \phi(v)$ for each $v \in VP \setminus \{u\}$. Delete all arcs of P from A. Apply plant(v) to each $v \in VP$ that is a root of the new forest.

The fact that this maintains (7.7) uses that if the starting vertex q of P is not a root of the original forest, then $q \neq u$ and $\phi(q)$ is reset from 0 to 1 — hence $\deg^{\text{out}}(q) + \phi(q)$ is not changed, and we maintain (7.7)(ii).

We estimate the running time. Throughout all iterations, ϕ increases at most m times (at most once in each application of (7.10)). Hence ϕ decreases at most m times. So the sum of the lengths of the paths P in (7.10) is at most $2m$. So A decreases at most $2m + 2n \log_2 p$ times (since each time we delete a root we delete at most $2 \log_2 p$ arcs). Therefore, A increases at most $2m + 2n \log_2 p + p$ times (since the final $|A|$ is less than p). This gives the running time bound. ∎

This implies for the shortest path problem:

Corollary 7.7a. *Given a digraph $D = (V, A)$, $s \in V$, and a length function $l : A \to \mathbb{Q}_+$, a shortest paths tree with root s can be found in time $O(m + n \log n)$.*

Proof. Directly from Dijkstra's method and Theorem 7.7. ∎

7.5. Further results and notes

7.5a. Weakly polynomial-time algorithms

The above methods all give a strongly polynomial-time algorithm for the shortest path problem, with best running time bound $O(m + n \log n)$. If we allow also the size of the numbers to occur in the running time bound, some other methods are of interest that are in some cases (when the lengths are small integers) faster than the above methods.

In Dijkstra's algorithm, we must select a $u \in U$ with $d(u)$ minimum. It was observed by Dial [1969] that partitioning U into 'buckets' according to the values of d gives a competitive running time bound. The method also gives the following result of Wagner [1976]:

Theorem 7.8. *Given a digraph $D = (V, A)$, $s \in V$, $l : A \to \mathbb{Z}_+$, and an upper bound Δ on $\max\{\mathrm{dist}_l(s, v) \mid v$ reachable from $s\}$, a shortest path tree rooted at s can be found in time $O(m + \Delta)$.*

Proof. Apply Dijkstra's algorithm as follows. Next to the function $d : U \to \mathbb{Z}_+ \cup \{\infty\}$, we keep doubly linked lists $L_0, \ldots, L_\Delta$ such that if $d(u) \leq \Delta$, then u is in $L_{d(u)}$. We keep also, for each $i = 0, \ldots, \Delta$, the first element u_i of L_i. If L_i is empty, then u_i is void. Moreover, we keep a 'current minimum' $\mu \in \{0, \ldots, \Delta\}$.

The initialization follows directly from the initialization of d: we set $L_0 := \{s\}$, $u_0 := s$, while L_i is empty and u_i void for $i = 1, \ldots, \Delta$. Initially, $\mu := 0$.

The iteration is as follows. If $L_\mu = \emptyset$ and $\mu \leq \Delta$, reset $\mu := \mu + 1$. If $L_\mu \neq \emptyset$, apply Dijkstra's iteration to u_μ: We remove u_μ from L_μ. When decreasing some $d(u)$ from d to d', we delete u from L_d (if $d \leq \Delta$) and insert it into $L_{d'}$ (if $d' \leq \Delta$).

We stop if $\mu = \Delta + 1$. With each removal or insertion, we can update the lists and the u_i in constant time. Hence we have the required running time. ∎

A consequence is the bound of Dial [1969]:

Corollary 7.8a. *Given a digraph $D = (V, A)$, $s \in V$, and $l : A \to \mathbb{Z}_+$, a shortest path tree rooted at s can be found in time $O(m+nL)$ where $L := \max\{l(a) \mid a \in A\}$.*

Proof. We can take $\Delta = nL$ in Theorem 7.8. ∎

One can derive from Theorem 7.8 also the following result of Gabow [1985b]:

Theorem 7.9. *Given a digraph $D = (V, A)$, $s \in V$, and a length function $l : A \to \mathbb{Z}_+$, a shortest paths tree rooted at s can be found in time $O(m \log_d L)$, where $d = \max\{2, m/n\}$, and $L := \max\{l(a) \mid a \in A\}$.*

Proof. For each $a \in A$, let $l'(a) := \lfloor l(a)/d \rfloor$. Recursively we find $\mathrm{dist}_{l'}(s, v)$ for all $v \in V$, in time $O(m \log_d L')$ where $L' := \lfloor L/d \rfloor$. Note that $\log_d L' \leq (\log_d L) - 1$. Now set

$$\tilde{l}(a) := l(a) + d \cdot \mathrm{dist}_{l'}(s, u) - d \cdot \mathrm{dist}_{l'}(s, v) \tag{7.11}$$

for each $a = (u, v) \in A$. Then $\tilde{l}(a) \geq 0$, since

$$l(a) \geq d \cdot l'(a) \geq d(\mathrm{dist}_{l'}(s, v) - \mathrm{dist}_{l'}(s, u)). \tag{7.12}$$

Moreover, $\mathrm{dist}_{\tilde{l}}(s, v) \leq nd$ for each v reachable from s, since if P is an $s - v$ path with $l'(P) = \mathrm{dist}_{l'}(s, v)$, then $\tilde{l}(P) = l(P) - dl'(P) \leq nd$. So by Theorem 7.8 we can find $\mathrm{dist}_{\tilde{l}}(s, v)$ for all $v \in V$ in time $O(m)$, since $nd \leq 2m$. As $\mathrm{dist}_l(s, v) = \mathrm{dist}_{\tilde{l}}(s, v) - d \cdot \mathrm{dist}_{l'}(s, v)$, we find the required data. ∎

(This improves a result of Hansen [1980a].)

7.5b. Complexity survey for shortest paths with nonnegative lengths

The following gives a survey of the development of the running time bound for the shortest path problem for a digraph $D = (V, A)$, $s, t \in V$, and nonnegative length-function l, where $n := |V|$, $m := |A|$, and $L := \max\{l(a) \mid a \in A\}$ (assuming l integer). As before, $*$ indicates an asymptotically best bound in the table.

	$O(n^4)$	Shimbel [1955]
	$O(n^2mL)$	Ford [1956]
	$O(nm)$	Bellman [1958], Moore [1959]
	$O(n^2 \log n)$	Dantzig [1958,1960], Minty (cf. Pollack and Wiebenson [1960]), Whiting and Hillier [1960]
	$O(n^2)$	Dijkstra [1959]
	$O(m + nL)$	Dial [1969] (cf. Wagner [1976], Filler [1976](=E.A. Dinits))
	$O(m \log(2 + (n^2/m)))$	Johnson [1972]
	$O(dn \log_d n + m + m \log_d(n^2/m))$	(for any $d \geq 2$) Johnson [1973b]
	$O(m \log_{m/n} n)$	Johnson [1973b,1977a], Tarjan [1983]
	$O(L + m \log\log L)$	van Emde Boas, Kaas, and Zijlstra [1977]
	$O(m \log\log L + n \log L \log\log L)$	Johnson [1977b]
	$O(\min_{k\geq 2}(nkL^{1/k} + m \log k))$	Denardo and Fox [1979]
	$O(m \log L)$	Hansen [1980a]
$*$	$O(m \log\log L)$	Johnson [1982], Karlsson and Poblete [1983]
$*$	$O(m + n \log n)$	Fredman and Tarjan [1984,1987][2]
	$O(m \log_{m/n} L)$	Gabow [1983b,1985b]
$*$	$O(m + n\sqrt{\log L})$	Ahuja, Mehlhorn, Orlin, and Tarjan [1990]

Fredman and Willard [1990,1994] gave an $O(m + n\frac{\log n}{\log\log n})$-time algorithm for shortest paths with nonnegative lengths, utilizing nonstandard capabilities of a RAM like addressing. This was extended to $O(m + n\sqrt{\log n \log\log n})$ by Raman [1996], to $O(m \log\log n)$ and $O(m \log\log L)$ by Thorup [1996,2000b] (cf. Hagerup [2000]), and to $O(m+n(\log L \log\log L)^{1/3})$ by Raman [1997]. For undirected graphs,

[2] Aho, Hopcroft, and Ullman [1983] (p. 208) claimed to give an $O(m + n \log n)$-time shortest path algorithm based on 2-heaps, but they assume that, after resetting a value, the heap can be restored in constant time.

a bound of $O(m)$ was achieved by Thorup [1997,1999,2000a]. Related results were given by Pettie and Ramachandran [2002b].

The expected complexity of Dijkstra's algorithm is investigated by Noshita, Masuda, and Machida [1978], Noshita [1985], Cherkassky, Goldberg, and Silverstein [1997,1999], Goldberg [2001a,2001b], and Meyer [2001].

In the special case of *planar* directed graphs:

	$O(n\sqrt{\log n})$	Frederickson [1983b,1987b]
*	$O(n)$	Klein, Rao, Rauch, and Subramanian [1994], Henzinger, Klein, Rao, and Subramanian [1997]

For the all-pairs shortest paths problem with nonnegative lengths one has:

	$O(n^4)$	Shimbel [1955]
	$O(n^3 \log n)$	Leyzorek, Gray, Johnson, Ladew, Meaker, Petry, and Seitz [1957]
	$O(n^2 m)$	Bellman [1958], Moore [1959]
	$O(n^3 \log n)$	Dantzig [1958,1960], Minty (cf. Pollack and Wiebenson [1960]), Whiting and Hillier [1960]
	$O(n^3)$	Dijkstra [1959]
	$O(nm + n^2 L)$	Dial [1969] (cf. Wagner [1976])
	$O(nm \log n)$	Johnson [1972]
	$O(n^3(\log\log n/\log n)^{1/3})$	Fredman [1976]
	$O(nm \log_{m/n} n)$	Johnson [1973b,1977a], Tarjan [1983]
	$O(nL + nm \log\log L)$	van Emde Boas, Kaas, and Zijlstra [1977]
	$O(nm \log\log L + n^2 \log L \log\log L)$	Johnson [1977b]
	$O(nm \log L)$	Hansen [1980a]
*	$O(nm \log\log L)$	Johnson [1982], Karlsson and Poblete [1983]
*	$O(n(m + n\log n))$	Fredman and Tarjan [1984,1987]
	$O(nm \log_{m/n} L)$	Gabow [1983b,1985b]
*	$O(nm + n^2\sqrt{\log L})$	Ahuja, Mehlhorn, Orlin, and Tarjan [1990]
*	$O(n^3(\log\log n/\log n)^{1/2})$	Takaoka [1992a,1992b]
*	$\widetilde{O}(n^{\frac{5\omega-3}{\omega+1}} L + n^{\frac{3+\omega}{2}} L^{\frac{\omega-1}{2}})$	Galil and Margalit [1997a,1997b]
*	$\widetilde{O}(n^{\omega} L^{\frac{\omega+1}{2}})$	*undirected* Galil and Margalit [1997a,1997b]

Here ω is any real such that any two $n \times n$ matrices can be multiplied by $O(n^\omega)$ arithmetic operations (e.g. $\omega = 2.376$). Moreover, $f = \widetilde{O}(g)$ if $f = O(g \log^k g)$ for some k. (At the negative side, Kerr [1970] showed that matrix multiplication of $n \times n$ matrices with addition and multiplication replaced by minimization and addition, requires time $\Omega(n^3)$.)

Spira [1973] gave an $O(n^2 \log^2 n)$ expected time algorithm for all-pairs shortest paths with nonnegative lengths. This was improved to $O(n^2 \log n \log\log n)$ by Takaoka and Moffat [1980], to $O(n^2 \log n \log^* n)$ by Bloniarz [1980,1983] (defining $\log^* n := \min\{i \mid \log^{(i)} n \leq 1\}$, where $\log^{(0)} n := n$ and $\log^{(i+1)} n := \log(\log^{(i)} n)$), and to $O(n^2 \log n)$ by Moffat and Takaoka [1985,1987]. Related work includes Carson and Law [1977], Frieze and Grimmett [1985], Hassin and Zemel [1985], Walley and Tan [1995], and Mehlhorn and Priebe [1997].

Yen [1972] (cf. Williams and White [1973]) described an all-pairs shortest paths method (based on Dijkstra's method) using $\frac{1}{2}n^3$ additions and n^3 comparisons. Nakamori [1972] gave a lower bound on the number of operations. Yao, Avis, and Rivest [1977] gave a lower bound of $\Omega(n^2 \log n)$ for the time needed for the all-pairs shortest paths problem.

Karger, Koller, and Phillips [1991,1993] and McGeoch [1995] gave an $O(n(m^* + n \log n))$ algorithm for all-pairs shortest paths, where m^* is the number of arcs that belong to at least one shortest path. See also Yuval [1976] and Romani [1980] for relations between all-pairs shortest paths and matrix multiplication.

Frederickson [1983b,1987b] showed that in a *planar* directed graph, with nonnegative lengths, the all-pairs shortest paths problem can be solved in $O(n^2)$ time.

7.5c. Further notes

Spira and Pan [1973,1975], Shier and Witzgall [1980], and Tarjan [1982] studied the sensitivity of shortest paths trees under modifying arc lengths. Fulkerson and Harding [1977] studied the problem of lengthening the arc lengths within a given budget (where each arc has a given cost for lengthening the arc length) so as to maximize the distance from a given source to a given sink. They reduced this problem to a parametric minimum-cost flow problem. Land and Stairs [1967] and Hu [1968] studied decomposition methods for finding all-pairs shortest paths (cf. Farbey, Land, and Murchland [1967], Hu and Torres [1969], Yen [1971b], Shier [1973], and Blewett and Hu [1977]).

Frederickson [1989,1995] gave a strongly polynomial-time algorithm to find an $O(n)$ encoding of shortest paths between all pairs in a directed graph with nonnegative length function. (It extends earlier work of Frederickson [1991] for planar graphs.)

Algorithms for finding the k shortest paths between pairs of vertices in a directed graph were given by Clarke, Krikorian, and Rausen [1963], Yen [1971a], Minieka [1974], Weigand [1976], Lawler [1977], Shier [1979], Katoh, Ibaraki, and Mine [1982], Byers and Waterman [1984], Perko [1986], Chen [1994], and Eppstein [1994b,1999].

Mondou, Crainic, and Nguyen [1991] gave a survey of shortest paths methods, with computational results, and Raman [1997] on 'recent' results on shortest paths with nonnegative lengths.

Books covering shortest path methods for nonnegative lengths include Berge [1973b], Aho, Hopcroft, and Ullman [1974,1983], Christofides [1975], Lawler [1976b], Minieka [1978], Even [1979], Hu [1982], Papadimitriou and Steiglitz [1982], Smith [1982], Sysło, Deo, and Kowalik [1983], Tarjan [1983], Gondran and Minoux [1984], Rockafellar [1984], Nemhauser and Wolsey [1988], Bazaraa, Jarvis, and Sherali [1990], Chen [1990], Cormen, Leiserson, and Rivest [1990], Lengauer [1990], Ahuja, Magnanti, and Orlin [1993], Cook, Cunningham, Pulleyblank, and Schrijver [1998], Jungnickel [1999], Mehlhorn and Näher [1999], and Korte and Vygen [2000].

Chapter 8

Shortest paths: arbitrary lengths

We now go over to the shortest path problem for the case where negative lengths are allowed, but where each directed circuit has nonnegative length (with no restriction, the problem is NP-complete). The basic algorithm here is the Bellman-Ford method.
In this chapter, graphs can be assumed to be simple. If not mentioned explicitly, *length* is taken with respect to a given function l.

8.1. Shortest paths with arbitrary lengths but no negative circuits

If lengths of arcs may take negative values, finding a shortest $s-t$ path is NP-complete — see Theorem 8.11 below. Negative-length directed circuits seem to be the source of the trouble: if no negative-length directed circuits exist, there is a polynomial-time algorithm — mainly due to the fact that running into loops cannot give shortcuts. So a shortest *walk* (nonsimple path) exists and yields a shortest *path*.

We first observe that if no negative-length directed circuits exists, then the existence of a shortest paths tree is easy:

Theorem 8.1. *Let $D=(V,A)$ be a digraph, let $s\in V$, and let $l:A\to\mathbb{R}$ be such that each directed circuit reachable from s has nonnegative length. Then there exists a shortest paths tree with root s.*

Proof. As the proof of Theorem 7.2. ∎

8.2. Potentials

The following observation of Gallai [1958b] is very useful. Let $D=(V,A)$ be a digraph and let $l:A\to\mathbb{R}$. A function $p:V\to\mathbb{R}$ is called a *potential* if for each arc $a=(u,v)$:

$$l(a)\geq p(v)-p(u). \tag{8.1}$$

Theorem 8.2. *Let $D = (V, A)$ be a digraph and let $l : A \to \mathbb{R}$ be a length function. Then there exists a potential if and only if each directed circuit has nonnegative length. If moreover l is integer, the potential can be taken integer.*

Proof. *Suffiency.* Suppose that a function p as described exists. Let $C = (v_0, a_1, v_1, \ldots, a_m, v_m)$ be a directed circuit ($v_m = v_0$). Then

$$l(C) = \sum_{i=1}^{m} l(a_i) \geq \sum_{i=1}^{m} (p(v_i) - p(v_{i-1})) = 0. \tag{8.2}$$

Necessity. Suppose that each directed circuit has nonnegative length. For each $t \in V$, let $p(t)$ be the minimum length of any path ending at t (starting wherever). This function satisfies the required condition. ▮

Theorem 8.2 gives a good characterization for the problem of deciding if there exists a negative-length directed circuit.

A potential is useful in transforming a length function to a nonnegative length function: if we define $\tilde{l}(a) := l(a) - p(v) + p(u)$ for each arc $a = (u, v)$, then we obtain a nonnegative length function $\tilde{l}$ such that each $s - t$ path is shortest with respect to l if and only if it is shortest with respect to $\tilde{l}$. So once we have a potential p, we can find shortest paths with Dijkstra's algorithm. This can be used for instance in finding shortest paths between all pairs of vertices — see Section 8.4.

One can also formulate a min-max relation in terms of functions that are potentials on an appropriate subgraph. This result is sometimes called the 'max-potential min-work theorem' (Duffin [1962]).

Theorem 8.3. *Let $D = (V, A)$ be a digraph, let $s, t \in V$ and let $l : A \to \mathbb{R}$. Then $\mathrm{dist}_l(s, t)$ is equal to the maximum value of $p(t) - p(s)$, where $p : V \to \mathbb{R}$ is such that $l(a) \geq p(v) - p(u)$ for each arc $a = (u, v)$ traversed by at least one $s - t$ walk. If l is integer, we can restrict p to be integer.*

Proof. Let $p(v) := \mathrm{dist}_l(s, v)$ if v belongs to at least one $s - t$ walk, and $p(v) := 0$ otherwise. This p is as required. ▮

The following observation can also be of use, for instance when calculating shortest paths by linear programming:

Theorem 8.4. *Let $D = (V, A)$ be a digraph, let $s \in V$ be such that each vertex of D is reachable from s, and let $l : A \to \mathbb{R}$ be such that each directed circuit has nonnegative length. Let p be a potential with $p(s) = 0$ and $\sum_{v \in V} p(v)$ maximal. Then $p(t) = \mathrm{dist}_l(s, t)$ for each $t \in V$.*

Proof. One easily shows that for any potential p with $p(s) = 0$ one has $p(t) \leq \text{dist}_l(s,t)$ for each $t \in V$. As $\text{dist}_l(s,\cdot)$ is a potential, the theorem follows. ∎

8.3. The Bellman-Ford method

Also in the case of a length function without negative-length directed circuit, there is a polynomial-time shortest path algorithm, the *Bellman-Ford method* (Shimbel [1955], Ford [1956], Bellman [1958], Moore [1959]). Again, it finds a shortest paths tree for any root s.

To describe the method, define for $t \in V$ and $k \geq 0$:

(8.3) $d_k(t) :=$ minimum length of any $s-t$ walk traversing at most k arcs,

setting $d_k(t) := \infty$ if no such walk exists.

Clearly, if there is no negative-length directed circuit reachable from s, the distance from s to t is equal to $d_n(v)$, where $n := |V|$.

Algorithmically, the function d_0 is easy to set: $d_0(s) = 0$ and $d_0(t) = \infty$ if $t \neq s$. Next $d_1, d_2, \ldots$ can be successively computed by the following rule:

$$d_{k+1}(t) = \min\{d_k(t), \min_{(u,t)\in A}(d_k(u) + l(u,t))\} \tag{8.4}$$

for all $t \in V$.

This method gives us the distance from s to t. It is not difficult to derive a method finding a shortest paths tree with root s. Thus:

Theorem 8.5. *Given a digraph $D = (V,A)$, $s \in V$, and a length function $l : A \to \mathbb{Q}$ such that each directed circuit reachable from s has nonnegative length, a shortest paths tree rooted at s can be found in time $O(nm)$.*

Proof. There are at most n iterations, each of which can be performed in time $O(m)$. ∎

A negative-length directed circuit can be detected similarly:

Theorem 8.6. *Given a digraph $D = (V,A)$, $s \in V$, and a length function $l : A \to \mathbb{Q}$, a directed circuit of negative length reachable from s (if any exists) can be found in time $O(nm)$.*

Proof. If $d_n \neq d_{n-1}$, then $d_n(t) < d_{n-1}(t)$ for some $t \in V$. So the algorithm finds an $s-t$ walk P of length $d_n(t)$, traversing n arcs. As P traverses n arcs, it contains a directed circuit C. Removing C gives an $s-t$ walk P' with less than n arcs. So $l(P') \geq d_{n-1}(t) > d_n(t) = l(P)$ and hence $l(C) < 0$.

If $d_n = d_{n-1}$, then there is no negative-length directed circuit reachable from s. ∎

Also a potential can be found with the Bellman-Ford method:

Theorem 8.7. *Given a digraph $D = (V, A)$ and a length function $l : A \to \mathbb{Q}$ such that each directed circuit has nonnegative length, a potential can be found in time $O(nm)$.*

Proof. Extend D by a new vertex s and arcs (s, v) for $v \in V$, each of length 0. Then setting $p(v)$ equal to the distance from s to v (which can be determined with the Bellman-Ford method) gives a potential. ∎

We remark that the shortest path problem for *undirected* graphs, for length functions without negative-length circuits, also can be solved in polynomial time. However, the obvious reduction — replacing every undirected edge uv by two arcs (u, v) and (v, u) each of length $l(uv)$ — may yield a negative-length directed circuit. So in this case, the undirected case does not reduce to the directed case, and we cannot apply the Bellman-Ford method. The undirected problem can yet be solved in polynomial time, with the methods developed for the matching problem — see Section 29.2.

8.4. All-pairs shortest paths

Let $D = (V, A)$ be a digraph and $l : A \to \mathbb{Q}$ be a length function such that each directed circuit has nonnegative length. By applying $|V|$ times the Bellman-Ford method one can find shortest $s-t$ paths for all $s, t \in V$. As the Bellman-Ford method takes time $O(nm)$, this makes an $O(n^2 m)$ algorithm.

A more efficient algorithm, the *Floyd-Warshall method* was described by Floyd [1962b], based on an idea of Warshall [1962], earlier found by Kleene [1956], Roy [1959], and McNaughton and Yamada [1960]: Order the vertices of D (arbitrarily) as $v_1, \ldots, v_n$. Define for $s, t \in V$ and $k \in \{0, \ldots, n\}$:

(8.5) $d_k(s, t) :=$ minimum length of an $s - t$ walk using only vertices in $\{s, t, v_1, \ldots, v_k\}$.

Clearly, $d_0(s, t) = l(s, t)$ if $(s, t) \in A$, while $d_0(s, t) = \infty$ otherwise. Moreover:

(8.6) $$d_{k+1}(s, t) = \min\{d_k(s, t), d_k(s, v_{k+1}) + d_k(v_{k+1}, t)\}$$

for all $s, t \in V$ and $k < n$.

This gives:

Theorem 8.8. *Given a digraph $D = (V, A)$ and a length function $l : A \to \mathbb{Q}$ with no negative-length directed circuit, all distances $\mathrm{dist}_l(s, t)$ can be determined in time $O(n^3)$.*

Proof. Note that $\mathrm{dist}_l = d_n$ and that d_n can be determined in n iterations, each taking $O(n^2)$ time. ∎

The Floyd-Warshall method can be adapted so as to find for all $s \in V$, a shortest $s-t$ paths tree rooted at s.

A faster method was observed by Johnson [1973b,1977a] and Bazaraa and Langley [1974]. Combined with the Fibonacci heap implementation of Dijkstra's algorithm, it gives all-pairs shortest paths in time $O(n(m + n \log n))$, which is, if $n \log n = O(m)$, of the same order as the Bellman-Ford for *single*-source shortest path. The idea is to preprocess the data by a potential function, so as to make the length function nonnegative, and next to apply Dijkstra's method:

Theorem 8.9. *Given a digraph $D = (V, A)$ and a length function $l : A \to \mathbb{Q}$ with no negative-length directed circuit, a family $(T_s \mid s \in V)$ of shortest paths trees T_s rooted at s can be found in time $O(n(m + n \log n))$.*

Proof. With the Bellman-Ford method one finds a potential p in time $O(nm)$ (Theorem 8.7). Set $\tilde{l}(a) := l(a) - p(v) + p(u)$ for each arc $a = (u, v)$. So $\tilde{l}(a) \geq 0$ for each arc a. Next with Dijkstra's method, using Fibonacci heaps, one can determine for each $s \in V$ a shortest paths tree T_s for $\tilde{l}$, in time $O(m + n \log n)$ (Corollary 7.7a). As these are shortest paths trees also for l, we have the current theorem. ∎

8.5. Finding a minimum-mean length directed circuit

Let $D = (V, A)$ be a directed graph (with n vertices) and let $l : A \to \mathbb{R}$. The *mean length* of a directed cycle (directed closed walk) $C = (v_0, a_1, v_1, \ldots, a_t, v_t)$ with $v_t = v_0$ and $t > 0$ is $l(C)/t$. Karp [1978] gave the following polynomial-time method for finding a directed cycle of minimum mean length. For each $v \in V$ and each $k = 0, 1, 2, \ldots$, let $d_k(v)$ be the minimum length of a walk with exactly k arcs, ending at v. So for each v one has

$$d_0(v) = 0 \text{ and } d_{k+1}(v) = \min\{d_k(u) + l(a) \mid a = (u, v) \in \delta^{\text{in}}(v)\}. \tag{8.7}$$

Now Karp [1978] showed:

Theorem 8.10. *The minimum mean length of a directed cycle in D is equal to*

$$\min_{v \in V} \max_{0 \leq k \leq n-1} \frac{d_n(v) - d_k(v)}{n - k}. \tag{8.8}$$

Proof. We may assume that the minimum mean length is 0, since adding ε to the length of each arc increases both minima in the theorem by ε. So we must show that (8.8) equals 0.

First, let minimum (8.8) be attained by v. Let P_n be a walk with n arcs ending at v, of length $d_n(v)$. So P_n can be decomposed into a path P_k, say, with k arcs ending at v, and a directed cycle C with $n - k$ arcs (for

some $k < n$). Hence $d_n(v) = l(P_n) = l(P_k) + l(C) \geq l(P_k) \geq d_k(v)$ and so $d_n(v) - d_k(v) \geq 0$. Therefore, (8.8) is nonnegative.

To see that it is 0, let $C = (v_0, a_1, v_1, \ldots, a_t, v_t)$ be a directed cycle of length 0. Then $\min_r d_r(v_0)$ is attained by some r with $n - t \leq r < n$ (as it is attained by some $r < n$ (since each circuit has nonnegative length), and as we can add C to the shortest walk ending at v_0). Fix this r.

Let $v := v_{n-r}$, and split C into walks

(8.9) $$P := (v_0, a_1, v_1, \ldots, a_{n-r}, v_{n-r}) \text{ and}$$
$$Q := (v_{n-r}, a_{n-r+1}, v_{n-r+1}, \ldots, a_t, v_t).$$

Then $d_n(v) \leq d_r(v_0) + l(P)$, and therefore for each k:

(8.10) $$d_k(v) + l(Q) \geq d_{k+(t-(n-r))}(v_0) \geq d_r(v_0) \geq d_n(v) - l(P).$$

This implies $d_n(v) - d_k(v) \leq l(C) = 0$. So the minimum (8.8) is at most 0. ∎

Algorithmically, it gives:

Corollary 8.10a. *A minimum-mean length directed circuit can be found in time $O(nm)$.*

Proof. See the method above. ∎

Notes. Karp and Orlin [1981] and Karzanov [1985c] gave generalizations. Orlin and Ahuja [1992] gave an $O(\sqrt{n}\, m \log(nL))$ algorithm for the minimum-mean length directed circuit problem (cf. McCormick [1993]). Early work on this problem includes Lawler [1967], Shapiro [1968], and Fox [1969].

8.6. Further results and notes

8.6a. Complexity survey for shortest path without negative-length circuits

The following gives a survey of the development of the running time bound for the shortest path problem for a digraph $D = (V, A)$, $s, t \in V$, and $l : A \to \mathbb{Z}$ (without negative-length directed circuits), where $n := |V|$, $m := |A|$, $L := \max\{|l(a)| \mid a \in A\}$, and $L' := \max\{-l(a) \mid a \in A\}$ (assuming $L' \geq 2$). As before, $*$ indicates an asymptotically best bound in the table.

	$O(n^4)$	Shimbel [1955]
	$O(n^2 mL)$	Ford [1956]
$*$	$O(nm)$	Bellman [1958], Moore [1959]
	$O(n^{3/4} m \log L')$	Gabow [1983b,1985b]

≫

continued

	$O(\sqrt{n}\, m \log(nL'))$	Gabow and Tarjan [1988b,1989]
*	$O(\sqrt{n}\, m \log L')$	Goldberg [1993b,1995]

(Gabow [1983b,1985b] and Gabow and Tarjan [1988b,1989] give bounds with L instead of L', but Goldberg [1995] mentioned that an anonymous referee of his paper observed that L can be replaced by L'.)

Kolliopoulos and Stein [1996,1998b] proved a bound of $o(n^3)$ for the average-case complexity.

For the special case of *planar* directed graphs:

	$O(n^{3/2})$	Lipton, Rose, and Tarjan [1979]
	$O(n^{4/3} \log(nL'))$	Klein, Rao, Rauch, and Subramanian [1994], Henzinger, Klein, Rao, and Subramanian [1997]
*	$O(n \log^3 n)$	Fakcharoenphol and Rao [2001]

For the all-pairs shortest paths problem, with no negative-length directed circuits, one has:

	$O(n^4)$	Shimbel [1955]
	$O(n^3 \log n)$	Leyzorek, Gray, Johnson, Ladew, Meaker, Petry, and Seitz [1957]
	$O(n^2 m)$	Bellman [1958], Moore [1959]
	$O(n^3)$	Floyd [1962b]
*	$O(n \cdot \mathrm{SP}_+(n, m, L))$	Johnson [1973b,1977a], Bazaraa and Langley [1974]
	$O(nm + n^3(\log\log n/\log n)^{1/3})$	Fredman [1976]
	$O(nm \log\log L + n^2 \log L \log\log L)$	Johnson [1977b]
	$O(nL + nm \log\log L)$	van Emde Boas, Kaas, and Zijlstra [1977]
*	$O(nm \log\log L)$	Johnson [1982]
	$O(nm \log_{m/n} L)$	Gabow [1985b]
*	$O(nm + n^2\sqrt{\log L})$	Ahuja, Mehlhorn, Orlin, and Tarjan [1990]
*	$O((nL)^{\frac{3+\omega}{2}} \log^3 n)$	Alon, Galil, and Margalit [1991, 1997], Galil and Margalit [1997a, 1997b]

Here $\mathrm{SP}_+(n, m, L)$ denotes the time needed to find a shortest path in a digraph with n vertices and m arcs, with *nonnegative* integer lengths on the arcs, each at most L. ω is any real such that any two $n \times n$ matrices can be multiplied by $O(n^\omega)$ arithmetic operations (e.g. $\omega = 2.376$).

Frederickson [1983b,1987b] showed that for *planar* directed graphs, the all-pairs shortest paths problem, with no negative-length directed circuits, can be solved in $O(n^2)$ time.

8.6b. NP-completeness of the shortest path problem

In full generality — that is, not requiring that each directed circuit has nonnegative length — the shortest path problem is NP-complete, even if each arc has length -1. Equivalently, finding a longest path in a graph (with unit length arcs) is NP-complete. This is a result of E.L. Lawler and R.E. Tarjan (cf. Karp [1972b]).

This directly follows from the NP-completeness of finding a Hamiltonian path in a graph. Let $D = (V, A)$ be a digraph. (A directed path P is called *Hamiltonian* if each vertex of D is traversed exactly once.)

We show the NP-completeness of the *directed Hamiltonian path problem*: Given a digraph $D = (V, A)$ and $s, t \in V$, is there a Hamiltonian $s - t$ path?

Theorem 8.11. *The directed Hamiltonian path problem is* NP*-complete.*

Proof. We give a polynomial-time reduction of the partition problem (Section 4.11) to the directed Hamiltonian path problem. Let $\mathcal{C} = \{C_1, \ldots, C_m\}$ be a collection of subsets of the set $X = \{1, \ldots, k\}$. Introduce vertices $r_0, r_1, \ldots, r_m, 0, 1, \ldots, k$.

For each $i = 1, \ldots, m$, we do the following. Let $C_i = \{j_1, \ldots, j_t\}$. We construct a digraph on the vertices r_{i-1}, r_i, $j_h - 1, j_h$ (for $h = 1, \ldots, t$) and $3t$ new vertices, as in Figure 8.1. Moreover, we make an arc from r_m to 0.

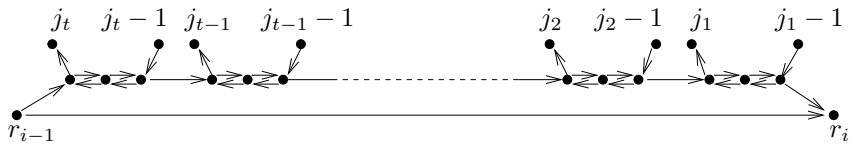


Figure 8.1

Let D be the digraph arising in this way. Then it is not difficult to check that there exists a subcollection $\mathcal{C}'$ of $\mathcal{C}$ that partitions X if and only if D has a directed Hamiltonian $r_0 - k$ path P. (Take: $(r_{i-1}, r_i) \in P \iff C_i \in \mathcal{C}'$.) ∎

Hence:

Corollary 8.11a. *Given a digraph $D = (V, A)$ and $s, t \in V$, finding a longest $s - t$ path is* NP*-complete.*

Proof. This follows from the fact that there exists an $s - t$ path of length $|V| - 1$ if and only if there is a directed Hamiltonian $s - t$ path. ∎

From this we derive the NP-completeness of the *undirected Hamiltonian path problem*: Given a graph $G = (V, E)$ and $s, t \in V$, does G have a Hamiltonian $s - t$ path? (R.E. Tarjan (cf. Karp [1972b])).

Corollary 8.11b. *The undirected Hamiltonian path problem is* NP*-complete.*

Proof. We give a polynomial-time reduction of the directed Hamiltonian path problem to the undirected Hamiltonian path problem. Let D be a digraph. Replace each vertex v by three vertices v', v'', v''', and make edges $\{v', v''\}$ and $\{v'', v'''\}$. Moreover, for each arc (v_1, v_2) of D, make an edge $\{v_1''', v_2'\}$. Delete the vertices s', s'', t', t'''. This makes the undirected graph G. One easily checks that D has a directed Hamiltonian $s-t$ path if and only if G has an (undirected) Hamiltonian $s'''-t'$ path. ∎

Again it implies:

Corollary 8.11c. *Given an undirected graph $G=(V,E)$ and $s,t \in V$, finding a longest $s-t$ path is* NP*-complete.*

Proof. This follows from the fact that there exists an $s-t$ path of length $|V|-1$ if and only if there is a Hamiltonian $s-t$ path. ∎

Notes. Corollary 8.11b implies that finding a Hamiltonian circuit in an undirected graph is NP-complete: just add a new vertex r and edges rs and rt. This reduces finding a Hamiltonian $s-t$ path in the original graph to finding a Hamiltonian circuit in the extended graph.

Also the directed Hamiltonian circuit problem is NP-complete, as the undirected version can be reduced to it by replacing each edge uv by two oppositely oriented arcs (u,v) and (v,u).

8.6c. Nonpolynomiality of Ford's method

The method originally described by Ford [1956] consists of the following. Given a digraph $D=(V,A)$, $s,t \in V$, and a length function $l: A \to \mathbb{Q}$, define $d(s) := 0$ and $d(v) := \infty$ for all $v \neq s$; next perform the following iteratively:

(8.11) choose an arc (u,v) with $d(v) > d(u) + l(u,v)$, and reset $d(v) := d(u) + l(u,v)$.

Stop if no such arc exists.

If there are no negative-length directed circuits, this is a finite method, since at each iteration $\sum_v d(v)$ decreases, while it is at least $\sum_v \mathrm{dist}_l(s,v)$ and it is an integer multiple of the g.c.d. of the $l(a)$.

In fact, it can be shown that the number of iterations is at most $2^{|V|}$, if l is nonnegative. Moreover, if l is arbitrary (without negative directed circuit), there are at most $2|V|^2 L$ iterations, where $L := \max\{|l(a)| \mid a \in A\}$.

However, Johnson [1973a,1973b] showed that the number of iterations is $\Omega(n2^n)$, even if we prescribe to choose (u,v) in (8.11) with $d(u)$ minimal. For nonnegative l, Johnson [1973b,1977a] showed that the number of iterations is $\Omega(2^n)$ if we prescribe no selection rule of u.

8.6d. Shortest and longest paths in acyclic graphs

Let $D = (V, A)$ be a digraph. A subset C of A is called a *directed cut* if there is a subset U of V with $\emptyset \neq U \neq V$ such that $\delta^{\text{out}}(U) = C$ and $\delta^{\text{in}}(U) = \emptyset$. So each directed cut is a cut.

It is easy to see that, if D is acyclic, then a set B of arcs is contained in a directed cut if and only if no two arcs in B are contained in a directed path. Similarly, if D is acyclic, a set B of arcs is contained in a directed path if and only if no two arcs in B are contained in a directed cut.

Theorem 8.12. *Let $D = (V, A)$ be an acyclic digraph and let $s, t \in V$. Then the maximum length of an $s - t$ path is equal to the minimum number of directed $s - t$ cuts covering all arcs that are on at least one $s - t$ path.*

Proof. Any $s - t$ path of length k needs at least k directed $s - t$ cuts to be covered, so the maximum cannot exceed the minimum.

To see equality, let for each $v \in V$, $d(v)$ be equal to the length of a longest $s - v$ path. Let $k := d(t)$. For $i = 1, \ldots, k$, let $U_i := \{v \in V \mid d(v) < i\}$. Then the $\delta^{\text{out}}(U_i)$ form k directed $s - t$ cuts covering all arcs that are on at least one $s - t$ path. ∎

One similarly shows for paths not fixing its ends (Vidyasankar and Younger [1975]):

Theorem 8.13. *Let $D = (V, A)$ be an acyclic digraph. Then the maximum length of any path is equal to the minimum number of directed cuts covering A.*

Proof. Similar to the proof above. ∎

Also weighted versions hold, and may be derived similarly. A weighted version of Theorem 8.12 is:

Theorem 8.14. *Let $D = (V, A)$ be an acyclic digraph, let $s, t \in V$ and let $l : A \to \mathbb{Z}_+$ be a length function. Then the maximum length of an $s - t$ path is equal to the minimum number of directed $s - t$ cuts covering each arc a that is on at least one $s - t$ path, at least $l(a)$ times.*

Proof. Any $s - t$ path of length k needs at least k directed $s - t$ cuts to be covered appropriately, so the maximum cannot exceed the minimum.

To see equality, let for each $v \in V$, $d(v)$ be equal to the length of a longest $s - v$ path. Let $k := d(t)$. For $i = 1, \ldots, k$, let $U_i := \{v \in V \mid d(v) < i\}$. Then the $\delta^{\text{out}}(U_i)$ form k directed $s - t$ cuts covering each arc a on any $s - t$ path at least $l(a)$ times. ∎

Similarly, a weighted version of Theorem 8.13 is:

Theorem 8.15. *Let $D = (V, A)$ be an acyclic digraph and $l : A \to \mathbb{Z}_+$ be a length function. Then the maximum length of any path is equal to the minimum number of directed cuts such that any arc a is in at least $l(a)$ of these directed cuts.*

Proof. Similar to the proof above. ∎

In acyclic graphs one can find shortest paths in *linear* time (Morávek [1970]):

Theorem 8.16. *Given an acyclic digraph $D = (V, A)$, $s, t \in V$, and a length function $l : A \to \mathbb{Q}$, a shortest $s - t$ path can be found in time $O(m)$.*

Proof. First order the vertices reachable from s topologically as $v_1, \ldots, v_n$ (cf. Corollary 6.5b). So $v_1 = s$. Set $d(v_1) := 0$ and determine

$$d(v) := \min\{d(u) + l(u, v) \mid (u, v) \in \delta^{\text{in}}(v)\} \tag{8.12}$$

for $v = v_2, \ldots, v_n$ (in this order). Then for each v reachable from s, $d(v)$ is the distance from s to v. ∎

Note that this implies that also a *longest* path in an acyclic digraph can be found in linear time.

Johnson [1973b] showed that, in a not necessarily acyclic digraph, an $O(m)$-time algorithm for the single-source shortest path problem exists if the number of directed circuits in any strongly connected component is bounded by a constant. Related work was reported by Wagner [2000].

More on longest paths and path covering in acyclic graphs can be found in Chapter 14.

8.6e. Bottleneck shortest path

Pollack [1960] observed that several of the shortest path algorithms can be modified to the following maximum-capacity path problem. For any digraph $D = (V, A)$ and 'capacity' function $c : A \to \mathbb{Q}$, the *capacity* of a path P is the minimum of the capacities of the arcs in P. (This is also called sometimes the *reliability* of P — cf. Section 50.6c.)

Then the *maximum-capacity path problem* (also called the *maximum reliability problem*), is:

(8.13) given: a digraph $D = (V, A)$, $s, t \in V$, and a 'capacity' function $c : A \to \mathbb{Q}$;
find: an $s - t$ path of maximum capacity.

To this end, one should appropriately replace min by max and + by min in these algorithms. Applying this to Dijkstra's algorithm gives, with Fibonacci heaps, a running time of $O(m + n \log n)$.

In fact, the following 'bottleneck' min-max relation holds (Fulkerson [1966]):

Theorem 8.17. *Let $D = (V, A)$ be a digraph, let $s, t \in V$, and let $c : A \to \mathbb{R}$ be a capacity function. Then:*

$$\max_P \min_{a \in AP} c(a) = \min_C \max_{a \in C} c(a), \tag{8.14}$$

where P ranges over all $s - t$ paths and C over all $s - t$ cuts.

Proof. To see $\leq$ in (8.14), let P be an $s-t$ path and let C be an $s-t$ cut. Since P and C have at least one arc in common, say a_0, we have $\min_{a\in AP} c(a) \leq c(a_0) \leq \max_{a\in C} c(a)$.

To see $\geq$ in (8.14), let $\gamma := \max_P \min_{a\in AP} c(a)$. Let $A' := \{a \in A \mid c(a) \leq \gamma\}$. Then A' intersects each $s-t$ path. So A' contains an $s-t$ cut C. Therefore, $c(a) \leq \gamma$ for all $a \in C$; that is, $\max_{a\in C} c(a) \leq \gamma$. Hence $\min_C \max_{a\in C} c(a) \leq \gamma$. ∎

It is easy to solve the bottleneck shortest path problem by binary search in time $O(m \log L)$, where $L := \|l\|_\infty$ assuming l integer. This was improved by Gabow [1985b] to $O(m \log_n L)$.

8.6f. Further notes

Further analyses of shortest path methods were given by Pollack and Wiebenson [1960], Hoffman and Winograd [1972], Tabourier [1973], Pape [1974], Kershenbaum [1981], Glover, Glover, and Klingman [1984], Pallottino [1984], Glover, Klingman, and Phillips [1985], Glover, Klingman, Phillips, and Schneider [1985], Desrochers [1987], Bertsekas [1991], Goldfarb, Hao, and Kai [1991], Sherali [1991], Cherkassky, Goldberg, and Radzik [1994,1996], and Cherkassky and Goldberg [1999] (negative circuit detection).

Spirakis and Tsakalidis [1986] gave an average-case analysis of an $O(nm)$-time negative circuit detecting algorithm, and Tarjan [1982] a sensitivity analysis of shortest paths trees.

Fast approximation algorithms for shortest paths were given by Klein and Sairam [1992], Cohen [1994b,2000], Aingworth, Chekuri, and Motwani [1996], Dor, Halperin, and Zwick [1996,2000], Cohen and Zwick [1997,2001], and Aingworth, Chekuri, Indyk, and Motwani [1999].

Dantzig [1957] observed that the shortest path problem can be formulated as a linear programming problem, and hence can be solved with the simplex method. Edmonds [1970a] showed that this may take exponentially many pivot steps, even for nonnegative arc lengths. On the other hand, Dial, Glover, Karney, and Klingman [1979] and Zadeh [1979] gave pivot rules that solve the shortest path problem with nonnegative arc lengths in $O(n)$ pivots. For arbitrary length Akgül [1985] and Goldfarb, Hao, and Kai [1990b] gave strongly polynomial simplex algorithms. Akgül [1993] gave an algorithm using $O(n^2)$ pivots, yielding an $O(n(m + n \log n))$-time algorithm. An improvement to $O(nm)$ (with $(n-1)(n-2)/2$ pivots) was given by Goldfarb and Jin [1999b]. Related work was done by Dantzig [1963], Orlin [1985], and Ahuja and Orlin [1988,1992].

Computational results were presented by Pape [1974,1980], Golden [1976], Carson and Law [1977], Kelton and Law [1978], van Vliet [1978], Denardo and Fox [1979], Dial, Glover, Karney, and Klingman [1979], Glover, Glover, and Klingman [1984], Imai and Iri [1984], Glover, Klingman, Phillips, and Schneider [1985], Gallo and Pallottino [1988], Mondou, Crainic, and Nguyen [1991], Goldberg and Radzik [1993] (Bellman-Ford method), Cherkassky, Goldberg, and Radzik [1996], Goldberg and Silverstein [1997], and Zhan and Noon [1998].

Frederickson [1987a,1991] gave an algorithm that gives a succint encoding of all pairs shortest path information in a directed planar graph (with arbitrary lengths, but no negative directed circuits).

Surveys and bibliographies on shortest paths were presented by Pollack and Wiebenson [1960], Murchland [1967a], Dreyfus [1969], Gilsinn and Witzgall [1973], Pierce [1975], Yen [1975], Lawler [1976b], Golden and Magnanti [1977], Deo and Pang [1984], Gallo and Pallottino [1986], and Mondou, Crainic, and Nguyen [1991].

Books covering shortest path methods include Christofides [1975], Lawler [1976b], Even [1979], Hu [1982], Papadimitriou and Steiglitz [1982], Smith [1982], Sysło, Deo, and Kowalik [1983], Tarjan [1983], Gondran and Minoux [1984], Nemhauser and Wolsey [1988], Bazaraa, Jarvis, and Sherali [1990], Chen [1990], Cormen, Leiserson, and Rivest [1990], Lengauer [1990], Ahuja, Magnanti, and Orlin [1993], Cook, Cunningham, Pulleyblank, and Schrijver [1998], Jungnickel [1999], Mehlhorn and Näher [1999], and Korte and Vygen [2000].

8.6g. Historical notes on shortest paths

Compared with other combinatorial optimization problems like the minimum spanning tree, assignment, and transportation problems, research on the shortest path problem started relatively late. This might be due to the fact that the problem is elementary and relatively easy, which is also illustrated by the fact that at the moment that the problem came into the focus of interest, several researchers independently developed similar methods. Yet, the problem has offered some substantial difficulties, as is illustrated by the fact that heuristic, nonoptimal approaches have been investigated (cf. for instance Rosenfeld [1956], who gave a heuristic approach for determining an optimal trucking route through a given traffic congestion pattern).

Search methods

Depth-first search methods were described in the 19th century in order to traverse all lanes in a maze without knowing its plan. Wiener [1873] described the following method:

> Man markire sich daher den Weg, den man zurücklegt nebst dem Sinne, in welchem es geschieht. Sobald man auf einen schon markirten Weg stösst, kehre man um und durchschreite den schon beschriebenen Weg in umgekehrtem Sinne. Da man, wenn man nicht ablenkte, denselben hierbei in seiner ganzen Ausdehnung nochmals zurücklegen würde, so muss mann nothwendig hierbei auf einen noch nicht markirten einmündenden Weg treffen, den man dann verfolge, bis man wieder auf einen markirten trifft. Hier kehre man wieder um und verfahre wie vorher. Es werden dadurch stets neue Wegtheile zu den beschriebenen zugefügt, so dass man nach einer endlichen Zeit das ganze Labyrinth durchwandern würde und so jedenfalls den Ausgang fände, wenn er nicht schon vorher erreicht worden wäre.[3]

[3] One therefore marks the road that one traverses together with the direction in which it happens. As soon as one hits a road already marked, one turns and traverses the road already followed in opposite direction. As one, if one would not deviate, would traverse it to its whole extent another time, by this one should necessarily meet a road running to a not yet marked one, which one next follows, until one again hits a marked one. Here one turns again and proceeds as before. In that road always new road parts are added to those already followed, so that after a finite time one would walk through the whole labyrinth and in this road in any case would find the exit, if it would not have been reached already before.

In his book *Recréations mathématiques*, Lucas [1882] described a method due to C.P. Trémaux to traverse all lanes of a maze exactly twice, starting at a vertex A: First traverse an arbitrary lane starting at A. Next apply the following rule iteratively when arriving through a lane L at a vertex N:

(8.15) if you did not visit N before, traverse next an arbitrary other lane at N, except if L is the only lane at N, in which case you return through L;
if you have visited N before, return through L, except if you have traversed L already twice; in that case traverse another lane at N not traversed before; if such a lane does not exist, traverse a lane at L that you have traversed before once.

The method stops if no lane can be chosen by this rule at N. It is not hard to show that then one is at the starting vertex A and that all lanes of the maze have been traversed exactly twice (if the maze is connected).

A simpler rule was given by Tarry [1895]:

> Tout labyrinthe peut être parcouru en une seule course, en passant deux fois en sens contraire par chacune des allées, sans qu'il soit nécessaire d'en connaître le plan.
> Pour résoudre ce problème, il suffit d'observer cette règle unique:
> *Ne reprendre l'allée initiale qui a conduit à un carrefour pour la première fois que lorsqu'on ne peut pas faire autrement.*[4]

This is equivalent to depth-first search.

Alternate routing

Path problems were also studied at the beginning of the 1950s in the context of 'alternate routing', that is, finding a second shortest route if the shortest route is blocked. This applies to freeway usage (cf. Trueblood [1952]), but also to telephone call routing. At that time making long-distance calls in the U.S.A. was automatized, and alternate routes for telephone calls over the U.S. telephone network nation-wide should be found automatically:

> When a telephone customer makes a long-distance call, the major problem facing the operator is how to get the call to its destination. In some cases, each toll operator has two main routes by which the call can be started towards this destination. The first-choice route, of course, is the most direct route. If this is busy, the second choice is made, followed by other available choices at the operator's discretion. When telephone operators are concerned with such a call, they can exercise choice between alternate routes. But when operator or customer toll dialing is considered, the choice of routes has to be left to a machine. Since the "intelligence" of a machine is limited to previously "programmed" operations, the choice of routes has to be decided upon, and incorporated in, an automatic alternate routing arrangement.

(Jacobitti [1955], cf. Myers [1953], Clos [1954], and Truitt [1954]).

[4] Each maze can be traversed in one single run, by passing each of the corridors twice in opposite direction, without that it is necessary to know its plan.
To solve this problem, it suffices to observe this only rule:
Retake the initial corridor that has led to a crossing for the first time only when one cannot do otherwise.

Matrix methods

Matrix methods were developed to study relations in networks, like finding the transitive closure of a relation; that is, identifying in a digraph the pairs of vertices s, t such that t is reachable from s. Such methods were studied because of their application to communication nets (including neural nets) and to animal sociology (e.g. peck rights).

The matrix methods consist of representing the relation by a matrix, and then taking iterative matrix products to calculate the transitive closure. This was studied by Landahl and Runge [1946], Landahl [1947], Luce and Perry [1949], Luce [1950], Lunts [1950,1952], and by A. Shimbel.

Shimbel's interest in matrix methods was motivated by their applications to neural networks. He analyzed with matrices which sites in a network can communicate to each other, and how much time it takes. To this end, let S be the $0,1$ matrix indicating that if $S_{i,j} = 1$, then there is direct communication from i to j (including $i = j$). Shimbel [1951] observed that the positive entries in S^t correspond to the pairs between which there exists communication in t steps. An *adequate* communication system is one for which S^t is positive for some t. One of the other observations of Shimbel [1951] is that in an adequate communication system, the time it takes that all sites have all information, is equal to the minimum value of t for which S^t is positive. (A related phenomenon was observed by Luce [1950].)

Shimbel [1953] mentioned that the distance from i to j is equal to the number of zeros in the i, j position in the matrices $S^0, S^1, S^2, \ldots, S^t$. So essentially he gave an $O(n^4)$ algorithm to find all distances in a (unit-length) digraph.

Shortest paths

The basic methods for the shortest path problem are the Bellman-Ford method and Dijkstra's method. The latter one is faster but is restricted to nonnegative length functions. The former method only requires that there is no directed circuit of negative length.

The general framework for both methods is the following scheme, described in this general form by Ford [1956]. Keep a provisional distance function d. Initially, set $d(s) := 0$ and $d(v) := \infty$ for each $v \neq s$. Next, iteratively,

(8.16) choose an arc (u, v) with $d(v) > d(u) + l(u, v)$ and reset $d(v) := d(u) + l(u, v)$.

If no such arc exists, d is the distance function.

The difference in the methods is the rule by which the arc (u, v) with $d(v) > d(u) + l(u, v)$ is chosen. The Bellman-Ford method consists of considering all arcs consecutively and applying (8.16) where possible, and repeating this (at most $|V|$ rounds suffice). This is the method described by Shimbel [1955], Bellman [1958], and Moore [1959].

Dijkstra's method prescribes to choose an arc (u, v) with $d(u)$ smallest (then each arc is chosen at most once, if the lengths are nonnegative). This was described by Leyzorek, Gray, Johnson, Ladew, Meaker, Petry, and Seitz [1957] and Dijkstra [1959]. A related method, but slightly slower than Dijkstra's method when implemented, was given by Dantzig [1958], G.J. Minty, and Whiting and Hillier [1960], and chooses an arc (u, v) with $d(u) + l(u, v)$ smallest.

Parallel to this, a number of further results were obtained on the shortest path problem, including a linear programming approach and 'good characterizations'.

We now describe the developments in greater detail.

The Bellman-Ford method: Shimbel

In April 1954, Shimbel [1955] presented at the Symposium on Information Networks in New York some observations on calculating distances, which amount to describing a 'min-addition' algebra and a method which later became known as the Bellman-Ford method. He introduced:

> Arithmetic
> For any arbitrary real or infinite numbers x and y
>
> $$x + y \equiv \min(x, y) \text{ and}$$
> $$xy \equiv \text{the algebraic sum of } x \text{ and } y.$$

He extended this arithmetic to the matrix product. Calling the distance matrix associated with a given length matrix S the 'dispersion', he stated:

> It follows trivially that S^k $k \geq 1$ is a matrix giving the shortest paths from site to site in S given that $k-1$ other sites may be traversed in the process. It also follows that for any S there exists an integer k such that $S^k = S^{k+1}$. Clearly, the dispersion of S (let us label it $D(S)$) will be the matrix S^k such that $S^k = S^{k+1}$.

Although Shimbel did not mention it, one trivially can take $k \leq |V|$, and hence the method yields an $O(n^4)$ algorithm to find the distances between all pairs of vertices.

Leyzorek, Gray, Johnson, Ladew, Meaker, Petry, and Seitz [1957] noted that Shimbel's method can be speeded up by calculating S^k by iteratively raising the current matrix to the square (in the min-addition matrix algebra). This solves the all-pairs shortest paths problem in time $O(n^3 \log n)$.

The Bellman-Ford method: Ford

In a RAND paper dated 14 August 1956, Ford [1956] described a method to find a shortest path from P_0 to P_N, in a network with vertices $P_0, \ldots, P_N$, where l_{ij} denotes the length of an arc from i to j:

> Assign initially $x_0 = 0$ and $x_i = \infty$ for $i \neq 0$. Scan the network for a pair P_i and P_j with the property that $x_i - x_j > l_{ji}$. For this pair replace x_i by $x_j + l_{ji}$. Continue this process. Eventually no such pairs can be found, and x_N is now minimal and represents the minimal distance from P_0 to P_N.

Ford next argues that the method terminates. It was shown by Johnson [1973a, 1973b,1977a] that Ford's liberal selection rule can require exponential time.

In their book *Studies in the Economics of Transportation*, Beckmann, McGuire, and Winsten [1956] showed that the distance matrix $D = (d_{i,j})$ is the unique matrix satisfying

$$\begin{aligned} &d_{i,i} = 0 \text{ for all } i; \\ &d_{i,k} = \min_j (l_{i,j} + d_{j,k}) \text{ for all } i, k \text{ with } i \neq k. \end{aligned} \tag{8.17}$$

The Bellman-Ford method: Bellman

We next describe the work of Bellman on shortest paths. After publishing several papers on dynamic programming (in a certain sense a generalization of shortest path methods), Bellman [1958] eventually focused on the shortest path problem by itself. He described the following 'functional equation approach' (originating from dynamic programming) for the shortest path problem, which is the same as that of Shimbel [1955].

Bellman considered N cities, numbered $1, \ldots, N$, every two of which are linked by a direct road, together with an $N \times N$ matrix $T = (t_{i,j})$, where $t_{i,j}$ is the time required to travel from i to j (not necessarily symmetric). Find a path between 1 and N which consumes minimum time. First, Bellman remarked that the problem is finite:

> Since there are only a finite number of paths available, the problem reduces to choosing the smallest from a finite set of numbers. This direct, or enumerative, approach is impossible to execute, however, for values of N of the order of magnitude of 20.

Next he gave a 'functional equation approach':

> The basic method is that of successive approximations. We choose an initial sequence $\{f_i^{(0)}\}$, and then proceed iteratively, setting
>
> $$f_i^{(k+1)} = \operatorname*{Min}_{j \neq i} (t_{ij} + f_j^{(k)}], \qquad i = 1, 2, \cdots, N-1,$$
>
> $$f_N^{(k+1)} = 0,$$
>
> for $k = 0, 1, 2 \cdots,$.

For the initial function $f_i^{(0)}$, Bellman proposed (upon a suggestion of F. Haight) to take $f_i^{(0)} = t_{i,N}$ for all i. Bellman observed that, for each fixed i, starting with this choice of $f_i^{(0)}$ gives that $f_i^{(k)}$ is monotonically nonincreasing in k, and states:

> It is clear from the physical interpretation of this iterative scheme that at most $(N-1)$ iterations are required for the sequence to converge to the solution.

Since each iteration can be done in time $O(N^2)$, the algorithm takes time $O(N^3)$. Bellman also remarks:

> It is easily seen that the iterative scheme discussed above is a feasible method for either hand or machine computation for values of N of the order of magnitude of 50 or 100.

In a footnote, Bellman says:

> *Added in proof* (*December* 1957): After this paper was written, the author was informed by Max Woodbury and George Dantzig that the particular iterative scheme discussed in Sec. 5 had been obtained by them from first principles.

Bellman [1958] mentioned that one could also start with $f_i^{(0)} = \min_{j \neq i} t_{i,j}$ (if $i \neq N$) and $f_N^{(0)} = 0$. In that case for each fixed i, the value of $f_i^{(k)}$ is monotonically *nondecreasing* in k, and converges to the distance from i to N. (Indeed, $f_i^{(k)}$ is equal to the shortest length of all those paths starting at i that have either exactly $k+1$ arcs, or have at most k arcs and end at N.)

The Bellman-Ford method: Moore

At the International Symposium on the Theory of Switching at Harvard University in April 1957, Moore [1959] of Bell Laboratories presented a paper 'The shortest path through a maze':

> The methods given in this paper require no foresight or ingenuity, and hence deserve to be called algorithms. They would be especially suited for use in a machine, either a special-purpose or a general-purpose digital computer.

The motivation of Moore was the routing of toll telephone traffic. He gave algorithms A, B, and C, and D.

First, Moore considered the case of an undirected graph $G = (V, E)$ with no length function, where a path from vertex A to vertex B should be found with a minimum number of edges. Algorithm A is: first give A label 0. Next do the following for $k = 0, 1, \ldots$: give label $k+1$ to all unlabeled vertices that are adjacent to some vertex labeled k. Stop as soon as vertex B is labeled.

> If it were done as a program on a digital computer, the steps given as single steps above would be done serially, with a few operations of the computer for each city of the maze; but, in the case of complicated mazes, the algorithm would still be quite fast compared with trial-and-error methods.

In fact, a direct implementation of the method would yield an algorithm with running time $O(m)$. It is essentially breadth-first search. Algorithms B and C differ from A in a more economical labeling (by fewer bits).

Moore's algorithm D finds a shortest route for the case where each edge of the graph has a nonnegative length. This method gives a refinement of Bellman's method described above: (i) it extends to the case that not all pairs of vertices have a direct connection; that is, if there is an underlying graph $G = (V, E)$ with length function; (ii) at each iteration only those $d_{i,j}$ are considered for which u_i has been decreased in the previous iteration.

The method has running time $O(nm)$. Moore observed that the algorithm is suitable for parallel implementation, yielding a decrease in the running time bound to $O(n\Delta(G))$, where $\Delta(G)$ is the maximum degree of G. He concluded:

> The origin of the present methods provides an interesting illustration of the value of basic research on puzzles and games. Although such research is often frowned upon as being frivolous, it seems plausible that these algorithms might eventually lead to savings of very large sums of money by permitting more efficient use of congested transportation or communication systems. The actual problems in communication and transportation are so much complicated by timetables, safety requirements, signal-to-noise ratios, and economic requirements that in the past those seeking to solve them have not seen the basic simplicity of the problem, and have continued to use trial-and-error procedures which do not always give the true shortest path. However, in the case of a simple geometric maze, the absence of these confusing factors permitted algorithms A, B, and C to be obtained, and from them a large number of extensions, elaborations, and modifications are obvious. The problem was first solved in connection with Claude Shannon's maze-solving machine. When this machine was used with a maze which had more than one solution, a visitor asked why it had not been built to always find the shortest path. Shannon and I each attempted to find economical methods of doing this by machine. He found several methods suitable for analog computation, and I obtained these algorithms. Months later the applicability of these ideas to practical problems in communication and transportation systems was suggested.

Among the further applications of his method, Moore described the example of finding the fastest connections from one station to another in a given railroad timetable (cf. also Levin and Hedetniemi [1963]). A similar method was given by Minty [1958].

Berge [1958b] described a breadth-first search method similar to Moore's algorithm A, to find the shortest paths from a given vertex a, for unit lengths, but he described it more generally for directed graphs: let $A(0) := \{a\}$; if $A(k)$ has been found, let $A(k+1)$ be the set of vertices x for which there is a $y \in A(k)$ with (y, x) an arc and with $x \notin A(i)$ for all $i \leq k$. One directly finds a shortest $a-b$ path from the $A(k)$. This gives an $O(m)$ algorithm.

D.A. D'Esopo (cf. the survey of Pollack and Wiebenson [1960]) proposed the following sharpening of Moore's version of the Bellman-Ford method, by indexing the vertices during the algorithm. First define index$(s) := 1$, and let $i := 1$. Then apply the following iteratively:

(8.18) Let v be the vertex with index i. For each arc (v, w) leaving v, reset $d(w) := d(v)+l(v, w)$ if it decreases $d(w)$; if w is not indexed give it the smallest unused index; if some $d(w)$ has been reset with index$(w) < i$, choose the w minimizing index(w), and let $i :=$index(w); otherwise, let $i := i + 1$.

In May 1958, Hoffman and Pavley [1959b] reported, at the Western Joint Computer Conference in Los Angeles, the following computing time for finding the distances between all pairs of vertices by Moore's algorithm (with nonnegative lengths):

> It took approximately three hours to obtain the minimum paths for a network of 265 vertices on an IBM *704*.

Linear programming and transportation

Orden [1955] observed that the shortest path problem is a special case of the transshipment problem: let be given an $n \times n$ matrix $(c_{i,j})$ and a vector $g \in \mathbb{R}^n$

(8.19)
$$\begin{aligned} &\text{minimize } \textstyle\sum_{i,j} c_{i,j} x_{i,j} \\ &\text{subject to } \sum_{j=1}^{n} (x_{i,j} - x_{j,i}) = g_i \text{ for } i = 1, \dots, n \\ &\text{and } x_{i,j} \geq 0 \text{ for } i, j = 1, \dots n, \end{aligned}$$

and showed that it can be reduced to a 'transportation problem', and hence to a linear programming problem. If one wants to find a shortest $1-n$ path, set $g_1 = 1$, $g_n = -1$, and $g_i = 0$ for all other i.

In a paper presented at the Summer 1955 meeting of ORSA at Los Angeles, Dantzig [1957] formulated the shortest path problem as an integer linear programming problem 'very similar to the system for the assignment problem', and similar to Orden's formulation. Dantzig observed that replacing the condition $x_{i,j} \geq 0$ by $x_{i,j} \in \{0, 1\}$ does not change the minimum value. (Dantzig assumed $d_{i,j} = d_{j,i}$ for all i, j.)

He described a graphical procedure for the simplex method applied to this problem. Let T be a rooted tree on $\{1, \dots, n\}$, with root 1. For each $i = 1, \dots, n$,

let u_i be equal to the length of the path from 1 to i in T. Now if $u_j \leq u_i + d_{i,j}$ for all i, j, then for each i, the $1 - i$ path in T is a shortest path. If $u_j > u_i + d_{i,j}$, replace the arc of T entering j by the arc (i, j), and iterate with the new tree.

Trivially, this process terminates (as $\sum_{j=1}^{n} u_j$ decreases at each iteration, and as there are only finitely many rooted trees). (Edmonds [1970a] showed that the method may take exponential time.) Dantzig illustrated his method by an example of sending a package from Los Angeles to Boston.

Good characterizations

Robacker [1956b] observed that the minimum length of a $P_0 - P_n$ path in a graph N is equal to the maximum number of disjoint $P_0 - P_n$ cuts:

> the maximum number of mutually disjunct cuts of N is equal to the length of the shortest chain of N from P_0 to P_n.

Gallai [1958b] noticed that if the length function $l : A \to \mathbb{Z}$ on the arcs of a digraph (V, A) gives no negative-length directed circuits, then there is a 'potential' $p : V \to \mathbb{Z}$ with $l(u, v) \geq p(v) - p(u)$ for each arc (u, v).

Case Institute of Technology 1957

In the *First Annual Report* of the project *Investigation of Model Techniques*, carried out by the Case Institute of Technology in Cleveland, Ohio for the Combat Development Department of the Army Electronic Proving Ground, Leyzorek, Gray, Johnson, Ladew, Meaker, Petry, and Seitz [1957] describe (rudimentarily) a shortest path algorithm similar to Dijkstra's algorithm:

> (1) All the links joined to the origin, a, may be given an outward orientation. ...
> (2) Pick out the link or links radiating from a, $a_{a\alpha}$, with the smallest delay. ... Then it is impossible to pass from the origin to any other node in the network by any "shorter" path than $a_{a\alpha}$. Consequently, the minimal path to the general node α is $a_{a\alpha}$.
> (3) All of the other links joining α may now be directed outward. Since $a_{a\alpha}$ must necessarily be the minimal path to α, there is no advantage to be gained by directing any other links toward α. ...
> (4) Once α has been evaluated, it is possible to evaluate immediately all other nodes in the network whose minimal values do not exceed the value of the second-smallest link radiating from the origin. Since the minimal values of these nodes are less than the values of the second-smallest, third-smallest, and all other links radiating directly from the origin, only the smallest link, $a_{a\alpha}$, can form a part of the minimal path to these nodes. Once a minimal value has been assigned to these nodes, it is possible to orient all other links except the incoming link in an outward direction.
> (5) Suppose that all those nodes whose minimal values do not exceed the value of the second-smallest link radiating from the origin have been evaluated. Now it is possible to evaluate the node on which the second-smallest link terminates. At this point, it can be observed that if conflicting directions are assigned to a link, in accordance with the rules which have been given for direction assignment, that link may be ignored. It will not be a part of the minimal path to either of the two nodes it joins. ...
> Following these rules, it is now possible to expand from the second-smallest link as well as the smallest link so long as the value of the third-smallest link radiating from the origin is not exceeded. It is possible to proceed in this way until the entire network has been solved.

(In this quotation we have deleted sentences referring to figures.)

Leyzorek, Gray, Johnson, Ladew, Meaker, Petry, and Seitz [1957] also described a speedup of solving the all-pairs shortest paths problem by matrix-multiplication:

> This process of multiplication may be simplified somewhat by squaring the original structure matrix to obtain a dispersion matrix which is the second power of the structure matrix; squaring the second-power matrix to obtain the fourth power of the structure matrix; and so forth.

This gives an $O(n^3 \log n)$-time all-pairs shortest paths algorithm.

Analog computing

In a reaction to the linear programming approach of Dantzig [1957] discussed above, Minty [1957] proposed an 'analog computer' for the shortest path problem:

> Build a string model of the travel network, where knots represent cities and string lengths represent distances (or costs). Seize the knot 'Los Angeles' in your left hand and the knot 'Boston' in your right and pull them apart. If the model becomes entangled, have an assistant untie and re-tie knots until the entanglement is resolved. Eventually one or more paths will stretch tight — they then are alternative shortest routes.
> Dantzig's 'shortest-route tree' can be found in this model by weighting the knots and picking up the model by the knot 'Los Angeles'.
> It is well to label the knots since after one or two uses of the model their identities are easily confused.

A similar method was proposed by Bock and Cameron [1958] (cf. Peart, Randolph, and Bartlett [1960]). The method was extended to the directed case by Klee [1964].

Rapaport and Abramson [1959] described an electric analog computer for solving the shortest path problem.

Dantzig's $O(n^2 \log n)$ algorithm

Dantzig [1958,1960] gave an $O(n^2 \log n)$ algorithm for the shortest path problem with nonnegative length function. A set X is updated throughout, together with a function $d : X \to \mathbb{Q}_+$. Initially $X = \{s\}$ and $d(s) = 0$. Then do the following iteratively:

(8.20) for each $v \in X$, let w_v be a vertex not in X with $d(w_v)$ minimal. Choose a $v \in X$ minimizing $d(v) + l(v, w_v)$. Add w_v to X and set $d(w_v) := d(v) + l(v, w_v)$.

Stop if $X = V$.

Note that throughout the iterations, the function d is only extended, and not updated. Dantzig assumed

> (a) that one can write down without effort for each node the arcs leading to other nodes in increasing order of length and (b) that it is no effort to ignore an arc of the list if it leads to a node that has been reached earlier.

Indeed, in a preprocessing the arcs can be ordered in time $O(n^2 \log n)$, and, for instance by using doubly linked lists, an arc can be deleted from the appropriate list in time $O(1)$. As each iteration can be done in time $O(n)$ (identifying a v

minimizing $d(v) + l(v, w_v)$ and deleting all arcs entering w_v from each list of arcs leaving x for $x \in X$), Dantzig's method can be performed in time $O(n^2 \log n)$.

Dantzig [1958,1960] mentioned that, beside Bellman, Moore, Ford, and himself, also D. Gale and Fulkerson proposed shortest path methods, 'in informal conversations'.

The same method as that of Dantzig (however without the observations concerning storing the outgoing arcs from any vertex in a list) was given by G.J. Minty (cf. Pollack and Wiebenson [1960]) and by Whiting and Hillier [1960].

Dijkstra's $O(n^2)$ algorithm

Dijkstra [1959] gave an $O(n^2)$ method which is slightly different from that of Dantzig [1958,1960]. Let $D = (V, A)$ be a graph and let a length function $l : A \to \mathbb{R}_+$ be given. Dijkstra's method consists of repeatedly updating a set X and a function $d : V \to \mathbb{R}_+$ as follows.

Initially, set $X = \emptyset$, $d(s) = 0$, $d(v) = \infty$ if $v \neq s$. Next move s into X. Then do the following iteratively: Let v be the vertex just moved into X;

(8.21) for each arc (v, w) with $w \notin X$, reset $d(w) := d(v) + l(v, w)$ if this would decrease $d(w)$. Choose $v' \notin X$ with minimum $d(v')$, and move v' into X.

Stop if no such v' exists.

Since each iteration can be done in time $O(n)$ and since there are at most $|V|$ iterations, the algorithm runs in time $O(n^2)$. Dijkstra states:

> The solution given above is to be preferred to the solution by L.R. Ford [3] as described by C. Berge [4], for, irrespective of the number of branches, we need not store the data for all branches simultaneously but only those for the branches in sets I and II, and this number is always less than n. Furthermore, the amount of work to be done seems to be considerably less.

(Dijkstra's references [3] and [4] are Ford [1956] and Berge [1958b].)

Dijkstra's method is easier to implement (as an $O(n^2)$ algorithm) than Dantzig's, since we need not store the information in lists: in order to find $v' \notin X$ minimizing $d(v')$ we can just scan all vertices that are not in X.

Whiting and Hillier [1960] described the same method as Dijkstra.

Heaps

The 2-heap was introduced by Williams [1964] (describing it as an array, with subroutines to insert and extract elements of the heap), as a major improvement to the sorting algorithm 'treesort' of Floyd [1962a]. The 2-heap of Williams was next extended by Floyd [1964] to the sorting algorithm 'treesort3'.

In an erratum of 24 October 1968 to a report of the London Business School, Murchland [1967b] seems to be the first to use heaps for finding shortest paths, although he concludes to a time bound of $O(n^2 \log n)$ only — worse than Dijkstra's bound $O(n^2)$. E.L. Johnson [1972] improved Murchland's method to $O(m \log(n^2/m))$. He also considers the k-heap for arbitrary k ('k-tree').

In his Ph.D. thesis, D.B. Johnson [1973b], using a sharper analysis and k-heaps, obtains an $O((nd + m) \log_d n)$-time algorithm, implying algorithms with running

time $O(m \log n)$ and $O(n^{1+\varepsilon} + m)$ (for each $\varepsilon > 0$) (published in D.B. Johnson [1977a]). Tarjan [1983] observed that taking $d := m/n$ gives $O(m \log_{m/n} n)$. Next, Fredman and Tarjan [1984,1987] showed that Fibonacci heaps give $O(m + n \log n)$.

All pairs: Roy, Warshall, Floyd, Dantzig

Based on a study of Kleene [1951,1956], McNaughton and Yamada [1960] gave a formula to calculate a 'regular expression' associated with a 'state graph' (essentially describing all paths from a given source) that is quite similar to the fast method of Roy [1959] and Warshall [1962] to compute the transitive closure $\overline{A}$ of a digraph $D = (V, A)$: Assume that the vertices are ordered $1, \ldots, n$. First set $\widetilde{A} := A$. Next, for $k = 1, \ldots, n$, add to $\widetilde{A}$ all pairs (i, j) for which both (i, k) and (k, j) belong to $\widetilde{A}$. The final $\widetilde{A}$ equals $\overline{A}$. This gives an $O(n^3)$ algorithm, which is faster than iterative matrix multiplication.

Floyd [1962b] extended this method to an algorithm to find all distances $d(i, j)$ given a length function $l : V \times V \to \mathbb{Q}_+$: First set $d(i, j) := l(i, j)$ for all i, j. Next, for $k = 1, \ldots, n$, reset, for all i, j, $d(i, j) := d(i, k) + d(k, j)$ if it decreases $d(i, j)$. This gives an $O(n^3)$ transitive closure algorithm for finding the distances between all pairs of vertices.

Dantzig [1967] proposed a variant of this method. For i, j, k with $i \leq k$ and $j \leq k$, let $d^k_{i,j}$ be the length of the shortest $i - j$ path in the graph induced by $\{1, \ldots, k\}$. Then there is an easy iterative scheme to determine the $d^k_{i,j}$ from the $d^{k-1}_{i,j}$: first $d^k_{i,k} = \min_{j<k}(d^{k-1}_{i,j} + l_{j,k})$ and $d^k_{k,i} = \min_{j<k}(l_{k,j} + d^{k-1}_{j,i})$. Next, for all $i, j < k$, $d^k_{i,j} = \min(d^{k-1}_{i,j}, d^k_{i,k} + d^k_{k,j})$.

Negative lengths

Ford and Fulkerson [1962] seem to be the first to observe that the Bellman-Ford method also works for arbitrary lengths as long as each directed circuit has nonnegative length. It is also implicit in the paper of Iri [1960].

PERT and CPM

The application of shortest path (and other) methods in the form of PERT (*Program Evaluation and Review Technique*, originally called *Program Evaluation Research Task*) started in 1958, and was reported by Malcolm, Roseboom, Clark, and Fazar [1959]. The use of the *Critical Path Method* (CPM) was described by Kelley [1957, 1961], and Kelley and Walker [1959].

The kth shortest path

Bock, Kantner, and Haynes [1957,1958] described a method to find the kth shortest path in a graph, based essentially on enumerating. Hoffman and Pavley [1959a] described an adaptation of Dantzig's tree method to obtain the kth shortest path. Bellman and Kalaba [1960] gave a method to find the kth shortest paths from a given vertex simultaneously to all other vertices. Also Pollack [1961b] described a method for the kth shortest path problem, especially suitable if k is small. A survey was given by Pollack [1961a].

Bottleneck path problems

Pollack [1960] modified the shortest path algorithm so as to obtain a path with maximum capacity (the capacity of a path is equal to the minimum of the capacities of the arcs in the path). Related work was done by Amara, Lindgren, and Pollack [1961].

Fanning out from both ends

Berge and Ghouila-Houri [1962] and Dantzig [1963] proposed to speed up Dijkstra's method by fanning out at both ends simultaneously. Berge and Ghouila-Houri [1962] proposed to stop as soon as a vertex is permanently labeled from both ends; however, one may see that this need not yield a shortest path.

Dantzig [1963] proposed to add an arc (s, v) with length $d(s, v)$ as soon as v is permanently labeled when fanning out from s, and similarly add an arc (w, t) with length $d(w, t)$ if a vertex w is labeled permanently when fanning out from t:

> The algorithm terminates whenever the fan of one of the problems reaches its terminal in the other.

Chapter 9

Disjoint paths

Having done with *shortest* paths, we now arrive at *disjoint* paths. We consider disjoint $s-t$ paths, where s and t are the same for all paths. The more general problem where we prescribe for each path a (possibly different) pair of ends, will be discussed in Part VII.
Menger's theorem equates the maximum number of disjoint $s-t$ paths to the minimum size of a cut separating s and t. There are several variants of Menger's theorem, all about equivalent: undirected, directed, vertex-disjoint, arc- or edge-disjoint. The meaning of 'cut' varies accordingly.
Next to Menger's min-max relation, we consider the algorithmic side of disjoint $s-t$ paths. This will be an extract from the related maximum flow algorithms to be discussed in the next chapter. (Maximum integer flow can be viewed as the capacitated version of disjoint $s-t$ paths.)

9.1. Menger's theorem

Menger [1927] gave a min-max theorem for the maximum number of disjoint $S-T$ paths in an undirected graph. It was observed by Grünwald [1938] (= T. Gallai) that the theorem also holds for directed graphs. We follow the proof given by Göring [2000].

Recall that a path is an $S-T$ *path* if it runs from a vertex in S to a vertex in T. A set C of vertices is called $S-T$ *disconnecting* if C intersects each $S-T$ path (C may intersect $S \cup T$).

Theorem 9.1 (Menger's theorem (directed vertex-disjoint version))**.** *Let $D=(V,A)$ be a digraph and let $S,T \subseteq V$. Then the maximum number of vertex-disjoint $S-T$ paths is equal to the minimum size of an $S-T$ disconnecting vertex set.*

Proof. Obviously, the maximum does not exceed the minimum. Equality is shown by induction on $|A|$, the case $A=\emptyset$ being trivial.

Let k be the minimum size of an $S-T$ disconnecting vertex set. Choose $a=(u,v) \in A$. If each $S-T$ disconnecting vertex set in $D-a$ has size at least k, then inductively there exist k vertex-disjoint $S-T$ paths in $D-a$, hence in D.

So we can assume that $D-a$ has an $S-T$ disconnecting vertex set C of size $\leq k-1$. Then $C\cup\{u\}$ and $C\cup\{v\}$ are $S-T$ disconnecting vertex sets of D of size k.

Now each $S-(C\cup\{u\})$ disconnecting vertex set B of $D-a$ has size at least k, as it is $S-T$ disconnecting in D. Indeed, each $S-T$ path P in D intersects $C\cup\{u\}$, and hence P contains an $S-(C\cup\{u\})$ path in $D-a$. So P intersects B.

So by induction, $D-a$ contains k disjoint $S-C\cup\{u\}$ paths. Similarly, $D-a$ contains k disjoint $C\cup\{v\}-T$ paths. Any path in the first collection intersects any path in the second collection only in C, since otherwise $D-a$ contains an $S-T$ path avoiding C.

Hence, as $|C|=k-1$, we can pairwise concatenate these paths to obtain disjoint $S-T$ paths, inserting arc a between the path ending at u and starting at v. ▮

A consequence of this theorem is a variant on *internally vertex-disjoint* $s-t$ paths, that is, $s-t$ paths having no vertex in common except for s and t. Recall that a set U of vertices is called an $s-t$ *vertex-cut* if $s,t\notin U$ and each $s-t$ path intersects U.

Corollary 9.1a (Menger's theorem (directed internally vertex-disjoint version)). *Let $D=(V,A)$ be a digraph and let s and t be two nonadjacent vertices of D. Then the maximum number of internally vertex-disjoint $s-t$ paths is equal to the minimum size of an $s-t$ vertex-cut.*

Proof. Let $D':=D-s-t$ and let S and T be the sets of outneighbours of s and of inneighbours of t, respectively. Then Theorem 9.1 applied to D',S,T gives the corollary. ▮

In turn, Theorem 9.1 follows from Corollary 9.1a by adding two new vertices s and t and arcs (s,v) for all $v\in S$ and (v,t) for all $v\in T$.

Also an arc-disjoint version can be derived (where paths are *arc-disjoint* is they have no arc in common). This version was first formulated by Dantzig and Fulkerson [1955,1956] for directed graphs and by Kotzig [1956] for undirected graphs.

Recall that a set C of arcs is an $s-t$ *cut* if $C=\delta^{\rm out}(U)$ for some subset U of V with $s\in U$ and $t\notin U$.

Corollary 9.1b (Menger's theorem (directed arc-disjoint version)). *Let $D=(V,A)$ be a digraph and $s,t\in V$. Then the maximum number of arc-disjoint $s-t$ paths is equal to the minimum size of an $s-t$ cut.*

Proof. Let $L(D)$ be the line digraph of D and let $S:=\delta^{\rm out}_A(s)$ and $T:=\delta^{\rm in}_A(t)$. Then Theorem 9.1 for $L(D),S,T$ implies the corollary. Note that a minimum-size set of arcs intersecting each $s-t$ path necessarily is an $s-t$ cut. ▮

The internally vertex-disjoint version of Menger's theorem can be derived in turn from the arc-disjoint version: make digraph D' as follows from D: replace any vertex v by two vertices v', v'' and make an arc (v', v''); moreover, replace each arc (u, v) by (u'', v'). Then Corollary 9.1b for D', s'', t' gives Corollary 9.1a for D, s, t.

Similar theorems hold for *undirected* graphs. The undirected vertex-disjoint version follows immediately from Theorem 9.1 by replacing each undirected edge by two oppositely oriented arcs. Next, the undirected edge-disjoint version follows from the undirected vertex-disjoint version applied to the line graph (like the proof of Corollary 9.1b).

9.1a. Other proofs of Menger's theorem

The proof above of Theorem 9.1b was given by Göring [2000], which curtails the proof of Pym [1969a], which by itself is a simplification of a proof of Dirac [1966]. The basic idea (decomposition into two subproblems determined by a minimum-size cut) is due to Menger [1927], for the undirected vertex-disjoint version. (Menger's original proof contains a hole, closed by Kőnig [1931] — see Section 9.6e.)

Hajós [1934] gave a different proof for the undirected vertex-disjoint case, based on intersections and unions of sets determining a cut. Also the proofs given in Nash-Williams and Tutte [1977] are based on this. We give the first of their proofs.

Let $G = (V, E)$ be an undirected graph and let $s, t \in V$. Suppose that the minimum size of an $s - t$ vertex-cut is k. We show by induction on $|E|$ that there exist k vertex-disjoint $s - t$ paths.

The statement is trivial if each edge is incident with at least one of s and t. So we can consider an edge $e = xy$ incident with neither s nor t.

We can assume that $G - e$ has an $s - t$ vertex-cut C of size $k - 1$ — otherwise the statement follows by induction. Similarly, we can assume that G/e has an $s - t$ vertex-cut of size $k - 1$ — otherwise the statement follows by induction again. Necessarily, this cut contains the new vertex obtained by contracting e. Hence G has an $s - t$ vertex-cut C' of size k containing both x and y.

Now let C_s be the set of vertices in $C \cup C'$ that are reachable in G from s by a path with no internal vertex in $C \cup C'$. Similarly, let C_t be the set of vertices in $C \cup C'$ that are reachable in G from t by a path with no internal vertex in $C \cup C'$.

Trivially, C_s and C_t are $s - t$ vertex-cuts, and $C_s \cup C_t \subseteq C \cup C'$. Moreover, $C_s \cap C_t \subseteq C \cap C'$, since for any $v \in C_s \cap C_t$ there is an $s - t$ path P intersecting $C \cup C'$ only in v. As $x, y \in C'$, P does not traverse edge e. Hence $v \in C \cap C'$.

Therefore,

$$|C_s| + |C_t| \leq |C| + |C'| \leq 2k - 1, \tag{9.1}$$

contradicting the fact that C_s and C_t each have size at least k.

An augmenting path proof for the directed vertex-disjoint version was given by Grünwald [1938] (= T. Gallai), and for the directed arc-disjoint version by Ford and Fulkerson [1955,1957b] — see Section 9.2. (O'Neil [1978] gave a proof similar to that of Grünwald [1938].)

More proof ideas were given by Halin [1964,1968,1989], Hajós [1967], McCuaig [1984], and Böhme, Göring, and Harant [2001].

9.2. Path packing algorithmically

A specialization of the maximum flow algorithm of Ford and Fulkerson [1955, 1957b] (to be discussed in the next chapter) yields a polynomial-time algorithm to find a maximum number of disjoint $s-t$ paths and a minimum-size $s-t$ cut.

Define for any digraph D and any path P in D:

(9.2) $D \leftarrow P :=$ the digraph arising from D by reversing the orientation of each arc occurring in P.

Note that if P is an $s-t$ path in $D=(V,A)$, then for each $U \subseteq V$ with $s \in U$, $t \notin U$, we have

$$\delta^{\mathrm{out}}_{A'}(U) = \delta^{\mathrm{out}}_{A}(U) - 1, \tag{9.3}$$

where A' is the arc set of $D \leftarrow P$.

Determine $D_0, D_1, \ldots$ as follows.

(9.4) Set $D_0 := D$. If D_k has been found and contains an $s-t$ path P, set $D_{k+1} := D_k \leftarrow P$. If D_k contains no $s-t$ path we stop.

The path P is called an *augmenting path*.

Now finding a minimum-size $s-t$ cut is easy: let U be the set of vertices reachable in the final D_k from s. Then $\delta^{\mathrm{out}}_A(U)$ is a minimum-size $s-t$ cut, by (9.3).

Also a maximum packing of $s-t$ paths can be derived. Indeed, the set B of arcs of D that are reversed in the final D_k contains k arc-disjoint $s-t$ paths in D. This can be seen as follows.

Let B_i be the set of arcs of D that are reversed in D_i, added with i parallel arcs from t to s. We show by induction on i that (V, B_i) is Eulerian. For $i=0$, this is trivial. Suppose that it has been proved for i. Let P be the $s-t$ path in D_i with $D_{i+1} = D_i \leftarrow P$. Then $(V, B_i \cup AP \cup \{(t,s)\})$ is Eulerian. Since B_{i+1} arises from $B_i \cup AP \cup \{(t,s)\}$ by deleting pairs a, a^{-1} with $a \in B_i$ and $a^{-1} \in AP$, also (V, B_{i+1}) is Eulerian.

A consequence is that k arc-disjoint $s-t$ paths in B can be found in linear time.

Since an $s-t$ path in D_k can be found in time $O(m)$, and since there are at most $|A|$ arc-disjoint $s-t$ paths, one has:

Theorem 9.2. *A maximum collection of arc-disjoint $s-t$ paths and a minimum-size $s-t$ cut can be found in time $O(m^2)$.*

Proof. See above. ∎

Similarly one has for the vertex-disjoint variant:

Theorem 9.3. *A maximum collection of internally vertex-disjoint $s-t$ paths and a minimum-size $s-t$ vertex-cut can be found in time $O(nm)$.*

Proof. Apply the reduction described after Corollary 9.1b. In this case the number of iterations is at most $|V|$. ∎

One similarly derives for a fixed number k of arc-disjoint paths:

Corollary 9.3a. *Given a digraph $D = (V, A)$, $s, t \in V$, and a natural number k, we can find k arc-disjoint $s - t$ paths (if they exist) in time $O(km)$.*

Proof. Directly from the fact that the path P can be found in time $O(m)$. ∎

9.3. Speeding up by blocking path packings

The algorithm might be speeded up by selecting, at each iteration, not just *one* path P, but several arc-disjoint paths $P_1, \ldots, P_l$ in D_i at one go, and setting

$$D_{i+1} := D_i \leftarrow P_1 \leftarrow \cdots \leftarrow P_l. \tag{9.5}$$

This might reduce the number of iterations — but of course this should be weighed against the increase in complexity of each iteration.

Such a speedup is obtained by a method of Dinits [1970] as follows. For any digraph $D = (V, A)$ and $s, t \in V$, let $\mu(D)$ denote the minimum length of an $s - t$ path in D. If no such path exists, set $\mu(D) = \infty$. If we choose the paths $P_1, \ldots, P_l$ in such a way that $\mu(D_{i+1}) > \mu(D_i)$, then the number of iterations clearly is not larger than $|V|$ (as $\mu(D_i) < |V|$ if finite).

We show that a collection $P_1, \ldots, P_l$ with the property that $\mu(D \leftarrow P_1 \leftarrow \cdots \leftarrow P_l) > \mu(D)$ indeed can be found quickly, namely in linear time.

To that end, call a collection of arc-disjoint $s - t$ paths $P_1, \ldots, P_l$ *blocking* if D contains no $s - t$ path arc-disjoint from $P_1, \ldots, P_l$. This is weaker than a maximum number of arc-disjoint paths, but a blocking collection can be found in linear time (Dinits [1970]):

Theorem 9.4. *Given an acyclic digraph $D = (V, A)$ and $s, t \in V$, a blocking collection of arc-disjoint $s - t$ paths can be found in time $O(m)$.*

Proof. With depth-first search we can find in time $O(|A'|)$ a subset A' of A and an $s - t$ path P_1 in A' such that no arc in $A' \setminus AP_1$ is contained in any $s - t$ path: scan s (cf. (6.2)) and stop as soon as t is reached; let A' be the set of arcs considered so far (as D is acyclic).

Next we find (recursively) a blocking collection $P_2, \ldots, P_k$ of arc-disjoint $s - t$ paths in the graph $D' := (V, A \setminus A')$. Then $P_1, \ldots, P_k$ is blocking in D. For suppose that D contains an $s - t$ path Q that is arc-disjoint from $P_1, \ldots, P_k$. Then $AQ \cap A' \neq \emptyset$, since $P_2, \ldots, P_k$ is blocking in D'. So AQ intersects AP_1, a contradiction. ∎

We also need the following. Let $\alpha(D)$ denote the set of arcs contained in at least one shortest $s-t$ path. Then:

Theorem 9.5. *Let $D=(V,A)$ be a digraph and let $s,t\in V$. Define $D':=(V,A\cup\alpha(D)^{-1})$. Then $\mu(D')=\mu(D)$ and $\alpha(D')=\alpha(D)$.*

Proof. It suffices to show that $\mu(D)$ and $\alpha(D)$ are invariant if we add a^{-1} to D for one arc $a\in\alpha(D)$. Suppose not. Then there is a directed $s-t$ path P in $A\cup\{a^{-1}\}$ traversing a^{-1}, of length at most $\mu(D)$. As $a\in\alpha(D)$, there is an $s-t$ path Q traversing a, of length $\mu(D)$. Hence $AP\cup AQ\setminus\{a,a^{-1}\}$ contains an $s-t$ path of length less than $\mu(D)$, a contradiction. ▮

The previous two theorems imply:

Corollary 9.5a. *Given a digraph $D=(V,A)$ and $s,t\in V$, a collection of arc-disjoint $s-t$ paths $P_1,\dots,P_l$ with $\mu(D\leftarrow P_1\leftarrow\cdots\leftarrow P_l)>\mu(D)$ can be found in time $O(m)$.*

Proof. Let $\widetilde{D}=(V,\alpha(D))$. (Note that $\alpha(D)$ can be identified in time $O(m)$ and that $(V,\alpha(D_f))$ is acyclic.) By Theorem 9.4, we can find in time $O(m)$ a blocking collection $P_1,\dots,P_l$ in $\widetilde{D}$. Define:

$$D':=(V,A\cup\alpha(D)^{-1}) \text{ and } D'':=D\leftarrow P_1\leftarrow\cdots\leftarrow P_l. \tag{9.6}$$

We show $\mu(D'')>\mu(D)$. As D'' is a subgraph of D', we have $\mu(D'')\geq\mu(D')=\mu(D)$, by Theorem 9.5. Suppose that $\mu(D'')=\mu(D')$. Then $\alpha(D'')\subseteq\alpha(D')=\alpha(D)$ (again by Theorem 9.5). Hence, as $\alpha(D'')$ contains an $s-t$ path (of length $\mu(D'')$), $\alpha(D)$ contains an $s-t$ path arc-disjoint from $P_1,\dots,P_l$. This contradicts the fact that $P_1,\dots,P_l$ is blocking in $\widetilde{D}$. ▮

This gives us the speedup in finding a maximum packing of $s-t$ paths:

Corollary 9.5b. *Given a digraph $D=(V,A)$ and $s,t\in V$, a maximum number of arc-disjoint $s-t$ paths and a minimum-size $s-t$ cut can be found in time $O(nm)$.*

Proof. Directly from Corollary 9.5a with the iterations (9.5). ▮

9.4. A sometimes better bound

Since $\mu(D_i)$ is at most $|V|$ (as long as it is finite), the number k of iterations is at most $|V|$. But Karzanov [1973a], Tarjan [1974e], and Even and Tarjan [1975] showed that an alternative, often tighter bound on k holds.

To see this, it is important to observe that, for each i, the set of arcs of D_i that are reversed in the final D_k (compared with D_i) forms a maximum number of arc-disjoint $s-t$ paths in D_i.

Theorem 9.6. *If $\mu(D_{i+1}) > \mu(D_i)$ for each $i < k$, then $k \leq 2|A|^{1/2}$. If moreover D is simple, then $k \leq 2|V|^{2/3}$.*

Proof. Let $p := \lfloor |A|^{1/2} \rfloor$. Then each $s-t$ path in D_p has length at least $p+1 \geq |A|^{1/2}$. Hence D_p contains at most $|A|/|A|^{1/2} = |A|^{1/2}$ arc-disjoint $s-t$ paths. Therefore $k - p \leq |A|^{1/2}$, and hence $k \leq 2|A|^{1/2}$.

If D is simple, let $p := \lfloor |V|^{2/3} \rfloor$. Then each $s-t$ path in D_p has length at least $p+1 \geq |V|^{2/3}$. Then D_p contains at most $|V|^{2/3}$ arc-disjoint $s-t$ paths. Indeed, let U_i denote the set of vertices at distance i from s in D_p. Then

$$\sum_{i=0}^{p} (|U_i| + |U_{i+1}|) \leq 2|V|. \tag{9.7}$$

Hence there is an $i \leq p$ with $|U_i| + |U_{i+1}| \leq 2|V|^{1/3}$. This implies $|U_i| \cdot |U_{i+1}| \leq \frac{1}{4}(|U_i| + |U_{i+1}|)^2 \leq |V|^{2/3}$. So D_p contains at most $|V|^{2/3}$ arc-disjoint $s-t$ paths. Therefore $k - p \leq |V|^{2/3}$, and hence $k \leq 2|V|^{2/3}$. ▮

This gives the following time bounds (Karzanov [1973a], Tarjan [1974e], Even and Tarjan [1975]):

Corollary 9.6a. *Given a digraph $D = (V, A)$ and $s, t \in V$, a maximum number of arc-disjoint $s-t$ paths and a minimum-size $s-t$ cut can be found in time $O(m^{3/2})$. If D is simple, the paths and the cut can be found also in time $O(n^{2/3}m)$.*

Proof. Directly from Corollary 9.5a and Theorem 9.6. ▮

(Related work was presented in Ahuja and Orlin [1991].)

9.5. Complexity of the vertex-disjoint case

If we are interested in *vertex*-disjoint paths, the results can be sharpened. Recall that if $D = (V, A)$ is a digraph and $s, t \in V$, then the problem of finding a maximum number of internally vertex-disjoint $s-t$ paths can be reduced to the arc-disjoint case by replacing each vertex $v \neq s, t$ by two vertices v', v'', while each arc with head v is redirected to v' and each arc with tail v is redirected from v''; moreover, an arc (v', v'') is added.

By Corollary 9.6a, this construction directly yields algorithms for vertex-disjoint paths with running time $O(m^{3/2})$ and $O(n^{2/3}m)$. But one can do better. Note that, with this construction, each of the digraphs D_i has the property that each vertex has indegree at most 1 or outdegree at most 1. Under this condition, the bound in Theorem 9.6 can be improved to $2|V|^{1/2}$:

Theorem 9.7. *If each vertex $v \neq s, t$ has indegree or outdegree equal to 1, and if $\mu(D_{i+1}) > \mu(D_i)$ for each $i \leq k$, then $k \leq 2|V|^{1/2}$.*

Proof. Let $p := \lceil |V|^{1/2} \rceil$. Then each $s-t$ path in D_p has length at least $p+1$. Let U_i be the set of vertices at distance i from s in D_i. Then $\sum_{i=1}^{p} |U_i| \leq |V|$. This implies that $|U_i| \leq |V|^{1/2}$ for some i. Hence D_i has at most $|V|^{1/2}$ arc-disjoint $s-t$ paths. So $k+1-p \leq |V|^{1/2}$. Hence $k \leq 2|V|^{1/2}$. ▮

This gives, similarly to Corollary 9.6a, another result of Karzanov [1973a], Tarjan [1974e], and Even and Tarjan [1975] (which can be derived also from Theorem 16.4 due to Hopcroft and Karp [1971,1973] and Karzanov [1973b], with the method of Hoffman [1960] given in Section 16.7c):

Corollary 9.7a. *Given a digraph $D = (V, A)$ and $s, t \in V$, a maximum number of internally vertex-disjoint $s-t$ paths and a minimum-size $s-t$ vertex-cut can be found in time $O(n^{1/2}m)$.*

Proof. Directly from Corollary 9.5a and Theorem 9.7. ▮

In fact one can reduce n in this bound to the minimum number $\tau(D)$ of vertices intersecting each arc of D (this bound will be used in deriving bounds for bipartite matching (Theorem 16.5)):

Theorem 9.8. *Given a digraph $D = (V, A)$ and $s, t \in V$, a maximum number of internally vertex-disjoint $s-t$ paths and a minimum-size $s-t$ vertex-cut can be found in time $O(\tau(D)^{1/2}m)$.*

Proof. Similar to Corollary 9.7a, by taking $p := \lfloor \tau(D)^{1/2} \rfloor$ in Theorem 9.7: Finding D_p takes $O(pm)$ time. Let W be a set of vertices intersecting each arc of D, of size $\tau(D)$. In D_p there are at most $2\tau(D)^{1/2}$ internally vertex-disjoint $s-t$ paths, since each $s-t$ path contains at least $p/2$ vertices in W. ▮

9.6. Further results and notes

9.6a. Complexity survey for the disjoint $s-t$ paths problem

For finding arc-disjoint $s-t$ paths we have the following survey of running time bounds ($*$ indicates an asymptotically best bound in the table):

	$O(m^2)$	Ford and Fulkerson [1955,1957b]
$*$	$O(nm)$	Dinits [1970]
$*$	$O(m^{3/2})$	Karzanov [1973a], Tarjan [1974e], Even and Tarjan [1975]
$*$	$O(n^{2/3}m)$	Karzanov [1973a], Tarjan [1974e], Even and Tarjan [1975] *D simple*

≫

continued

*	$O(k^2n)$	Nagamochi and Ibaraki [1992a] finding k arc-disjoint paths
*	$O(kn^{5/3})$	Nagamochi and Ibaraki [1992a] finding k arc-disjoint paths
*	$O(kn^2)$	Nagamochi and Ibaraki [1992a] finding k arc-disjoint paths; *D simple*
*	$O(k^{3/2}n^{3/2})$	Nagamochi and Ibaraki [1992a] finding k arc-disjoint paths; *D simple*

For *undirected* simple graphs, Goldberg and Rao [1997b,1999] gave an $O(n^{3/2}m^{1/2})$ bound, and Karger and Levine [1998] gave $(m+nk^{3/2})$ and $O(nm^{2/3}k^{1/6})$ bounds, where k is the number of paths.

For vertex-disjoint paths:

	$O(nm)$	Grünwald [1938], Ford and Fulkerson [1955, 1957b]
	$O(\sqrt{n}\,m)$	Karzanov [1973a], Tarjan [1974e], Even and Tarjan [1975]
*	$O(\sqrt{n}\,m\log_n(n^2/m))$	Feder and Motwani [1991,1995]
*	$O(k^2n)$	Nagamochi and Ibaraki [1992a] finding k vertex-disjoint paths
*	$O(kn^{3/2})$	Nagamochi and Ibaraki [1992a] finding k vertex-disjoint paths

For edge-disjoint $s-t$ paths in simple undirected *planar* graphs:

	$O(n^2\log n)$	Itai and Shiloach [1979]
	$O(n^2)$	Cheston, Probert, and Saxton [1977]
	$O(n^{3/2}\log n)$	Johnson and Venkatesan [1982]
	$O(n\log^2 n)$	Reif [1983] (minimum-size $s-t$ cut), Hassin and Johnson [1985] (edge-disjoint $s-t$ paths)
	$O(n\log n\log^* n)$	Frederickson [1983b]
	$O(n\log n)$	Frederickson [1987b]
*	$O(n)$	Weihe [1994a,1997a]

For arc-disjoint $s-t$ paths in simple *directed* planar graphs:

$O(n^{3/2}\log n)$	Johnson and Venkatesan [1982]
$O(n^{4/3}\log^2 n)$	Klein, Rao, Rauch, and Subramanian [1994], Henzinger, Klein, Rao, and Subramanian [1997]

≫

continued

	$O(n \log n)$	Weihe [1994b,1997b]
*	$O(n)$	Brandes and Wagner [1997]

For vertex-disjoint $s-t$ paths in *undirected* planar graphs:

	$O(n \log n)$	Suzuki, Akama, and Nishizeki [1990]
*	$O(n)$	Ripphausen-Lipa, Wagner, and Weihe [1993b,1997]

Orlova and Dorfman [1972] and Hadlock [1975] showed, with matching theory, that in planar undirected graphs also a *maximum*-size cut can be found in polynomial-time (Barahona [1990] gave an $O(n^{3/2} \log n)$ time bound) — see Section 29.1. Karp [1972b] showed that in general finding a maximum-size cut is NP-complete — see Section 75.1a.

9.6b. Partially disjoint paths

For any digraph $D = (V, A)$ and any $B \subseteq A$, call two paths *disjoint on* B if they have no common arc in B. One may derive from Menger's theorem a more general min-max relation for such partially disjoint paths:

Theorem 9.9. *Let $D = (V, A)$ be a digraph, $s, t \in V$, and $B \subseteq A$. Then the maximum number of $s-t$ paths such that any two are disjoint on B is equal to the minimum size of an $s-t$ cut contained in B.*

Proof. If there is no $s-t$ cut contained in B, then clearly the maximum is infinite. If B contains any $s-t$ cut, let k be its minimum size. Replace any arc $a \in A \setminus B$ by $|A|$ parallel arcs. Then by Menger's theorem there exist k arc-disjoint $s-t$ paths in the extended graph. This gives k $s-t$ paths in the original graph that are disjoint on B. ▮

The construction given in this proof can also be used algorithmically, but making $|A|$ parallel arcs takes $\Omega(m^2)$ time. However, one can prove:

Theorem 9.10. *Given a digraph $D = (V, A)$, $s, t \in V$, and $B \subseteq A$, a maximum number of $s-t$ paths such that any two are disjoint on B can be found in time $O(nm)$.*

Proof. The theorem follows from Corollary 10.11a below. ▮

9.6c. Exchange properties of disjoint paths

Disjoint paths have a number of exchange properties that imply a certain matroidal structure (cf. Section 39.4).

Let $D = (V, A)$ be a directed graph and let $X, Y \subseteq V$. Call X *linked to* Y if $|X| = |Y|$ and D has $|X|$ vertex-disjoint $X - Y$ paths. Note that the following

theorem follows directly from the algorithm for finding a maximum set of disjoint paths (since any vertex in $S \cup T$, once covered by the disjoint paths, remains covered during the further iterations):

Theorem 9.11. *Let $D = (V, A)$ be a digraph, let $S, T \subseteq V$, and suppose that $X \subseteq S$ and $Y \subseteq T$ are linked. Then there exists a maximum number of vertex-disjoint $S-T$ paths covering $X \cup Y$.*

Proof. Directly from the algorithm. ▮

The following result, due to Perfect [1968], says that for two distinct maximum packings $\mathcal{P}, \mathcal{Q}$ of $S-T$ paths there exists a maximum packing of $S-T$ paths whose starting vertices are equal to those of $\mathcal{P}$ and whose end vertices are equal to those of $\mathcal{Q}$.

Theorem 9.12. *Let $D = (V, A)$ be a digraph and $S, T \subseteq V$. Let k be the maximum number of disjoint $S - T$ paths. Let $X \subseteq S$ be linked to $Y \subseteq T$, and let $X' \subseteq S$ be linked to $Y' \subseteq T$, with $|X| = |X'| = k$. Then X is linked to Y'.*

Proof. Let C be a minimum-size vertex set intersecting each $S - T$ path. So by Menger's theorem, $|C| = |X| = |Y| = |X'| = |Y'| = k$. Let $P'_1, \ldots, P'_k$ be vertex-disjoint $X - Y$ paths. Similarly, let $P''_1, \ldots, P''_k$ be vertex-disjoint $X' - Y'$ paths. We may assume that, for each i, P'_i and P''_i have a vertex in C in common. Let P_i be the path obtained by traversing P'_i until it reaches C, after which it traverses P''_i. Then $P_1, \ldots, P_k$ are vertex-disjoint $X - Y'$ paths. ▮

The previous two theorems imply:

Corollary 9.12a. *Let $D = (V, A)$ be a digraph, let X' be linked to Y', and let X'' be linked to Y''. Then there exist X and Y with $X' \subseteq X \subseteq X' \cup X''$ and $Y'' \subseteq Y \subseteq Y' \cup Y''$ such that X is linked to Y.*

Proof. Directly from Theorems 9.11 and 9.12. ▮

Other proofs of this corollary were given by Pym [1969b,1969c], Brualdi and Pym [1971], and McDiarmid [1975b].

9.6d. Further notes

Lovász, Neumann-Lara, and Plummer [1978] proved the following on the maximum number of disjoint paths of bounded length. Let $G = (V, E)$ be an undirected graph, let s and t be two distinct and nonadjacent vertices, and let $k \geq 2$. Then the minimum number of vertices ($\neq s, t$) intersecting all $s - t$ paths of length at most k is at most $\lfloor \frac{1}{2}k \rfloor$ times the maximum number of internally vertex-disjoint $s - t$ paths each of length at most k. (A counterexample to a conjecture on this raised by Lovász, Neumann-Lara, and Plummer [1978] was given by Boyles and Exoo [1982]. Related results are given by Galil and Yu [1995].)

On the other hand, when taking lower bounds on the path lengths, Montejano and Neumann-Lara [1984] showed that, in a directed graph, the minimum number

of vertices ($\neq s,t$) intersecting all $s-t$ paths of length at least k is at most $3k-5$ times the maximum number of internally vertex-disjoint such paths. For $k=3$, the factor was improved to 3 by Hager [1986] and to 2 by Mader [1989].

Egawa, Kaneko, and Matsumoto [1991] gave a version of Menger's theorem in which vertex-disjoint and edge-disjoint are mixed: Let $G=(V,E)$ be an undirected graph, let $s,t \in V$, and $k,l \in \mathbb{Z}_+$. Then G contains l disjoint edge sets, each containing k vertex-disjoint $s-t$ paths if and only if for each $U \subseteq V \setminus \{s,t\}$ there exist $l(k-|U|)$ edge-disjoint $s-t$ paths in $G-U$. Similarly, for directed graphs. The proof is by reduction to Menger's theorem using integer flow theory.

Bienstock and Diaz [1993] showed that the problem of finding a minimum-weight subset of edges intersecting all $s-t$ cuts of size at most k is polynomial-time solvable if k is fixed, while it is NP-complete if k is not fixed.

Motwani [1989] investigated the expected running time of Dinits' disjoint paths algorithm.

Extensions of Menger's theorem to the infinite case were given by P. Erdős (cf. Kőnig [1932]), Grünwald [1938] (= T. Gallai), Dirac [1960,1963,1973], Halin [1964], McDiarmid [1975b], Podewski and Steffens [1977], Aharoni [1983a,1987], and Polat [1991].

Halin [1964], Lovász [1970b], Escalante [1972], and Polat [1976] made further studies of (the lattice of) $s-t$ cuts.

9.6e. Historical notes on Menger's theorem

The topologist Karl Menger published his theorem in an article called *Zur allgemeinen Kurventheorie* (On the general theory of curves) (Menger [1927]) in the following form:

> Satz β. *Ist K ein kompakter regulär-eindimensionaler Raum, welcher zwischen den beiden endlichen Mengen P und Q n-punktig zusammenhängend ist, dann enthält K n paarweise fremde Bögen, von denen jeder einen Punkt von P und einen Punkt von Q verbindet.*[5]

It can be formulated equivalently in terms of graphs as: Let $G=(V,E)$ be an undirected graph and let $P,Q \subseteq V$. Then the maximum number of disjoint $P-Q$ paths is equal to the minimum size of a set W of vertices such that each $P-Q$ path intersects W.

The result became known as the *n-chain theorem*. Menger's interest in this question arose from his research on what he called 'curves': a *curve* is a connected compact topological space X with the property that for each $x \in X$ and each neighbourhood N of x there exists a neighbourhood $N' \subseteq N$ of x with $\mathrm{bd}(N')$ totally disconnected. (Here bd stands for boundary; a space is *totally disconnected* if each point forms an open set.)

The curve is called *regular* if for each $x \in X$ and each neighbourhood N of x there exists a neighbourhood $N' \subseteq N$ of x with $|\mathrm{bd}(N')|$ finite. The *order* of a point $x \in X$ is equal to the minimum natural number n such that for each neighbourhood N of x there exists a neighbourhood $N' \subseteq N$ of x satisfying $|\mathrm{bd}(N')| \leq n$.

According to Menger:

[5] Theorem β. *If K is a compact regularly one-dimensional space which is n-point connected between the two finite sets P and Q, then K contains n pairwise disjoint curves, each of which connects a point in P and a point in Q.*

> Eines der wichtigsten Probleme der Kurventheorie ist die Frage nach den Beziehungen zwischen der Ordnungszahl eines Punktes der regulären Kurve K und der Anzahl der im betreffenden Punkt zusammenstossenden und sonst fremden Teilbögen von K.[6]

In fact, Menger used 'Satz β' to show that if a point in a regular curve K has order n, then there exists a topological *n-leg* with p as top; that is, K contains n arcs $P_1, \ldots, P_n$ such that $P_i \cap P_j = \{p\}$ for all i, j with $i \neq j$.

The proof idea is as follows. There exists a series $N_1 \supset N_2 \supset \cdots$ of open neighbourhoods of p such that $N_1 \cap N_2 \cap \cdots = \{p\}$ and $|\mathrm{bd}(N_i)| = n$ for all $i = 1, 2, \ldots$ and such that

$$|\mathrm{bd}(N)| \geq n \text{ for each neighbourhood } N \subseteq N_1. \tag{9.8}$$

This follows quite directly from the definition of order.

Now Menger showed that we may assume that the space $G_i := \overline{N_i} \setminus N_{i+1}$ is a (topological) graph. For each i, let $Q_i := \mathrm{bd}(N_i)$. Then (9.8) gives with Menger's theorem that there exist n disjoint paths $P_{i,1}, \ldots, P_{i,n}$ in G such that each $P_{i,j}$ runs from Q_i to Q_{i+1}. Properly connecting these paths for $i = 1, 2, \ldots$ we obtain n arcs forming the required n-leg.

It was however noticed by Kőnig [1932] that Menger's proof of 'Satz β' is incomplete. Menger applied induction on $|E|$, where E is the edge set of the graph G. Menger first claimed that one easily shows that $|E| \geq n$, and that if $|E| = n$, then G consists of n disjoint edges connecting P and Q. He stated that if $|E| > n$, then there exists a vertex $s \notin P \cup Q$, or in his words (where the 'Grad' denotes $|E|$):

> Wir nehmen also an, der irreduzibel n-punktig zusammenhängende Raum K' besitze den Grad $g(> n)$. Offenbar enthält dann K' ein punktförmiges Stück s, welches in der Menge $P + Q$ nicht enthalten ist.[7]

Indeed, as Menger showed, if such a vertex s exists one is done: If s is contained in no set W intersecting each $P - Q$ path with $|W| = n$, then we can delete s and the edges incident with s without decreasing the minimum in the theorem. If s is contained in some set W intersecting each $P - Q$ path such that $|W| = n$, then we can split G into two subgraphs G_1 and G_2 that intersect in W in such a way that $P \subseteq G_1$ and $Q \subseteq G_2$. By the induction hypothesis, there exist n disjoint $P - W$ paths in G_1 and n disjoint $W - Q$ paths in G_2. By pairwise sticking these paths together at W we obtain paths as required.

However, such a vertex s need not exist. It might be that V is the disjoint union of P and Q in such a way that each edge connects P and Q, and that there are more than n edges. In that case, G is a bipartite graph, with colour classes P and Q, and what should be shown is that G contains a matching of size n. This is a nontrivial basis of the proof.

At the meeting of 26 March 1931 of the Eötvös Loránd Matematikai és Fizikai Társulat (Loránd Eötvös Mathematical and Physical Society) in Budapest, Kőnig [1931] presented a new result that formed the missing basis for Menger's theorem:

---

[6] One of the most important problems of the theory of curves is the question of the relations between the order of a point of a regular curve K and the number of subarcs of K meeting in that point and disjoint elsewhere.

[7] Thus we assume that the irreducibly n-point-connected space K' has degree $g(> n)$. Obviously, in that case K' contains a point-shaped piece s, that is not contained in the set $P + Q$.

> Páros körüljárású graphban az éleket kimerítő szögpontok minimális száma megegyezik a páronként közös végpontot nem tartalmazó élek maximális számával.[8]

In other words, in a bipartite graph $G = (V, E)$, the maximum size of a matching is equal to the minimum number of vertices needed to cover all edges, which is Kőnig's matching theorem — see Theorem 16.2.

Kőnig did not mention in his 1931 paper that this result provided the missing element in Menger's proof, although he finishes with:

> Megemlítjük végül, hogy eredményeink szorosan összefüggnek FROBENIUSnak determinánsokra és MENGERnek graphokra vonatkozó némely vizsgálatával. E kapcsolatokra másutt fogunk kiterjeszkedni.[9]

'Elsewhere' is Kőnig [1932], in which paper he gave a full proof of Menger's theorem. The hole in Menger's original proof is discussed in a footnote:

> Der Beweis von MENGER enthält eine Lücke, da es vorausgesetzt wird (S. 102, Zeile 3–4) daß „K' ein punktförmiges Stück s enthält, welches in der Menge $P+Q$ nicht enthalten ist", während es recht wohl möglich ist, daß —mit der hier gewählten Bezeichnungsweise ausgedrückt—jeder Knotenpunkt von G zu $H_1 + H_2$ gehört. Dieser—keineswegs einfacher—Fall wurde in unserer Darstellung durch den Beweis des Satzes 13 erledigt. Die weiteren—hier folgenden—Überlegungen, die uns zum Mengerschen Satz führen werden, stimmen im Wesentlichen mit dem—sehr kurz gefaßten—Beweis von MENGER überein. In Anbetracht der Allgemeinheit und Wichtigkeit des Mengerschen Satzes wird im Folgenden auch dieser Teil ganz ausführlich und den Forderungen der *rein-kombinatorischen* Graphentheorie entsprechend dargestellt.
> [Zusatz bei der Korrektur, 10.V.1933] Herr MENGER hat die Freundlichkeit gehabt—nachdem ich ihm die Korrektur meiner vorliegenden Arbeit zugeschickt habe—mir mitzuteilen, daß ihm die oben beanstandete Lücke seines Beweises schon bekannt war, daß jedoch sein vor Kurzem erschienenes Buch *Kurventheorie* (Leipzig, 1932) einen vollkommen lückenlosen und rein kombinatorischen Beweis des Mengerschen Satzes (des "n-Kettensatzes") enthält. Mir blieb dieser Beweis bis jetzt unbekannt.[10]

---

[8] In an even circuit graph, the minimal number of vertices that exhaust the edges agrees with the maximal number of edges that pairwise do not contain any common end point.

[9] We finally mention that our results are closely connected to some investigations of FROBENIUS on determinants and of MENGER on graphs. We will enlarge on these connections elsewhere.

[10] The proof of MENGER contains a hole, as it is assumed (page 102, line 3–4) that 'K' contains a point-shaped piece s that is not contained in the set $P + Q$', while it is quite well possible that—expressed in the notation chosen here—every node of G belongs to $H_1 + H_2$. This—by no means simple—case is settled in our presentation by the proof of Theorem 13. The further arguments following here that will lead us to Menger's theorem, agree essentially with the—very briefly couched—proof of MENGER. In view of the generality and the importance of Menger's theorem, also this part is exhibited in the following very extensively and conforming to the progress of the *purely combinatorial* graph theory.

[Added in proof, 10 May 1933] Mr. MENGER has had the kindness—after I have sent him the galley proofs of my present work—to inform me that the hole in his proof objected above, was known to him already, but that his, recently appeared, book *Kurventheorie* (Leipzig, 1932) contains a completely holeless and purely combinatorial proof of the Menger theorem (the 'n-chain theorem'). As yet, this proof remained unknown to me.

The book *Kurventheorie* (Curve Theory) mentioned is Menger [1932b], which contains a complete proof of Menger's theorem. Menger did not refer to any hole in his original proof, but remarked:

> Über den n-Kettensatz für Graphen und die im vorangehenden zum Beweise verwendete Methode vgl. Menger (Fund. Math. *10*, 1927, S. 101 f.). Die obige detaillierte Ausarbeitung und Darstellung stammt von Nöbeling.[11]

In his book *Theorie der endlichen und unendlichen Graphen* (Theory of finite and infinite graphs), Kőnig [1936] called his theorem *ein wichtiger Satz* (an important theorem), and he emphasized the chronological order of the proofs of Menger's theorem and of Kőnig's theorem (which is implied by Menger's theorem):

> Ich habe diesen Satz 1931 ausgesprochen und bewiesen, s. Kőnig [9 und 11]. 1932 erschien dann der erste lückenlose Beweis des Mengerschen Graphensatzes, von dem in §4 die Rede sein wird und welcher als eine Verallgemeinerung dieses Satzes 13 (falls dieser *nur für endliche* Graphen formuliert wird) angesehen werden kann.[12]

([9 und 11] are Kőnig [1931] and Kőnig [1932].)

In his reminiscences on the origin of the n-arc theorem, Menger [1981] wrote:

> In the spring of 1930, I came through Budapest and met there a galaxy of Hungarian mathematicians. In particular, I enjoyed making the acquaintance of Dénes Kőnig, for I greatly admired the work on set theory of his father, the late Julius Kőnig—to this day one of the most significant contributions to the continuum problem—and I had read with interest some of Dénes' papers. Kőnig told me that he was about to finish a book that would include all that was known about graphs. I assured him that such a book would fill a great need; and I brought up my n-Arc Theorem which, having been published as a lemma in a curve-theoretical paper, had not yet come to his attention. Kőnig was greatly interested, but did not believe that the theorem was correct. "This evening," he said to me in parting, "I won't go to sleep before having constructed a counterexample." When we met again the next day he greeted me with the words, "A sleepless night!" and asked me to sketch my proof for him. He then said that he would add to his book a final section devoted to my theorem. This he did; and it is largely thanks to Kőnig's valuable book that the n-Arc Theorem has become widely known among graph theorists.

Related work

In a paper presented 7 May 1927 to the American Mathematical Society, Rutt [1927, 1929] gave the following variant of Menger's theorem, suggested by J.R. Kline. Let $G = (V, E)$ be a planar graph and let $s, t \in V$. Then the maximum number of internally vertex-disjoint $s-t$ paths is equal to the minimum number of vertices in $V \setminus \{s, t\}$ intersecting each $s-t$ path.

---

[11] On the n-chain theorem for graphs and the method used in the foregoing for the proof, compare Menger (Fund. Math. *10*, 1927, p. 101 ff.). The detailed elaboration and explanation above originates from Nöbeling.

[12] I have enunciated and proved this theorem in 1931, see Kőnig [9 and 11]. Next, in 1932, the first holeless proof of the Menger theorem appeared, of which will be spoken in §4 and which can be considered as a generalization of this Theorem 13 (in case this is formulated *only for finite* graphs).

In fact, the theorem follows quite easily from Menger's version of his theorem by deleting s and t and taking for P and Q the sets of neighbours of s and t respectively. (Rutt referred to Menger and gave an independent proof of the theorem.)

This construction was also observed by Knaster [1930] who showed that Menger's theorem would follow from Rutt's theorem for general (not necessarily planar) graphs. A similar theorem was published by Nöbeling [1932], using Menger's result.

A result implied by Menger's theorem was presented by Whitney [1932a] on 28 February 1931 to the American Mathematical Society: a graph is n-connected if and only if any two vertices are connected by n internally disjoint paths. While referring to the papers of Menger and Rutt, Whitney gave a direct proof. Kőnig [1932] remarked on Whitney's theorem:

> Das interessante Hauptresultat einer Abhandlung von WHITNEY [10], nämlich sein Theorem 7, folgt unmittelbar aus diesem Mengerschen Satz, jedoch, wie es scheint, nicht umgekehrt.[13]

In the 1930s, other proofs of Menger's theorem were given by Hajós [1934] and Grünwald [1938] (= T. Gallai) — the latter paper gives an essentially algorithmic proof based on augmenting paths, and it observes, in a footnote, that the theorem also holds for directed graphs:

> Die ganze Betrachtung lässt sich auch bei orientierten Graphen durchführen und liefert dann eine Verallgemeinerung des Mengerschen Satzes.[14]

The arc-disjoint version of Menger's theorem seems to be first shown by Ford and Fulkerson [1954,1956b] and Kotzig [1956] for undirected graphs and by Dantzig and Fulkerson [1955,1956] for directed graphs.

In his dissertation for the degree of Academical Doctor, Kotzig [1956] defined, for any undirected graph G and vertices u, v of G, $\sigma_G(u, v)$ to be the minimum size of a $u - v$ cut. Then he states:

> Veta 35. Nech G je l'ubovol'ný graf obsahujúci uzly $u \neq v$, o ktorých platí $\sigma_G(u, v) = k > 0$, potom existuje systém ciest $\{C_1, C_2, \ldots, C_k\}$ taký že každa cesta spojuje uzly u, v a žiadne dve rôzne cesty systému nemajú spoločnej hrany. Takýto systém ciest v G existuje len vtedy, keď je $\sigma_G(u, v) \geq k$.[15]

In Theorems 33 and 34 of the dissertation, methods are developed for the proof of Theorem 35. The method is to consider a minimal graph satisfying the cut condition, and next to orient it so as to make a directed graph in which each vertex w (except u and v) has indegree = outdegree, while u has outdegree k and indegree 0. This then yields the required paths.

Although the dissertation has several references to Kőnig's book, which contains the undirected vertex-disjoint version of Menger's theorem, Kotzig did not link his

---

[13] The interesting main result of an article of WHITNEY [10], namely his Theorem 7, follows immediately from this theorem of Menger, however, as it seems, not conversely.

[14] The whole argument lets itself carry out also for oriented graphs and then yields a generalization of Menger's theorem.

[15] Theorem 35. Let G be an arbitrary graph containing vertices $u \neq v$ for which $\sigma_G(u, v) = k > 0$, then there exists a system of paths $\{C_1, C_2, \ldots, C_k\}$ such that each path connects vertices u, v and no two distinct paths have an edge in common. Such a system of paths in G exists only if $\sigma_G(u, v) \geq k$.

result to that of Menger. (Kotzig [1961a] gave a proof of the directed arc-disjoint version of Menger's theorem, without reference to Menger.)

We refer to the historical notes on maximum flows in Section 10.8e for further notes on the work of Dantzig, Ford, and Fulkerson on Menger's theorem.

Chapter 10

Maximum flow

An $s-t$ flow is defined as a nonnegative real-valued function on the arcs of a digraph satisfying the 'flow conservation law' at each vertex $\neq s,t$. In this chapter we consider the problem of finding a maximum-value flow subject to a given capacity function. Basic results are Ford and Fulkerson's max-flow min-cut theorem and their augmenting path algorithm to find a maximum flow.

Each $s-t$ flow is a nonnegative linear combination of incidence vectors of $s-t$ paths and of directed circuits. Moreover, an integer flow is an integer such combination. This makes flows tightly connected to disjoint paths. Thus, maximum integer flow corresponds to a capacitated version of a maximum packing of disjoint paths, and the max-flow min-cut theorem is equivalent to Menger's theorem on disjoint paths.

Distinguishing characteristic of flow is however that it is not described by a combination of paths but by a function on the arcs. This promotes the algorithmic tractability.

In this chapter, graphs can be assumed to be simple.

10.1. Flows: concepts

Let $D=(V,A)$ be a digraph and let $s,t\in V$. A function $f:A\to\mathbb{R}$ is called a *flow from s to t*, or an *$s-t$ flow*, if:

$$\begin{array}{lll} \text{(i)} & f(a)\geq 0 & \text{for each } a\in A,\\ \text{(ii)} & f(\delta^{\text{out}}(v))=f(\delta^{\text{in}}(v)) & \text{for each } v\in V\setminus\{s,t\}. \end{array} \tag{10.1}$$

Condition (10.1)(ii) is called the *flow conservation law*: the amount of flow entering a vertex $v\neq s,t$ should be equal to the amount of flow leaving v.

The *value* of an $s-t$ flow f is, by definition:

$$\text{value}(f):=f(\delta^{\text{out}}(s))-f(\delta^{\text{in}}(s)). \tag{10.2}$$

So the value is the net amount of flow leaving s. This is equal to the net amount of flow entering t (this follows from (10.5) below).

Let $c:A\to\mathbb{R}_+$ be a *capacity* function. We say that a flow f is *under c* (or *subject to c*) if

$$f(a)\leq c(a) \text{ for each } a\in A. \tag{10.3}$$

A *maximum $s-t$ flow*, or just a *maximum flow*, is an $s-t$ flow under c, of maximum value. The *maximum flow problem* is to find a maximum flow.

By compactness and continuity, a maximum flow exists. It will follow from the results in this chapter (in particular, Theorem 10.4), that if the capacities are rational, then there exists a rational-valued maximum flow.

It will be convenient to make an observation on general functions $f : A \to \mathbb{R}$. For any $f : A \to \mathbb{R}$, the *excess function* is the function $\text{excess}_f : \mathcal{P}(V) \to \mathbb{R}$ defined by

$$\text{excess}_f(U) := f(\delta^{\text{in}}(U)) - f(\delta^{\text{out}}(U)) \tag{10.4}$$

for $U \subseteq V$. Set $\text{excess}_f(v) := \text{excess}_f(\{v\})$ for $v \in V$. Then:

Theorem 10.1. *Let $D = (V, A)$ be a digraph, let $f : A \to \mathbb{R}$, and let $U \subseteq V$. Then:*

$$\text{excess}_f(U) = \sum_{v \in U} \text{excess}_f(v). \tag{10.5}$$

Proof. This follows directly by counting, for each $a \in A$, the multiplicity of $f(a)$ at both sides of (10.5). ▮

To formulate a min-max relation, define the *capacity* of a cut $\delta^{\text{out}}(U)$ by $c(\delta^{\text{out}}(U))$. Then:

Theorem 10.2. *Let $D = (V, A)$ be a digraph, $s, t \in V$, and $c : A \to \mathbb{R}_+$. Then*

$$\text{value}(f) \leq c(\delta^{\text{out}}(U)), \tag{10.6}$$

for each $s-t$ flow $f \leq c$ and each $s-t$ cut $\delta^{\text{out}}(U)$. Equality holds in (10.6) if and only if $f(a) = c(a)$ for each $a \in \delta^{\text{out}}(U)$ and $f(a) = 0$ for each $a \in \delta^{\text{in}}(U)$.

Proof. Using (10.5) we have

$$\begin{aligned} \text{value}(f) &= -\text{excess}_f(s) = -\text{excess}_f(U) = f(\delta^{\text{out}}(U)) - f(\delta^{\text{in}}(U)) \\ &\leq c(\delta^{\text{out}}(U)), \end{aligned} \tag{10.7}$$

with equality if and only if $f(\delta^{\text{out}}(U)) = c(\delta^{\text{out}}(U))$ and $f(\delta^{\text{in}}(U)) = 0$. ▮

Finally, we consider a concept that turns out to be important in studying flows. Let $D = (V, A)$ be a digraph. For each $a = (u, v) \in A$, let $a^{-1} := (v, u)$. Define

$$A^{-1} := \{a^{-1} \mid a \in A\}. \tag{10.8}$$

Fix a lower bound function $d : A \to \mathbb{R}$ and an upper bound function $c : A \to \mathbb{R}$. Then for any $f : A \to \mathbb{R}$ satisfying $d \leq f \leq c$ we define

$$A_f := \{a \mid a \in A, f(a) < c(a)\} \cup \{a^{-1} \mid a \in A, f(a) > d(a)\}. \tag{10.9}$$

Clearly, A_f depends not only on f, but also on D, d, and c, but in the applications below D, d, and c are fixed, while f is variable. The digraph

$$D_f = (V, A_f) \tag{10.10}$$

is called the *residual graph* of f. So D_f is a subgraph of the directed graph $(V, A \cup A^{-1})$. As we shall see, the residual graph is very useful in studying flows and circulations, both theoretically and algorithmically.

In the context of flows we take $d = \mathbf{0}$. We observe:

Corollary 10.2a. *Let f be an $s-t$ flow in D with $f \leq c$. Suppose that D_f has no $s-t$ path. Define U as the set of vertices reachable in D_f from s. Then* value$(f) = c(\delta_A^{\rm out}(U))$. *In particular, f has maximum value.*

Proof. We apply Theorem 10.2. For each $a \in \delta_A^{\rm out}(U)$, one has $a \notin A_f$, and hence $f(a) = c(a)$. Similarly, for each $a \in \delta_A^{\rm in}(U)$ one has $a^{-1} \notin A_f$, and hence $f(a) = 0$. So value$(f) = c(\delta_A^{\rm out}(U))$ and f has maximum value by Theorem 10.2. ∎

Any directed path P in D_f gives an undirected path in $D = (V, A)$. Define $\chi^P \in \mathbb{R}^A$ by:

$$\chi^P(a) := \begin{cases} 1 & \text{if } P \text{ traverses } a, \\ -1 & \text{if } P \text{ traverses } a^{-1}, \\ 0 & \text{if } P \text{ traverses neither } a \text{ nor } a^{-1}, \end{cases} \tag{10.11}$$

for $a \in A$.

10.2. The max-flow min-cut theorem

The following theorem was proved by Ford and Fulkerson [1954,1956b] for the undirected case and by Dantzig and Fulkerson [1955,1956] for the directed case. (According to Robacker [1955a], the max-flow min-cut theorem was conjectured first by D.R. Fulkerson.)

Theorem 10.3 (max-flow min-cut theorem)**.** *Let $D = (V, A)$ be a digraph, let $s, t \in V$, and let $c : A \to \mathbb{R}_+$. Then the maximum value of an $s-t$ flow subject to c is equal to the minimum capacity of an $s-t$ cut.*

Proof. Let f be an $s-t$ flow subject to c, of maximum value. By Theorem 10.2, it suffices to show that there is an $s-t$ cut $\delta^{\rm out}(U)$ with capacity equal to value(f).

Consider the residual graph D_f (for lower bound $d := \mathbf{0}$). Suppose that it contains an $s-t$ path P. Then $f' := f + \varepsilon\chi^P$ is again an $s-t$ flow subject to c, for $\varepsilon > 0$ small enough, with value(f') = value$(f) + \varepsilon$. This contradicts the maximality of value(f).

So D_f contains no $s-t$ path. Let U be the set of vertices reachable in D_f from s. Then $\text{value}(f) = c(\delta^{\text{out}}(U))$ by Corollary 10.2a. ∎

This 'constructive' proof method is implied by the algorithm of Ford and Fulkerson [1955,1957b], to be discussed below.

Moreover, one has (Dantzig and Fulkerson [1955,1956])[16]:

Corollary 10.3a (integrity theorem)**.** *If c is integer, there exists an integer maximum flow.*

Proof. Directly from the proof of the max-flow min-cut theorem, where we can take $\varepsilon = 1$. ∎

10.3. Paths and flows

The following observation gives an important link between flows at one side and paths at the other side.

Let $D = (V, A)$ be a digraph, let $s, t \in V$, and let $f : A \to \mathbb{R}_+$ be an $s-t$ flow. Then f is a nonnegative linear combination of at most $|A|$ vectors χ^P, where P is a directed $s-t$ path or a directed circuit. If f is integer, we can take the linear combination integer-scalared.

Conversely, if $P_1, \ldots, P_k$ are $s-t$ paths in D, then $f := \chi^{AP_1} + \cdots + \chi^{AP_k}$ is an integer $s-t$ flow of value k.

With this observation, Corollary 10.3a implies the arc-disjoint version of Menger's theorem (Corollary 9.1b). Conversely, Corollary 10.3a (the integrity theorem) can be derived from the arc-disjoint version of Menger's theorem by replacing each arc a by $c(a)$ parallel arcs.

10.4. Finding a maximum flow

The proof idea of the max-flow min-cut theorem can also be used algorithmically to find a maximum $s-t$ flow, as was shown by Ford and Fulkerson [1955,1957b]. Let $D = (V, A)$ be a digraph and $s, t \in V$ and let $c : A \to \mathbb{Q}_+$ be a 'capacity' function.

Initially set $f := \mathbf{0}$. Next apply the following *flow-augmenting algorithm* iteratively:

(10.12) let P be a directed $s-t$ path in D_f and reset $f := f + \varepsilon\chi^P$, where ε is as large as possible so as to maintain $\mathbf{0} \le f \le c$.

If no such path exists, the flow f is maximum, by Corollary 10.2a.

The path P is called a *flow-augmenting path* or an *f-augmenting path*, or just an *augmenting path*.

[16] The name 'integrity theorem' was used by Ford and Fulkerson [1962].

As for termination, we have:

Theorem 10.4. *If all capacities $c(a)$ are rational, the algorithm terminates.*

Proof. If all capacities are rational, there exists a natural number K such that $Kc(a)$ is an integer for each $a \in A$. (We can take for K the l.c.m. of the denominators of the $c(a)$.)

Then in the flow-augmenting iterations, every $f_i(a)$ and every ε is a multiple of $1/K$. So at each iteration, the flow value increases by at least $1/K$. Since the flow value cannot exceed $c(\delta^{\text{out}}(\{s\}))$, there are only finitely many iterations. ∎

If we delete the rationality condition, this theorem is not maintained — see Section 10.4a. On the other hand, in Section 10.5 we shall see that if we always choose a *shortest possible* flow-augmenting path, then the algorithm terminates in a polynomially bounded number of iterations, regardless whether the capacities are rational or not.

10.4a. Nontermination for irrational capacities

Ford and Fulkerson [1962] showed that Theorem 10.4 is not maintained if we allow arbitrary real-valued capacities. The example is as follows.

Let $D = (V, A)$ be the complete directed graph on 8 vertices, with $s, t \in V$. Let $A_0 = \{a_1, a_2, a_3\}$ consist of three disjoint arcs of D, each disjoint from s and t. Let r be the positive root of $r^2 + r - 1 = 0$; that is, $r = (-1 + \sqrt{5})/2 < 1$. Define a capacity function c on A by

$$c(a_1) := 1,\ c(a_2) := 1,\ c(a_3) := r, \tag{10.13}$$

and $c(a)$ at least

$$q := \frac{1}{1-r} = 1 + r + r^2 + \cdots \tag{10.14}$$

for each $a \in A \setminus A_0$. Apply the flow-augmenting algorithm iteratively as follows.

In step 0, choose, as flow-augmenting path, the $s-t$ path of length 3 traversing a_1. After this step, the flow f satisfies, for $k = 1$:

(10.15)
(i) f has value $1 + r + r^2 + \cdots + r^{k-1}$,
(ii) $\{c(a) - f(a) \mid a \in A_0\} = \{0, r^{k-1}, r^k\}$,
(iii) $f(a) \le 1 + r + r^2 + \cdots + r^{k-1}$ for each $a \in A$.

We describe the further steps. In each step k, for $k \ge 1$, the input flow f satisfies (10.15). Choose a flow-augmenting path P in D_f that contains the arc $a \in A_0$ satisfying $c(a) - f(a) = 0$ in backward direction, and the other two arcs in A_0 in forward direction; all other arcs of P are arcs of D in forward direction. Since $r^k < r^{k-1}$, and since $(1 + r + \cdots + r^{k-1}) + r^k < q$, the flow augmentation increases the flow value by r^k. Since $r^{k-1} - r^k = r^{k+1}$, the new flow satisfies (10.15) with k replaced by $k + 1$.

We can keep iterating this, making the flow value converge to $1+r+r^2+r^3+\cdots = q$. So the algorithm does not terminate, and the flow value does not converge to the optimum value, since, trivially, the maximum flow value is more than q.

(Zwick [1995] gave the smallest directed graph (with 6 vertices and 8 arcs) for which the algorithm (with irrational capacities) need not terminate.)

10.5. A strongly polynomial bound on the number of iterations

We saw in Theorem 10.4 that the number of iterations in the maximum flow algorithm is finite, if all capacities are rational. But if we choose as our flow-augmenting path P in the auxiliary graph D_f an *arbitrary* $s-t$ path, the number of iterations yet can get quite large. For instance, in the graph in Figure 10.1 the number of iterations, at an unfavourable choice of paths, can become $2 \cdot 10^k$, so exponential in the size of the input data (which is $O(k)$).

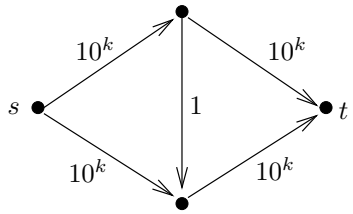


Figure 10.1

However, if we choose always a *shortest* $s-t$ path in D_f as our flow-augmenting path P (that is, with a minimum number of arcs), then the number of iterations is at most $|V| \cdot |A|$ (also if capacities are irrational). This was shown by Dinits [1970] and Edmonds and Karp [1972]. (The latter remark that this refinement 'is so simple that it is likely to be incorporated innocently into a computer implementation.')

To see this bound on the number of iterations, let again, for any digraph $D = (V, A)$ and $s, t \in V$, $\mu(D)$ denote the minimum length of an $s-t$ path. Moreover, let $\alpha(D)$ denote the set of arcs contained in at least one shortest $s-t$ path. Recall that by Theorem 9.5:

(10.16) for $D' := (V, A \cup \alpha(D)^{-1})$, one has $\mu(D') = \mu(D)$ and $\alpha(D') = \alpha(D)$.

This implies the result of Dinits [1970] and Edmonds and Karp [1972]:

Theorem 10.5. *If we choose in each iteration a shortest $s-t$ path in D_f as flow-augmenting path, the number of iterations is at most $|V| \cdot |A|$.*

Proof. If we augment flow f along a shortest $s-t$ path P in D_f, obtaining flow f', then $D_{f'}$ is a subgraph of $D' := (V, A_f \cup \alpha(D_f)^{-1})$. Hence $\mu(D_{f'}) \geq \mu(D') = \mu(D_f)$ (by (10.16)). Moreover, if $\mu(D_{f'}) = \mu(D_f)$, then $\alpha(D_{f'}) \subseteq \alpha(D') = \alpha(D_f)$ (again by (10.16)). As at least one arc in P belongs to D_f but not to $D_{f'}$, we have a strict inclusion. Since $\mu(D_f)$ increases at most $|V|$ times and, as long as $\mu(D_f)$ does not change, $\alpha(D_f)$ decreases at most $|A|$ times, we have the theorem. ∎

Since a shortest path can be found in time $O(m)$ (Theorem 6.3), this gives:

Corollary 10.5a. *A maximum flow can be found in time $O(nm^2)$.*

Proof. Directly from Theorem 10.5. ∎

10.6. Dinits' $O(n^2m)$ algorithm

Dinits [1970] observed that one can speed up the maximum flow algorithm, by not augmenting simply along *paths* in D_f, but along *flows* in D_f. The approach is similar to that of Section 9.3 for path packing.

To describe this, define a capacity function c_f on A_f by, for each $a \in A$:

$$\begin{array}{ll} c_f(a) := c(a) - f(a) & \text{if } a \in A_f \text{ and} \\ c_f(a^{-1}) := f(a) & \text{if } a^{-1} \in A_f. \end{array} \tag{10.17}$$

Then for any flow g in D_f subject to c_f,

$$f'(a) := f(a) + g(a) - g(a^{-1}) \tag{10.18}$$

gives a flow f' in D subject to c. (We define $g(a)$ or $g(a^{-1})$ to be 0 if a or a^{-1} does not belong to A_f.)

Now we shall see that, given a flow f in D, one can find in time $O(m)$ a flow g in D_f such that the flow f' arising by (10.18) satisfies $\mu(D_{f'}) > \mu(D_f)$. It implies that there are at most n iterations.

The basis of the method is the concept of 'blocking flow'. An $s-t$ flow f is called *blocking* if for each $s-t$ flow f' with $f \leq f' \leq c$ one has $f' = f$.

Theorem 10.6. *Given an acyclic graph $D = (V, A)$, $s, t \in V$, and a capacity function $c : A \to \mathbb{Q}_+$, a blocking $s-t$ flow can be found in time $O(nm)$.*

Proof. By depth-first search we can find, in time $O(|A'|)$, a subset A' of A and an $s-t$ path P in A' such that no arc in $A' \setminus AP$ is contained in any $s-t$ path: just scan s (cf. (6.2)) until t is reached; then A' is the set of arcs considered so far.

Let f be the maximum flow that can be sent along P, and reset $c := c - f$. Delete all arcs in $A' \setminus AP$ and all arcs a with $c(a) = 0$, and recursively find

a blocking $s-t$ flow f' in the new network. Then $f'+f$ is a blocking $s-t$ flow for the original data, as is easily checked.

The running time of the iteration is $O(n+t)$, where t is the number of arcs deleted. Since there are at most $|A|$ iterations and since at most $|A|$ arcs can be deleted, we have the required running time bound. ∎

Hence we have an improvement on the running time for finding a maximum flow:

Corollary 10.6a. *A maximum flow can be found in time $O(n^2m)$.*

Proof. It suffices to describe an $O(nm)$ method to find, for given flow f, a flow f' with $\mu(D_{f'}) > \mu(D_f)$.

Find a blocking flow g in $(V, \alpha(D_f))$. (Note that $\alpha(D_f)$ can be determined in $O(m)$ time.) Let $f'(a) := f(a)+g(a)-g(a^{-1})$, taking values 0 if undefined. Then $D_{f'}$ is a subgraph of $D' := (V, A_f \cup \alpha(D_f)^{-1})$, and hence by (10.16), $\mu(D_{f'}) \geq \mu(D') = \mu(D_f)$. If $\mu(D_{f'}) = \mu(D_f)$, $D_{f'}$ has a path P of length $\mu(D_f)$, which (again (10.16)) should also be a path in $\alpha(D_f)$. But then g could have been increased along this path, contradicting the fact that g is blocking in D_f. ∎

10.6a. Karzanov's $O(n^3)$ algorithm

Karzanov [1974] gave a faster algorithm to find a blocking flow, thus speeding up the maximum flow algorithm. We give the short proof of Malhotra, Kumar, and Maheshwari [1978] (see also Cherkasskiĭ [1979] and Tarjan [1984]).

Theorem 10.7. *Given an acyclic digraph $D = (V, A)$, $s, t \in V$, and a capacity function $c : A \to \mathbb{Q}_+$, a blocking $s-t$ flow can be found in time $O(n^2)$.*

Proof. First order the vertices reachable from s as $s = v_1, v_2, \ldots, v_{n-1}, v_n$ *topologically*; that is, if $(v_i, v_j) \in A$, then $i < j$. This can be done in time $O(m)$ (see Corollary 6.5b).

We describe the algorithm recursively. Consider the minimum of the values $c(\delta^{\text{in}}(v))$ for all $v \in V \setminus \{s\}$ and $c(\delta^{\text{out}}(v))$ for all $v \in V \setminus \{t\}$. Let the minimum be attained by v_i and $c(\delta^{\text{out}}(v_i))$ (without loss of generality). Define $f(a) := c(a)$ for each $a \in \delta^{\text{out}}(v_i)$ and $f(a) := 0$ for all other a.

Next for $j = i+1, \ldots, n-1$, redefine $f(a)$ for each $a \in \delta^{\text{out}}(v_j)$ such that $f(a) \leq c(a)$ and such that $f(\delta^{\text{out}}(v_j)) = f(\delta^{\text{in}}(v_j))$. By the minimality of $c(\delta^{\text{out}}(v_i))$, we can always do this, as initially $f(\delta^{\text{in}}(v_j)) \leq c(\delta^{\text{out}}(v_i)) \leq c(\delta^{\text{out}}(v_j))$. We do this in such a way that finally $f(a) \in \{0, c(a)\}$ for all but at most one a in $\delta^{\text{out}}(v_j)$.

After that, for $j = i, i-1, \ldots, 2$, redefine similarly $f(a)$ for $a \in \delta^{\text{in}}(v_j)$ such that $f(a) \leq c(a)$, $f(\delta^{\text{in}}(v_j)) = f(\delta^{\text{out}}(v_j))$, and $f(a) \in \{0, c(a)\}$ for all but at most one a in $\delta^{\text{in}}(v_j)$.

If $v_i \in \{s, t\}$ we stop, and f is a blocking flow. If $v_i \notin \{s, t\}$, set $c'(a) := c(a) - f(a)$ for each $a \in A$, and delete all arcs a with $c'(a) = 0$ and delete v_i and all arcs incident with v_i, thus obtaining the directed graph $D' = (V', A')$. Obtain

(recursively) a blocking flow f' in D' subject to the capacity function c'. Define $f''(a) := f(a) + f'(a)$ for $a \in A'$ and $f''(a) = f(a)$ for $a \in A \setminus A'$. Then f'' is a blocking flow in D.

This describes the algorithm. The correctness can be seen as follows. If $v_i \in \{s,t\}$ the correctness is immediate. If $v_i \notin \{s,t\}$, suppose that f'' is not a blocking flow in D, and let P be an $s-t$ path in D with $f''(a) < c(a)$ for each arc a in P. Then each arc of P belongs to A', since $f''(a) = f(a) = c(a)$ for each $a \in A \setminus (A' \cup \delta^{\text{in}}(v_i))$. So for each arc a of P one has $c'(a) = c(a) - f(a) > f''(a) - f(a) = f'(a)$. This contradicts the fact that f' is a blocking flow in D'.

The running time of the algorithm is $O(n^2)$, since the running time of the iteration is $O(n + |A \setminus A'|)$, and since there are at most $|V|$ iterations. ∎

Theorem 10.7 improves the running time for finding a maximum flow as follows:

Corollary 10.7a. *A maximum flow can be found in time $O(n^3)$.*

Proof. Similar to the proof of Corollary 10.6a. ∎

Sharper blocking flow algorithms were found by Cherkasskiĭ [1977a] ($O(n\sqrt{m})$), Galil [1978,1980a] ($O((nm)^{2/3})$), Shiloach [1978] and Galil and Naamad [1979,1980] ($O(m \log^2 n)$), Sleator [1980] and Sleator and Tarjan [1981,1983a] ($O(m \log n)$), and Goldberg and Tarjan [1990] ($O(m \log(n^2/m))$), each yielding a maximum flow algorithm with running time bound a factor of n higher.

An alternative approach finding a maximum flow in time $O(nm \log(n^2/m))$, based on the 'push-relabel' method, was developed by Goldberg [1985,1987] and Goldberg and Tarjan [1986,1988a], and is described in the following section.

10.7. Goldberg's push-relabel method

The algorithms for the maximum flow problem described above are all based on flow augmentation. The basis is updating a flow f until D_f has no $s-t$ path. Goldberg [1985,1987] and Goldberg and Tarjan [1986,1988a] proposed a different, in a sense dual, method, the 'push-relabel' method: update a 'preflow' f, maintaining the property that D_f has no $s-t$ path, until f is a flow. (Augmenting flow methods are 'primal' as they maintain feasibility of the primal linear program, while the push-relabel method maintains feasibility of the dual linear program.)

Let $D = (V, A)$ be a digraph, $s, t \in V$, and $c : A \to \mathbb{Q}_+$. A function $f : A \to \mathbb{Q}$ is called an $s-t$ *preflow*, or just a *preflow*, if

(10.19) (i) $0 \le f(a) \le c(a)$ for each $a \in A$,
(ii) $\text{excess}_f(v) \ge 0$ for each vertex $v \ne s$.

(Preflows were introduced by Karzanov [1974]. excess_f was defined in Section 10.1.)

Condition (ii) says that at each vertex $v \ne s$, the outgoing preflow does not exceed the ingoing preflow. For any preflow f, call a vertex v *active* if

$v \neq t$ and $\text{excess}_f(v) > 0$. So f is an $s-t$ flow if and only if there are no active vertices.

The *push-relabel method* consists of keeping a pair f, p, where f is a preflow and $p : V \to \mathbb{Z}_+$ such that

(10.20) (i) if $(u,v) \in A_f$, then $p(v) \geq p(u) - 1$,
(ii) $p(s) = n$ and $p(t) = 0$.

Note that for any given f, such a function p exists if and only if D_f has no $s-t$ path. Hence, if a function p satisfying (10.20) exists and f is an $s-t$ flow, then f is an $s-t$ flow of maximum value (Corollary 10.2a).

Initially, f and p are set by:

(10.21) $f(a) := c(a)$ if $a \in \delta^{\text{out}}(s)$ and $f(a) := 0$ otherwise;
$p(v) := n$ if $v = s$ and $p(v) := 0$ otherwise.

Next, while there exist active vertices, choose an active vertex u maximizing $p(u)$, and apply the following iteratively, until u is inactive:

(10.22) choose an arc $(u,v) \in A_f$ with $p(v) = p(u) - 1$ and *push* over (u,v); if no such arc exists, *relabel* u.

Here to *push* over $(u,v) \in A_f$ means:

(10.23) if $(u,v) \in A$, reset $f(u,v) := f(u,v) + \varepsilon$, where $\varepsilon := \min\{c(u,v) - f(u,v), \text{excess}_f(u)\}$;
if $(v,u) \in A$, reset $f(v,u) := f(v,u) - \varepsilon$, where $\varepsilon := \min\{f(v,u), \text{excess}_f(u)\}$.

To *relabel* u means:

(10.24) reset $p(u) := p(u) + 1$.

Note that if A_f has no arc (u,v) with $p(v) = p(u) - 1$, then we can relabel u without violating (10.20).

This method terminates, since:

Theorem 10.8. *The number of pushes is $O(n^3)$ and the number of relabels is $O(n^2)$.*

Proof. First we show:

(10.25) throughout the process, $p(v) < 2n$ for each $v \in V$.

Indeed, if v is active, then D_f contains a $v-s$ path (since f can be decomposed as a sum of incidence vectors of $s-v$ paths, for $v \in V$, and of directed circuits). So by (10.20)(i), $p(v) - p(s) \leq \text{dist}_{D_f}(v,s) < n$. As $p(s) = n$, we have $p(v) < 2n$. This gives (10.25), which directly implies:

(10.26) the number of relabels is at most $2n^2$.

To estimate the number of pushes, call a push (10.23) *saturating* if after the push one has $f(u,v) = c(u,v)$ (if $(u,v) \in A$) or $f(v,u) = 0$ (if $(v,u) \in A$). Then:

(10.27) the number of saturating pushes is $O(nm)$.

For consider any arc $a = (u,v) \in A$. If we increase $f(a)$, then $p(v) = p(u) - 1$, while if we decrease $f(a)$, then $p(u) = p(v) - 1$. So meantime $p(v)$ should have been relabeled at least twice. As p is nondecreasing (in time), by (10.25) we have (10.27).

Finally:

(10.28) the number of nonsaturating pushes is $O(n^3)$.

Between any two relabels the function p does not change. Hence there are $O(n)$ nonsaturating pushes, as each of them makes an active vertex v maximizing $p(v)$ inactive (while possibly a vertex v' with $p(v') < p(v)$ is activated). With (10.26) this gives (10.28). ∎

There is an efficient implementation of the method:

Theorem 10.9. *The push-relabel method finds a maximum flow in time* $O(n^3)$.

Proof. We order the vertex set V as a doubly linked list, in order of increasing value $p(v)$. Moreover, for each $u \in V$ we keep the set L_u of arcs (u,v) in A_f with $p(v) = p(u) - 1$, ordered as a doubly linked list. We also keep with each vertex v the value $\mathrm{excess}_f(v)$, and we keep linked lists of arcs of D incident with v.

Throughout the iterations, we choose an active vertex u maximizing $p(u)$, and we process u, until u becomes inactive. Between any two relabelings, this searching takes $O(n)$ time, since as long as we do not relabel, we can continue searching the list V in order. As we relabel $O(n^2)$ times, we can do the searching in $O(n^3)$ time.

Suppose that we have found an active vertex u maximizing $p(u)$. We next push over each of the arcs in L_u. So finding an arc $a = (u,v)$ for pushing takes time $O(1)$. If it is a saturating push, we can delete (u,v) from L_u in time $O(1)$. Moreover, we can update $\mathrm{excess}_f(u)$ and $\mathrm{excess}_f(v)$ in time $O(1)$. Therefore, as there are $O(n^3)$ pushes, they can be done in $O(n^3)$ time.

We decide to relabel u if $L_u = \emptyset$. When relabeling, updating the lists takes $O(n)$ time: When we reset $p(u)$ from i to $i+1$, then for each arc (u,v) or (v,u) of D, we add (u,v) to L_u if $p(v) = i$ and $(u,v) \in A_f$, and we remove (v,u) from L_v if $p(v) = i+1$ and $(v,u) \in A_f$; moreover, we move u to its new rank in the list V. This all takes $O(n)$ time. Therefore, as there are $O(n^2)$ relabels, they can be done in $O(n^3)$ time. ∎

Further notes on the push-relabel method. If we allow any active vertex u to be chosen for (10.22) (not requiring maximality of $p(u)$), then the bounds of

$O(n^2)$ on the number of relabels and $O(nm)$ on the number of saturating pushes are maintained, while the number of nonsaturating pushes is $O(n^2m)$.

A first-in first-out selection rule was studied by Goldberg [1985], also yielding an $O(n^3)$ algorithm. Theorem 10.9 (using the largest-label selection) is due to Goldberg and Tarjan [1986,1988a], who also showed an implementation of the push-relabel method with dynamic trees, taking $O(nm\log(n^2/m))$ time. Cheriyan and Maheshwari [1989] and Tunçel [1994] showed that the bound on the number of pushes in Theorem 10.8 can be improved to $O(n^2\sqrt{m})$, yielding an $O(n^2\sqrt{m})$ running time bound. Further improvements are given in Ahuja and Orlin [1989] and Ahuja, Orlin, and Tarjan [1989]. The worst-case behaviour of the push-relabel method was studied by Cheriyan and Maheshwari [1989].

10.8. Further results and notes

10.8a. A weakly polynomial bound

Edmonds and Karp [1972] considered the following *fattest augmenting path* rule: choose a flow-augmenting path for which the flow value increase is maximal. They showed that, if all capacities are integer, it terminates in at most $1 + m'\log\phi$ iterations, where ϕ is the maximum flow value and where m' is the maximum number of arcs in any $s-t$ cut. This gives a maximum flow algorithm of running time $O(n^2m\log nC)$, where C is the maximum capacity (assuming all capacities are integer). (For irrational capacities, Queyranne [1980] showed that the method need not terminate.)

Edmonds and Karp [1970,1972] and Dinits [1973a] introduced the idea of *capacity-scaling*, which gives the following stronger running time bound:

Theorem 10.10. *For integer capacities, a maximum flow can be found in time* $O(m^2\log C)$.

Proof. Let $L := \lceil\log_2 C\rceil + 1$. For $i = L, L-1, \dots, 0$, we can obtain a maximum flow f' for capacity function $c' := \lfloor c/2^i\rfloor$, from a maximum flow f'' for capacity function $c'' := \lfloor c/2^{i+1}\rfloor$ as follows. Observe that the maximum flow value for c' differs by at most m from that of the maximum flow value ϕ for $2c''$. For let $\delta^{\mathrm{out}}(U)$ be a cut with $2c''(\delta^{\mathrm{out}}(U)) = \phi$. Then $c'(\delta^{\mathrm{out}}(U)) - \phi \le |\delta^{\mathrm{out}}(U)| \le m$. So a maximum flow with respect to c' can be obtained from $2f''$ by at most m augmenting path iterations. As each augmenting path iteration can be done in $O(m)$ time, and as $\lfloor c/2^L\rfloor = 0$, we have the running time bound given. ∎

With methods similar to those used in Corollary 10.6a, the bound in Theorem 10.10 can be improved to $O(nm\log C)$, a result of Dinits [1973a] and Gabow [1985b]. To see this, observe that the proof of Theorem 10.6 also yields:

Theorem 10.11. *Given an acyclic graph* $D = (V, A)$, $s, t \in V$, *and a capacity function* $c : A \to \mathbb{Z}_+$, *an integer blocking flow* f *can be found in time* $O(n\phi + m)$, *where* ϕ *is the value of* f.

Proof. Consider the proof of Theorem 10.6. We do at most ϕ iterations, while each iteration takes $O(n+t)$ time, where t is the number of arcs deleted. ∎

Hence, similarly to Corollary 10.6a one has:

Corollary 10.11a. *For integer capacities, a maximum flow can be found in time $O(n(\phi+m))$, where ϕ is the maximum flow value.*

Proof. Similar to the proof of Corollary 10.6a. ∎

Therefore,

Corollary 10.11b. *For integer capacities, a maximum flow can be found in time $O(nm\log C)$.*

Proof. In the proof of Theorem 10.10, a maximum flow with respect to c' can be obtained from $2f''$ in time $O(nm)$ (by Corollary 10.11a), since the maximum flow value in the residual graph $D_{f''}$ is at most m. ∎

10.8b. Complexity survey for the maximum flow problem

Complexity survey ($*$ indicates an asymptotically best bound in the table):

$O(n^2mC)$	Dantzig [1951a] simplex method
$O(nmC)$	Ford and Fulkerson [1955,1957b] augmenting path
$O(nm^2)$	Dinits [1970], Edmonds and Karp [1972] shortest augmenting path
$O(n^2m\log nC)$	Edmonds and Karp [1972] fattest augmenting path
$O(n^2m)$	Dinits [1970] shortest augmenting path, layered network
$O(m^2\log C)$	Edmonds and Karp [1970,1972] capacity-scaling
$O(nm\log C)$	Dinits [1973a], Gabow [1983b,1985b] capacity-scaling
$O(n^3)$	Karzanov [1974] (preflow push); cf. Malhotra, Kumar, and Maheshwari [1978], Tarjan [1984]
$O(n^2\sqrt{m})$	Cherkasskiĭ [1977a] blocking preflow with long pushes
$O(nm\log^2 n)$	Shiloach [1978], Galil and Naamad [1979,1980]
$O(n^{5/3}m^{2/3})$	Galil [1978,1980a]

≫

continued

	$O(nm \log n)$	Sleator [1980], Sleator and Tarjan [1981,1983a] dynamic trees
*	$O(nm \log(n^2/m))$	Goldberg and Tarjan [1986,1988a] push-relabel+dynamic trees
	$O(nm + n^2 \log C)$	Ahuja and Orlin [1989] push-relabel + excess scaling
	$O(nm + n^2\sqrt{\log C})$	Ahuja, Orlin, and Tarjan [1989] Ahuja-Orlin improved
*	$O(nm \log((n/m)\sqrt{\log C} + 2))$	Ahuja, Orlin, and Tarjan [1989] Ahuja-Orlin improved + dynamic trees
*	$O(n^3/\log n)$	Cheriyan, Hagerup, and Mehlhorn [1990,1996]
	$O(n(m + n^{5/3} \log n))$	Alon [1990] (derandomization of Cheriyan and Hagerup [1989,1995])
	$O(nm + n^{2+\varepsilon})$	(for each $\varepsilon > 0$) King, Rao, and Tarjan [1992]
*	$O(nm \log_{m/n} n + n^2 \log^{2+\varepsilon} n)$	(for each $\varepsilon > 0$) Phillips and Westbrook [1993,1998]
*	$O(nm \log_{\frac{m}{n \log n}} n)$	King, Rao, and Tarjan [1994]
*	$O(m^{3/2} \log(n^2/m) \log C)$	Goldberg and Rao [1997a,1998]
*	$O(n^{2/3} m \log(n^2/m) \log C)$	Goldberg and Rao [1997a,1998]

Here $C := \|c\|_\infty$ for integer capacity function c. For a complexity survey for unit capacities, see Section 9.6a.

Research problem: Is there an $O(nm)$-time maximum flow algorithm?

For the special case of *planar* undirected graphs:

	$O(n^2 \log n)$	Itai and Shiloach [1979]
	$O(n \log^2 n)$	Reif [1983] (minimum cut), Hassin and Johnson [1985] (maximum flow)
	$O(n \log n \log^* n)$	Frederickson [1983b]
*	$O(n \log n)$	Frederickson [1987b]

For *directed* planar graphs:

	$O(n^{3/2} \log n)$	Johnson and Venkatesan [1982]
	$O(n^{4/3} \log^2 n \log C)$	Klein, Rao, Rauch, and Subramanian [1994], Henzinger, Klein, Rao, and Subramanian [1997]
*	$O(n \log n)$	Weihe [1994b,1997b]

Itai and Shiloach [1979] and Hassin [1981b] showed that if s and t both are on the outer boundary, then a shortest path algorithm applied to the dual gives an $O(n \log n)$ algorithm for finding a minimum-capacity $s-t$ cut and a maximum-value $s-t$ flow, also for the directed case. This extends earlier work of Hu [1969].

Khuller, Naor, and Klein [1993] studied the lattice structure of the integer $s-t$ flows in a planar directed graph. More on planar maximum flow can be found in Khuller and Naor [1990,1994].

10.8c. An exchange property

Dinits [1973b] and Minieka [1973] observed the following analogue of Theorem 9.12:

Theorem 10.12. *Let $D = (V, A)$ be a digraph, let $s, t \in V$, let $c : A \to \mathbb{R}_+$, and let f_1 and f_2 be maximum $s-t$ flows subject to c. Then there exists a maximum $s-t$ flow f subject to c such that $f(a) = f_1(a)$ for each arc a incident with s and $f(a) = f_2(a)$ for each arc a incident with t.*

Proof. Let $\delta^{\text{out}}(U)$ be an $s-t$ cut of minimum capacity, with $U \subseteq V$ and $s \in U$,$t \notin U$. So f_1 and f_2 coincide on $\delta^{\text{out}}(U)$ and on $\delta^{\text{in}}(U)$. Define $f(a) := f_1(a)$ if a is incident with U and $f(a) := f_2(a)$ if a is incident with $V \setminus U$. This defines a maximum $s-t$ flow as required. ∎

This was also shown by Megiddo [1974], who used it to prove the following. Let $D = (V, A)$ be a directed graph, let $c : A \to \mathbb{R}_+$ be a capacity function, and let $s, t \in V$, where s is a source, and t is a sink. An $s-t$ flow $f \leq c$ is called *source-optimal* if the vector $(f(a) \mid a \in \delta^{\text{out}}(s))$ is lexicographically maximal among all $s-t$ flows subject to c (ordering the $a \in \delta^{\text{out}}(s)$ by nonincreasing value of $f(a)$). The maximum flow algorithm implies that a source-optimal $s-t$ flow is a maximum-value $s-t$ flow.

One similarly defines *sink-optimal*, and Theorem 10.12 implies that there exists an $s-t$ flow that is both source- and sink-optimal. The proof shows that this flow can be found by combining a source-optimal and a sink-optimal flow appropriately.

As Megiddo showed, a source-optimal flow can be found iteratively, by updating a flow f (starting with $f = \mathbf{0}$), by determining an arc $a \in \delta^{\text{out}}(s)$ with $f(a) = 0$, on which $f(a)$ can be increased most. Making this increase, gives the next f. Stop if no increase is possible anymore.

10.8d. Further notes

Simplex method. The maximum flow problem is a linear programming problem, and hence it can be solved with the simplex method of Dantzig [1951b] (this paper includes an anti-cycling rule based on perturbation). This was elaborated by Fulkerson and Dantzig [1955a,1955b]. A direct, combinatorial anti-cycling rule for flow problems was given by Cunningham [1976]. Goldfarb and Hao [1990] gave a pivot rule that leads to at most nm pivots, yielding an algorithm of running time $O(n^2 m)$. Goldberg, Grigoriadis, and Tarjan [1991] showed that with the help of dynamic trees there is an $O(nm \log n)$ implementation. See also Gallo, Grigoriadis, and Tarjan [1989], Plotkin and Tardos [1990], Goldfarb and Hao [1991], Orlin, Plotkin,

and Tardos [1993], Ahuja and Orlin [1997], Armstrong and Jin [1997], Goldfarb and Chen [1997], Tarjan [1997], Armstrong, Chen, Goldfarb, and Jin [1998], and Hochbaum [1998].

Worst-case analyses of maximum flow algorithms were given by Zadeh [1972, 1973b] (shortest augmenting path rule), Dinits [1973b] (shortest augmenting path rule), Tarjan [1974e], Even and Tarjan [1975] (Dinits $O(n^2m)$ algorithm), Baratz [1977] (Karzanov's $O(n^3)$ algorithm), Galil [1981], Cheriyan [1988] (push-relabel method), Cheriyan and Maheshwari [1989] (push-relabel method), and Martel [1989] (push-relabel method). For further analysis of maximum flow algorithms, see Tucker [1977a] and Ahuja and Orlin [1991].

Computational studies were reported by Cherkasskiĭ [1979], Glover, Klingman, Mote, and Whitman [1979,1980,1984], Hamacher [1979] (Karzanov's method), Cheung [1980], Imai [1983b], Goldfarb and Grigoriadis [1988] (Dinits' method and the simplex method), Derigs and Meier [1989] (push-relabel method), Alizadeh and Goldberg [1993] (push-relabel in parallel), Anderson and Setubal [1993] (push-relabel), Gallo and Scutellà [1993], Nguyen and Venkateswaran [1993] (push-relabel), and Cherkassky and Goldberg [1995,1997] (push-relabel method). Consult also Johnson and McGeoch [1993].

A probabilistic analysis was presented by Karp, Motwani, and Nisan [1993]. A randomized approximation algorithm for minimum $s-t$ cut was given by Benczúr and Karger [1996].

Fulkerson [1959b] gave a labeling algorithm for finding the minimum cost of capacities to be added to make an $s-t$ flow of given value possible. Wollmer [1964] studied which k arcs to remove from a capacitated digraph so as to reduce the maximum $s-t$ flow value as much as possible. McCormick [1997] studied the problem of computing 'least infeasible' flows. Akers [1960] described the effect of ΔY operations on max-flow computations.

Ponstein [1972] gave another rule guaranteeing termination of the augmenting path iterations in Ford and Fulkerson's algorithm. Karp [1972b] showed that the *maximum*-cut problem is NP-complete — see Section 75.1a. Work on flows with small capacities was reported by Fernández-Baca and Martel [1989] and Ahuja and Orlin [1991]. Decomposition algorithms for locating minimal cuts were studied by Jarvis and Tufekci [1982]. The kth best cut algorithm was given by Hamacher [1982].

The problem of determining a flow along odd paths in an undirected graph was considered by Schrijver and Seymour [1994] — see Section 29.11e.

For an in-depth survey of network flows, see Ahuja, Magnanti, and Orlin [1993]. Other surveys were given by Ford and Fulkerson [1962], Dantzig [1963], Busacker and Saaty [1965], Fulkerson [1966], Hu [1969,1982], Iri [1969], Frank and Frisch [1971], Berge [1973b], Adel'son-Vel'skiĭ, Dinits, and Karzanov [1975] (for a review, see Goldberg and Gusfield [1991]), Christofides [1975], Lawler [1976b], Bazaraa and Jarvis [1977], Minieka [1978], Even [1979], Jensen and Barnes [1980], Papadimitriou and Steiglitz [1982], Smith [1982], Chvátal [1983], Sysło, Deo, and Kowalik [1983], Tarjan [1983], Gondran and Minoux [1984], Rockafellar [1984], Tarjan [1986], Nemhauser and Wolsey [1988], Ahuja, Magnanti, and Orlin [1989,1991], Chen [1990], Cormen, Leiserson, and Rivest [1990], Goldberg, Tardos, and Tarjan [1990], Frank [1995], Cook, Cunningham, Pulleyblank, and Schrijver [1998], Jungnickel [1999], Mehlhorn and Näher [1999], and Korte and Vygen [2000].

Golden and Magnanti [1977] gave a bibliography and Slepian [1968] discussed the algebraic theory of flows.

10.8e. Historical notes on maximum flow

The problem of sending flow through a network was considered by Kantorovich [1939]. In fact, he considered multicommodity flows — see the historical notes in Section 70.13g.

The foundations for one-commodity maximum flow were laid during the period November 1954-December 1955 at RAND Corporation in Santa Monica, California. We will review the developments in a chronological order, by the date of the RAND Reports.

In their basic report *Maximal Flow through a Network* dated 19 November 1954, Ford and Fulkerson [1954,1956b] showed the max-flow min-cut theorem for undirected graphs:

> Theorem 1. (Minimal cut theorem). The maximal flow value obtainable in a network N is the minimum of $v(D)$ taken over all disconnecting sets D.

(Robacker [1955a] wrote that the max-flow min-cut theorem was conjectured first by Fulkerson.)

Ford and Fulkerson were motivated by flows and cuts in railway networks — see below. In the same report, also a simple algorithm was described for the maximum flow problem in case the graph, added with an extra edge connecting s and t, is planar.

The authors moreover observed that the maximum flow problem is a special case of a linear programming problem and that hence it can be solved by Dantzig's simplex method.

In a report of 1 January 1955 (revised 15 April 1955), Dantzig and Fulkerson [1955,1956] showed that the max-flow min-cut theorem can also be deduced from the duality theorem of linear programming (they mention that also A.J. Hoffman did this), they generalized it to the directed case, and they observed, using results of Dantzig [1951a], that if the capacities are integer, there is an integer maximum flow (the 'integrity theorem'). Hence (as they mention) Menger's theorem follows as a consequence. A simple computational method for the maximum flow problem based on the simplex method was described in a report of 1 April 1955 by Fulkerson and Dantzig [1955a,1955b].

Conversely, in a report of 26 May 1955, Robacker [1955a] derived the undirected max-flow min-cut theorem from the undirected vertex-disjoint version of Menger's theorem.

Boldyreff's heuristic

While the maximum flow algorithms found so far were derived from the simplex method, the quest for combinatorial methods remained vivid. A heuristic for the maximum flow problem, the 'flooding technique', was presented by Boldyreff [1955c, 1955b] on 3 June 1955 at the New York meeting of the Operations Research Society of America (published as a RAND Report of 5 August 1955 (Boldyreff [1955a])). The method is intuitive and the author did not claim generality (we quote from Boldyreff [1955b]):

> It has been previously assumed that a highly complex railway transportation system, too complicated to be amenable to analysis, can be represented by a much simpler model. This was accomplished by representing each complete railway operating division by a point, and by joining pairs of such points by arcs (lines) with traffic carrying capacities equal to the maximum possible volume of traffic (expressed in some convenient unit, such as trains per day) between the corresponding operating divisions.
> In this fashion, a network is obtained consisting of three sets of points — points of origin, intermediate or junction points, and the terminal points (or points of destination) — and a set of arcs of specified traffic carrying capacities, joining these points to each other.

Boldyreff's arguments for designing a heuristic procedure are formulated as follows:

> In the process of searching for the methods of solving this problem the following objectives were used as a guide:
> **1.** That the solution could be obtained quickly, even for complex networks.
> **2.** That the method could be explained easily to personnel without specialized technical training and used by them effectively.
> **3.** That the validity of the solution be subject to easy, direct verification.
> **4.** That the method would not depend on the use of high-speed computing or other specialized equipment.

Boldyreff's 'flooding technique' pushes as much flow as possible greedily through the network. If at some vertex a 'bottleneck' arises (i.e., more trains arrive than can be pushed further through the network), it is eliminated by returning the excess trains to the origin.

The method is empirical, not using backtracking, and not leading to an optimum solution in all cases:

> Whenever arbitrary decisions have to be made, ordinary common sense is used as a guide. At each step the guiding principle is to move forward the maximum possible number of trains, and to maintain the greatest flexibility for the remaining network.

Boldyreff speculates:

> In dealing with the usual railway networks a single flooding, followed by removal of bottlenecks, should lead to a maximal flow.

In the abstract of his lecture, Boldyreff [1955c] mentions:

> The mechanics of the solutions is formulated as a simple game which can be taught to a ten-year-old boy in a few minutes.

In his article, Boldyreff [1955b] gave as example the model of a real, comprehensive, railway transportation system with 41 vertices and 85 arcs:

> The total time of solving the problem is less than thirty minutes.

His closing remarks are:

> Finally there is the question of a systematic formal foundation, the comprehensive mathematical basis for empiricism and intuition, and the relation of the present techniques to other processes, such as, for instance, the multistage decision process (a suggestion of Bellman's).
> All this is reserved for the future.

Ford and Fulkerson's motivation: The Harris-Ross report

In their first report on maximum flow, *Maximal Flow through a Network*, Ford and Fulkerson [1954,1956b] mentioned that the maximum flow problem was formulated by T.E. Harris as follows:

> Consider a rail network connecting two cities by way of a number of intermediate cities, where each link of the network has a number assigned to it representing its capacity. Assuming a steady state condition, find a maximal flow from one given city to the other.

Later, in their book *Flows in Networks*, Ford and Fulkerson [1962] gave a more precise reference[17]:

> It was posed to the authors in the spring of 1955 by T.E. Harris, who, in conjunction with General F.S. Ross (Ret.), had formulated a simplified model of railway traffic flow, and pinpointed this particular problem as the central one suggested by the model [11].

Ford and Fulkerson's reference [11] here is the secret report by Harris and Ross [1955] entitled *Fundamentals of a Method for Evaluating Rail Net Capacities*, dated 24 October 1955[18], and written for the Air Force. The report was downgraded to 'unclassified' on 21 May 1999.

Unlike what Ford and Fulkerson write, the interest of Harris and Ross was not to find a maximum flow, but rather a minimum cut ('interdiction') of the Soviet railway system. We quote:

> Air power is an effective means of interdicting an enemy's rail system, and such usage is a logical and important mission for this Arm.
> As in many military operations, however, the success of interdiction depends largely on how complete, accurate, and timely is the commander's information, particularly concerning the effect of his interdiction-program efforts on the enemy's capability to move men and supplies. This information should be available at the time the results are being achieved.
> The present paper describes the fundamentals of a method intended to help the specialist who is engaged in estimating railway capabilities, so that he might more readily accomplish this purpose and thus assist the commander and his staff with greater efficiency than is possible at present.

In the Harris-Ross report, first much attention is given to modelling a railway network: taking each railway junction as a vertex would give a too refined network (for their purposes). Therefore, Harris and Ross proposed to take 'railway divisions' (organizational units based on geographical areas) as vertices, and to estimate the capacity of the connections between any two adjacent railway divisions. In an interview with Alexander [1996], Harris remembered:

> We were studying rail transportation in consultation with a retired army general, Frank Ross, who had been chief of the Army's Transportation Corps in Europe. We thought of modeling a rail system as a network. At first it didn't make sense, because there's no reason why the crossing point of two lines should be a special

[17] There seems to be some discrepancy between the date of the RAND Report of Ford and Fulkerson (19 November 1954) and the date mentioned in the quotation (spring of 1955).

[18] In their book, Ford and Fulkerson incorrectly date the Harris-Ross report 24 October 1956.

> sort of node. But Ross realized that, in the region we were studying, the "divisions" (little administrative districts) should be the nodes. The link between two adjacent nodes represents the total transportation capacity between them. This made a reasonable and manageable model for our rail system.

The Harris-Ross report stresses that specialists remain needed to make up the model (which seems always a good strategy to get new methods accepted):

> It is not the purpose that the highly specialized individual who estimates track and network capacities should be replaced by a novice with a calculating machine. Rather, it is accepted that the evaluation of track capacities remains a task for the specialist.
>
> [...]
>
> The ability to estimate with relative accuracy the capacity of single railway lines is largely an art. Specialists in this field have no authoritative text (insofar as the authors are informed) to guide their efforts, and very few individuals have either the experience or talent for this type of work. The authors assume that this job will continue to be done by the specialist.

The authors next disputed the naive belief that a railway network is just a set of disjoint through lines, and that cutting these lines would imply cutting the network:

> It is even more difficult and time-consuming to evaluate the capacity of a railway network comprising a multitude of rail lines which have widely varying characteristics. Practices among individuals engaged in this field vary considerably, but all consume a great deal of time. Most, if not all, specialists attack the problem by viewing the railway network as an aggregate of through lines.
> The authors contend that the foregoing practice does not portray the full flexibility of a large network. In particular it tends to gloss over the fact that even if every one of a set of independent through lines is made inoperative, there may exist alternative routings which can still move the traffic.
> This paper proposes a method that departs from present practices in that it views the network as an aggregate of railway operating divisions. All trackage capacities within the divisions are appraised, and these appraisals form the basis for estimating the capability of railway operating divisions to receive trains from and concurrently pass trains to each neighboring division in 24-hour periods.

Whereas experts are needed to set up the model, to solve it is routine (when having the 'work sheets'):

> The foregoing appraisal (accomplished by the expert) is then used in the preparation of comparatively simple work sheets that will enable relatively inexperienced assistants to compute the results and thus help the expert to provide specific answers to the problems, based on many assumptions, which may be propounded to him.

While Ford and Fulkerson flow-augmenting path algorithm for the maximum flow problem was not found yet, the Harris-Ross report suggests applying Boldyreff's flooding technique described above. The authors preferred this above the simplex method for maximum flow:

> The calculation would be cumbersome; and, even if it could be performed, sufficiently accurate data could not be obtained to justify such detail.

However, later in the report their assessment of the simplex method is more favourable:

> These methods do not require elaborate computations and can be performed by a relatively untrained person.

The Harris-Ross report applies Boldyreff's flooding technique to a network model of the Soviet and East European railways. For the data it refers to several secret reports of the Central Intelligence Agency (C.I.A.) on sections of the Soviet and East European railway networks. After the aggregation of railway divisions to vertices, the network has 44 vertices and 105 (undirected) edges.

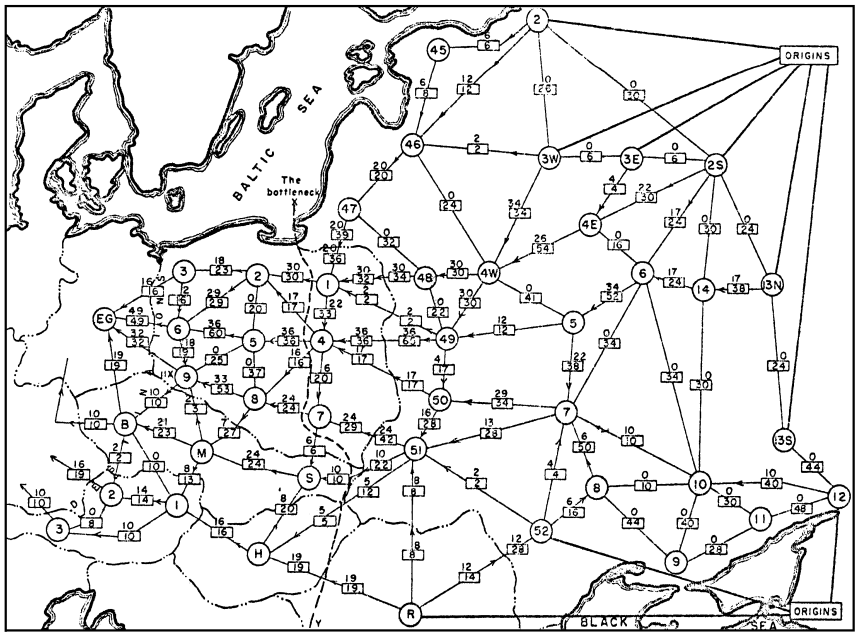


Figure 10.2

From Harris and Ross [1955]: Schematic diagram of the railway network of the Western Soviet Union and East European countries, with a maximum flow of value 163,000 tons from Russia to Eastern Europe, and a cut of capacity 163,000 tons indicated as 'The bottleneck'.

The application of the flooding technique to the problem is displayed step by step in an appendix of the report, supported by several diagrams of the railway network. (Also work sheets are provided, to allow for future changes in capacities.) It yields a flow of value 163,000 tons from sources in the Soviet Union to destinations in East European 'satellite' countries, together with a cut with a capacity of, again, 163,000 tons. So the flow value and the cut capacity are equal, hence optimum. In the report, the minimum cut is indicated as 'the bottleneck' (Figure 10.2).

Further developments

Soon after the Harris-Ross report, Ford and Fulkerson [1955,1957b] presented in a RAND Report of 29 December 1955 their 'very simple algorithm' for the maximum flow problem, based on finding 'augmenting paths' as described in Section 10.4 above. The algorithm finds in a finite number of steps a maximum flow, if all capacities have integer values. We quote:

> This problem is of course a linear programming problem, and hence may be solved by Dantzig's simplex algorithm. In fact, the simplex computation for a problem of this kind is particularly efficient, since it can be shown that the sets of equations one solves in the process are always triangular [2]. However, for the flow problem, we shall describe what appears to be a considerably more efficient algorithm; it is, moreover, readily learned by a person with no special training, and may easily be mechanized for handling large networks. We believe that problems involving more than 500 nodes and 4,000 arcs are within reach of present computing machines.

(Reference [2] is Dantzig and Fulkerson [1955].)

In the RAND Report, Ford and Fulkerson [1955] mention that Boldyreff's flooding technique might give a good starting flow, but in the final paper (Ford and Fulkerson [1957b]) this suggestion has been omitted.

An alternative proof of the max-flow min-cut theorem was given by Elias, Feinstein, and Shannon [1956] ('manuscript received by the PGIT, July 11,1956'), who claimed that the result was known by workers in communication theory:

> This theorem may appear almost obvious on physical grounds and appears to have been accepted without proof for some time by workers in communication theory. However, while the fact that this flow cannot be exceeded is indeed almost trivial, the fact that it can actually be achieved is by no means obvious. We understand that proofs of the theorem have been given by Ford and Fulkerson and Fulkerson and Dantzig. The following proof is relatively simple, and we believe different in principle.

The proof of Elias, Feinstein, and Shannon is based on a reduction technique similar to that used by Menger [1927] in proving his theorem.

Chapter 11

Circulations and transshipments

Circulations and transshipments are variants of flows. Circulations have no source or sink — so flow conservation holds in each vertex — while transshipments have several sources and sinks — so *any* nonnegative function is a transshipment (however, the problem is to find a transshipment with prescribed excess function).
Problems on circulations and transshipments can be reduced to flow problems, or can be treated with similar methods.

11.1. A useful fact on arc functions

Recall that for any digraph $D = (V, A)$ and any $f : A \to \mathbb{R}$, the *excess* function $\text{excess}_f : V \to \mathbb{R}$ is defined by

$$\text{excess}_f(v) := f(\delta^{\text{in}}(v)) - f(\delta^{\text{out}}(v)) \tag{11.1}$$

for $v \in V$.

The following theorem of Gallai [1958a,1958b] will turn out to be useful in this chapter:

Theorem 11.1. *Let $D = (V, A)$ be a digraph and let $f : A \to \mathbb{R}_+$. Then f is a nonnegative linear combination of at most $|A|$ vectors χ^P, where P is a directed path or circuit. If P is a path, it starts at a vertex v with $\text{excess}_f(v) < 0$ and ends at a vertex with $\text{excess}_f(v) > 0$. If f is integer, we can take the linear combination integer-scalared. The combination can be found in $O(nm)$ time.*

Proof. We may assume that $\text{excess}_f = \mathbf{0}$, since we can add a new vertex u, for each $v \in V$ with $\text{excess}_f(v) > 0$, an arc (v, u), and for each $v \in V$ with $\text{excess}_f(v) < 0$, an arc (u, v). Define $f(v, u) := \text{excess}_f(v)$ and $f(u, v) := -\text{excess}_f(v)$ for any new arc (u, v) or (v, u). Then the new f satisfies $\text{excess}_f = \mathbf{0}$, and a decomposition for the new f gives a decomposition for the original f.

Define $A' := \{a \mid f(a) > 0\}$. We apply induction on $|A'|$. We may assume that $A' \neq \emptyset$. Then A' contains a directed circuit, C say. Let τ be the minimum

of the $f(a)$ for $a \in AC$ and let $f' := f - \tau\chi^C$. Then the theorem follows by induction.

Since we can find C in $O(n)$ time, we can find the decomposition in $O(nm)$ time. ∎

11.2. Circulations

Let $D = (V, A)$ be a digraph. A function $f : A \to \mathbb{R}$ is called a *circulation* if

$$f(\delta^{\text{in}}(v)) = f(\delta^{\text{out}}(v)) \tag{11.2}$$

for each vertex $v \in V$. So now the flow conservation law holds in *each* vertex v. By Theorem 11.1,

(11.3) each nonnegative circulation is a nonnegative linear combination of incidence vectors of directed circuits; each nonnegative integer circulation is the sum of incidence vectors of directed circuits.

Hoffman [1960] mentioned that he proved the following characterization of the existence of circulations in 1956[19]:

Theorem 11.2 (Hoffman's circulation theorem)**.** *Let $D = (V, A)$ be a digraph and let $d, c : A \to \mathbb{R}$ with $d \le c$. Then there exists a circulation f satisfying $d \le f \le c$ if and only if*

$$d(\delta^{\text{in}}(U)) \le c(\delta^{\text{out}}(U)) \tag{11.4}$$

for each subset U of V. If moreover d and c are integer, f can be taken integer.

Proof. To see necessity of (11.4), suppose that a circulation f satisfying $d \le f \le c$ exists. Then for each $U \subseteq V$,

$$d(\delta^{\text{in}}(U)) \le f(\delta^{\text{in}}(U)) = f(\delta^{\text{out}}(U)) \le c(\delta^{\text{out}}(U)). \tag{11.5}$$

To see sufficiency, choose a function f satisfying $d \le f \le c$ and minimizing $\|\text{excess}_f\|_1$. Let $S := \{v \in V \mid \text{excess}_f(v) > 0\}$ and $T := \{v \in V \mid \text{excess}_f(v) < 0\}$. Suppose that $S \neq \emptyset$. Let $D_f = (V, A_f)$ be the residual graph (defined in (10.9)). If D_f contains an $S-T$ path P, we can modify f along P so as to reduce $\|\text{excess}_f\|_1$. So D_f contains no $S-T$ path. Let U be the set of vertices reachable in D_f from S. Then for each $a \in \delta^{\text{out}}_A(U)$ we have $a \notin A_f$ and hence $f(a) = c(a)$. Similarly, for each $a \in \delta^{\text{in}}_A(U)$ we have $a^{-1} \notin A_f$ and hence $f(a) = d(a)$. Therefore,

$$\begin{aligned} d(\delta^{\text{in}}(U)) - c(\delta^{\text{out}}(U)) &= f(\delta^{\text{in}}(U)) - f(\delta^{\text{out}}(U)) = \text{excess}_f(U) \\ &= \text{excess}_f(S) > 0, \end{aligned} \tag{11.6}$$

[19] In fact, A.J. Hoffman [1960] attributes it to A.H. Hoffman, but this is a misprint (A.J. Hoffman, personal communication 1995).

contradicting (11.4). ∎

(One can derive this theorem also from the max-flow min-cut theorem, with the methods described in Section 11.6 below.)

Theorem 11.2 implies that any circulation can be rounded:

Corollary 11.2a. *Let $D = (V, A)$ be a digraph and let $f : A \to \mathbb{R}$ be a circulation. Then there exists an integer circulation f' with $\lfloor f(a) \rfloor \leq f'(a) \leq \lceil f(a) \rceil$ for each arc a.*

Proof. Take $d := \lfloor f \rfloor$ and $c := \lceil f \rceil$ in Theorem 11.2. ∎

Another consequence is:

Corollary 11.2b. *Let $D = (V, A)$ be a digraph, let $k \in \mathbb{Z}_+$ (with $k \geq 1$), and let $f : A \to \mathbb{Z}$ be a circulation. Then $f = f_1 + \cdots + f_k$ where each f_i is an integer circulation satisfying*

$$\lfloor \tfrac{1}{k} f \rfloor \leq f_i \leq \lceil \tfrac{1}{k} f \rceil. \tag{11.7}$$

Proof. By induction on k. Define $d := \lfloor \frac{1}{k} f \rfloor$ and $c := \lceil \frac{1}{k} f \rceil$. It suffices to show that there exists an integer circulation f_k such that

$$d \leq f_k \leq c \text{ and } (k-1)d \leq f - f_k \leq (k-1)c, \tag{11.8}$$

equivalently,

$$\begin{aligned} &\max\{d(a), f(a) - (k-1)c(a)\} \leq f_k(a) \\ &\leq \min\{c(a), f(a) - (k-1)d(a)\} \end{aligned} \tag{11.9}$$

for each $a \in A$. Since these bounds are integer, by Corollary 11.2a it suffices to show that there is any circulation obeying these bounds. For that we can take $\frac{1}{k} f$. ∎

This corollary implies that the set of circulations f satisfying $d \leq f \leq c$ for some integer bounds d, c, has the integer decomposition property.

11.3. Flows with upper and lower bounds

We can derive from Corollary 11.2a that flows can be rounded similarly:

Corollary 11.2c. *Let $D = (V, A)$ be a digraph, let $s, t \in V$, and let f be an $s - t$ flow of value $k \in \mathbb{Z}$. Then there exists an integer $s - t$ flow f' of value k with $\lfloor f(a) \rfloor \leq f'(a) \leq \lceil f(a) \rceil$ for each arc a.*

Proof. Add an arc (t, s) and define $f(t, s) := k$. We obtain a circulation to which we can apply Corollary 11.2a. Deleting the new arc, we obtain an $s - t$ flow as required. ∎

According to Berge [1958b], A.J. Hoffman showed the following on the existence of a flow obeying both an upper bound (capacity) and a lower bound (demand):

Corollary 11.2d. *Let $D = (V, A)$ be a digraph, let $s, t \in V$, and let $d, c : A \to \mathbb{R}_+$ with $d \leq c$. Then there exists an $s-t$ flow f with $d \leq f \leq c$ if and only if*

$$c(\delta^{\text{out}}(U)) \geq d(\delta^{\text{in}}(U)) \tag{11.10}$$

for each $U \subseteq V$ not separating s and t. If moreover d and c are integer, f can be taken integer.

Proof. Necessity being direct, we show sufficiency. Identify s and t. By Theorem 11.2, there exists a circulation in the shrunk network. This gives a flow as required in the original network. ∎

(Berge [1958b] wrote that this result was shown by Hoffman with linear programming techniques, and was reduced to network theory by L.R. Ford, Jr.)

Moreover, a min-max relation for the maximum value of a flow obeying upper and lower bounds can be derived:

Corollary 11.2e. *Let $D = (V, A)$ be a digraph, let $s, t \in V$, and let $d, c : A \to \mathbb{R}_+$ with $d \leq c$, such that there exists an $s-t$ flow f with $d \leq f \leq c$. Then the maximum value of an $s-t$ flow f with $d \leq f \leq c$ is equal to the minimum value of*

$$c(\delta^{\text{out}}(U)) - d(\delta^{\text{in}}(U)) \tag{11.11}$$

taken over $U \subseteq V$ with $s \in U$ and $t \notin U$. If d and c are integer, the maximum is attained by an integer flow f.

Proof. Let μ be the minimum value of (11.11). Add to D an arc (t, s), with $d(t, s) = c(t, s) = \mu$. Then the extended network has a circulation by Theorem 11.2. Indeed, condition (11.4) for U not separating s and t follows from (11.10). If $s \in U$, $t \notin U$, (11.4) follows from the definition of μ. If $s \notin U$, $t \in U$, then $\mu \geq \text{value}(f)$. Hence

$$\begin{aligned} &\mu + c(\delta^{\text{out}}(U)) - d(\delta^{\text{in}}(U)) \geq \mu + f(\delta^{\text{out}}(U)) - f(\delta^{\text{in}}(U)) \\ &= \mu - \text{value}(f) \geq 0. \end{aligned} \tag{11.12}$$

Therefore, the original network has a flow as required. ∎

11.4. b-transshipments

Let $D = (V, A)$ be a digraph and let $b \in \mathbb{R}^V$. A function $f : A \to \mathbb{R}$ is called a *b-transshipment* if $\text{excess}_f = b$. (So each function $f : A \to \mathbb{R}$ is a

b-transshipment for some b. excess_f is defined in Section 10.1.) By reduction to Hoffman's circulation theorem, one may characterize the existence of a b-transshipment obeying given upper and lower bounds on the arcs:

Corollary 11.2f. *Let $D = (V, A)$ be a digraph, let $d, c : A \to \mathbb{R}$ with $d \leq c$, and let $b : V \to \mathbb{R}$ with $b(V) = 0$. Then there exists a b-transshipment f with $d \leq f \leq c$ if and only if*

$$c(\delta^{\text{in}}(U)) - d(\delta^{\text{out}}(U)) \geq b(U) \tag{11.13}$$

for each $U \subseteq V$. If moreover b, c, and d are integer, f can be taken integer.

Proof. The corollary can be reduced to Hoffman's circulation theorem (Theorem 11.2). Add a new vertex u, and for each $v \in V$ an arc (v, u) with $d(v, u) := c(v, u) := b(v)$. Then a function f as required exists if and only if the extended graph has a circulation f' satisfying $d \leq f' \leq c$. The condition in Hoffman's circulation theorem is equivalent to (11.13). ▮

Conversely, Hoffman's circulation theorem is the special case $b = \mathbf{0}$. The special case $d = \mathbf{0}$ is the following result of Gale [1956,1957]:

Corollary 11.2g (Gale's theorem)**.** *Let $D = (V, A)$ be a digraph and let $c : A \to \mathbb{R}$ and $b : V \to \mathbb{R}$ with $b(V) = 0$. Then there exists a b-transshipment f satisfying $\mathbf{0} \leq f \leq c$ if and only if*

$$c(\delta^{\text{in}}(U)) \geq b(U) \tag{11.14}$$

for each $U \subseteq V$. If moreover b and c are integer, f can be taken integer.

Proof. Take $d = \mathbf{0}$ in Corollary 11.2f. ▮

The proof of Gale is by reduction to the max-flow min-cut theorem. Conversely, the max-flow min-cut theorem follows easily: if $s, t \in V$ and ϕ is the minimum capacity of an $s - t$ cut, then set $b(s) := -\phi$, $b(t) := \phi$, and $b(v) := 0$ for each $v \neq s, t$. As (11.14) is satisfied, by Gale's theorem there exists a b-transshipment f with $\mathbf{0} \leq f \leq c$. This is an $s - t$ flow of value ϕ.

Taking $c = \boldsymbol{\infty}$ in Gale's theorem gives the following result of Rado [1943]:

Corollary 11.2h. *Let $D = (V, A)$ be a digraph and let $b : V \to \mathbb{R}$ with $b(V) = 0$. Then there exists a b-transshipment $f \geq \mathbf{0}$ if and only if $b(U) \leq 0$ for each $U \subseteq V$ with $\delta^{\text{in}}(U) = \emptyset$.*

Proof. This is Gale's theorem (Corollary 11.2g) for $c = \boldsymbol{\infty}$. ▮

11.5. Upper and lower bounds on excess_f

Instead of equality constraints on excess_f one may put upper and lower bounds b and a. This has the following characterization:

Corollary 11.2i. *Let $D = (V, A)$ be a digraph, let $d, c : A \to \mathbb{R}$ with $d \leq c$, and let $a, b : V \to \mathbb{R}$ with $a \leq b$. Then there exists a z-transshipment f with $d \leq f \leq c$ for some z with $a \leq z \leq b$ if and only if*

$$c(\delta^{\text{in}}(U)) - d(\delta^{\text{out}}(U)) \geq \max\{a(U), -b(V \setminus U)\} \tag{11.15}$$

for each $U \subseteq V$. If moreover a, b, c, and d are integer, f can be taken integer.

Proof. The corollary can be reduced to Hoffman's circulation theorem: Add a new vertex u, and for each $v \in V$ an arc (v, u) with $d(v, u) := a(v)$ and $c(v, u) := b(v)$. Then a function f as required exists if and only if the extended graph has a circulation f' satisfying $d \leq f' \leq c$. ∎

This characterization can be formulated equivalently as:

Corollary 11.2j. *Let $D = (V, A)$ be a digraph, let $d, c : A \to \mathbb{R}$ with $d \leq c$, and let $a, b : V \to \mathbb{R}$ with $a \leq b$. Then there exists a z-transshipment f with $d \leq f \leq c$ for some z with $a \leq z \leq b$ if and only if there exists a z-transshipment f' with $d \leq f' \leq c$ for some $z \geq a$ and there exists a z-transshipment f'' with $d \leq f'' \leq c$ for some $z \leq b$.*

Proof. Directly from Corollary 11.2i, since (11.15) can be split into a condition on a and one on b. ∎

For $d = \mathbf{0}$, Corollary 11.2i gives a result of Fulkerson [1959a]:

Corollary 11.2k. *Let $D = (V, A)$ be a digraph, let $c : A \to \mathbb{R}_+$, and let $a, b : V \to \mathbb{R}$ with $a \leq b$. Then there exists a z-transshipment f satisfying $\mathbf{0} \leq f \leq c$, for some z with $a \leq z \leq b$ if and only if*

$$c(\delta^{\text{in}}(U)) \geq \max\{a(U), -b(V \setminus U)\} \tag{11.16}$$

for each $U \subseteq V$. If moreover a, b, and c are integer, f can be taken integer.

Proof. Directly from Corollary 11.2i, taking $d = \mathbf{0}$. ∎

11.6. Finding circulations and transshipments algorithmically

Algorithmic and complexity results for circulations and transshipments follow directly from those for the maximum flow problem, by the following construction.

Let $D = (V, A)$ be a digraph and let $d, c : A \to \mathbb{Q}$ with $d \leq c$. Then a circulation f satisfying $d \leq f \leq c$ can be found as follows. Give each arc a a new capacity

$$c'(a) := c(a) - d(a). \tag{11.17}$$

Add two new vertices s and t. For each $v \in V$ with $\text{excess}_d(v) > 0$, add an arc (s, v) with capacity $c'(s, v) := \text{excess}_d(v)$. For each $v \in V$ with $\text{excess}_d(v) < 0$, add an arc (v, t) with capacity $c'(v, t) := -\text{excess}_d(v)$. This makes the extended graph $D' = (V', A')$.

Then D has a circulation f satisfying $d \leq f \leq c$ if and only if D' has an $s - t$ flow $f' \leq c'$ of value

$$\sum_{\substack{v \in V \\ \text{excess}_d(v) > 0}} \text{excess}_d(v) \tag{11.18}$$

(by taking $f(a) = f'(a) + d(a)$ for each $a \in A$).

This yields:

Theorem 11.3. *If a maximum flow can be found in time* $\text{MF}(n, m)$*, then a circulation can be found in time* $O(\text{MF}(n, m))$*.*

Proof. Apply the above construction. ▮

If, in addition, functions $a, b : V \to \mathbb{Q}$ are given, we can reduce the problem of finding a transshipment f satisfying $d \leq f \leq c$ and $a \leq \text{excess}_f \leq b$ to finding a circulation in a slightly larger graph — see the proof of Corollary 11.2i. This gives:

Corollary 11.3a. *If a maximum flow can be found in time* $\text{MF}(n, m)$*, then (given* a, b, d, c*) a* z*-transshipment* f *satisfying* $d \leq f \leq c$ *and* $a \leq z \leq b$ *can be found in time* $O(\text{MF}(n, m))$*.*

Proof. Reduce the problem with the construction of Corollary 11.2i to the circulation problem, and use Theorem 11.3. ▮

11.6a. Further notes

The results on flows, circulations, and transshipments extend directly to the case where also each vertex has an upper and/or lower bound on the amount of flow traversing that vertex. We can reduce this to the cases considered above by splitting any vertex v into two vertices v' and v'', adding an arc from v' to v'' with bounds equal to the vertex bounds, and replacing any arc (u, v) by (u'', v').

The results of this chapter also apply to characterizing the existence of a subgraph $D' = (V, A')$ of a given graph $D = (V, A)$, where D' has prescribed bounds on the indegrees and outdegrees (cf. Hakimi [1965]).

Chapter 12

Minimum-cost flows and circulations

Minimum-cost flows can be seen to generalize both shortest path and maximum flow. A shortest $s-t$ path can be deduced from a minimum-cost $s-t$ flow of value 1, while a maximum $s-t$ flow is a minimum-cost $s-t$ flow if we take cost -1 on arcs leaving s and 0 on all other arcs (assuming no arc enters s).
Minimum-cost flows, circulations, and transshipments are closely related, and when describing algorithms, we will choose the most suitable variant. *In this chapter, graphs can be assumed to be simple.*

12.1. Minimum-cost flows and circulations

Let $D=(V,A)$ be a digraph and let $k: A \to \mathbb{R}$, called the *cost* function. For any function $f: A \to \mathbb{R}$, the *cost* of f is, by definition,

$$\text{cost}(f) := \sum_{a \in A} k(a) f(a). \tag{12.1}$$

The *minimum-cost $s-t$ flow problem* is: given a digraph $D=(V,A)$, $s,t \in V$, a 'capacity' function $c: A \to \mathbb{Q}_+$, a 'cost' function $k: A \to \mathbb{Q}$, and a value $\phi \in \mathbb{Q}_+$, find an $s-t$ flow $f \leq c$ of value ϕ that minimizes $\text{cost}(f)$. This problem includes the problem of finding a maximum-value $s-t$ flow that has minimum cost among all maximum-value $s-t$ flows.

Related is the *minimum-cost circulation problem:* given a digraph $D=(V,A)$, a 'demand' function $d: A \to \mathbb{Q}$, a 'capacity' function $c: A \to \mathbb{Q}$, and a 'cost' function $k: A \to \mathbb{Q}$, find a circulation f subject to $d \leq f \leq c$, minimizing $\text{cost}(f)$.

One can easily reduce the minimum-cost flow problem to the minimum-cost circulation problem: just add an arc $a_0 := (t,s)$ with $d(a_0) := c(a_0) := \phi$ and $k(a_0) := 0$. Also, let $d(a) := 0$ for each arc $a \neq a_0$. Then a minimum-cost circulation in the extended digraph gives a minimum-cost flow of value ϕ in the original digraph.

Also the problem of finding a maximum-value $s-t$ flow can be reduced easily to a minimum-cost circulation problem in the extended digraph: now

define $d(a_0) := 0$, $c(a_0) := \infty$, and $k(a_0) := -1$. Moreover, set $k(a) := 0$ for each $a \neq a_0$. Then a minimum-cost circulation gives a maximum-value $s-t$ flow.

Edmonds and Karp [1970,1972] showed that the minimum-cost circulation problem is solvable in polynomial time. Their algorithm is based on a technique called *capacity-scaling* and can be implemented to run in $O(m(m + n\log n)\log C)$ time, where $C := \|c\|_\infty$ (assuming c integer). So it is weakly polynomial-time. They raised the question of the existence of a *strongly* polynomial-time algorithm.

Tardos [1985a] answered this question positively. Her algorithm has resulted in a stream of further research on strongly polynomial-time algorithms for the minimum-cost circulation problem. It stood at the basis of the strongly polynomial-time algorithms discussed in this chapter.

12.2. Minimum-cost circulations and the residual graph D_f

It will be useful again to consider the residual graph $D_f = (V, A_f)$ of $f : A \to \mathbb{R}$ (with respect to d and c), where

$$A_f := \{a \mid a \in A, f(a) < c(a)\} \cup \{a^{-1} \mid a \in A, f(a) > d(a)\}. \tag{12.2}$$

Here $a^{-1} := (v, u)$ if $a = (u, v)$.

We extend any cost function k to A^{-1} by defining the *cost* $k(a^{-1})$ of a^{-1} by:

$$k(a^{-1}) := -k(a) \tag{12.3}$$

for each $a \in A$.

We also use the following notation. Any directed circuit C in D_f gives an undirected circuit in $D = (V, A)$. We define $\chi^C \in \mathbb{R}^A$ by:

$$\chi^C(a) := \begin{cases} 1 & \text{if } C \text{ traverses } a, \\ -1 & \text{if } C \text{ traverses } a^{-1}, \\ 0 & \text{if } C \text{ traverses neither } a \text{ nor } a^{-1}, \end{cases} \tag{12.4}$$

for $a \in A$.

Given $D = (V, A)$, $d, c : A \to \mathbb{R}$, a circulation f in D is called *feasible* if $d \leq f \leq c$. The following observation is fundamental[20]:

Theorem 12.1. *Let $D = (V, A)$ be a digraph and let $d, c, k : A \to \mathbb{R}$. Let $f : A \to \mathbb{R}$ be a feasible circulation. Then f has minimum cost among all feasible circulations if and only if each directed circuit of D_f has nonnegative cost.*

[20] The idea goes back to Tolstoĭ [1930,1939] (for the transportation problem), and was observed also by Robinson [1949,1950] (for the transportation problem), Gallai [1957, 1958b], Busacker and Gowen [1960], Fulkerson [1961], and Klein [1967].

Proof. *Necessity.* Let C be a directed circuit in D_f of negative cost. Then for small enough $\varepsilon > 0$, $f' := f + \varepsilon\chi^C$ is again a circulation satisfying $d \le f' \le c$. Since $\text{cost}(f') < \text{cost}(f)$, f is not minimum-cost.

Sufficiency. Suppose that each directed circuit in D_f has nonnegative cost. Let f' be any feasible circulation. Then $f' - f$ is a circulation, and hence

$$f' - f = \sum_{j=1}^{m} \lambda_j \chi^{C_j} \tag{12.5}$$

for some directed circuits $C_1, \ldots, C_m$ in D_f and $\lambda_1, \ldots, \lambda_m > 0$. Hence

$$\text{cost}(f') - \text{cost}(f) = \text{cost}(f' - f) = \sum_{j=1}^{m} \lambda_j k(C_j) \ge 0. \tag{12.6}$$

So $\text{cost}(f') \ge \text{cost}(f)$. ∎

This directly implies a strong result: optimality of a given feasible circulation f can be checked in polynomial time, namely in time $O(nm)$ (with the Bellman-Ford method). It also implies the following good characterization (Gallai [1957,1958b], Ford and Fulkerson [1962]; for the simpler case of a symmetric cost function satisfying the triangle inequality it was shown by Kantorovich [1942], Kantorovich and Gavurin [1949], and Koopmans and Reiter [1951]):

Corollary 12.1a. *Let $D = (V, A)$ be a digraph, let $d, c, k : A \to \mathbb{R}$, and let f be a feasible circulation. Then f is minimum-cost if and only if there exists a function $p : V \to \mathbb{R}$ such that*

$$\begin{aligned} &k(a) \ge p(v) - p(u) \text{ if } f(a) < c(a), \\ &k(a) \le p(v) - p(u) \text{ if } f(a) > d(a), \end{aligned} \tag{12.7}$$

for each arc $a = (u, v) \in A$.

Proof. Directly from Theorem 12.1 with Theorem 8.2. ∎

From this characterization, a min-max relation for minimum-cost circulations can be derived — see Section 12.5b. It also follows directly from the duality theorem of linear programming — see Chapter 13.

12.3. Strongly polynomial-time algorithm

Theorem 12.1 gives us a method to improve a given circulation f:

(12.8) Choose a negative-cost directed circuit C in the residual graph D_f, and reset $f := f + \tau\chi^C$ where τ is maximal subject to $d \le f \le c$. If no such directed circuit exists, f is a minimum-cost circulation.

It is not difficult to see that for rational data this leads to a finite algorithm.

However, if we just select circuits in an arbitrary fashion, the algorithm may take exponential time, as follows by application to the maximum flow problem given in Figure 10.1 (by adding an arc from t to s of cost -1). Zadeh [1973a,1973b] showed that several strategies of selecting a circuit do not lead to a strongly polynomial-time algorithm.

Goldberg and Tarjan [1988b,1989] were able to prove that one obtains a strongly polynomial-time algorithm if one chooses in (12.8) a directed circuit C of minimum mean cost, that is, one minimizing

$$\frac{k(C)}{|C|}. \tag{12.9}$$

(We identify C with its set AC of arcs.) In Corollary 8.10a we saw that such a circuit can be found in time $O(nm)$.

Note that if we formulate a maximum $s-t$ flow problem as a minimum-cost circulation problem, by adding an arc (t,s) of cost -1 and capacity $+\infty$, then the minimum-mean cost cycle-cancelling algorithm reduces to the shortest augmenting path method of Dinits [1970] and Edmonds and Karp [1972] (Section 10.5).

We now prove the result of Goldberg and Tarjan (as usual, $n := |V|$, $m := |A|$):

Theorem 12.2. *Choosing a minimum-mean cost cycle C in* (12.8)*, the number of iterations is at most $4nm^2\lceil \ln n\rceil$.*

Proof. Let $f_0, f_1, \ldots$ be the circulations found. For each $i \geq 0$, define $A_i := A_{f_i}$, let ε_i be *minus* the minimum of (12.9) in (V, A_i), and let C_i be the directed circuit in A_i chosen to obtain f_{i+1} (taking circuits as arc sets). So

$$k(C_i) = -\varepsilon_i |C_i|. \tag{12.10}$$

ε_i is the smallest value such that if we would add ε_i to the cost of each arc of A_i, then each directed circuit has nonnegative cost. So ε_i is the smallest value for which there exists a function $p_i : V \to \mathbb{Q}$ such that

$$k(a) + \varepsilon_i \geq p_i(v) - p_i(u) \text{ for each } a = (u,v) \in A_i. \tag{12.11}$$

The proof of the theorem is based on the following two facts on the decrease of the ε_i:

$$\text{(i) } \varepsilon_{i+1} \leq \varepsilon_i \text{ and (ii) } \varepsilon_{i+m} \leq (1 - \tfrac{1}{n})\varepsilon_i \tag{12.12}$$

(assuming in (ii) that we reach iteration $i+m$). To prove (12.12), we may assume that $i = 0$ and $p_0 = \mathbf{0}$. Then $k(a) \geq -\varepsilon_0$ for each $a \in A_0$, with equality if $a \in C_0$.

Since $A_1 \subseteq A_0 \cup C_0^{-1}$ and since each arc in C_0^{-1} has cost $\varepsilon_0 \geq 0$, we know that $k(a) \geq -\varepsilon_0$ for each $a \in A_1$. Hence $\varepsilon_1 \leq \varepsilon_0$. This proves (12.12)(i).

To prove (ii), we may assume $\varepsilon_m > 0$. We first show that at least one of the directed circuits $C_0, \ldots, C_{m-1}$ contains an arc a with $k(a) \geq 0$. Otherwise

each A_h arises from A_{h-1} by deleting at least one negative-cost arc and adding only positive-cost arcs. This implies that A_m contains no negative-cost arc, and hence f_m has minimum cost; so $\varepsilon_m = 0$, contradicting our assumption.

Let h be the smallest index such that C_h contains an arc a with $k(a) \geq 0$. So all negative-cost arcs in C_h also belong to A_0, and hence have cost at least $-\varepsilon_0$. So $k(C_h) \geq -(|C_h|-1)\varepsilon_0$ and therefore $\varepsilon_h = -k(C_h)/|C_h| \leq (1-\frac{1}{n})\varepsilon_0$. This proves (12.12).

Now define

$$t := 2nm\lceil \ln n \rceil. \tag{12.13}$$

Then by (12.12)(ii):

$$\varepsilon_t \leq (1-\frac{1}{n})^{2n\lceil \ln n \rceil}\varepsilon_0 < \varepsilon_0/2n, \tag{12.14}$$

since $(1-\frac{1}{n})^n < e^{-1}$ and $e^{-2\ln n} = n^{-2} \leq \frac{1}{2n}$.

We finally show that for each i there exists an arc a in C_i such that $a \notin C_h$ for each $h \geq i+t$. Since $|A \cup A^{-1}| = 2m$, this implies that the number of iterations is at most $4mt$, as required.

To prove this, we may assume that $i = 0$ and that $p_t = \mathbf{0}$. As $k(C_0) = -\varepsilon_0|C_0|$, C_0 contains an arc a_0 with $k(a_0) \leq -\varepsilon_0 < -2n\varepsilon_t$. Without loss of generality, $a_0 \in A$.

Suppose that $f_h(a_0) \neq f_t(a_0)$ for some $h > t$. Since $k(a_0) < -2n\varepsilon_t \leq -\varepsilon_t$, we have that $a_0 \notin A_t$ (by (12.11)). So $f_t(a_0) = c(a_0)$, and hence $f_h(a_0) < f_t(a_0)$. Then, by (11.3) applied to $f_t - f_h$, A_h has a directed circuit C containing a_0 such that A_t contains C^{-1}. By (12.11), $-k(a) = k(a^{-1}) \geq -\varepsilon_t$ for each $a \in C$. This gives (using (12.12)(i)):

$$\begin{aligned} k(C) &= k(a_0) + k(C \setminus \{a_0\}) < -2n\varepsilon_t + (|C|-1)\varepsilon_t \leq -n\varepsilon_t \\ &\leq -n\varepsilon_h \leq -|C|\varepsilon_h, \end{aligned} \tag{12.15}$$

contradicting the definition of ε_h. ▮

This gives for finding a minimum-cost circulation:

Corollary 12.2a. *A minimum-cost circulation can be found in $O(n^2m^3 \log n)$ time. If d and c are integer, an integer minimum-cost circulation is found.*

Proof. Directly from Theorem 12.2 and Corollary 8.10a. Note that if d and c are integer and we start with an integer circulation f_0, all further circulations obtained by (12.8) are integer. ▮

So we have the theorem of Tardos [1985a]:

Corollary 12.2b. *A minimum-cost circulation can be found in strongly polynomial time. If d and c are integer, an integer circulation is found.*

Proof. Directly from Corollary 12.2a. ▮

Notes. Goldberg and Tarjan [1988b,1989] showed that, with the help of dynamic trees, the running time of the minimum-mean cost cycle-cancelling method can be improved to $O(nm \log n \min\{\log(nK), m \log n\})$, where $K := \|k\|_\infty$, assuming k to be integer.

Weintraub [1974] showed that if we take always a directed circuit C in D_f such that, by resetting f to $f + \tau\chi^C$ as in (12.8), the cost decreases most, then the number of iterations (12.8) is polynomially bounded. However, finding such a circuit is NP-complete (finding a Hamiltonian circuit in a directed graph is a special case). Weintraub [1974] also proposed a heuristic of finding a short (negative) circuit by finding a minimum-cost set of vertex-disjoint circuits in D_f (by solving an assignment problem), and choosing the shortest among them. Barahona and Tardos [1989] showed that this also leads to a (weakly) polynomial-time algorithm.

12.4. Related problems

Corollary 12.2b concerns solving the minimization problem in the following LP-duality equation, where M denotes the $V \times A$ incidence matrix of D:

$$\begin{aligned}(12.16)\qquad &\min\{k^\mathsf{T}x \mid d \le x \le c, Mx = \mathbf{0}\}\\ &= \max\{z_1^\mathsf{T}d - z_2^\mathsf{T}c \mid z_1, z_2 \ge \mathbf{0}, \exists y : z_1^\mathsf{T} - z_2^\mathsf{T} + y^\mathsf{T}M = k^\mathsf{T}\}.\end{aligned}$$

It implies that also the maximization problem can be solved in strongly polynomial time:

Corollary 12.2c. *The maximization problem in* (12.16) *can be solved in strongly polynomial time. If k is integer, an integer optimum solution is found.*

Proof. Let x be an optimum solution of the minimization problem, that is, a minimum-cost circulation. Since x is extreme, the digraph D_x has no negative-cost directed circuits. Hence we can find a function ('potential') $y : V \to \mathbb{Q}$ such that $k(u,v) \ge y(v) - y(u)$ if $x(u,v) < c(u,v)$ and $k(u,v) \le y(v) - y(u)$ if $x(u,v) > d(u,v)$, in strongly polynomial time (Theorem 8.7). If k is integer, we find an integer y.

Let z_1 and z_2 be the unique vectors with $z_1, z_2 \ge \mathbf{0}$, $z_1^\mathsf{T} - z_2^\mathsf{T} = k^\mathsf{T} - y^\mathsf{T}M$ and $z_1(a)z_2(a) = 0$ for each $a \in A$. So $z_1(a) = 0$ if $x(a) > d(a)$ and $z_2(a) = 0$ if $x(a) < c(a)$. Hence

$$(12.17)\qquad z_1^\mathsf{T}d - z_2^\mathsf{T}c = z_1^\mathsf{T}x - z_2^\mathsf{T}x = (k^\mathsf{T} - y^\mathsf{T}M)x = k^\mathsf{T}x.$$

So z_1, z_2 form an optimum solution for the maximization problem. ∎

By an easy construction, Corollary 12.2b implies that a more general problem is solvable in strongly polynomial time:

(12.18) input: a digraph $D = (V, A)$ and functions $a, b : V \to \mathbb{Q}$ and $d, c, k : A \to \mathbb{Q}$,

find: a z-transshipment x with $a \leq z \leq b$ and $d \leq x \leq c$, minimizing $k^{\mathsf{T}}x$.

Corollary 12.2d. *Problem* (12.18) *is solvable in strongly polynomial time. If a, b, d, and c are integer, an integer optimum solution is found.*

Proof. Extend D by a new vertex u and arcs (v,u) for each $v \in V$. Extend d, c, and k by defining $d(v,u) := a(v)$, $c(v,u) := b(v)$ and $k(v,u) := 0$ for each $v \in V$. By Corollary 12.2b, we can find a minimum-cost circulation x with $d \leq x \leq c$ in the extended digraph in strongly polynomial time. It gives a b-transshipment in the original graph as required. ▮

For later reference, we derive that also the dual problem (in the LP sense) can be solved in strongly polynomial time. If M denotes $V \times A$ incidence matrix of D, problem (12.18) corresponds to the minimum in the LP-duality equation:

$$\begin{aligned}(12.19)\qquad &\min\{k^{\mathsf{T}}x \mid d \leq x \leq c, a \leq Mx \leq b\}\\ &= \max\{y_1^{\mathsf{T}}b - y_2^{\mathsf{T}}a + z_1^{\mathsf{T}}d - z_2^{\mathsf{T}}c \mid y_1, y_2, z_1, z_2 \geq \mathbf{0},\\ &\qquad\qquad (y_1 - y_2)^{\mathsf{T}}M + (z_1 - z_2)^{\mathsf{T}} = k^{\mathsf{T}}\}.\end{aligned}$$

Corollary 12.2e. *An optimum solution for the maximum in* (12.19) *can be found in strongly polynomial time. If k is integer, we find an integer optimum solution.*

Proof. By reduction to Corollary 12.2c, using a reduction similar to that given in the proof of Corollary 12.2d. ▮

12.4a. A dual approach

The approach above consists of keeping a feasible circulation, and throughout improving its cost. A dual approach can best be described in terms of b-transshipments: we keep a b'-transshipment f such that D_f has no negative-cost directed circuits, and improve b' until $b' = b$. This can be studied with the concept of 'extreme function'.

Let $D = (V,A)$ be a digraph and let $d, c, k : A \to \mathbb{R}$ be given, the lower bound function, the capacity function, and the cost function, respectively. Let $f : A \to \mathbb{R}$ be such that $d \leq f \leq c$. We call f *extreme* if $\text{cost}(f') \geq \text{cost}(f)$ for each function f' satisfying $d \leq f' \leq c$ and $\text{excess}_{f'} = \text{excess}_f$; in other words, setting $b := \text{excess}_f$, f is a minimum-cost b-transshipment subject to $d \leq f \leq c$. (excess_f is defined in Section 10.1.)

Note that the concept of extreme depends on k, d, and c. So it might be better to define a function to be extreme *with respect to* k, d, and c. However, when considering extreme functions f, the functions k, d, and c are generally fixed, or follow from the context.

Again it will be useful to consider the residual graph $D_f = (V, A_f)$ of f (with respect to d and c), where

$$A_f := \{a \mid a \in A, f(a) < c(a)\} \cup \{a^{-1} \mid a \in A, f(a) > d(a)\}. \tag{12.20}$$

Here $a^{-1} := (v, u)$ if $a = (u, v)$.

We extend k to $A^{-1} := \{a^{-1} \mid a \in A\}$ by defining

$$k(a^{-1}) := -k(a) \tag{12.21}$$

for each $a \in A$. We call $k(a^{-1})$ the *cost* of a^{-1}.

We also use the following notation. Any directed path P in D_f gives an undirected path in $D = (V, A)$. We define $\chi^P \in \mathbb{R}^A$ by:

$$\chi^P(a) := \begin{cases} 1 & \text{if } P \text{ traverses } a, \\ -1 & \text{if } P \text{ traverses } a^{-1}, \\ 0 & \text{if } P \text{ traverses neither } a \text{ nor } a^{-1}, \end{cases} \tag{12.22}$$

for $a \in A$.

Theorem 12.1 for minimum-cost circulations can be directly extended to extreme functions:

Theorem 12.3. *Let $D = (V, A)$ be a digraph and let $d, c, k, f : A \to \mathbb{R}$ with $d \leq f \leq c$. Then f is extreme if and only if each directed circuit of D_f has nonnegative cost.*

Proof. Like the proof of Theorem 12.1. ▮

This implies that the optimality of a given feasible solution f of a b-transshipment problem can be checked in polynomial time, namely in time $O(nm)$ (with the Bellman-Ford method). It also implies the following good characterization (Kantorovich [1942], Gallai [1957,1958b], Ford and Fulkerson [1962]):

Corollary 12.3a. *Let $D = (V, A)$ be a digraph and let $d, c, k, f : A \to \mathbb{R}$ with $d \leq f \leq c$. Then f is extreme if and only if there exists a function $p : V \to \mathbb{R}$ such that*

$$\begin{aligned} &k(a) \geq p(v) - p(u) \text{ if } f(a) < c(a), \\ &k(a) \leq p(v) - p(u) \text{ if } f(a) > d(a), \end{aligned} \tag{12.23}$$

for each arc $a = (u, v) \in A$.

Proof. Directly from Theorem 12.3 with Theorem 8.2. ▮

As for the algorithmic side, the following observation (Jewell [1958], Busacker and Gowen [1960], Iri [1960]) is very useful in analyzing algorithms ('This theorem may properly be regarded as the central one concerning minimal cost flows' — Ford and Fulkerson [1962]):

Theorem 12.4. *Let $D = (V, A)$ be a digraph and let $d, c, k, f : A \to \mathbb{R}$ with $d \leq f \leq c$ and with f extreme. Let P be a minimum-cost $s-t$ path in D_f, for some $s, t \in V$, and let $\varepsilon > 0$ be such that $f' := f + \varepsilon\chi^P$ satisfies $d \leq f' \leq c$. Then f' is extreme again.*

Proof. Let f'' satisfy $d \leq f'' \leq c$ and $\text{excess}_{f''} = \text{excess}_{f'}$. Then by Theorem 11.1,

$$f'' - f = \sum_{i=1}^{n} \mu_i \chi^{P_i} + \sum_{j=1}^{m} \lambda_j \chi^{C_j}, \tag{12.24}$$

where $P_1, \ldots, P_n$ are $s-t$ paths in D_f, $C_1, \ldots, C_m$ are directed circuits in D_f, $\mu_1, \ldots, \mu_n > 0$, and $\lambda_1, \ldots, \lambda_m > 0$, with $\sum_i \mu_i = \varepsilon$. Then

$$\text{cost}(f'' - f) = \sum_i \mu_i \cdot \text{cost}(P_i) + \sum_j \lambda_j \cdot \text{cost}(C_j) \geq \sum_i \mu_i \tau = \varepsilon\tau, \tag{12.25}$$

where $\tau := \text{cost}(P)$. As $\text{cost}(f' - f) = \varepsilon\tau$, we have $\text{cost}(f'' - f) \geq \text{cost}(f' - f)$, and therefore $\text{cost}(f'') \geq \text{cost}(f')$. ∎

We will refer to updating f to $f + \varepsilon\chi^P$ as in Theorem 12.4 as to *sending a flow of value ε over P*.

Also the following observation is useful in algorithms (Edmonds and Karp [1970], Tomizawa [1971]):

Theorem 12.5. *In Theorem* 12.4*, if p is a potential for D_f such that $p(t) - p(s) = \text{dist}_k(s,t)$, then p is also a potential for $D_{f'}$.*

Proof. Choose $a = (u,v) \in A_{f'}$. If $a \in A_f$, then $p(v) \leq p(u) + k(a)$. If $a \notin A_f$, then a^{-1} is traversed by P, and hence $p(u) = p(v) + k(a^{-1}) = p(v) - k(a)$. Therefore $p(v) \leq p(u) + k(a)$. ∎

These theorems lead to the following minimum-cost $s-t$ flow algorithm due to Ford and Fulkerson [1958b], Jewell [1958], Busacker and Gowen [1960], and Iri [1960] (an equivalent 'primal-dual' algorithm was given by Fujisawa [1959]).

Let be given $D = (V, A)$, $s, t \in V$, and $c, k : A \to \mathbb{Q}_+$, the capacity and cost function, respectively.

Algorithm for minimum-cost $s-t$ flow
Starting with $f = \mathbf{0}$ apply the following iteratively:
Iteration: Let P be an $s-t$ path in D_f minimizing $k(P)$. Reset $f := f + \varepsilon\chi^P$, where ε is maximal subject to $\mathbf{0} \leq f + \varepsilon \cdot \chi^P \leq c$.

Termination of this algorithm for rational capacities follows similarly as for the maximum flow algorithm (Theorem 10.4).

One may use the Bellman-Ford method to obtain the path P, since D_f has no negative-cost directed circuits (by Theorems 12.3 and 12.4). However, using a trick of Edmonds and Karp [1970] and Tomizawa [1971], one can use Dijkstra's algorithm, since by Theorem 12.5 we can maintain a potential that makes all lengths (= costs) nonnegative. This leads to the following theorem (where $\text{SP}_+(n,m,K)$ denotes the time needed to find a shortest path in a digraph with n vertices, m arcs, and *nonnegative* integer lengths, each at most K):

Theorem 12.6. *For $c, k : A \to \mathbb{Z}_+$ and $\phi \in \mathbb{Z}_+$, a minimum-cost $s-t$ flow $f \leq c$ of value ϕ can be found in time $O(\phi \cdot \text{SP}_+(n,m,K))$, where $K := \|k\|_\infty$.*

Proof. Note that each iteration consists of finding a shortest path in D_f. Simultaneously we can find a potential for D_f satisfying $p(t) - p(s) = \text{dist}_k(s,t)$. Since by Theorem 12.5, p is a potential also for $D_{f'}$, we can perform each iteration in time $\text{SP}_+(n,m,K)$. ∎

12.4b. A strongly polynomial-time algorithm using capacity-scaling

The algorithm given in Theorem 12.6 is not polynomial-time, but several improvements leading to a polynomial-time algorithm have been found. Orlin [1988,1993] gave the currently fastest strongly polynomial-time algorithm for minimum-cost circulation, which is based on this dual approach: while keeping an extreme b'-transshipment, it throughout improves b', until $b' = b$. This implies an algorithm for minimum-cost circulation.

Let $D = (V, A)$ be a digraph, let $k : A \to \mathbb{Q}_+$ be a cost function, and let $b : V \to \mathbb{Q}$ be such that there exists a nonnegative b-transshipment. For any $f : A \to \mathbb{Q}$ define $\mathrm{def}_f : V \to \mathbb{Q}$ by

(12.26) $\mathrm{def}_f := b - \mathrm{excess}_f$.

So $\mathrm{def}_f(v)$ is the 'deficiency' of f at v. Then $\mathrm{def}_f(V) = b(V) - \mathrm{excess}_f(V) = 0$.

The algorithm determines a sequence of functions $f_i : A \to \mathbb{Q}_+$ and rationals β_i $(i = 0, 1, 2, \ldots)$. Initially, set $f_0 := \mathbf{0}$ and $\beta_0 := \|b\|_\infty$. If f_i and β_i have been found, we find f_{i+1} and β_{i+1} by the following iteration (later referred to as *iteration* i).

Let A_i be the set of arcs a with $f_i(a) > 12n\beta_i$ and let $\mathcal{K}_i$ be the collection of weak components of the digraph (V, A_i). We are going to update a function g_i starting with $g_i := f_i$.

(12.27) If there exists a component $K \in \mathcal{K}_i$ and distinct $u, v \in K$ with $|\mathrm{def}_{g_i}(u)| \geq |\mathrm{def}_{g_i}(v)| > 0$, then update g_i by sending a flow of value $|\mathrm{def}_{g_i}(v)|$ from v to u or conversely along a path in A_i, so as to make $\mathrm{def}_{g_i}(v)$ equal to 0.

(In (12.32) it is shown that this is possible, and that it does not modify D_{g_i}, hence g_i remains extreme.) We iterate (12.27), so that finally each $K \in \mathcal{K}_i$ contains at most one vertex u with $\mathrm{def}_{g_i}(u) \neq 0$.

Next do the following repeatedly, as long as there exists a $u \in V$ with $|\mathrm{def}_{g_i}(u)| > \frac{n-1}{n}\beta_i$:

(12.28) If $\mathrm{def}_{g_i}(u) > \frac{n-1}{n}\beta_i$, then there exists a $v \in V$ such that $\mathrm{def}_{g_i}(v) < -\frac{1}{n}\beta_i$ and such that u reachable from v in the residual graph D_{g_i}. Update g_i by sending a flow of value β_i along a minimum-cost $v - u$ path in D_{g_i}.
If $\mathrm{def}_{g_i}(u) < -\frac{n-1}{n}\beta_i$, proceed symmetrically.

(The existence of v in (12.28) follows from the assumption that there exists a nonnegative b-transshipment, f say, by applying Theorem 11.1 to $f - g_i$. The fact that we can send a flow of value β_i in the residual graph follows from (12.36).)

When we cannot apply (12.27) anymore, we define $f_{i+1} := g_i$. Let $T := \|\mathrm{def}_{f_{i+1}}\|_\infty$. If $T = 0$ we stop. Otherwise, define:

(12.29) $$\beta_{i+1} := \begin{cases} \frac{1}{2}\beta_i & \text{if } T \geq \frac{1}{12n}\beta_i, \\ T & \text{if } 0 < T < \frac{1}{12n}\beta_i, \end{cases}$$

and iterate.

Theorem 12.7. *The algorithm stops after at most n iterations of* (12.27) *and at most $O(n \log n)$ iterations of* (12.28).

Proof. Throughout the proof we assume $n \geq 2$. We first observe that for each i:

(12.30) $$\|\mathrm{def}_{f_{i+1}}\|_\infty \leq \frac{n-1}{n}\beta_i,$$

since otherwise we could have applied (12.28) to the final $g_i(= f_{i+1})$. This implies that for each i:

(12.31) $$\|\mathrm{def}_{f_i}\|_\infty \leq 2\beta_i.$$

This is direct for $i = 0$. If $\beta_{i+1} = \frac{1}{2}\beta_i$, then, by (12.30), $\|\mathrm{def}_{f_{i+1}}\|_\infty \leq \frac{n-1}{n}\beta_i \leq \beta_i = 2\beta_{i+1}$. If $\beta_{i+1} < \frac{1}{2}\beta_i$, then $\|\mathrm{def}_{f_{i+1}}\|_\infty = T = \beta_{i+1} \leq 2\beta_{i+1}$. This proves (12.31).

We next show, that, for any i:

(12.32) in the iterations (12.27) and (12.28), for any $a \in A_i$ the value of $g_i(a)$ remains more than $6n\beta_i$.

In each iteration (12.27), for any arc $a \in A_i$, the value of $g_i(a)$ changes by at most $\|\mathrm{def}_{f_i}\|_\infty$, which is at most $2\beta_i$ (by (12.31)). For any fixed i, we apply (12.27) at most n times. So the value of $g_i(a)$ on any arc $a \in A_i$ changes by at most $2n\beta_i$.

In the iterations (12.28), the value of $g_i(a)$ changes by at most $4n\beta_i$. To see this, consider the sum

(12.33) $$\sum_{\substack{v \in V \\ |\mathrm{def}_{g_i}(v)| > \frac{n-1}{n}\beta_i}} |\mathrm{def}_{g_i}(v)|.$$

In each iteration (12.28), this sum decreases by at least $\frac{n-1}{n}\beta_i$, which is at least $\frac{1}{2}\beta_i$. On the other hand, $g_i(a)$ changes by at most β_i. Since (12.33) initially is at most $\|\mathrm{def}_{g_i}\|_1 \leq \|\mathrm{def}_{f_i}\|_1 \leq 2n\beta_i$, we conclude that in the iterations (12.28), $g_i(a)$ changes by at most $4n\beta_i$.

Concluding, in the iterations (12.27) and (12.28), any $g_i(a)$ changes by at most $6n\beta_i$. Since at the beginning of these iterations we have $g_i(a) > 12n\beta_i$ for $a \in A_i$, this proves (12.32).

(12.32) implies that in iteration (12.27) we can make $\mathrm{def}_{g_i}(v)$ equal to 0. (After that it will remain 0.) Hence, iteration (12.27) can be applied at most n times in total (over all i), since each time the number of vertices v with $\mathrm{def}_{f_i}(v) \neq 0$ drops.

(12.32) also implies:

(12.34) each f_i is extreme.

This is clearly true if $i = 0$ (since the cost function k is nonnegative). Suppose that f_i is extreme. Then also g_i is extreme initially, and remains extreme during the iterations (12.27) (since by (12.32) the residual graph D_{g_i} does not change during the iterations (12.27)). Moreover, also during the iterations (12.28) the function g_i remains extreme, since we send flow over a minimum-cost path in D_{g_i} (Theorem 12.4). This proves (12.34).

Directly from (12.32) we have, for each i:

(12.35) $$A_i \subseteq A_{i+1},$$

since $\beta_{i+1} \leq \frac{1}{2}\beta_i$. This implies that each set in $\mathcal{K}_i$ is contained in some set in $\mathcal{K}_{i+1}$.

Next, throughout iteration i,

(12.36) If $a \in A \setminus A_i$, then $\beta_i | g_i(a)$.

The proof is by induction on i, the case $i = 0$ being trivial (since for $i = 0$ we do not apply (12.27), as $A_0 = \emptyset$). Suppose that we know (12.36). Choose $a \in A \setminus A_{i+1}$. Then $a \in A \setminus A_i$ by (12.35). Hence $\beta_i | g_i(a)$, and so $\beta_i | f_{i+1}(a)$. If $f_{i+1}(a) > 0$ and $\beta_{i+1} < \frac{1}{2}\beta_i$, then $\beta_i > 12nT = 12n\|\mathrm{def}_{f_{i+1}}\|_\infty$, and hence

$$f_{i+1}(a) \geq \beta_i > 12n\|\mathrm{def}_{f_{i+1}}\|_\infty = 12n\beta_{i+1}, \tag{12.37}$$

contradicting the fact that a does not belong to A_{i+1}.

So $f_{i+1}(a) = 0$ or $\beta_{i+1} = \frac{1}{2}\beta_i$. This implies that $\beta_{i+1}|f_{i+1}(a)$. In iteration $i+1$, only flow packages of size β_{i+1} are sent over arc a (since $a \notin A_{i+1}$). Therefore, throughout iteration $i+1$ we have $\beta_{i+1}|g_{i+1}(a)$, which proves (12.36).

So in iteration (12.28) we indeed can send a flow of value β_i in the residual graph D_{g_i}. Since f_{i+1} is equal to the final g_i, (12.36) also implies:

(12.38) if $a \in A \setminus A_i$, then $\beta_i | f_{i+1}(a)$.

Next we come to the kernel in the proof, which gives two bounds on $b(K)$ for $K \in \mathcal{K}_i$:

Claim 1. For each i and each $K \in \mathcal{K}_i$:

(12.39)
- (i) $|b(K)| \leq 13n^3\beta_i$;
- (ii) *suppose $i > 0$, $K \in \mathcal{K}_{i-1}$, and (12.28) is applied to a vertex u in K; then $|b(K)| \geq \frac{1}{n}\beta_i$.*

Proof of Claim 1. I. We first show (12.39)(i). If $i = 0$, then $|b(K)| \leq n\|b\|_\infty = n\beta_0 \leq 13n^3\beta_0$. If $i > 0$, then, by (12.31),

$$|\mathrm{def}_{f_i}(K)| \leq n\|\mathrm{def}_{f_i}\|_\infty \leq 2n\beta_i. \tag{12.40}$$

Moreover, since $f_i(a) \leq 12n\beta_i$ for each $a \in \delta^{\mathrm{in}}(K)$,

$$|\mathrm{excess}_{f_i}(K)| \leq 12n\beta_i \cdot |\delta^{\mathrm{in}}(K)| \leq 12n^3\beta_i. \tag{12.41}$$

Hence

$$|b(K)| \leq |\mathrm{def}_{f_i}(K)| + |\mathrm{excess}_{f_i}(K)| \leq 2n\beta_i + 12n^3\beta_i \leq 13n^3\beta_i. \tag{12.42}$$

This proves (12.39)(i).

II. Next we show (12.39)(ii). Since $K \in \mathcal{K}_{i-1} \cap \mathcal{K}_i$, u is the only vertex in K with $\mathrm{def}_{f_i}(u) \neq 0$. So, in iteration i, we do not apply (12.27) to a vertex in K. Moreover, by applying (12.28), $|\mathrm{def}_{g_i}(K)|$ does not increase. This gives

$$\frac{n-1}{n}\beta_i < |\mathrm{def}_{g_i}(K)| \leq |\mathrm{def}_{f_i}(K)| \leq \frac{n-1}{n}\beta_{i-1}. \tag{12.43}$$

The first inequality holds as we apply (12.28) to u, and the last inequality follows from (12.30).

To prove (12.39)(ii), first assume $\beta_i = \frac{1}{2}\beta_{i-1}$. Since $\mathrm{def}_{f_i}(K) = \mathrm{def}_{f_i}(u)$, we have, by (12.43),

$$\frac{n-1}{2n}\beta_{i-1} = \frac{n-1}{n}\beta_i \leq |\mathrm{def}_{f_i}(K)| \leq \frac{n-1}{n}\beta_{i-1}. \tag{12.44}$$

So $|\mathrm{def}_{f_i}(K)|/\beta_{i-1}$ has distance at least $1/2n$ to $\mathbb{Z}$. Since $f_i(a) \equiv 0 \pmod{\beta_{i-1}}$ by (12.38), we have

$$|b(K)|/\beta_{i-1} \equiv |\mathrm{def}_{f_i}(K)|/\beta_{i-1} \pmod 1. \tag{12.45}$$

Hence also $|b(K)|/\beta_{i-1}$ has distance at least $1/2n$ to $\mathbb{Z}$. So $|b(K)| \geq \frac{1}{2n}\beta_{i-1} \geq \frac{1}{n}\beta_i$, as required.

Second assume $\beta_i < \frac{1}{2}\beta_{i-1}$. Then $\beta_i = \|\mathrm{def}_{f_i}\|_\infty < \frac{1}{12n}\beta_{i-1}$. Now as $K \in \mathcal{K}_i$, we have for each $a \in \delta(K)$: $0 \leq f_i(a) < 12n\beta_i < \beta_{i-1}$, while $f_i(a) \equiv 0 \pmod{\beta_{i-1}}$ by (12.38). So $f_i(a) = 0$ for each $a \in \delta(K)$. Hence by (12.43),

$$|b(K)| = |\mathrm{def}_{f_i}(K)| \geq \frac{n-1}{n}\beta_i \geq \frac{1}{n}\beta_i, \tag{12.46}$$

which proves (12.39)(ii). *End of Proof of Claim 1*

Define $\mathcal{K}^* := \bigcup_i \mathcal{K}_i$, and consider any $K \in \mathcal{K}^*$. Let I be the set of i with $K \in \mathcal{K}_i$. Let t be the smallest element of I. Let λ_K be the number of components in $\mathcal{K}_{t-1}$ contained in K. (Set $\lambda_K := 1$ if $t = 0$.) Then in iteration t, (12.28) is applied at most $4\lambda_K$ times to a vertex u in K, since (at the start of applying (12.28))

$$|\mathrm{def}_{g_t}(K)| = |\mathrm{def}_{f_t}(K)| \leq \lambda_K \|\mathrm{def}_{f_t}\|_\infty \leq 2\lambda_K \beta_t \tag{12.47}$$

(by (12.31)). In any further iteration $i > t$ with $i \in I$, (12.28) is applied at most twice to u (again by (12.31)).

We estimate now the number of iterations $i \in I$ in which (12.28) is applied to $u \in K$. Consider the smallest such i with $i > t$. Then:

$$\text{if } j > i + \log_2(13n^4), \text{ then } j \notin I. \tag{12.48}$$

For suppose to the contrary that $K \in \mathcal{K}_j$. Since $\beta_i \geq 2^{j-i}\beta_j > 13n^4\beta_j$, Claim 1 gives the contradiction $b(K) \leq 13n^3\beta_j < \frac{1}{n}\beta_i \leq b(K)$. This proves (12.48).

So (12.28) is applied at most $4\lambda_K + 2\log_2(13n^4)$ times to a vertex $u \in K$, in iterations $i \in I$. Since

$$\sum_{K \in \mathcal{K}^*} \lambda_K \leq |\mathcal{K}^*| < 2n \tag{12.49}$$

(as $\mathcal{K}^*$ is laminar — cf. Theorem 3.5), (12.28) is applied at most $8n + 4n\log_2(13n^4)$ times in total. ∎

This bound on the number of iterations gives:

Corollary 12.7a. *A minimum-cost nonnegative b-transshipment can be found in time $O(n \log n(m + n \log n))$.*

Proof. Directly from Theorem 12.7, since any iteration (12.27) or (12.28) takes $O(m + n \log n)$ time, using Fibonacci heaps (Corollary 7.7a) and maintaining a potential as in Theorem 12.5. ∎

We can derive a bound for finding a minimum-cost circulation:

Corollary 12.7b. *A minimum-cost circulation can be found in time $O(m \log n(m + n \log n))$.*

Proof. The minimum-cost circulation problem can be reduced to the minimum-cost transshipment problem as follows. Let $D = (V, A)$, $d, c, k : A \to \mathbb{Q}$ be the input for the minimum-cost circulation problem. Define

$$b(v) := d(\delta^{\mathrm{out}}(v)) - d(\delta^{\mathrm{in}}(v)) \tag{12.50}$$

for each $v \in V$. Then any minimum-cost b-transshipment x satisfying $\mathbf{0} \leq x \leq c - d$ gives a minimum-cost circulation $x' := x + d$ satisfying $d \leq x' \leq c$. So we can assume $d = \mathbf{0}$.

Now replace each arc $a = (u, v)$ by three arcs (u, u_a), (v_a, u_a), and (v_a, v), where u_a and v_a are new vertices. This makes the digraph D' say.

Define $b(u_a) := c(a)$ and $b(v_a) := -c(a)$. Moreover, define a cost function k' on the arcs of D' by $k'(u, u_a) := k(a)$, $k'(v_a, u_a) := 0$, $k'(v_a, v) := 0$ if $k(a) \geq 0$, and $k'(u, u_a) := 0$, $k'(v_a, u_a) := -k(a)$, $k'(v_a, v) := 0$ if $k(a) < 0$. Then a minimum-cost b-transshipment $x \geq \mathbf{0}$ in D' gives a minimum-cost b-transshipment x satisfying $\mathbf{0} \leq x \leq c$ in the original digraph D.

By Theorem 12.7, a minimum-cost b-transshipment $x \geq \mathbf{0}$ in D' can be found by finding $O(n \log n)$ times a shortest path in a residual graph D'_x. While this digraph has $2m + n$ vertices, it can be reduced in $O(m)$ time to finding a shortest path in an auxiliary digraph with $O(n)$ vertices only. Hence again it takes $O(m + n \log n)$ time by using Fibonacci heaps (Corollary 7.7a) and maintaining a potential as in Theorem 12.5. ∎

12.5. Further results and notes

12.5a. Complexity survey for minimum-cost circulation

Complexity survey for minimum-cost circulation ($*$ indicates an asymptotically best bound in the table):

	$O(n^4CK)$	Ford and Fulkerson [1958b] labeling
	$O(m^3C)$	Yakovleva [1959], Minty [1960], Fulkerson [1961] out-of-kilter method
	$O(nm^2C)$	Busacker and Gowen [1960], Iri [1960] successive shortest paths
$*$	$O(nC \cdot \mathrm{SP}_+(n, m, K))$	Edmonds and Karp [1970], Tomizawa [1971] successive shortest paths with nonnegative lengths using vertex potentials
	$O(nK \cdot \mathrm{MF}(n, m, C))$	Edmonds and Karp [1972]
$*$	$O(m \log C \cdot \mathrm{SP}_+(n, m, K))$	Edmonds and Karp [1972] capacity-scaling
	$O(nm \log(nC))$	Dinits [1973a] capacity-scaling
	$O(n \log K \cdot \mathrm{MF}(n, m, C))$	Röck [1980] (cf. Bland and Jensen [1992]) cost-scaling
	$O(m^2 \log n \cdot \mathrm{MF}(n, m, C))$	Tardos [1985a]
	$O(m^2 \log n \cdot \mathrm{SP}_+(n, m, K))$	Orlin [1984a], Fujishige [1986]
	$O(n^2 \log n \cdot \mathrm{SP}_+(n, m, K))$	Galil and Tardos [1986,1988]
	$O(n^3 \log(nK))$	Goldberg and Tarjan [1987], Bertsekas and Eckstein [1988]
$*$	$O(n^{5/3}m^{2/3} \log(nK))$	Goldberg and Tarjan [1987] generalized cost-scaling

≫

continued

	$O(nm \log n \log(nK))$	Goldberg and Tarjan [1987] generalized cost-scaling; Goldberg and Tarjan [1988b,1989] minimum-mean cost cycle-cancelling
*	$O(m \log n \cdot \mathrm{SP}_+(n,m,K))$	Orlin [1988,1993]
*	$O(nm \log(n^2/m) \log(nK))$	Goldberg and Tarjan [1990] generalized cost-scaling
*	$O(nm \log\log C \log(nK))$	Ahuja, Goldberg, Orlin, and Tarjan [1992] double scaling
*	$O(n \log C(m + n \log n))$	*circulations with lower bounds only* Gabow and Tarjan [1989]
*	$O((m^{3/2}C^{1/2} + \gamma \log \gamma) \log(nK))$	Gabow and Tarjan [1989]
*	$O((nm + \gamma \log \gamma) \log(nK))$	Gabow and Tarjan [1989]

Here $K := \|k\|_\infty$, $C := \|c\|_\infty$, and $\gamma := \|c\|_1$, for integer cost function k and integer capacity function c. Moreover, $\mathrm{SP}_+(n,m,K)$ denotes the running time of any algorithm finding a shortest path in a digraph with n vertices, m arcs, and nonnegative integer length function l with $K = \|l\|_\infty$. Similarly, $\mathrm{MF}(n,m,C)$ denotes the running time of any algorithm finding a maximum flow in a digraph with n vertices, m arcs, and nonnegative integer capacity function c with $C = \|c\|_\infty$.

Complexity survey for minimum-cost nonnegative transshipment:

*	$O(n \log B \cdot \mathrm{SP}_+(n,m,K))$	Edmonds and Karp [1970,1972]
	$O(n^2 \log n \cdot \mathrm{SP}_+(n,m,K))$	Galil and Tardos [1986,1988]
*	$O(n \log n \cdot \mathrm{SP}_+(n,m,K))$	Orlin [1988,1993]

Here $B := \|b\|_\infty$ for integer b.

12.5b. Min-max relations for minimum-cost flows and circulations

From Corollary 12.1a, the following min-max equality for minimum-cost circulation can be derived. The equality also follows directly from linear programming duality and total unimodularity. Both approaches were considered by Gallai [1957,1958a, 1958b].

Theorem 12.8. *Let $D = (V, A)$ be a digraph and let $c, d, k : A \to \mathbb{R}$. Then the minimum of $\sum_{a \in A} k(a)f(a)$ taken over all circulations f in D satisfying $d \le f \le c$ is equal to the maximum value of*

$$\sum_{a \in A} (y(a)d(a) - z(a)c(a)), \tag{12.51}$$

where $y, z : A \to \mathbb{R}_+$ are such that there exists a function $p : V \to \mathbb{R}$ with the property that

(12.52) $y(a) - z(a) = k(a) - p(v) + p(u)$

for each arc $a = (u, v)$ of D.

If d and c are integer, we can take f integer. If k is integer, we can take y and z integer.

Proof. The minimum is not less than the maximum, since if f is any circulation in D with $d \le f \le c$ and y, z, p satisfy (12.52), then

$$\sum_{a\in A} k(a)f(a) = \sum_{a=(u,v)\in A} (k(a)+p(u)-p(v))f(a) = \sum_{a\in A}(y(a)-z(a))f(a) \ge \sum_{a\in A}(y(a)d(a) - z(a)c(a)). \tag{12.53}$$

To see equality, let f be a minimum-cost circulation. By Corollary 12.1a, there is a function $p : V \to \mathbb{R}$ satisfying (12.7). Define for each arc $a = (u, v)$:

$$y(a) := \max\{0, k(a) - p(v) + p(u)\}, \quad z(a) := \max\{0, -k(a) + p(v) - p(u)\}. \tag{12.54}$$

So y and z satisfy (12.52). Moreover, we have by (12.7) that $y(a)(f(a) - d(a)) = 0$ and $z(a)(c(a) - f(a)) = 0$ for each arc a. Hence we have equality throughout in (12.53). ∎

We consider a special case (Gallai [1957,1958a,1958b]). Let $D = (V, A)$ be a strongly connected digraph, let $d : A \to \mathbb{Z}_+$, and let $k : A \to \mathbb{Z}_+$ be a cost function, with $k(C) \ge 0$ for each directed circuit C. Then:

(12.55) the minimum cost $\sum_{a\in A} k(a)f(a)$ of an integer circulation f in D with $f \ge d$ is equal to the maximum value of $\sum_{a\in A} d(a)y(a)$ where $y : A \to \mathbb{Z}_+$ with $y(C) = k(C)$ for each directed circuit C in D.

A consequence of this applies to the 'directed Chinese postman problem': given a strongly connected directed graph, find a shortest directed closed path traversing each arc at least once. If we take unit length, we obtain the following. The minimum number of arcs in any closed directed path traversing each arc at least once is equal to the maximum value of

$$|A| + \sum_{U\in\mathcal{U}} (|\delta^{\text{in}}(U)| - |\delta^{\text{out}}(U)|), \tag{12.56}$$

where $\mathcal{U}$ is a collection of subsets U of V such that the $\delta^{\text{out}}(U)$ are disjoint.

This equality follows from (12.55), by taking $d = \mathbf{1}$ and $k = \mathbf{1}$: then there is a $p : V \to \mathbb{Z}$ with $y(a) = 1 - p(v) + p(u)$ for each arc $a = (u, v)$. So $p(v) \le p(u) + 1$ for each arc $a = (u, v)$. Taking $U_i := \{v \in V \mid p(v) \le i\}$ for each $i \in \mathbb{Z}$ gives the required cuts $\delta^{\text{out}}(U_i)$.

12.5c. Dynamic flows

A minimum-cost flow algorithm (in disguised form) was given by Ford and Fulkerson [1958b]. They considered the following 'dynamic flow' problem. Let $D = (V, A)$ be a digraph and let $r, s \in V$ (for convenience we assume that r is a source and s is a

sink of D). Let $c : A \to \mathbb{Z}_+$ be a capacity function. Moreover, let a 'traversal time' function $\tau : A \to \mathbb{Z}_+$ be given, and a 'time limit' T.

The problem now is to send a maximum amount of flow from r to s, such that, for each arc a, at each time unit at most $c(a)$ flow is sent over a; it takes $\tau(a)$ time to traverse a. All flow is sent from r at one of the times $1, 2, \ldots, T$, while it reaches s at time at most T.

More formally, for any arc $a = (u, v)$ and any $t \in \{1, 2, \ldots, T\}$, let $x(a, t)$ denote the amount of flow sent from u over a at time t, reaching v at time $t + \tau(a)$. A first constraint is:

$$0 \le x(a,t) \le c(a) \tag{12.57}$$

for each $a \in A$ and each $t \in \{1, \ldots, T\}$. Next a flow conservation law can be formulated. Flow may 'wait' at any vertex until there is capacity enough to be transmitted further. This can be described as follows:

$$\sum_{a \in \delta^{\mathrm{in}}(v)} \sum_{t=1}^{t'-\tau(a)} x(a,t) \ge \sum_{a \in \delta^{\mathrm{out}}(v)} \sum_{t=1}^{t'} x(a,t) \tag{12.58}$$

for each $v \in V \setminus \{r, s\}$ and each $t' \in \{1, \ldots, T\}$. We maximize the amount of flow reaching s not later than time T; that is, we

$$\text{maximize} \sum_{a \in \delta^{\mathrm{in}}(s)} \sum_{t=1}^{T-\tau(a)} x(a,t). \tag{12.59}$$

Since we may assume that we do not send flow from r that will not reach s, we may assume that we have equality in (12.58) if $t' = T$.

As Ford and Fulkerson [1958b] observed, this 'dynamic flow' problem can be transformed to a 'static' flow problem as follows. Let D' be the digraph with vertices all pairs (v, t) with $v \in V$ and $t \in \{1, \ldots, T\}$, and arcs

$$\begin{aligned} &\text{(i)} \ ((u,t),(v,t+\tau(a))) \text{ for each } a = (u,v) \in A \text{ and } t \in \{1, \ldots, T-\tau(a)\}, \\ &\text{(ii)} \ ((v,t),(v,t+1)) \text{ for each } v \in V \text{ and } t \in \{1, \ldots, T-1\}. \end{aligned} \tag{12.60}$$

Let any arc of type (i) have capacity $c(a)$ and let any arc of type (ii) have capacity $+\infty$. Then the maximum dynamic flow problem is equivalent to finding a maximum flow in the new network from $(r, 1)$ to (s, T).

By this construction, a maximum dynamic flow can be found by solving a maximum flow problem in the large graph D'. Ford and Fulkerson [1958b] however described an alternative way of finding a dynamic flow that has a number of advantages. First of all, no 'large' graph D' has to be constructed (and the final algorithm can be modified with the scaling method of Edmonds and Karp [1972] to a method that is polynomial also in $\log T$). Second, the solution can be represented as a relatively small number of paths over which flow is transmitted repeatedly. Finally, the method shows that at intermediate vertices hold-over of flows is not necessary (that is, all arcs of type (12.60)(ii) with $v \ne r, s$ can be deleted).

Ford and Fulkerson [1958b] showed that a solution of the dynamic flow problem can be found by solving the following problem:

$$\begin{aligned} &\text{maximize} \sum_{a \in \delta^{\mathrm{in}}(s)} Tx(a) - \sum_{a \in A} \tau(a)x(a), \\ &\text{where } x \text{ is an } r-s \text{ flow satisfying } \mathbf{0} \le x \le c. \end{aligned} \tag{12.61}$$

This is equivalent to a minimum-cost flow problem, with cost $k(a) := \tau(a) - T$ for $a \in \delta^{\text{in}}(s)$, and $k(a) := \tau(a)$ for all other a. Note that there are arcs of negative cost (generally), and that the value of the flow is not prescribed. So by adding an arc (s, r) we obtain a minimum-cost circulation problem.

How is problem (12.61) related to the dynamic flow problem? Given an optimum solution $x : A \to \mathbb{Z}_+$ of (12.61), there exist $r - s$ paths $P_1, \ldots, P_m$ in D such that

$$x \geq \sum_{i=1}^{m} \chi^{P_i} \tag{12.62}$$

where m is the value of x. (We identify a path P and its set of arcs.) For any path P, let $\tau(P)$ be the traversal time of P (= the sum of the traversal times of the arcs in P). Then $\tau(P_i) \leq T$ for each i, since otherwise we could replace x by $x - \chi^{P_i}$, while increasing the objective value in (12.61).

Now send, for each $i = 1, \ldots, m$, a flow of value 1 along P_i at times $1, \ldots, T - \tau(P_i)$. It is not difficult to describe this in terms of the $x(a, t)$, yielding a feasible solution for the dynamic flow problem, of value

$$\sum_{i=1}^{m} (T - \tau(P_i)) \geq mT - \sum_{a \in A} \tau(a) x(a), \tag{12.63}$$

which is the optimum value of (12.61).

In fact, this dynamic flow is optimum. Indeed, by Theorem 12.8 (alternatively, by LP-duality and total unimodularity), the optimum value of (12.61) is equal to that of:

$$\text{minimize} \sum_{a \in A} c(a) y(a) \tag{12.64}$$

where $y : A \to \mathbb{Z}_+$ such that there exists $p : V \to \mathbb{Z}$ satisfying:

$$p(u) - p(v) + y(a) \geq -\tau(a) \text{ for each } a = (u, v) \in A, \tag{12.65}$$

where $p(r) = 0$ and $p(s) = T$.

Now if $x(a, t)$ is a feasible solution of the dynamic flow problem, then by (12.58), (12.57) and (12.63),

$$\begin{aligned}
\sum_{a \in \delta^{\text{in}}(s)} \sum_{t=1}^{T-\tau(a)} x(a,t) &\leq \sum_{a \in \delta^{\text{in}}(s)} \sum_{t=1}^{T-\tau(a)} x(a,t) \\
&+ \sum_{v \neq s} \Big(\sum_{a \in \delta^{\text{in}}(v)} \sum_{t=1}^{p(v)-\tau(a)} x(a,t) - \sum_{a \in \delta^{\text{out}}(v)} \sum_{t=1}^{p(v)} x(a,t) \Big) \\
&= \sum_{a=(u,v) \in A} \Big(\sum_{t=1}^{p(v)-\tau(a)} x(a,t) - \sum_{t=1}^{p(u)} x(a,t) \Big) \\
&\leq \sum_{a=(u,v) \in A} \sum_{t=p(u)+1}^{p(v)-\tau(a)} x(a,t) \leq \sum_{a=(u,v) \in A} c(a)(p(v) - \tau(a) - p(u)) \\
&\leq \sum_{a \in A} c(a) y(a).
\end{aligned} \tag{12.66}$$

Therefore, the dynamic flow constructed is optimum.

Ford and Fulkerson [1958b] described a method for solving (12.61) which essentially consists of repeatedly finding a shortest $r - s$ path in the residual graph, making costs nonnegative by translating the cost with the help of the current potential p (this is 'Routine I' of Ford and Fulkerson [1958b]). In this formulation, it is a primal-dual method.

The method of Ford and Fulkerson [1958b] improves the algorithm of Ford [1956] for the dynamic flow problem. More on this and related problems can be found in Wilkinson [1971], Minieka [1973], Orlin [1983,1984b], Aronson [1989], Burkard, Dlaska, and Klinz [1993], Hoppe and Tardos [1994,1995,2000], Klinz and Woeginger [1995], Fleischer and Tardos [1998], and Fleischer [1998b,1999c,2001b,2001a].

12.5d. Further notes

The minimum-cost flow problem is a linear programming problem, and hence it can be solved with the primal simplex method or the dual simplex method. Strongly polynomial *dual* simplex algorithms for minimum-cost flow have been given by Orlin [1985] ($O(m^3)$ pivots) and Plotkin and Tardos [1990] ($O(m^2/\log n)$ pivots) (cf. Orlin, Plotkin, and Tardos [1993]). No pivot rule is known however that finds a minimum-cost flow with the *primal* simplex method in polynomial time. Partial results were found by Goldfarb and Hao [1990] and Tarjan [1991].

Further work on the primal simplex method applied to the minimum-cost flow problem is discussed by Dantzig [1963], Gassner [1964], Johnson [1966b], Grigoriadis and Walker [1968], Srinivasan and Thompson [1973], Glover, Karney, and Klingman [1974], Glover, Karney, Klingman, and Napier [1974], Glover, Klingman, and Stutz [1974], Ross, Klingman, and Napier [1975], Cunningham [1976,1979], Bradley, Brown, and Graves [1977], Gavish, Schweitzer, and Shlifer [1977], Barr, Glover, and Klingman [1978,1979], Mulvey [1978a], Kennington and Helgason [1980], Chvátal [1983], Cunningham and Klincewicz [1983], Gibby, Glover, Klingman, and Mead [1983], Grigoriadis [1986], Ahuja and Orlin [1988,1992], Goldfarb, Hao, and Kai [1990a], Tarjan [1991,1997], Eppstein [1994a,2000], Orlin [1997], and Sokkalingam, Sharma, and Ahuja [1997].

Further results on the dual simplex method applied to minimum-cost flows are given by Dantzig [1963], Helgason and Kennington [1977b], Armstrong, Klingman, and Whitman [1979], Orlin [1984a], Ikura and Nemhauser [1986], Adler and Cosares [1990], Plotkin and Tardos [1990], Orlin, Plotkin, and Tardos [1993], Eppstein [1994a,2000], and Armstrong and Jin [1997].

Further algorithmic work is presented by Briggs [1962], Pla [1971] (dual out-of-kilter), Barr, Glover, and Klingman [1974] (out-of-kilter), Hassin [1983,1992], Bertsekas [1985], Kapoor and Vaidya [1986] (interior-point method), Bertsekas and Tseng [1988] ('relaxation method'), Masuzawa, Mizuno, and Mori [1990] (interior-point method), Cohen and Megiddo [1991], Bertsekas [1992] ('auction algorithm'), Norton, Plotkin, and Tardos [1992], Wallacher and Zimmermann [1992] (combinatorial interior-point method), Ervolina and McCormick [1993a,1993b], Fujishige, Iwano, Nakano, and Tezuka [1993], McCormick and Ervolina [1994], Hadjiat and Maurras [1997], Goldfarb and Jin [1999a], McCormick and Shioura [2000a,2000b] (cycle canceling), Shigeno, Iwata, and McCormick [2000] (cycle- and cut-canceling), and Vygen [2000].

Worst-case studies are made by Zadeh [1973a,1973b,1979] (cf. Niedringhaus and Steiglitz [1978]), Adel'son-Vel'skiĭ, Dinits, and Karzanov [1975], Dinits and Karzanov [1974], Radzik and Goldberg [1991,1994] (minimum-mean cost cycle-cancelling), and Hadjiat [1998] (dual out-of-kilter).

Computational studies were presented by Glover, Karney, and Klingman [1974] (simplex method), Glover, Karney, Klingman, and Napier [1974], Harris [1976], Karney and Klingman [1976], Bradley, Brown, and Graves [1977], Helgason and Kennington [1977b] (dual simplex method), Ali, Helgason, Kennington, and Lall [1978], Mulvey [1978b] (simplex method), Armstrong, Klingman, and Whitman [1979], Monma and Segal [1982] (simplex method), Gibby, Glover, Klingman, and Mead [1983], Grigoriadis [1986] (simplex method), Ikura and Nemhauser [1986] (dual simplex method), Bertsekas and Tseng [1988], Bland and Jensen [1992], Bland, Cheriyan, Jensen, and Ladányi [1993], Fujishige, Iwano, Nakano, and Tezuka [1993], Goldberg [1993a,1997] (push-relabel and successive approximation), Goldberg and Kharitonov [1993] (push-relabel), and Resende and Veiga [1993] (interior-point). Consult also Johnson and McGeoch [1993].

Bein, Brucker, and Tamir [1985] and Hoffman [1988] considered minimum-cost circulation for series-parallel digraphs. Wagner and Wan [1993] gave a polynomial-time simplex method for the maximum k-flow problem (with a profit for every unit of flow sent, and a cost for every unit capacity added to any arc a), which can be reduced to a minimum-cost circulation problem.

Maximum flows where the cost may not exceed a given 'budget' were considered by Fulkerson [1959b] and Ahuja and Orlin [1995].

'Unsplittable' flows (with one source and several sinks, where all flow from the source to any sink follows the same path) were investigated by Kleinberg [1996, 1998], Kolliopoulos and Stein [1997,1998a,1999,2002], Srinivasan [1997], Dinitz, Garg, and Goemans [1998,1999], Skutella [2000,2002], Azar and Regev [2001], Erlebach and Hall [2002], and Kolman and Scheideler [2002].

For generalized flows (with 'gains' on arcs), see Jewell [1962], Fujisawa [1963], Eisemann [1964], Mayeda and van Valkenburg [1965], Charnes and Raike [1966], Onaga [1966,1967], Glover, Klingman, and Napier [1972], Maurras [1972], Glover and Klingman [1973], Grinold [1973], Truemper [1977], Minieka [1978], Elam, Glover, and Klingman [1979], Jensen and Barnes [1980], Gondran and Minoux [1984], Kapoor and Vaidya [1986], Bertsekas and Tseng [1988], Ruhe [1988], Vaidya [1989c], Goldberg, Tardos, and Tarjan [1990], Goldberg, Plotkin, and Tardos [1991], Cohen and Megiddo [1994], Goldfarb and Jin [1996], Tseng and Bertsekas [1996,2000], Goldfarb, Jin, and Orlin [1997], Radzik [1998], Tardos and Wayne [1998], Oldham [1999,2001], Wayne [1999,2002], Wayne and Fleischer [1999], Fleischer and Wayne [2002], and Goldfarb, Jin, and Lin [2002].

For convex costs,see Charnes and Cooper [1958], Beale [1959], Shetty [1959], Berge [1960b], Minty [1960,1961,1962], Tụy [1963,1964], Menon [1965], Hu [1966], Weintraub [1974], Jensen and Barnes [1980], Kennington and Helgason [1980], Dembo and Klincewicz [1981], Hassin [1981a], Klincewicz [1983], Ahuja, Batra, and Gupta [1984], Minoux [1984,1986], Rockafellar [1984], Florian [1986], Bertsekas, Hosein, and Tseng [1987], Katsura, Fukushima, and Ibaraki [1989], Karzanov and McCormick [1995,1997], Tseng and Bertsekas [1996,2000], and Ahuja, Hochbaum, and Orlin [1999].

Concave costs were studied by Zangwill [1968], Rothfarb and Frisch [1970], Daeninck and Smeers [1977], Jensen and Barnes [1980], Graves and Orlin [1985], and Erickson, Monma, and Veinott [1987].

The basic references on network flows are the books of Ford and Fulkerson [1962] (historical) and Ahuja, Magnanti, and Orlin [1993]. Minimum-cost flow problems are discussed also in the books of Hu [1969,1982], Iri [1969], Frank and Frisch [1971], Potts and Oliver [1972], Adel'son-Vel'skiĭ, Dinits, and Karzanov [1975] (for a review, see Goldberg and Gusfield [1991]), Christofides [1975], Lawler [1976b], Murty [1976], Bazaraa and Jarvis [1977], Minieka [1978], Jensen and Barnes [1980], Kennington and Helgason [1980], Phillips and Garcia-Diaz [1981], Swamy and Thulasiraman [1981], Papadimitriou and Steiglitz [1982], Smith [1982], Chvátal [1983], Sysło, Deo, and Kowalik [1983], Tarjan [1983], Gondran and Minoux [1984], Rockafellar [1984], Derigs [1988a], Nemhauser and Wolsey [1988], Bazaraa, Jarvis, and Sherali [1990], Chen [1990], Cook, Cunningham, Pulleyblank, and Schrijver [1998], Jungnickel [1999], and Korte and Vygen [2000].

Survey papers include Glover and Klingman [1977], Ahuja, Magnanti, and Orlin [1989,1991], Goldberg, Tardos, and Tarjan [1990], and Frank [1995]. A bibliography was given by Golden and Magnanti [1977].

The history of the minimum-cost flow, circulation, and transshipment problems is closely intertwined with that of the transportation problem — see Section 21.13e.

Chapter 13

Path and flow polyhedra and total unimodularity

A large part of the theory of paths and flows can be represented geometrically by polytopes and polyhedra, and can be studied with methods from geometry and linear programming. Theorems like Menger's theorem, the max-flow min-cut theorem, and Hoffman's circulation theorem can be derived and interpreted with elementary polyhedral tools.
This can be done with the help of the total unimodularity of the incidence matrices of directed graphs, and of the more general network matrices. It again yields proofs of basic flow theorems and implies the polynomial-time solvability of flow problems.

13.1. Path polyhedra

Let $D = (V, A)$ be a digraph and let $s, t \in V$. The $s - t$ *path polytope* $P_{s-t \text{ path}}(D)$ is the convex hull of the incidence vectors in $\mathbb{R}^A$ of $s-t$ paths in D. (We recall that paths are simple, by definition.) So $P_{s-t \text{ path}}(D)$ is a polytope in the space $\mathbb{R}^A$. Since finding a maximum-length $s-t$ path in D is NP-complete, we may not expect to have a decent description of the inequalities determining $P_{s-t \text{ path}}(D)$ (cf. Corollary 5.16a). That is, the separation problem for $P_{s-t \text{ path}}(D)$ is NP-hard.

However, if we extend the $s-t$ path polytope to its dominant, it becomes more tractable. This leads to an illuminating geometric framework, in which the (easy) max-potential min-work theorem (Theorem 7.1) and the (more difficult) max-flow min-cut theorem (Theorem 10.3) show up as polars of each other, and can be derived from each other. This duality forms a prototype for many other dual theorems and problems.

The dominant $P^{\uparrow}_{s-t \text{ path}}(D)$ of $P_{s-t \text{ path}}(D)$ is the set of vectors $x \in \mathbb{R}^A$ with $x \geq y$ for some $y \in P_{s-t \text{ path}}(D)$. So

$$P^{\uparrow}_{s-t \text{ path}}(D) = P_{s-t \text{ path}}(D) + \mathbb{R}^A_+. \tag{13.1}$$

An alternative way of describing this polyhedron is as the set of all capacity functions $c : A \to \mathbb{R}_+$ for which there exists a flow $x \leq c$ of value 1.

It is not difficult to derive from the (easy) max-potential min-work theorem that the following inequalities determine $P^{\uparrow}_{s-t \text{ path}}(D)$:

(13.2) (i) $x(a) \geq 0$ for each $a \in A$,
(ii) $x(C) \geq 1$ for each $s-t$ cut C.

Theorem 13.1. *$P^{\uparrow}_{s-t\text{ path}}(D)$ is determined by* (13.2).

Proof. Clearly, each vector in $P^{\uparrow}_{s-t\text{ path}}(D)$ satisfies (13.2). So $P^{\uparrow}_{s-t\text{ path}}(D)$ is contained in the polyhedron Q determined by (13.2). Suppose that the reverse inclusion does not hold. Then there is an $l \in \mathbb{Z}^A$ such that the minimum of $l^{\mathsf{T}}x$ over $x \in Q$ is smaller than over $x \in P^{\uparrow}_{s-t\text{ path}}(D)$. If $l \notin \mathbb{Z}^A_+$, the minimum in both cases is $-\infty$; so $l \in \mathbb{Z}^A_+$. Then the minimum over $P^{\uparrow}_{s-t\text{ path}}(D)$ is equal to the minimum length k of an $s-t$ path, taking l as length function. By Theorem 7.1, there exist $s-t$ cuts $C_1, \ldots, C_k$ such that each arc a is in at most $l(a)$ of the C_i. Hence for any $x \in Q$ one has

$$l^{\mathsf{T}}x \geq \Big(\sum_{i=1}^{k} \chi^{C_i}\Big)^{\mathsf{T}} x = \sum_{i=1}^{k} x(C_i) \geq k, \tag{13.3}$$

by (13.2)(ii). So the minimum over Q is at least k, contradicting our assumption. ∎

So the characterization of the dominant $P^{\uparrow}_{s-t\text{ path}}(D)$ of the $s-t$ path polytope follows directly from the easy Theorem 7.1 (the max-potential min-work theorem).

Note that Theorem 13.1 is equivalent to:

Corollary 13.1a. *The polyhedron determined by* (13.2) *is integer.*

Proof. The vertices are integer, as they are incidence vectors of paths. ∎

Next, the theory of blocking polyhedra implies a similar result when we interchange 'paths' and 'cuts', thus deriving the max-flow min-cut theorem.

The *$s-t$ cut polytope* $P_{s-t\text{ cut}}(D)$ is the convex hull of the incidence vectors of $s-t$ cuts in D. Again, $P_{s-t\text{ cut}}(D)$ is a polytope in the space $\mathbb{R}^A$. Since finding a maximum-size $s-t$ cut in D is NP-complete (Theorem 75.1), we may not expect to have a decent description of inequalities determining $P_{s-t\text{ cut}}(D)$. That is, the separation problem for $P_{s-t\text{ cut}}(D)$ is NP-hard.

Again, a polyhedron that behaves more satisfactorily is the dominant $P^{\uparrow}_{s-t\text{ cut}}(D)$ of the $s-t$ cut polytope, which is the set of vectors $x \in \mathbb{R}^A$ with $x \geq y$ for some $y \in P_{s-t\text{ cut}}(D)$. That is,

$$P^{\uparrow}_{s-t\text{ cut}}(D) = P_{s-t\text{ cut}}(D) + \mathbb{R}^A_+. \tag{13.4}$$

Now the following inequalities determine $P^{\uparrow}_{s-t\text{ cut}}(D)$:

(13.5) (i) $x(a) \geq 0$ for $a \in A$,
(ii) $x(AQ) \geq 1$ for each $s-t$ path Q.

Corollary 13.1b. $P^{\uparrow}_{s-t\ \mathrm{cut}}(D)$ *is determined by* (13.5).

Proof. Directly with the theory of blocking polyhedra (Theorem 5.8) from Theorem 13.1. ∎

Equivalently:

Corollary 13.1c. *The polyhedron determined by* (13.5) *is integer.*

Proof. The vertices are integer, as they are incidence vectors of $s-t$ cuts. ∎

The two polyhedra are connected by the blocking relation:

Corollary 13.1d. *The polyhedra* $P^{\uparrow}_{s-t\ \mathrm{path}}(D)$ *and* $P^{\uparrow}_{s-t\ \mathrm{cut}}(D)$ *form a blocking pair of polyhedra.*

Proof. Directly from the above. ∎

With linear programming duality, this theorem implies the max-flow min-cut theorem:

Corollary 13.1e (max-flow min-cut theorem)**.** *Let* $D=(V,A)$*, let* $s,t\in V$ *and let* $c:A\to\mathbb{R}_+$ *be a capacity function. Then the maximum value of an* $s-t$ *flow* $f\le c$ *is equal to the minimum capacity of an* $s-t$ *cut.*

Proof. The minimum capacity of an $s-t$ cut is equal to the minimum of $c^{\mathsf{T}}x$ over $x\in P^{\uparrow}_{s-t\ \mathrm{cut}}(D)$ (by definition (13.4)). By Corollary 13.1b, this is equal to the minimum value μ of $c^{\mathsf{T}}x$ where x satisfies (13.5). By linear programming duality, μ is equal to the maximum value of $\sum_Q \lambda_Q$, where $\lambda_Q\ge 0$ for each $s-t$ path Q, such that

$$\sum_Q \lambda_Q \chi^{AQ} \le c. \tag{13.6}$$

Then $f:=\sum_Q \lambda_Q\chi^{AQ}$ is an $s-t$ flow of value μ. ∎

Thus the theory of blocking polyhedra links minimum-length paths and minimum-capacity cuts.

Algorithmic duality. The duality of paths and cuts can be extended to the polynomial-time solvability of the corresponding optimization problems. Indeed, Corollary 5.14a implies that the following can be derived from the fact that $P^{\uparrow}_{s-t\ \mathrm{path}}(D)$ and $P^{\uparrow}_{s-t\ \mathrm{cut}}(D)$ form a blocking pair of polyhedra (Corollary 13.1d):

(13.7) the minimum-length path problem is polynomial-time solvable
$\iff$
the minimum-capacity cut problem is polynomial-time solvable.

(Here the length and capacity functions are restricted to be nonnegative.)

By Theorem 5.15, one can also find a dual solution, which implies:

(13.8) the minimum-capacity cut problem is polynomial-time solvable
$\iff$
the maximum flow problem is polynomial-time solvable.

The statements in (13.7) by themselves are not surprising, since the polynomial-time solvability of neither of the problems has turned out to be hard, although finding a shortest path in polynomial time is easier than finding a maximum flow in polynomial time.

However, it is good to realize that the equivalence has been derived purely from the theoretical fact that the two polyhedra form a blocking pair. In further chapters we will see more sophisticated applications of this principle.

Dual integrality. The fact that $P^{\uparrow}_{s-t\ \mathrm{path}}(D)$ and $P^{\uparrow}_{s-t\ \mathrm{cut}}(D)$ form a blocking pair of polyhedra, is equivalent to the fact that the polyhedra determined by (13.2) and (13.5) each are integer (that is, have integer vertices only). More precisely, blocking polyhedra theory tells us:

(13.9) the polyhedron determined by (13.2) is integer $\iff$ the polyhedron determined by (13.5) is integer.

In other words, minimizing any linear function over (13.2) gives an integer optimum solution if and only if minimizing any linear function over (13.5) gives an integer optimum solution. Thus there is an equivalence of the existence of integer optimum solutions between two classes of linear programming problems. What can be said about the *dual* linear programs?

There is no general theorem known that links the existence of integer optimum dual solutions of blocking pairs of polyhedra. In fact, it is not the case that if two systems $Ax \leq b$ and $A'x \leq b'$ of linear inequalities represent a blocking pair of polyhedra, then the existence of integer optimum dual solutions for one system implies the existence of integer optimum dual solutions for the other. That is, the total dual integrality of $Ax \leq b$ is not equivalent to the total dual integrality of $A'x \leq b'$ (even not if one puts strong conditions on the two systems, like A, b, A', and b' being $0, 1$).

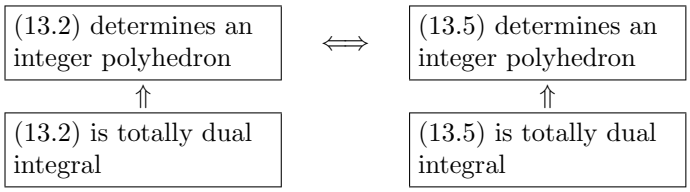
Yet, in the special case of paths and cuts the systems *are* totally dual integral, as follows directly from theorems proved in previous chapters. (In particular, total dual integrality of (13.5) amounts to the integrity theorem for flows.)

Theorem 13.2. *The systems* (13.2) *and* (13.5) *are totally dual integral.*

Proof. Total dual integrality of (13.2) is equivalent to Theorem 7.1, and total dual integrality of (13.5) is equivalent to Corollary 10.3a. ▮

By Theorem 5.22, total dual integrality of a system $Ax \leq b$ (with b integer) implies total *primal* integrality; that is, integrality of the polyhedron determined by $Ax \leq b$. So general polyhedral theory gives the following implications:

(13.10)



13.1a. Vertices, adjacency, and facets

Vertices of the dominant of the $s-t$ path polytope have a simple characterization:

Theorem 13.3. *A vector x is a vertex of $P^{\uparrow}_{s-t \text{ path}}$ if and only if $x = \chi^{\pi}$ for some $s-t$ path π.*

Proof. If $x = \chi^{\pi}$ for some $s-t$ path π, then x is a vertex of $P^{\uparrow}_{s-t \text{ path}}$, as for the length function l defined by $l(a) := 0$ if $a \in A\pi$ and $l(a) := 1$ otherwise, the path π is the unique shortest $s-t$ path.

Conversely, let x be a vertex. As x is integer, $x \geq \chi^{\pi}$ for some $s-t$ path π. Then χ^{π} and $2x - \chi^{\pi} = x + (x - \chi^{\pi})$ belong to $P^{\uparrow}_{s-t \text{ path}}$. As $x = (\chi^{\pi} + (2x - \chi^{\pi}))/2$, we have $x = \chi^{\pi}$. ▮

As for adjacency, one has:

Theorem 13.4. *Let π and π' be two distinct $s-t$ paths in D. Then χ^{π} and $\chi^{\pi'}$ are adjacent vertices of $P^{\uparrow}_{s-t \text{ path}}$ if and only if $A\pi \triangle A\pi'$ is an undirected circuit consisting of two internally vertex-disjoint directed paths.*

Proof. If $A\pi \triangle A\pi'$ is an undirected circuit consisting of two internally vertex-disjoint paths, define the length function l by $l(a) := 0$ if $a \in A\pi \cup A\pi'$ and $l(a) := 1$ otherwise. Then π and π' are the only two shortest $s-t$ paths.

Conversely, let χ^{π} and $\chi^{\pi'}$ be adjacent. Suppose that $A\pi \cup A\pi'$ contains an $s-t$ path π'' different from π and π'. Then $\chi^{\pi} + \chi^{\pi'} - \chi^{\pi''} = \chi^{\pi'''}$ for some $s-t$ path π''', contradicting the adjacency of χ^{π} and $\chi^{\pi'}$. This implies that $A\pi \triangle A\pi'$ is an undirected circuit consisting of two internally vertex-disjoint directed paths. ▮

Finally, for the facets we have:

Theorem 13.5. *Let C be an $s-t$ cut. Then the inequality $x(C) \geq 1$ determines a facet of $P^{\uparrow}_{s-t \text{ path}}$ if and only if C is an inclusionwise minimal $s-t$ cut.*

Proof. *Necessity.* Suppose that there is an $s-t$ cut $C' \subset C$. Then the inequalities $x \geq \mathbf{0}$ and $x(C') \geq 1$ imply $x(C) \geq 1$, and hence $x(C) \geq 1$ is not facet-inducing.

Sufficiency. If the inequality $x(C) \geq 1$ is not facet-inducing, it is a nonnegative linear combination of other inequalities in system (13.2). At least one of them is of the form $x(C') \geq 1$ for some $s-t$ cut C'. Then necessarily $C' \subset C$. ∎

Garg and Vazirani [1993,1995] characterized the vertices of and adjacency on a variant of the $s-t$ cut polytope.

13.1b. The $s-t$ connector polytope

There are a number of related polyhedra for which similar results hold. Call a subset A' of A an *$s-t$ connector* if A' contains the arc set of an $s-t$ path as a subset. The *$s-t$ connector polytope* $P_{s-t\text{ connector}}(D)$ is the convex hull of the incidence vectors of the $s-t$ connectors.

This polytope turns out to be determined by the following system of linear inequalities:

$$\begin{array}{lll} \text{(i)} & 0 \leq x(a) \leq 1 & \text{for each } a \in A, \\ \text{(ii)} & x(C) \geq 1 & \text{for each } s-t \text{ cut } C. \end{array} \tag{13.11}$$

Again, the fact that the $s-t$ connector polytope is contained in the polytope determined by (13.11) follows from the fact that χ^P satisfies (13.11) for each $s-t$ connector P.

Also in this case one has:

Theorem 13.6. *System* (13.11) *is totally dual integral.*

Proof. Directly from Theorem 13.2, using Theorem 5.23. ∎

It implies primal integrality:

Corollary 13.6a. *The $s-t$ connector polytope is equal to the solution set of* (13.11).

Proof. Directly from Theorem 13.6. ∎

The dimension of $P_{s-t\text{ connector}}(D)$ is easily determined:

Theorem 13.7. *Let A' be the set of arcs a for which there exists an $s-t$ path not traversing a. Then* $\dim P_{s-t\text{ connector}}(D) = |A'|$.

Proof. We use Theorem 5.6. Clearly, no inequality $x_a \geq 0$ is an implicit equality. Moreover, the inequality $x_a \leq 1$ is an implicit equality if and only if $a \in A \setminus A'$. For distinct arcs $a \in A \setminus A'$, these equalities are independent.

Suppose that there is a $U \subseteq V$ with $s \in U$, $t \notin U$, such that $x(\delta^{\text{out}}(U)) \geq 1$ is an implicit equality. Then $|\delta^{\text{out}}(U)| = 1$, since the all-one vector belongs to the polytope. So the arc in $\delta^{\text{out}}(U)$ belongs to $A \setminus A'$.

We conclude that the maximum number of independent implicit equalities is equal to $|A \setminus A'|$. Hence $\dim P = |A'|$. ∎

Let $D = (V, A)$ be a digraph, let $s, t \in V$, and let $k \in \mathbb{Z}_+$. By the results above, the convex hull P of the incidence vectors χ^B of those subsets B of A that contain k arc-disjoint $s-t$ paths, is determined by

$$\begin{array}{lll} (13.12) & \text{(i)} & 0 \leq x(a) \leq 1 \quad \text{for each } a \in A, \\ & \text{(ii)} & x(C) \geq k \quad \text{for each } s-t \text{ cut } C. \end{array}$$

L.E. Trotter, Jr observed that this polytope P has the *integer decomposition property*; that is, for each $l \in \mathbb{Z}_+$, any integer vector $x \in l \cdot P$ is the sum of l integer vectors in P.

Theorem 13.8. *The polytope P determined by* (13.12) *has the integer decomposition property.*

Proof. Let $l \in \mathbb{Z}_+$ and let $x \in l \cdot P$. Then there exists an integer $s-t$ flow $f \leq x$ of value $l \cdot k$ (by the max-flow min-cut theorem). We can assume that $x = f$. As $\frac{1}{l} f$ is an $s-t$ flow of value k, by Corollary 11.2c there exists an integer $s-t$ flow f' of value k with

$$(13.13) \qquad \lfloor \frac{1}{l} f(a) \rfloor \leq f'(a) \leq \lceil \frac{1}{l} f(a) \rceil$$

for each arc a. Then f' is an integer vector in P, since $f'(a) \leq 1$ for each arc a, as $f(a) \leq l$ for each arc a. Moreover, $f - f'$ is an integer vector belonging to $(l-1) \cdot P$, as $f - f'$ is an $s-t$ flow of value $(l-1) \cdot k$ and as $(f - f')(a) \leq l - 1$ for each arc a, since if $f(a) = l$, then $f'(a) = 1$ by (13.13). ∎

13.2. Total unimodularity

Let $D = (V, A)$ be a digraph. Recall that the $V \times A$ *incidence matrix* M of D is defined by $M_{v,a} := -1$ if a leaves v, $M_{v,a} := +1$ if a enters v, and $M_{v,a} := 0$ otherwise. So each column of M contains exactly one $+1$ and exactly one -1, while all other entries are 0. The following basic statement follows from a theorem of Poincaré [1900][21] (we follow the proofs of Chuard [1922] and Veblen and Franklin [1921]):

Theorem 13.9. *The incidence matrix M of any digraph D is totally unimodular.*

Proof. Let B be a square submatrix of M, of order k say. We prove that $\det B \in \{0, \pm 1\}$ by induction on k, the case $k = 1$ being trivial. Let $k > 1$. We distinguish three cases.

[21] Poincaré [1900] showed the total unimodularity of any $\{0, \pm 1\}$ matrix $M = (M_{i,j})$ with the property that for each k and all distinct row indices $i_1, \ldots, i_k$ and all distinct column indices $j_1, \ldots, j_k$, the product

$$M_{i_1,j_1} M_{i_1,j_2} M_{i_2,j_2} M_{i_2,j_3} \cdots M_{i_{k-1},j_{k-1}} M_{i_{k-1},j_k} M_{i_k,j_k} M_{i_k,j_1}$$

belongs to $\{0, 1\}$ if k is even and to $\{0, -1\}$ if k is odd. Incidence matrices of digraphs have this property.

Case 1: B has a column with only zeros. Then $\det B = 0$.

Case 2: B has a column with exactly one nonzero. Then we can write (up to permuting rows and columns):

$$B = \begin{pmatrix} \pm 1 & b^{\mathsf{T}} \\ \mathbf{0} & B' \end{pmatrix}, \tag{13.14}$$

for some vector b and matrix B'. Then by the induction hypothesis, $\det B' \in \{0, \pm 1\}$, and hence $\det B \in \{0, \pm 1\}$.

Case 3 : Each column of B contains two nonzeros. Then each column of B contains one $+1$ and one -1, while all other entries are 0. So each row of B adds up to 0, and hence $\det B = 0$. ▮

One can derive several results on circulations, flows, and transshipments from the total unimodularity of the incidence matrix of a digraph, like the max-flow min-cut theorem (see Section 13.2a below) and theorems characterizing the existence of a circulation or a b-transshipment (Theorem 11.2 and Corollary 11.2f). Moreover, min-max equalities for minimum-cost flow, circulation (cf. Theorem 12.8), and b-transshipment follow. We discuss some of the previous and some new results in the following sections.

13.2a. Consequences for flows

We show that the max-flow min-cut theorem can be derived from the total unimodularity of the incidence matrix of a digraph:

Corollary 13.9a (max-flow min-cut theorem)**.** *Let $D = (V, A)$, let $s, t \in V$, and let $c : A \to \mathbb{R}_+$ be a capacity function. Then the maximum value of an $s - t$ flow $f \leq c$ is equal to the minimum capacity of an $s - t$ cut.*

Proof. Since the maximum clearly cannot exceed the minimum, it suffices to show that there exists an $s - t$ flow $x \leq c$ and an $s - t$ cut, whose capacity is not more than the value of x.

Let M be the incidence matrix of D and let M' arise from M by deleting the rows corresponding to s and t. So the condition $M'x = \mathbf{0}$ means that the flow conservation law should hold at any vertex $v \neq s, t$.

Let w be the row of M corresponding to vertex t. So for any arc a, $w_a = +1$ if a enters t, $w_a = -1$ if a leaves t, and $w_a = 0$ otherwise.

Now the maximum value of an $s - t$ flow subject to c is equal to

$$\max\{w^{\mathsf{T}} x \mid \mathbf{0} \leq x \leq c; M'x = \mathbf{0}\}. \tag{13.15}$$

By LP-duality, this is equal to

$$\min\{y^{\mathsf{T}} c \mid y \geq \mathbf{0}; \exists z : y^{\mathsf{T}} + z^{\mathsf{T}} M' \geq w^{\mathsf{T}}\}. \tag{13.16}$$

Since M' is totally unimodular by Theorem 13.9 and since w is an integer vector, minimum (13.16) is attained by *integer* vectors y and z. Extend z by defining $z_t := -1$ and $z_s := 0$. Then $y^{\mathsf{T}} + z^{\mathsf{T}} M \geq \mathbf{0}$.

Now define

(13.17) $U := \{v \in V \mid z_v \geq 0\}$.

Then U is a subset of V containing s and not containing t.

It suffices to show that

(13.18) $c(\delta^{\text{out}}(U)) \leq y^{\mathsf{T}} c$,

since $y^{\mathsf{T}} c$ is equal to the maximum flow value (13.15).

To prove (13.18), it suffices to show that

(13.19) if $a = (u, v) \in \delta^{\text{out}}(U)$, then $y_a \geq 1$.

To see this, note that $z_u \geq 0$ and $z_v \leq -1$. Moreover, $y^{\mathsf{T}} + z^{\mathsf{T}} M \geq \mathbf{0}$ implies $y_a + z_v - z_u \geq 0$. So $y_a \geq z_u - z_v \geq 1$. This proves (13.19). ∎

It follows similarly that if all capacities are integers, then there exists a maximum *integer* flow; that is, we have the integrity theorem (Corollary 10.3a).

Let $D = (V, A)$ be a digraph and $s, t \in V$. The set of all $s - t$ flows of value 1 is a polyhedron $P_{s-t \text{ flow}}(D)$, determined by:

(13.20)
$$\begin{array}{lll} \text{(i)} & x(a) \geq 0 & \text{for each } a \in A, \\ \text{(ii)} & x(\delta^{\text{in}}(v)) = x(\delta^{\text{out}}(v)) & \text{for each } v \in V \setminus \{s, t\}, \\ \text{(iii)} & x(\delta^{\text{out}}(s)) - x(\delta^{\text{in}}(s)) = 1. & \end{array}$$

The set of all $s - t$ flows of value ϕ trivially equals $\phi \cdot P_{s-t \text{ flow}}(D)$.

The total unimodularity of M gives that, for integer ϕ, the intersection of $\phi \cdot P_{s-t \text{ flow}}(D)$ with an integer box $\{x \mid \mathbf{0} \leq x \leq c\}$ is an integer polytope. In other words:

Theorem 13.10. *Let $D = (V, A)$ be a digraph, $s, t \in V$, $c : A \to \mathbb{Z}$, and $\phi \in \mathbb{Z}_+$. Then the set of $s - t$ flows $x \leq c$ of value ϕ forms an integer polytope.*

Proof. Directly from the total unimodularity of the incidence matrix of a digraph (using Theorem 5.20). ∎

In particular this gives:

Corollary 13.10a. *Let $D = (V, A)$ be a digraph, $s, t \in V$, $c : A \to \mathbb{Z}$ and $\phi \in \mathbb{Z}$. If there exists an $s - t$ flow $x \leq c$ of value ϕ, then there exists an integer such flow.*

Proof. Directly from Theorem 13.10. ∎

Notes. A relation of $P_{s-t \text{ flow}}(D)$ with the polytope $P_{s-t \text{ path}}(D)$ is that

(13.21) $P_{s-t \text{ path}}(D) \subseteq P_{s-t \text{ flow}}(D) \subseteq P^{\uparrow}_{s-t \text{ path}}(D)$.

Hence

(13.22) $P^{\uparrow}_{s-t \text{ path}}(D) = P^{\uparrow}_{s-t \text{ flow}}(D)$.

Dantzig [1963] (pp. 352–366) showed that each vertex x of $P_{s-t \text{ flow}}(D)$ is the incidence vector χ^P of some $s - t$ path P. It can be shown that, if each arc of D is in some $s - t$ path, then $P_{s-t \text{ flow}}(D)$ is the topological closure of the convex hull of the vectors $\chi^P \in \mathbb{R}^A$ where P is an $s - t$ walk and where

(13.23) $\chi^P(a) :=$ number of times P traverses a,

for $a \in A$.

For two distinct $s-t$ paths, the vertices χ^P and $\chi^{P'}$ are adjacent if and only if the symmetric difference $AP \triangle AP'$ forms an undirected circuit consisting of two internally vertex disjoint directed paths.

Saigal [1969] proved that any two vertices of $P_{s-t \text{ flow}}(D)$ are connected by a path on the 1-skeleton of $P_{s-t \text{ flow}}(D)$ with at most $|A|-1$ edges — this implies the Hirsch conjecture for this class of polyhedra. (The Hirsch conjecture (cf. Dantzig [1963,1964]) says that the 1-skeleton of a polytope in $\mathbb{R}^n$ determined by m inequalities has diameter at most $m-n$.) In fact, Saigal showed more strongly that for any two feasible bases B, B' of (13.20), there is a series of at most $|A|-1$ pivots bringing B to B'. It amounts to the following. Call a spanning tree T of D *feasible* if it contains a directed $s-t$ path. Call two feasible spanning trees T, T' *adjacent* if $|AT \setminus AT'| = 1$. Then for any two feasible spanning trees T, T' there exists a sequence $T_0, \ldots, T_k$ of feasible spanning trees such that $T_0 = T$, $T_k = T'$, $k \leq |A|-1$, and T_{i-1} and T_i adjacent for $i = 1, \ldots, k$.

Rispoli [1992] showed that if D is the complete directed graph, then for each length function l and each vertex x_0 of (13.20), there is a path $x_0, x_1, \ldots, x_d$ on the 1-skeleton of (13.20), where $l^{\mathsf{T}}x_i \leq l^{\mathsf{T}}x_{i-1}$ for $i = 1, \ldots, d$, where x_d minimizes $l^{\mathsf{T}}x$ over (13.20), and where $d \leq \frac{2}{3}(|V|-1)$.

13.2b. Consequences for circulations

Another consequence is:

Corollary 13.10b. *Let $D = (V, A)$ be a digraph and let $c, d : A \to \mathbb{Z}$. Then the set of circulations x satisfying $d \leq x \leq c$ forms an integer polytope.*

Proof. Directly from the total unimodularity of the incidence matrix of a digraph. ∎

In particular this implies:

Corollary 13.10c. *Let $D = (V, A)$ be a digraph and let $c, d : A \to \mathbb{Z}$. If there exists a circulation x satisfying $d \leq x \leq c$, then there exists an integer such circulation.*

Proof. Directly from Corollary 13.10b. ∎

Another consequence is the integer decomposition property as in Corollary 11.2b.

13.2c. Consequences for transshipments

Let $D = (V, A)$ be a digraph, $d, c \in \mathbb{R}^A_+$, and $b \in \mathbb{R}^V$. The *b-transshipment polytope* is the set of all b-transshipments x with $d \leq x \leq c$. So it is equal to

(13.24) $P := \{x \in \mathbb{R}^A \mid d \leq x \leq c, Mx = b\}$,

where M is the $V \times A$ incidence matrix of D.

Again, the total unimodularity of M (Theorem 13.9) implies:

Theorem 13.11. *Let $D = (V, A)$ be a digraph, $b : V \to \mathbb{Z}$, and $c, d : A \to \mathbb{Z}$. Then the b-transshipment polytope is an integer polytope.*

Proof. Directly from the total unimodularity of the incidence matrix of a digraph. ∎

In particular this gives:

Corollary 13.11a. *Let $D = (V, A)$ be a digraph, $b : V \to \mathbb{Z}$, and $c, d : A \to \mathbb{Z}$. If there exists a b-transshipment x with $d \leq x \leq c$, then there exists an integer such b-transshipment.*

Proof. Directly from Theorem 13.11. ∎

Also Corollary 11.2f can be derived:

Theorem 13.12. *There exists a b-transshipment x satisfying $d \leq x \leq c$ if and only if $b(V) = 0$, $d \leq c$ and $c(\delta^{\text{in}}(U)) - d(\delta^{\text{out}}(U)) \geq b(U)$ for each $U \subseteq V$.*

Proof. Necessity being easy, we show sufficiency. If no b-transshipment as required exists, then by Farkas' lemma, there exist vectors $y \in \mathbb{R}^V$ and $z', z'' \in \mathbb{R}^A_+$ such that $y^{\mathsf{T}}M + z'^{\mathsf{T}} - z''^{\mathsf{T}} = \mathbf{0}$ and $y^{\mathsf{T}}b + z'^{\mathsf{T}}c - z''^{\mathsf{T}}d < 0$. By adding a multiple of $\mathbf{1}$ to y we can assume that $y \geq \mathbf{0}$ (since $\mathbf{1}^{\mathsf{T}}M = \mathbf{0}$ and $\mathbf{1}^{\mathsf{T}}b = 0$). Next, by scaling we can assume that $\mathbf{0} \leq y \leq \mathbf{1}$. As M is totally unimodular, we can assume moreover that y is integer. So $y = \chi^U$ for some $U \subseteq V$. Since $d \leq c$, we can assume that $z'(a) = 0$ or $z''(a) = 0$ for each $a \in A$. Hence $z' = \chi^{\delta^{\text{out}}(U)}$ and $z'' = \chi^{\delta^{\text{in}}(U)}$. Then $y^{\mathsf{T}}b + z'^{\mathsf{T}}c - z''^{\mathsf{T}}d < 0$ contradicts the condition for $V \setminus U$. ∎

For any digraph $D = (V, A)$ and $b \in \mathbb{R}^V$, let P_b denote the set of b-transshipments. So

$$P_b = \{x \mid Mx = b\} \tag{13.25}$$

where M is the $V \times A$ incidence matrix of D. Koopmans and Reiter [1951] characterized the dimension of the transshipment space:

(13.26) if P_b is nonempty, then it has dimension $|A| - |V| + k$, where k is the number of weak components of D.

(A *weak component* of a digraph is a component of the underlying undirected graph.)

To see (13.26), let $F \subseteq A$ form a spanning forest in the underlying undirected graph. So (V, F) has k weak components and contains no undirected circuit. Then $|F| = |V| - k$. Now each $x \in \mathbb{R}^{A \setminus F}$ can be extended uniquely to a b-transshipment $x \in \mathbb{R}^A$. Hence P_b has dimension $|A \setminus F| = |A| - |V| + k$.

Consider next the polyhedron

$$Q_b := \{x \in \mathbb{R}^A \mid \text{there exists a nonnegative } b\text{-transshipment } f \leq x\} \tag{13.27}$$

So

(13.28) $$Q_b = (P_b \cap \mathbb{R}^A_+) + \mathbb{R}^A_+.$$

By Gale's theorem (Corollary 11.2g), Q_b is determined by:

(13.29)
(i) $x(a) \geq 0$ for each $a \in A$,
(ii) $x(\delta^{\text{in}}(U)) \geq b(U)$ for each $U \subseteq V$.

Fulkerson and Weinberger [1975] showed that this system is TDI:

Theorem 13.13. *System* (13.29) *is TDI.*

Proof. Choose $w \in \mathbb{Z}^A_+$. We must show that the dual of minimizing $w^{\mathsf{T}}x$ over (13.29) has an integer optimum solution.

Let μ be the minimum value of $w^{\mathsf{T}}x$ over (13.29). As (13.29) determines Q_b, μ is equal to the minimum value of $w^{\mathsf{T}}x$ over $x \geq \mathbf{0}$, $Mx = b$, where M is the $V \times A$ incidence matrix of D. Since M is totally unimodular, this LP-problem has an integer optimum dual solution. That is, there exists a $y \in \mathbb{Z}^V$ such that $y^{\mathsf{T}}M \leq w^{\mathsf{T}}$ and $y^{\mathsf{T}}b = \mu$. We can assume that $y \geq \mathbf{0}$, since $\mathbf{1}^{\mathsf{T}}M = \mathbf{0}$ and $\mathbf{1}^{\mathsf{T}}b = 0$ (we can add a multiple of $\mathbf{1}$ to y). For each $i \in \mathbb{Z}_+$, let $U_i := \{v \mid y_v \geq i\}$. (So $U_i = \emptyset$ for i large enough.) Then

(13.30) $$\sum_{i=1}^{\infty} \chi^{\delta^{\text{out}}(U_i)} \leq w,$$

since for each arc $a = (u, v)$ we have $y_v - y_u \leq w(a)$, implying that the number of i such that a enters U_i is at most $\max\{0, y_v - y_u\}$, which is at most $w(a)$. So this gives a feasible integer dual solution to the problem of minimizing $w^{\mathsf{T}}x$ over (13.29). It is in fact optimum, since

(13.31) $$\sum_{i=1}^{\infty} b(U_i) = y^{\mathsf{T}}b = \mu.$$

This proves the theorem. ∎

This implies for primal integrality:

Corollary 13.13a. *If b is integer, then Q_b is integer.*

Proof. Directly from Theorem 13.13. ∎

Fulkerson and Weinberger [1975] also showed an integer decomposition theorem for nonnegative b-transshipments (it also follows directly from the total unimodularity of the incidence matrix M of D):

Theorem 13.14. *Let $D = (V, A)$, $b \in \mathbb{Z}^V$, and $k \in \mathbb{Z}_+$, with $k \geq 1$, and let $f : A \to \mathbb{Z}_+$ be a $k \cdot b$-transshipment. Then there exist b-transshipments $f_1, \ldots, f_k : A \to \mathbb{Z}_+$ such that $f = f_1 + \cdots + f_k$.*

Proof. It suffices to show that there exists a b-transshipment $g : A \to \mathbb{Z}_+$ such that $g \leq f$ — the theorem then follows by induction on k.

The existence of g follows from Gale's theorem (Corollary 11.2g), since $b(U) \leq f(\delta^{\text{in}}(U))$ for each $U \subseteq V$, as either $b(U) < 0$, or $b(U) \leq kb(U) \leq f(\delta^{\text{in}}(U))$. ∎

Theorem 13.14 implies the integer decomposition property for the polyhedron Q_b:

Corollary 13.14a. *If b is integer, the polyhedron Q_b has the integer decomposition property.*

Proof. Let $k \in \mathbb{Z}_+$ and let c be an integer vector in kQ_b. So $c \in Q_{k\cdot b}$, implying that there exists an integer $k \cdot b$-transshipment $f \leq c$. By Theorem 13.14, there exist integer b-transshipments $f_1, \ldots, f_k \geq \mathbf{0}$ with $f = f_1 + \cdots + f_k$. Define $f_1' := f_1 + (c - f)$. Then $f_1', f_2, \ldots, f_k \in Q_b$ and $c = f_1' + f_2 + \cdots + f_k$. ∎

If b is integer, we know:

$$Q_b = \text{conv.hull}\{f \mid f \text{ nonnegative integer } b\text{-transshipment}\} + \mathbb{R}_+^A. \tag{13.32}$$

Hence the blocking polyhedron $B(Q_b)$ of Q_b is determined by:

$$\begin{array}{lll} \text{(i)} & x(a) \geq 0 & \text{for each } a \in A, \\ \text{(ii)} & f^{\mathsf{T}}x \geq 1 & \text{for each nonnegative integer } b\text{-transshipment } f. \end{array} \tag{13.33}$$

Fulkerson and Weinberger [1975] derived from Corollary 13.14a that this system has the integer rounding property if b is integer:

Corollary 13.14b. *If b is integer, system* (13.33) *has the integer rounding property.*

Proof. Choose $c \in \mathbb{Z}^A$. Let

$$\mu := \max\{\sum_f z_f \mid z_f \geq 0, \sum_f z_f f \leq c\} \tag{13.34}$$

and let μ' be the maximum in which the z_f are restricted to nonnegative integers (here f ranges over minimal nonnegative integer b-transshipments). Let $k := \lfloor \mu \rfloor$. We must show that $\mu' = k$.

As $\mu = \min\{c^{\mathsf{T}}x \mid x \in B(Q_b)\}$, we know that $c \in \mu \cdot Q_b$. Hence, as $k \leq \mu$, $c \in kQ_b$. By Corollary 13.14a, there exist nonnegative integer b-transshipments $f_1, \ldots, f_k$ with $f_1 + \cdots + f_k \leq c$. Hence $\mu' \geq k$. Since $\mu' \leq \mu$, we have $\mu' = k$. ∎

Generally, we cannot restrict (13.33) to those f that form a vertex of Q_b while maintaining the integer rounding property, as was shown by Fulkerson and Weinberger [1975]. Trotter and Weinberger [1978] extended these results to b-transshipments with upper and lower bounds on the arcs. For related results, see Bixby, Marcotte, and Trotter [1987].

13.2d. Unions of disjoint paths and cuts

The total unimodularity of the incidence matrix of a digraph can also be used to derive min-max relations for the minimum number of arcs covered by l arc-disjoint $s - t$ paths:

Theorem 13.15. *Let $D = (V, A)$ be a digraph, $s, t \in V$, and $l \in \mathbb{Z}_+$. Then the minimum value of $|AP_1| + \cdots + |AP_l|$ where $P_1, \ldots, P_l$ are arc-disjoint $s - t$ paths is equal to the maximum value of*

$$\left|\bigcup \mathcal{C}\right| - \sum_{C \in \mathcal{C}} (|C| - l), \tag{13.35}$$

where $\mathcal{C}$ ranges over all collections of $s-t$ cuts.

Proof. The minimum in the theorem is equal to the minimum value of $\sum_{a \in A} f(a)$ subject to

$$\begin{array}{ll} 0 \leq f(a) \leq 1 & \text{for each } a \in A, \\ f(\delta^{\text{in}}(v)) - f(\delta^{\text{out}}(v)) = 0 & \text{for each } v \in V \setminus \{s,t\}, \\ f(\delta^{\text{in}}(t)) - f(\delta^{\text{out}}(t)) = l. & \end{array} \tag{13.36}$$

By LP-duality and total unimodularity of the constraint matrix, this minimum value μ is equal to the maximum value of $l \cdot p(t) - \sum_{a \in A} y(a)$, where $y \in \mathbb{Z}_+^A$ and $p \in \mathbb{Z}^V$ satisfy:

$$\begin{array}{l} p(s) = 0; \\ p(v) - p(u) - y(a) \leq 1 \text{ for each } a = (u,v) \in A. \end{array} \tag{13.37}$$

As $\mu \geq 0$, we know $p(t) \geq 0$. Let $r := p(t)$, and for $j = 1, \ldots, r$, let $U_j := \{v \in V \mid p(v) < j\}$ and $C_j := \delta^{\text{out}}(U_j)$. Then

$$\begin{aligned} \sum_{j=1}^{r} |C_j| &\leq \sum_{\substack{a=(u,v) \in A \\ p(v) > p(u) \\ p(v) \geq 0 \\ p(u) < r}} (p(v) - p(u)) \leq \sum_{\substack{a=(u,v) \in A \\ p(v) > p(u) \\ p(v) \geq 0 \\ p(u) < r}} (1 + y(a)) \\ &\leq \Big|\bigcup_{j=1}^{r} C_j\Big| + \sum_{a \in A} y(a) = \Big|\bigcup_{j=1}^{r} C_j\Big| + l \cdot r - \mu. \end{aligned} \tag{13.38}$$

So

$$\mu \leq \Big|\bigcup_{j=1}^{r} C_j\Big| - \sum_{j=1}^{r} (|C_j| - l), \tag{13.39}$$

and we have the required min-max equality. ∎

A similar formula holds for unions of arc-disjoint $s-t$ cuts:

Theorem 13.16. *Let $D = (V, A)$ be a digraph, $s, t \in V$, and $l \in \mathbb{Z}_+$. Then the minimum value of $|C_1| + \cdots + |C_l|$ where $C_1, \ldots, C_l$ are disjoint $s-t$ cuts is equal to the maximum value of*

$$\left|\bigcup \mathcal{P}\right| - \sum_{P \in \mathcal{P}} (|AP| - l), \tag{13.40}$$

where $\mathcal{P}$ ranges over all collections of $s-t$ paths.

Proof. By total unimodularity, the minimum size of the union of l disjoint $s-t$ cuts is equal to the minimum value of $\sum_{a \in A} x(a)$ where $x \in \mathbb{R}^A$ and $p \in \mathbb{R}^V$ such that

$$\begin{array}{ll} 0 \leq x(a) \leq 1 & \text{for each } a \in A, \\ p(v) - p(u) - x(a) \leq 0 & \text{for each } a = (u,v) \in A; \\ p(t) - p(s) = l. & \end{array} \tag{13.41}$$

By LP-duality and total unimodularity, this is equal to the maximum value of $l \cdot r - \sum_{a \in A} y(a)$, where $r \in \mathbb{Z}$, $y \in \mathbb{Z}_+^A$, $f \in \mathbb{Z}_+^A$ such that

$$\begin{aligned} & f(\delta^{\text{in}}(v)) - f(\delta^{\text{out}}(v)) = 0 && \text{for each } v \in V \setminus \{s,t\}, \\ & f(\delta^{\text{in}}(t)) - f(\delta^{\text{out}}(t)) = r, \\ & f(a) - y(a) \leq 1 && \text{for each } a \in A. \end{aligned} \tag{13.42}$$

As f is an $s-t$ flow of value r, it is the sum of the incidence vectors of $s-t$ paths $P_1, \ldots, P_r$, say. Then

$$\sum_{a \in A} y(a) \geq \sum_{j=1}^{r} |AP_j| - \Big| \bigcup_{j=1}^{r} AP_j \Big|. \tag{13.43}$$

Hence we have the required equality. ∎

One may derive similarly min-max formulas for the minimum number of vertices in l internally vertex-disjoint $s-t$ paths and for the minimum number of vertices in l disjoint $s-t$ vertex-cuts.

Minimum-cost flow methods also provide fast algorithms to find optimum unions of disjoint paths:

Theorem 13.17. *Given a digraph $D = (V, A)$, $s, t \in V$, and $l \in \mathbb{Z}_+$, a collection of arc-disjoint $s-t$ paths $P_1, \ldots, P_l$ minimizing $|AP_1| + \cdots + |AP_l|$ can be found in time $O(lm)$.*

Proof. Directly from Theorems 12.6 and 11.1 and Corollary 7.8a. ∎

Similarly for disjoint cuts:

Theorem 13.18. *Given a digraph $D = (V, A)$, $s, t \in V$, $l \in \mathbb{Z}_+$, and a length function $k : A \to \mathbb{Q}_+$, a collection of arc-disjoint $s-t$ paths $P_1, \ldots, P_l$ minimizing $k(P_1) + \cdots + k(P_l)$ can be found in time $O(l(m + n \log n))$.*

Complexity survey for finding k arc-disjoint $s-t$ paths of minimum total length ($*$ indicates an asymptotically best bound in the table):

$*$	$O(k \cdot \text{SP}(n, m, L))$	Ford and Fulkerson [1958b], Jewell [1958], Busacker and Gowen [1960], Iri [1960]
	$O(nL \cdot \text{DP}_k(n, m))$	Edmonds and Karp [1972]
$*$	$O(n \log L \cdot \text{DP}_k(n, m))$	Röck [1980] (cf. Bland and Jensen [1992])

Here $\text{DP}_k(n, m)$ denotes the time needed to find k arc-disjoint disjoint $s-t$ paths in a digraph with n vertices and m edges.

Suurballe and Tarjan [1984] described an $O(m \log_{m/n} n)$ algorithm for finding, in a digraph with nonnegative length function and fixed vertex s, for all v a pair of edge-disjoint $s-v$ paths P_v, Q_v with $\text{length}(P_v) + \text{length}(Q_v)$ minimum.

Gabow [1983b,1985b] described minimum-cost flow algorithms for networks with unit capacities. The running times are $O(m^{7/4} \log L)$ and, if D is simple, $O(n^{1/3} m^{3/2} \log L)$. For the vertex-disjoint case, he gave algorithms with running

time $O(n^{3/4}m \log L)$ and $O(nm \log_{2+\frac{m}{n}} L)$. Goldberg and Tarjan [1990] gave an $O(nm \log(nL))$ algorithm for minimum-cost flow with unit capacities. More complexity results follow from the table in Section 12.5a. Disjoint $s-t$ cuts were considered by Wagner [1990] and Talluri and Wagner [1994].

13.3. Network matrices

Let $D = (V, A)$ be a digraph and let $T = (V, A')$ be a directed tree. Let C be the $A' \times A$ matrix defined as follows. Take $a' \in A'$ and $a = (u, v) \in A$ and let P be the undirected $u - v$ path in T. Define

$$C_{a',a} := \begin{cases} +1 \text{ if } a' \text{ occurs in forward direction in } P, \\ -1 \text{ if } a' \text{ occurs in backward direction in } P, \\ \ \ 0 \text{ if } a' \text{ does not occur in } P. \end{cases} \tag{13.44}$$

Matrix C is called a *network matrix*, *generated by* $T = (V, A')$ and $D = (V, A)$.

Theorem 13.19. *Any submatrix of a network matrix is again a network matrix.*

Proof. Deleting column indexed by $a \in A$ corresponds to deleting a from $D = (V, A)$. Deleting the row indexed by $a' = (u, v) \in A'$ corresponds to contracting a' in the tree $T = (V, A')$ and identifying u and v in D. ∎

The following theorem is implicit in Tutte [1965a]:

Theorem 13.20. *A network matrix is totally unimodular.*

Proof. By Theorem 13.19, it suffices to show that any square network matrix C has determinant 0, 1, or -1. We prove this by induction on the size of C, the case of 1×1 matrices being trivial. We use notation as above.

Assume that $\det C \neq 0$. Let u be an end vertex of T and let a' be the arc in T incident with u. By reversing orientations, we can assume that each arc in A and A' incident with u, has u as tail. Then, by definition of C, the row indexed by a' contains only 0's and 1's.

Consider two 1's in row a'. That is, consider two columns indexed by arcs $a_1 = (u, v_1)$ and $a_2 = (u, v_2)$ in A. Subtracting column a_1 from column a_2, has the effect of resetting a_2 to (v_1, v_2). So after that, column a_2 has a 0 in position a'. Since this subtraction does not change the determinant, we can assume that there is exactly one arc in A incident with u; that is, row a' has exactly one nonzero. Then by expanding the determinant by row a', we obtain inductively that $\det C = \pm 1$. ∎

The incidence matrix of a digraph $D = (V, A)$ is a network matrix: add a new vertex u to D giving digraph $D' = (V \cup \{u\}, A)$. Let T be the directed

tree on $V \cup \{u\}$ with arcs (u, v) for $v \in V$. Then the network matrix generated by T and D' is equal to the incidence matrix of D.

Notes. Recognizing whether a given matrix is a network matrix has been studied by Gould [1958], Auslander and Trent [1959,1961], Tutte [1960,1965a,1967], Tomizawa [1976b], and Fujishige [1980a] (cf. Bixby and Wagner [1988] and Section 20.1 in Schrijver [1986b]).

Seymour [1980a] showed that all totally unimodular matrices can be obtained by glueing network matrices and copies of a certain 5×5 matrix together (cf. Schrijver [1986b] and Truemper [1992]).

13.4. Cross-free and laminar families

We now show how cross-free and laminar families of sets give rise to network matrices. The results in this section will be used mainly in Part V.

A family $\mathcal{C}$ of subsets of a finite set S is called *cross-free* if for all $X, Y \in \mathcal{C}$ one has

(13.45) $X \subseteq Y$ or $Y \subseteq X$ or $X \cap Y = \emptyset$ or $X \cup Y = S$.

$\mathcal{C}$ is called *laminar* if for all $X, Y \in \mathcal{C}$ one has

(13.46) $X \subseteq Y$ or $Y \subseteq X$ or $X \cap Y = \emptyset$.

So each laminar family is cross-free.

Cross-free families could be characterized geometrically as having a 'Venn-diagram' representation on the sphere without crossing lines. If the family is laminar we have such a representation in the plane.

A laminar collection $\mathcal{C}$ can be partitioned into 'levels': the ith level consists of all sets $X \in \mathcal{C}$ such that there are $i-1$ sets $Y \in \mathcal{C}$ satisfying $Y \supset X$. Then each level consists of disjoint sets, and for each set X of level $i+1$ there is a unique set of level i containing X.

Note that if $\mathcal{C}$ is a cross-free family, then adding, for each set $X \in \mathcal{C}$, the complement $S \setminus X$ to $\mathcal{C}$ maintains cross-freeness. Moreover, for any fixed $s \in S$, the family $\{X \in \mathcal{C} \mid s \notin X\}$ is laminar.

In order to relate cross-free families and directed trees, suppose that we have a directed tree $T = (V, A)$ and a function $\pi : S \to V$, for some set S. Then the pair T, π defines a family $\mathcal{C}$ of subsets of S as follows. Define for each arc $a = (u, v)$ of T the subset X_a of S by:

(13.47) $X_a :=$ the set of vertices in the weak component of $T - a$ containing v.

So X_a is the set of $s \in S$ for which arc a 'points' in the direction of $\pi(s)$ in T.

Let $\mathcal{C}_{T,\pi}$ be the family of sets X_a; that is,

(13.48) $\mathcal{C}_{T,\pi} := \{X_a \mid a \in A\}$.

If $\mathcal{C} = \mathcal{C}_{T,\pi}$, the pair T, π is called a *tree-representation* for $\mathcal{C}$. If moreover T is a rooted tree, then T, π is called a *rooted tree-representation* for $\mathcal{C}$.

It is easy to see that $\mathcal{C}_{T,\pi}$ is cross-free. Moreover, if T is a rooted tree, then $\mathcal{C}_{T,\pi}$ is laminar. In fact, each cross-free family has a tree-representation, and each laminar family has a rooted tree-representation, as is shown by the following theorem of Edmonds and Giles [1977]:

Theorem 13.21. *A family $\mathcal{C}$ of subsets of S is cross-free if and only if $\mathcal{C}$ has a tree-representation. Moreover, $\mathcal{C}$ is laminar if and only if $\mathcal{C}$ has a rooted tree-representation.*

Proof. As sufficiency of the conditions is easy, we show necessity. We first show that each laminar family $\mathcal{C}$ of subsets of a set S has a rooted tree-representation. The proof is by induction on $|\mathcal{C}|$, the case $\mathcal{C} = \emptyset$ being trivial. If $\mathcal{C} \neq \emptyset$, choose an inclusionwise minimal $X \in \mathcal{C}$. By induction, the family $\mathcal{C}' := \mathcal{C} \setminus \{X\}$ has a rooted tree-representation $T = (V, A)$, $\pi : S \to V$.

If $X = \emptyset$, then we can add to T a new arc from any vertex to a new vertex, to obtain a rooted tree-representation T', π of $\mathcal{C}$. So we can assume that $X \neq \emptyset$.

Now $|\pi(X)| = 1$, since if $\pi(x) \neq \pi(y)$ for some $x, y \in X$, then there is an arc a of T separating $\pi(x)$ and $\pi(y)$. Hence the set $X_a \in \mathcal{C}'$ contains one of $\pi(x)$ and $\pi(y)$, say $\pi(y)$. As $\mathcal{C}$ is laminar, this implies that X_a is properly contained in X, contradicting the minimality of X.

This proves that $|\pi(X)| = 1$. Let v be the vertex of T with $\pi(X) = \{v\}$. Augment T by a new vertex w and a new arc $b = (v, w)$. Reset $\pi(z) := w$ for each $z \in X$. Then the new tree and π form a rooted tree-representation for $\mathcal{C}$. This shows that each laminar family has a rooted tree-representation.

To see that each cross-free family $\mathcal{C}$ has a tree-representation, choose $s \in S$, and let $\mathcal{G}$ be obtained from $\mathcal{C}$ by replacing any set containing s by its complement. Then $\mathcal{G}$ is laminar, and hence it has a rooted tree-representation by the foregoing. Reversing arcs in the tree if necessary, it gives a tree-representation for $\mathcal{C}$. ∎

From Theorems 13.20 and 13.21 we derive the total unimodularity of certain matrices. Let $D = (V, A)$ be a directed graph and let $\mathcal{C}$ be a family of subsets of V. Let N be the $\mathcal{C} \times A$ matrix defined by:

$$N_{X,a} := \begin{cases} 1 & \text{if } a \text{ enters } X, \\ -1 & \text{if } a \text{ leaves } X, \\ 0 & \text{otherwise,} \end{cases} \tag{13.49}$$

for $X \in \mathcal{C}$ and $a \in A$.

Corollary 13.21a. *If $\mathcal{C}$ is cross-free, then N is a network matrix, and hence N is totally unimodular.*

Proof. Let $T = (W, B)$, $\pi : V \to W$ be a tree-representation for $\mathcal{C}$. Let $D' = (W, A')$ be the directed graph with

$$A' := \{(\pi(u), \pi(v)) \mid (u, v) \in A\}. \tag{13.50}$$

Then N is equal to the network matrix generated by T and D' (up to identifying any arc b of T with the set X_b in $\mathcal{C}$ determined by b, and any arc (u, v) of D with the arc $(\pi(u), \pi(v))$ of D'). Hence by Theorem 13.20, N is totally unimodular. ∎

Chapter 14

Partially ordered sets and path coverings

Partially ordered sets can be considered as a special type of networks, and several optimization problems on partially ordered sets can be handled with flow techniques. Basic theorem is Dilworth's min-max relation for the maximum size of an antichain.

14.1. Partially ordered sets

A *partially ordered set* is a pair $(S, \leq)$ where S is a set and where $\leq$ is a relation on S satisfying:

(14.1) (i) $s \leq s$,
(ii) if $s \leq t$ and $t \leq s$, then $s = t$,
(iii) if $s \leq t$ and $t \leq u$, then $s \leq u$,

for all $s, t, u \in S$. We put $s < t$ if $s \leq t$ and $s \neq t$. We restrict ourselves to *finite* partially ordered sets; that is, with S finite.

A subset C of S is called a *chain* if $s \leq t$ or $t \leq s$ for all $s, t \in C$. A subset A of S is called an *antichain* if $s \not\leq t$ and $t \not\leq s$ for all $s, t \in A$. Hence if C is a chain and A is an antichain, then

(14.2) $|C \cap A| \leq 1$.

First we notice the following easy min-max relation:

Theorem 14.1. *Let $(S, \leq)$ be a partially ordered set. Then the minimum number of antichains covering S is equal to the maximum size of a chain.*

Proof. That the maximum cannot be larger than the minimum follows easily from (14.2). To see that the two numbers are equal, define for any element $s \in S$ the *height* of s as the maximum size of any chain in S with maximum s. For any $i \in \mathbb{Z}_+$, let A_i denote the set of elements of height i. Let k be the maximum height of the elements of S. Then $A_1, \ldots, A_k$ are antichains covering S, and moreover there exists a chain of size k, since there exists an element of height k. ∎

This result can also be formulated in terms of graphs. Let $D = (V, A)$ be a digraph. A subset C of A is called a *directed cut* if there exists a subset U of V such that $\emptyset \neq U \neq V$, $\delta^{\text{out}}(U) = C$, and $\delta^{\text{in}}(U) = \emptyset$. Then Vidyasankar and Younger [1975] observed:

Corollary 14.1a. *Let $D = (V, A)$ be an acyclic digraph. Then the minimum number of directed cuts covering A is equal to the maximum length of a directed path.*

Proof. Define a partial order $\leq$ on A by: $a < a'$ if there exists a directed path traversing a and a', in this order. Applying Theorem 14.1 gives the Corollary. ∎

14.2. Dilworth's decomposition theorem

Dilworth [1950] proved that Theorem 14.1 remains true after interchanging the terms 'chain' and 'antichain', which is less simple to prove:

Theorem 14.2 (Dilworth's decomposition theorem)**.** *Let $(S, \leq)$ be a partially ordered set. Then the minimum number of chains covering S is equal to the maximum size of an antichain.*

Proof. That the maximum cannot be larger than the minimum follows easily from (14.2). To see that the two numbers are equal, we apply induction on $|S|$. Let α be the maximum size of an antichain and let A be an antichain of size α. Define

$$\begin{aligned} A^{\downarrow} &:= \{s \in S \mid \exists t \in A : s \leq t\}, \\ A^{\uparrow} &:= \{s \in S \mid \exists t \in A : s \geq t\}. \end{aligned} \tag{14.3}$$

Then $A^{\downarrow} \cap A^{\uparrow} = A$ and $A^{\downarrow} \cup A^{\uparrow} = S$ (otherwise we can augment A).

First assume that $A^{\downarrow} \neq S$ and $A^{\uparrow} \neq S$. Then, by induction, $A^{\downarrow}$ can be covered by α chains. Since $A \subseteq A^{\downarrow}$, each of these chains contains exactly one element in A. For each $s \in A$, let C_s denote the chain containing s. Similarly, there exist α chains C'_s (for $s \in A$) covering $A^{\uparrow}$, where C'_s contains s. Then for each $s \in A$, $C_s \cup C'_s$ forms a chain in S, and moreover these chains cover S.

So we may assume that $A^{\downarrow} = S$ or $A^{\uparrow} = S$ for each antichain A of size α. It means that each antichain A of size α is either the set of minimal elements of S or the set of maximal elements of S. Now choose a minimal element s and a maximal element t of S with $s \leq t$. Then the maximum size of an antichain in $S \setminus \{s, t\}$ is equal to $\alpha - 1$ (since each antichain in S of size α contains s or t). By induction, $S \setminus \{s, t\}$ can be covered by $\alpha - 1$ chains. Adding the chain $\{s, t\}$ yields a covering of S by α chains. ∎

Notes. This proof is due to Perles [1963]. Dilworth original proof is based on a different induction. For a proof using linear programming duality, see Dantzig and Hoffman [1956]. For a deduction of Dilworth's decomposition theorem from Kőnig's matching theorem (Theorem 16.2), see Fulkerson [1956], Ford and Fulkerson [1962] (pp. 61–64), and Mirsky and Perfect [1966]. Further proofs were given by Dilworth [1960], Tverberg [1967], and Pretzel [1979].

14.3. Path coverings

Dilworth's decomposition theorem can be formulated equivalently in terms of covering vertices of a digraph[22]:

Corollary 14.2a. *Let $D = (V, A)$ be an acyclic digraph. Then the minimum number of paths covering all vertices is equal to the maximum number of vertices no two of which belong to a directed path.*

Proof. Apply Dilworth's decomposition theorem to the partially ordered set $(V, \leq)$ where $u \leq v$ if and only if v is reachable in D from u. ▮

As for covering the arcs, we have:

Corollary 14.2b. *Let $D = (V, A)$ be an acyclic digraph. Then the minimum number of paths covering all arcs is equal to the maximum size of a directed cut.*

Proof. Apply Dilworth's decomposition theorem to the partially ordered set $(A, \leq)$ where $a < a'$ if and only if there exists a directed path traversing a and a', in this order. ▮

Similarly, for $s - t$ paths:

Corollary 14.2c. *Let $D = (V, A)$ be an acyclic digraph with exactly one source, s, and exactly one sink, t. Then the minimum number of $s - t$ paths covering A is equal to the maximum size of a directed $s - t$ cut.*

Proof. Apply Dilworth's decomposition theorem to the partially ordered set $(A, \leq)$ defined by: $a \leq a'$ if and only if there exists an $s - t$ path traversing a and a', in this order. ▮

If only a subset of the arcs has to be covered, one has more generally:

Corollary 14.2d. *Given an acyclic digraph $D = (V, A)$ and $B \subseteq A$, the minimum number of paths covering B is equal to the maximum of $|C \cap B|$ where C is a directed cut.*

[22] Gallai and Milgram [1960] claim to have found this result in 1947.

Proof. Consider the partially ordered set $(B, \leq)$ with $a < a'$ if there exists a directed path traversing a and a', in this order. Then for each chain K in $(B, \leq)$ there is a path in D covering K, and for each antichain L in $(B, \leq)$ there is a directed cut C in D with $L \subseteq C \cap B$. Hence the theorem follows from Dilworth's decomposition theorem. ▮

14.4. The weighted case

Dilworth's decomposition theorem has a self-refining nature, and implies a weighted version. Let $(S, \leq)$ be a partially ordered set. Let $\mathcal{C}$ and $\mathcal{A}$ denote the collections of chains and antichains in $(S, \leq)$, respectively. Let $w : S \to \mathbb{Z}_+$ be a 'weight' function. Then:

Theorem 14.3. *The maximum weight $w(A)$ of an antichain A is equal to the minimum size of a family of chains covering each element s exactly $w(s)$ times.*

Proof. Replace each element s of S by $w(s)$ copies, making the set S'. For any copy s' of s and t' of t, define $s' <' t'$ if and only if $s < t$. This gives the partially ordered set $(S', \leq')$. Note that the copies of one element of S form an antichain in S'.

Then the maximum weight $w(A)$ of an antichain A in S is equal to the maximum size $|A'|$ of an antichain A' in S'. By Dilworth's decomposition theorem, S' can be covered by a collection Λ of $|A'|$ chains. Replacing the elements of each chain by their originals in S, gives the required equality. ▮

In terms of digraphs this gives the following result of Gallai [1958a,1958b]:

Corollary 14.3a. *Let $D = (V, A)$ be an acyclic digraph and let S and T be subsets of V such that each vertex is on at least one $S - T$ path. Let $c \in \mathbb{Z}_+^V$. Then the minimum number k of $S - T$ paths $P_1, \ldots, P_k$ such that each vertex v is covered at least $c(v)$ times by the P_i is equal to the maximum of $c(U)$ where U is a set of vertices intersecting each $S - T$ path at most once.*

Proof. Directly from Theorem 14.3 by defining the partially ordered set $(V, \leq)$ by: $u \leq v$ if and only if there exists a $u - v$ path. ▮

Similarly, there is the following 'min-flow max-cut theorem':

Corollary 14.3b. *Let $D = (V, A)$ be an acyclic digraph with exactly one source, s, and exactly one sink, t. Let $d : A \to \mathbb{R}_+$. Then the minimum value of any $s - t$ flow f satisfying $f \geq d$ is equal to the maximum value of $d(C)$ where C is a directed $s - t$ cut. If d is integer, we can take f integer.*

Proof. Define a partial order $\leq$ on A by: $a < a'$ if there is an $s-t$ path traversing a and a' in this order. Then any chain in A is contained in some $s-t$ path and any antichain is contained in some directed $s-t$ cut. Hence this Corollary can be derived from Theorem 14.3 (using continuity, compactness, and scaling). ∎

A similar weighted variant of the easier Theorem 14.1 holds: Again, let $(S, \leq)$ be a partially ordered set. Let $w : S \to \mathbb{Z}_+$ be a 'weight' function.

Theorem 14.4. *The maximum weight $w(C)$ of any chain is equal to the minimum size of a family of antichains covering each element s exactly $w(s)$ times.*

Proof. Similar to the proof of Theorem 14.3. ∎

The following 'length-width inequality' follows similarly:

Theorem 14.5. *Let $(S, \leq)$ be a partially ordered set and let $l, w : S \to \mathbb{R}_+$. Then*

$$\max_{C \text{ chain}} l(C) \cdot \max_{A \text{ antichain}} w(A) \geq \sum_{s \in S} l(s)w(s). \tag{14.4}$$

Proof. We can assume that l and w are rational (by continuity), and hence integer. Let t be the maximum of $l(C)$ taken over all chains C.

For each $s \in S$, let $h(s)$ be the maximum of $l(C)$ taken over all chains C with maximum element s. For each $k \in \mathbb{Z}$, let A_k be the set of those elements $s \in S$ with $h(s) - l(s) < k \leq h(s)$.

Then each A_k is an antichain. Moreover, $A_k = \emptyset$ if $k > t$, and

$$\sum_{k=1}^{t} \chi^{A_k} = l. \tag{14.5}$$

Therefore,

$$\begin{aligned} \max_C l(C) \cdot \max_A w(A) = t \cdot \max_A w(A) \geq \sum_{k=1}^{t} w(A_k) = \sum_{k=1}^{t} w^{\mathsf{T}} \chi^{A_k} \\ = w^{\mathsf{T}} l, \end{aligned} \tag{14.6}$$

where C and A range over chains and antichains, respectively. ∎

14.5. The chain and antichain polytopes

Let $(S, \leq)$ be a partially ordered set. The *chain polytope* $P_{\text{chain}}(S)$ of S is the convex hull of the incidence vectors (in $\mathbb{R}^S$) of chains in S. Similarly,

the *antichain polytope* $P_{\text{antichain}}(S)$ of S is the convex hull of the incidence vectors (in $\mathbb{R}^S$) of antichains in S.

These two polytopes turn out to form an antiblocking pair of polyhedra. To see this, we first show:

Corollary 14.5a. *The chain polytope of S is determined by*

$$\text{(14.7)} \qquad \begin{array}{lll} \text{(i)} & 0 \le x_s \le 1 & \text{for each } s \in S, \\ \text{(ii)} & x(A) \le 1 & \text{for each antichain } A, \end{array}$$

and this system is TDI.

Proof. Directly from Theorem 14.4, by LP-duality: for any chain C, χ^C is a feasible solution of (14.7). An antichain family covering each element s precisely $w(s)$ times gives a dual feasible solution. As the minimum of $w(C)$ is equal to the maximum value of these dual feasible solutions (by Theorem 14.4), the linear program of minimizing $w^{\mathsf{T}}x$ over (14.7) has integer optimum primal and dual solutions. ▮

Similarly for the antichain polytope:

Theorem 14.6. *The antichain polytope of S is determined by the inequalities*

$$\text{(14.8)} \qquad \begin{array}{lll} \text{(i)} & 0 \le x_s \le 1 & \text{for each } s \in S, \\ \text{(ii)} & x(C) \le 1 & \text{for each chain } C, \end{array}$$

and this system is TDI.

Proof. Similar to the previous proof, now using Theorem 14.3. ▮

This implies that the chain and antichain polytope are related by the antiblocking relation:

Corollary 14.6a. $P^{\uparrow}_{\text{chain}}(S)$ *and* $P^{\uparrow}_{\text{antichain}}(S)$ *form an antiblocking pair of polyhedra.*

Proof. Directly from the previous results. ▮

14.5a. Path coverings algorithmically

When studying partially ordered sets $(S, \le)$ algorithmically, we should know how these are represented. Generally, giving all pairs (s, t) with $s \le t$ yields a large, redundant input. It suffices to give an acyclic digraph $D = (S, A)$ such that $s \le t$ if and only if t is reachable from s. So it is best to formulate the algorithmic results in terms of acyclic digraphs.

The strong polynomial-time solvability of the problems discussed below follow from the strong polynomial-time solvability of minimum-cost circulation. We give some better running time bounds.

Theorem 14.7. *Given an acyclic digraph $D = (V, A)$ and $B \subseteq A$, a minimum number of paths covering B can be found in time $O(nm)$.*

Proof. Add two vertices s and t to D, and, for each vertex v, make $\deg_B^{\text{in}}(v)$ parallel arcs from s to v and $\deg_B^{\text{out}}(v)$ parallel arcs from v to t. Let $D' = (V', A')$ be the extended graph. By Theorem 9.10, we can find in time $O(nm)$ a maximum collection $P_1, \ldots, P_k$ of $s-t$ paths that are disjoint on the set $A' \setminus A$ of new arcs.

We make another auxiliary graph $\widetilde{D} = (V, \widetilde{A})$ as follows. Each arc in B belongs to $\widetilde{A}$. Moreover, for each $a = (u, v) \in A$, we make r parallel arcs from u to v, where r is the number of times a is traversed by the P_i. (So if $a \in B$, there are $r+1$ parallel arcs from u to v.) This gives the acyclic graph $\widetilde{D}$. Now choose, repeatedly as long as possible, in $\widetilde{D}$ a path from a (current) source to a (current) sink and remove its arcs. This gives us a collection of paths in $\widetilde{D}$ and hence also in D, covering all arcs in B. We claim that it has minimum size.

For each i, let P_i' be the path obtained from P_i by deleting the first and last arc. For each $v \in V$, let $\sigma(v)$ be the number of P_i that start with (s, v) and let $\tau(v)$ be the number of P_i that and with (v, t).

Let $U \subseteq V$ give a minimum $s-t$ cut $\delta_{A'}^{\text{out}}(U \cup \{s\})$ in D' with $\delta_A^{\text{out}}(U) = \emptyset$. As $P_1, \ldots, P_k$ form a maximum $s-t$ path packing in D', we have for each $v \in V$:

$$\begin{aligned} &\sigma(v) \leq \delta_B^{\text{in}}(v), \text{ with equality if } v \in V \setminus U,\\ &\tau(v) \leq \delta_B^{\text{out}}(v), \text{ with equality if } v \in U. \end{aligned} \tag{14.9}$$

So

$$\begin{aligned} \deg_{\widetilde{A}}^{\text{in}}(v) &= \deg_B^{\text{in}}(v) + \sum_{i=1}^{k} \deg_{AP_i'}^{\text{in}}(v) \geq \sigma(v) + \sum_{i=1}^{k} \deg_{AP_i'}^{\text{in}}(v)\\ &= \sum_{i=1}^{k} \deg_{AP_i}^{\text{in}}(v), \end{aligned} \tag{14.10}$$

with equality if $v \in V \setminus U$. Similarly,

$$\begin{aligned} \deg_{\widetilde{A}}^{\text{out}}(v) &= \deg_B^{\text{out}}(v) + \sum_{i=1}^{k} \deg_{AP_i'}^{\text{out}}(v) \geq \tau(v) + \sum_{i=1}^{k} \deg_{AP_i'}^{\text{out}}(v)\\ &= \sum_{i=1}^{k} \deg_{AP_i}^{\text{out}}(v), \end{aligned} \tag{14.11}$$

with equality if $v \in U$. Hence, since $\deg_{AP_i}^{\text{in}}(v) = \deg_{AP_i}^{\text{out}}(v)$ for each $v \in V$ and each $i = 1, \ldots, k$:

$$\begin{aligned} &\deg_{\widetilde{D}}^{\text{in}}(v) \geq \deg_{\widetilde{D}}^{\text{out}}(v) \text{ for each } v \in U \text{ and}\\ &\deg_{\widetilde{D}}^{\text{in}}(v) \leq \deg_{\widetilde{D}}^{\text{out}}(v) \text{ for each } v \in V \setminus U. \end{aligned} \tag{14.12}$$

Now deleting any source-sink path P in $\widetilde{D}$ does not invalidate (14.12). Moreover, P runs from $V \setminus U$ to U. Since none of the arcs in $\delta_A^{\text{in}}(U)$ are traversed by any P_i, we know that P should use an arc of $B \cap \delta^{\text{in}}(U)$. So the number of paths found is at most $|B \cap \delta_A^{\text{in}}(U)|$. Therefore, by Corollary 14.2d, the paths form a minimum-size collection of paths covering B. ∎

The special case where *all* arcs must be covered, is:

Corollary 14.7a. *Given an acyclic digraph $D = (V, A)$, a minimum collection of paths covering all arcs can be found in time $O(nm)$.*

Proof. Directly from the foregoing, by taking $B := A$. ∎

The theorem also applies to vertex coverings:

Corollary 14.7b. *Given an acyclic digraph $D = (V, A)$, a minimum number of paths covering all vertices can be found in time $O(nm)$.*

Proof. Introduce for each vertex v of D vertices v' and v''. Define $V' := \{v' \mid v \in V\}$, $V'' := \{v'' \mid v \in V\}$, $A' := \{(v', v'') \mid v \in V\}$, and $A'' := \{(u'', v') \mid (u, v) \in A\}$.

Then a minimum cover of A' by paths in the new graph $(V' \cup V'', A' \cup A'')$ gives a minimum cover of V by paths in the original graph. ∎

It is trivial to extend the results to $s - t$ paths:

Corollary 14.7c. *Given an acyclic digraph $D = (V, A)$ and $s, t \in V$, a minimum collection of $s - t$ paths covering all arcs can be found in time $O(nm)$.*

Proof. We may assume that each arc of D is contained in at least one $s-t$ path. But then each path can be extended to an $s - t$ path, and hence a minimum collection of paths gives a minimum collection of $s - t$ paths. ∎

One similarly has for covering the vertices:

Corollary 14.7d. *Given an acyclic digraph $D = (V, A)$ and $s, t \in V$, a minimum collection of $s - t$ paths covering all vertices can be found in time $O(nm)$.*

Proof. Similar to the proof of Corollary 14.7b. ∎

These bounds are best possible, as the size of the output is $\Omega(nm)$. As for paths covering the arcs, this can be seen by taking vertices $v_1, \ldots, v_n$, with r parallel arcs from v_1 to v_2, r parallel arcs from v_{n-1} to v_n, and one arc from v_{i-1} to v_i for each $i = 2, \ldots, n - 1$. Then the number of arcs is $2r + n - 2$, while any minimum path covering of the arcs consists of r paths of length $n - 1$ each.

14.6. Unions of directed cuts and antichains

The following theorem is (in the terminology of partially ordered sets — see Corollary 14.8a) due to Greene and Kleitman [1976]. We follow the proof method of Fomin [1978] and Frank [1980a] based on minimum-cost circulations.

Theorem 14.8. *Let $D = (V, A)$ be an acyclic digraph, let $B \subseteq A$, and let $k \in \mathbb{Z}_+$. Then the maximum of $|B \cap \bigcup \mathcal{C}|$, where $\mathcal{C}$ is a collection of at most k directed cuts is equal to the minimum value of*

(14.13) $|B \setminus \bigcup \mathcal{P}| + k \cdot |\mathcal{P}|$,

where $\mathcal{P}$ is a collection of directed paths.

Proof. To see max $\leq$ min, let $\mathcal{C}$ be a collection of at most k directed cuts and let $\mathcal{P}$ be a collection of directed paths. Then, setting $\Gamma := \bigcup \mathcal{C}$ and $\Pi := \bigcup \mathcal{P}$,

(14.14) $|B \cap \Gamma| \leq |B \setminus \Pi| + |\Gamma \cap \Pi| \leq |B \setminus \Pi| + k \cdot |\mathcal{P}|$.

(Note that any directed path P intersects any directed cut in at most one edge; hence $|\Gamma \cap AP| \leq k$ for each $P \in \mathcal{P}$.)

In proving equality, we may assume that D has exactly one source, s say, and exactly one sink, t say. (Adding s to V, and all arcs (s, v) for $v \in V$ does not change the theorem. Similarly for adding a sink.)

Define for each $a \in A$, a capacity $c(a) := \infty$ and a cost $l(a) := 0$. For each arc $a = (u, v) \in B$, introduce a new arc $a' = (u, v)$ parallel to a, with $c(a) := 1$ and $l(a) := -1$. Finally, add an arc (t, s), with $c(t, s) := \infty$ and $l(t, s) := k$. This makes the digraph $\widetilde{D} = (V, \widetilde{A})$.

Let $f : \widetilde{A} \to \mathbb{Z}$ be a minimum-cost nonnegative circulation in $\widetilde{D}$ subject to c. As D_f has no negative-cost directed circuits (Theorem 12.1), there exists a function $p : V \to \mathbb{Z}$ such that for each $a = (u, v) \in A$:

(14.15) $p(v) \leq p(u)$, with equality if $f(a) \geq 1$.

Moreover, for each $a = (u, v) \in B$:

(14.16) $p(v) \leq p(u) - 1$ if $f(a') = 0$,
$p(v) \geq p(u) - 1$ if $f(a') = 1$.

Finally,

(14.17) $p(s) \leq p(t) + k$, with equality if $f(t, s) \geq 1$.

We may assume that $p(t) = 0$. So by (14.15), $p(s) \geq 0$ and by (14.17), $p(s) \leq k$. For each $i = 1, \ldots, p(s)$, let $U_i := \{v \in V \mid p(v) \geq i\}$. Then for each i, $\delta_A^{\text{out}}(U_i)$ is a directed $s - t$ cut, since $s \in U_i$, $t \notin U_i$, and no arc in A enters U_i: if $(u, v) \in A$ with $v \in U_i$, then $p(v) \geq i$, and hence by (14.15), $p(u) \geq i$, that is $u \in U_i$.

Let $\mathcal{C}$ be the collection of these directed cuts and let $\Gamma := \bigcup \mathcal{C}$. We can decompose f as a sum of incidence vectors of directed circuits in $\widetilde{D}$. Each of these circuits contains exactly one arc (t, s). Deleting it, and identifying any a' with a (for $a \in B$) gives a path collection $\mathcal{P}$ in D. Let $\Pi := \bigcup \mathcal{P}$.

Then $B \setminus \Pi = B \cap \Gamma \setminus \Pi$. For let $a = (u, v) \in B \setminus \Pi$. Then $f(a') = 0$, and hence by (14.16), $p(v) \leq p(u) - 1$. Hence $a \in \delta^{\text{out}}(U_i)$ for $i := p(u)$. So $a \in \Gamma$.

Moreover,

$$\begin{aligned} k \cdot |\mathcal{P}| &= (p(s) - p(t)) f(t, s) \\ &= \sum_{a=(u,v) \in A} (p(u) - p(v)) f(a) + \sum_{a=(u,v) \in B} (p(u) - p(v)) f(a') \\ &= \sum_{a=(u,v) \in B} (p(u) - p(v)) f(a') = |B \cap \Gamma \cap \Pi|. \end{aligned} \tag{14.18}$$

Thus $|B\setminus\Pi|+k\cdot|\mathcal{P}| = |B\setminus\Pi|+|B\cap\Gamma\cap\Pi| = |B\cap\Gamma\setminus\Pi|+|B\cap\Gamma\cap\Pi| = |B\cap\Gamma|$. ∎

This implies for partially ordered sets:

Corollary 14.8a. *Let $(S,\leq)$ be a partially ordered set and let $k\in\mathbb{Z}_+$. Then the maximum size of the union of k antichains is equal to the minimum value of*

$$\sum_{C\in\mathcal{C}} \min\{k,|C|\}, \tag{14.19}$$

where $\mathcal{C}$ ranges over partitions of S into chains.

Proof. This can be reduced to Theorem 14.8, by making a digraph $D=(V,A)$ as follows. Let for each $s\in S$, s' be a copy of s, and let $V := S\cup\{s' \mid s\in S\}$. Let A consist of all pairs (s,s') with $s\in S$ and all pairs (s',t) with $s<t$. Taking $B:=\{(s,s') \mid s\in S\}$ reduces Corollary 14.8a to Theorem 14.8. (For each arcs (s,s') in $B\setminus\bigcup\mathcal{P}$, we take a singleton $C=\{s\}$.) ∎

Corollary 14.8a can be stated in a slightly different form. For any partially ordered set $(S,\leq)$, any $Y\subseteq S$, and any $k\in\mathbb{Z}_+$, let

$$a_k(Y) := \max\{|Z| \;\big|\; Z\subseteq Y \text{ is the union of } k \text{ antichains}\}. \tag{14.20}$$

Then:

Corollary 14.8b. $a_k(S) = \min_{Y\subseteq S}(|S\setminus Y| + k\cdot a_1(Y))$.

Proof. The inequality $\leq$ follows from the fact that if Z is the union of k antichains and $Y\subseteq S$, then $|Z|\leq |Z\setminus Y| + a_k(Z\cap Y) \leq |S\setminus Y| + k\cdot a_1(Y)$.

To obtain equality, let $\mathcal{C}$ be a partition of S into chains attaining the minimum in Corollary 14.8a. Let $\mathcal{C}'$ be the collection of those chains $C\in\mathcal{C}$ with $|C|\geq k$. Let Y be the union of the chains in $\mathcal{C}'$. Then (14.19) is equal to $|S\setminus Y| + k|\mathcal{C}'|$. This is at least $|S\setminus Y| + k\cdot a_1(Y)$ (as $|\mathcal{C}'|\geq a_1(Y)$). Thus we have equality. ∎

Note that the proof of Theorem 14.8 gives a polynomial-time algorithm to find a maximum union of k directed cuts or antichains. For a proof of the results in this section based on LP-duality, see Hoffman and Schwartz [1977]. For other proofs, see Saks [1979]. For extensions, see Linial [1981] and Cameron [1986].

14.6a. Common saturating collections of chains

Greene and Kleitman [1976] also showed that for each h there is a chain partition $\mathcal{C}$ of a partially ordered set $(S,\leq)$ attaining the minimum of (14.19) both for $k=h$ and for $k=h+1$.

More generally, in terms of acyclic digraphs, there is the following result of Greene and Kleitman [1976] on the minimum in Theorem 14.8:

Theorem 14.9. *Let $D = (V, A)$ be an acyclic digraph, let $B \subseteq A$, and let $h \in \mathbb{Z}_+$. Then there is a collection $\mathcal{P}$ of directed paths attaining*

$$\min_{\mathcal{P}}(|B \setminus \bigcup \mathcal{P}| + k \cdot |\mathcal{P}|) \tag{14.21}$$

both for $k = h$ and for $k = h + 1$.

Proof. It suffices to show that in the proof of Theorem 14.8 the minimum-cost circulation f can be chosen such that it has minimum cost simultaneously with respect to the given cost function l, and with respect to the cost function l' which is the same as l except that $l'(t, s) = k + 1$.

For choose f such that it has minimum cost with respect to l, and with ${l'}^{\mathsf{T}} f$ as small as possible. Suppose that f does *not* have minimum cost with respect to l'. Then D_f has a directed circuit C with $l'(C) < 0$. As $l(C) \geq 0$, C traverses (s, t). So $l(C) - l'(C) = l(s, t) - l'(s, t) = 1$, and therefore $l(C) = 0$. So $f' := f + \chi^C$ is a feasible circulation with $l^{\mathsf{T}} f' = l^{\mathsf{T}} f$ and ${l'}^{\mathsf{T}} f' < {l'}^{\mathsf{T}} f$. This contradicts our assumption. ∎

For partially ordered sets it gives (using definition (14.20)):

Corollary 14.9a. *Let $(S, \leq)$ be a partially ordered set and let $k \in \mathbb{Z}_+$. Then there exists a chain partition $\mathcal{C}$ of S such that*

$$a_k(S) = \sum_{C \in \mathcal{C}} \min\{k, |C|\} \text{ and } a_{k+1}(S) = \sum_{C \in \mathcal{C}} \min\{k + 1, |C|\}. \tag{14.22}$$

Proof. Directly from Theorem 14.9. ∎

(For a linear programming proof and an extension, see Hoffman and Schwartz [1977]. For another proof, see Perfect [1984]. Denig [1981] showed that the common saturating chain collections determine a matroid.)

14.7. Unions of directed paths and chains

Results dual to those of the previous sections were obtained by Greene [1976] and Edmonds and Giles [1977]. They can be formulated by interchanging the terms 'chain' and 'antichain'. Again we follow the proof method of Fomin [1978] and Frank [1980a] based on minimum-cost flows.

Theorem 14.10. *Let $D = (V, A)$ be an acyclic digraph, let $B \subseteq A$, and let $k \in \mathbb{Z}_+$. Then the maximum of $|B \cap \bigcup \mathcal{P}|$, where $\mathcal{P}$ is a collection of the arc sets of at most k directed paths, is equal to the minimum value of*

$$|B \setminus \bigcup \mathcal{C}| + k \cdot |\mathcal{C}|, \tag{14.23}$$

where $\mathcal{C}$ is a collection of directed cuts.

Proof. The inequality max $\leq$ min is shown similarly as in Theorem 14.8. In proving the theorem, we may again assume that D has only one source, s say, and only one sink, t say.

To obtain equality, we again consider the extended graph $\widetilde{D}$ as in the proof of Theorem 14.8, with capacity c and cost l, except that we delete arc (t,s).

Let $f : \widetilde{A} \to \mathbb{Z}$ be a minimum-cost $s-t$ flow in $\widetilde{D}$ of value k subject to c. As D_f has no negative-cost directed circuits, there exists a function $p : V \to \mathbb{Z}$ such that for each $a = (u,v) \in A$:

$$p(v) \leq p(u), \text{ with equality if } f(a) \geq 1. \tag{14.24}$$

Moreover, for each $a = (u,v) \in B$:

$$\begin{aligned} &p(v) \leq p(u) - 1 \text{ if } f(a') = 0,\\ &p(v) \geq p(u) - 1 \text{ if } f(a') = 1. \end{aligned} \tag{14.25}$$

We may assume that $p(t) = 0$. By (14.24), $p(s) \geq 0$. For each $i = 1, \ldots, p(s)$, let $U_i := \{v \in V \mid p(v) \geq i\}$. Then for each i, $\delta^{\text{out}}(U_i)$ is a directed cut, since $s \in U_i$, $t \notin U_i$, and no arc in A enters U_i: if $(u,v) \in A$ with $v \in U_i$, then $p(v) \geq i$, and hence by (14.24), $p(u) \geq i$, that is $u \in U_i$.

Let $\mathcal{C}$ be the collection of these directed cuts and let $\Gamma := \bigcup \mathcal{C}$. We can decompose f as a sum of incidence vectors of k directed paths in $\widetilde{D}$. Identifying any a' with a (for $a \in B$) this gives a collection $\mathcal{P}$ of $s-t$ paths. Let $\Pi := \bigcup_{P \in \mathcal{P}} AP$.

Then $B \setminus \Pi \subseteq \Gamma$. For let $a = (u,v) \in B \setminus \Pi$. Then $f(a') = 0$, and hence by (14.25), $p(v) \leq p(u) - 1$. Hence $a \in \delta^{\text{out}}(U_i)$ for $i = p(u)$. So $a \in \Gamma$.

Moreover,

$$\begin{aligned} k \cdot |\mathcal{C}| &= k(p(s) - p(t))\\ &= \sum_{a=(u,v)\in A} (p(u) - p(v)) f(a) + \sum_{a=(u,v)\in B} (p(u) - p(v)) f(a')\\ &= \sum_{a \in B} (p(u) - p(v)) f(a') = |B \cap \Gamma \cap \Pi|. \end{aligned} \tag{14.26}$$

So $|B \setminus \Gamma| + k|\mathcal{C}| = |B \setminus \Gamma| + |B \cap \Gamma \cap \Pi| = |B \cap \Pi \setminus \Gamma| + |B \cap \Gamma \cap \Pi| = |B \cap \Pi|$. ∎

This implies for partially ordered sets:

Corollary 14.10a. *Let $(S, \leq)$ be a partially ordered set and let $k \in \mathbb{Z}_+$. Then the maximum size of the union of k chains is equal to the minimum value of*

$$\sum_{A \in \mathcal{A}} \min\{k, |A|\}, \tag{14.27}$$

where $\mathcal{A}$ ranges over partitions of S into antichains.

Proof. Similar to the proof of Corollary 14.8a. ▮

Like the result in the previous section, also this theorem can be stated in a different form. For any partially ordered set $(S, \leq)$, $Y \subseteq S$ and $k \in \mathbb{Z}_+$, let

$$c_k(Y) := \max\{|Z| \mid Z \subseteq Y \text{ is the union of } k \text{ chains}\}. \tag{14.28}$$

Then:

Corollary 14.10b. $c_k(S) = \min_{Y \subseteq S}(|S \setminus Y| + k \cdot c_1(Y))$.

Proof. Similar to the proof of Corollary 14.8b. ▮

Note that the proof method gives a polynomial-time algorithm to find a maximum union of k paths or chains. A weighted version was given by Edmonds and Giles [1977]. For an extension, see Hoffman [1983].

14.7a. Common saturating collections of antichains

Similar results to those in Section 14.6a were obtained for antichain partitions by Greene [1976]. Consider the proof of Theorem 14.10. By Theorem 12.5, there exist minimum-cost flows f and f' of values k and $k+1$ respectively and a function $p : V \to \mathbb{Z}$ that is both a potential for f and for f'.

This implies the following result of Greene [1976] on the minimum in Theorem 14.10:

Theorem 14.11. *Let $D = (V, A)$ be an acyclic digraph, let $B \subseteq A$, and let $h \in \mathbb{Z}_+$. Then there is a collection $\mathcal{C}$ of directed cuts attaining*

$$\min_{\mathcal{C}}(|B \setminus \bigcup \mathcal{C}| + k \cdot |\mathcal{C}|) \tag{14.29}$$

both for $k = h$ and for $k = h + 1$.

Proof. Directly from the foregoing observation. ▮

For partially ordered sets it gives (using definition (14.28)):

Corollary 14.11a. *Let $(S, \leq)$ be a partially ordered set and let $k \in \mathbb{Z}_+$. Then there exists an antichain partition $\mathcal{A}$ of S such that*

$$c_k(S) = \sum_{A \in \mathcal{A}} \min\{k, |A|\} \text{ and } c_{k+1}(S) = \sum_{A \in \mathcal{A}} \min\{k+1, |A|\}. \tag{14.30}$$

Proof. Directly from Theorem 14.11. ▮

More on this can be found in Perfect [1984]. For more on chain and antichain partitions, see Frank [1980a].

14.7b. Conjugacy of partitions

Greene [1976] showed that the numbers studied above give so-called 'conjugate' partitions, implying that in fact the results of Section 14.6 and those of 14.7 can be derived from each other.

Fix a partially ordered set $(S, \leq)$. For each $k = 0, 1, 2, \ldots$, let a_k be the maximum size of the union of k antichains in S and let c_k be the maximum size of the union of k chains in S.

Then Corollary 14.8a is equivalent to:

$$a_k = |S| + \min_{p \geq 0}(kp - c_p). \tag{14.31}$$

Similarly, Corollary 14.10a is equivalent to:

$$c_k = |S| + \min_{p \geq 0}(kp - a_p). \tag{14.32}$$

Define for each $k = 1, 2, \ldots$:

$$\alpha_k := a_k - a_{k-1} \text{ and } \gamma_k := c_k - c_{k-1}. \tag{14.33}$$

Trivially, each α_k and γ_k is nonnegative, and both $\alpha_1, \alpha_2, \ldots$ and $\gamma_1, \gamma_2, \ldots$ are partitions of the number $|S|$. In fact:

Theorem 14.12. *$\alpha_1 \geq \alpha_2 \geq \cdots$ and $\gamma_1 \geq \gamma_2 \geq \cdots$.*

Proof. For each $k \geq 1$, one has $\alpha_k \geq \alpha_{k+1}$; equivalently

$$a_{k+1} + a_{k-1} \leq 2a_k. \tag{14.34}$$

Indeed, by Corollary 14.8a there is a collection $\mathcal{C}$ of chains satisfying

$$a_k = \sum_{C \in \mathcal{C}} \min\{k, |C|\}. \tag{14.35}$$

Then

$$\begin{aligned} 2a_k = \sum_{C \in \mathcal{C}} 2\min\{k, |C|\} &\geq \sum_{C \in \mathcal{C}} (\min\{k-1, |C|\} + \min\{k+1, |C|\}) \\ &\geq a_{k-1} + a_{k+1}. \end{aligned} \tag{14.36}$$

The second part of the theorem is shown similarly (with Corollary 14.10a). ∎

In fact, the partitions $(\alpha_1, \alpha_2, \ldots)$ and $(\gamma_1, \gamma_2, \ldots)$ of $|S|$ are *conjugate*. To show this, we mention some of the theory of partitions of numbers.

Let $\nu_1 \geq \nu_2 \geq \cdots$ be integers forming a partition of the number n; that is, $\nu_1 + \nu_2 + \cdots = n$. (So $\nu_k = 0$ for almost all k.) The *conjugate* partition (ν_p^*) of (ν_k) is defined by:

$$\nu_p^* := \max\{k \mid \nu_k \geq p\} \tag{14.37}$$

for $p = 1, 2, \ldots$. Then it is easy to see that $\nu_1^* \geq \nu_2^* \geq \cdots$, and that

$$\text{for all } p, k \geq 1\text{: } p \leq \nu_k \iff k \leq \nu_p^*. \tag{14.38}$$

The conjugate partition can be interpreted in terms of the 'Young diagram'. The *Young diagram* F of (ν_k) is the collection of pairs (x, y) of natural numbers $x, y \geq 1$ satisfying $y \leq \nu_x$. So the Young diagram uniquely determines the sequence (ν_k). The number of pairs in the Young diagram is equal to n. Now the Young diagram F^* of the conjugate partition (ν_p^*) satisfies

(14.39) $F^* = \{(y,x) \mid (x,y) \in F\}$.

This follows directly from (14.38). It implies that the conjugate partition (ν_p^*) is again a partition of n, and that the conjugate of (ν_p^*) is (ν_k).

The following interprets conjugacy of partitions in terms of their partial sums. Let $\nu_1 \geq \nu_2 \geq \cdots$ and $\nu_1' \geq \nu_2' \geq \cdots$ be partitions of n. For each $k = 0, 1, \ldots$, let

(14.40) $n_k := \nu_1 + \cdots + \nu_k$ and $n_k' := \nu_1' + \cdots + \nu_k'$.

Then:

Lemma 14.13α. *(ν_k) and (ν_k') are conjugate partitions if and only if*

(14.41) $n_p' = n + \min_{k \geq 0}(pk - n_k)$

for each $p = 0, 1, 2, \ldots$.

Proof. First note that, for each $p = 1, 2, \ldots$,

(14.42) $\min_{k \geq 0}(pk - n_k)$ is attained by $k = \nu_p^*$.

Indeed, choose $k \geq 0$ attaining $\min_{k \geq 0}(pk - n_k)$, with k as large as possible. Then $(k+1)p - n_{k+1} > pk - n_k$, and hence $\nu_{k+1} < p$. Moreover, if $k \geq 1$, then $(k-1)p - n_{k-1} \geq pk - n_k$, and hence $\nu_k \geq p$, implying $\nu_p^* = k$. If $k = 0$, then $\nu_1 < p$, again implying $\nu_p^* = 0 = k$. This shows (14.42).

Moreover,

(14.43) $$\sum_{q=1}^{p} \nu_q^* = n + \min_{k \geq 0}(pk - n_k),$$

since

(14.44) $$\begin{aligned} \sum_{q=1}^{p} \nu_q^* &= \sum_{q=1}^{p} \max\{k \mid \nu_k \geq q\} = \sum_{q=1}^{p} \sum_{\substack{k=1 \\ \nu_k \geq q}}^{\infty} 1 = \sum_{k=1}^{\infty} \min\{\nu_k, p\} \\ &= \sum_{k=1}^{\infty} \nu_k - \sum_{k=1}^{\nu_p^*} (\nu_k - p) = n + p\nu_p^* - \sum_{k=1}^{\nu_p^*} \nu_k = n + p\nu_p^* - n_{\nu_p^*} \\ &= n + \min_{k \geq 0}(pk - n_k), \end{aligned}$$

by (14.42).

By (14.43), condition (14.41) is equivalent to

(14.45) $$n_p' = \sum_{q=1}^{p} \nu_q^* \text{ for } p = 0, 1, \ldots.$$

Hence it is equivalent to: $\nu_q' = \nu_q^*$ for each $q = 1, 2, \ldots$; that is to: (ν_k) and (ν_k') are conjugate. ∎

This yields the conjugacy of the α_k and γ_p:

Theorem 14.13. *(α_k) and (γ_p) are conjugate partitions of $|S|$.*

Proof. Directly from Lemma 14.13α and Corollary 14.10b. ∎

Lemma 14.13α gives the equivalence of the Corollaries 14.10b and 14.8b. For other proofs of the conjugacy of (α_k) and (γ_p), see Fomin [1978] and Frank [1980a].

14.8. Further results and notes

14.8a. The Gallai-Milgram theorem

Gallai and Milgram [1960] showed the following generalization of Dilworth's decomposition theorem. It applies to any directed graph, but generally is not a min-max relation.

Theorem 14.14 (Gallai-Milgram theorem)**.** *Let $D = (V, A)$ be a digraph and let $\alpha(D)$ be the maximum number of vertices that are pairwise nonadjacent in the underlying undirected graph. Then V can be partitioned into $\alpha(D)$ directed paths.*

Proof. For any partition Π of V into directed paths, let C_Π be the set of end vertices of the paths in Π. For any subset U of V, let $\alpha(U)$ be the maximum number of pairwise nonadjacent vertices in U. We show by induction on $|V|$ that

(14.46) for each partition Π of V into directed paths there is a partition Π' into directed paths with $C_{\Pi'} \subseteq C_\Pi$ and $|C_{\Pi'}| \leq \alpha(V)$.

This implies the theorem.

Let Π be a partition of V into directed paths. To prove (14.46), we may assume that C_Π is inclusionwise minimal among all such partitions.

If C_Π is a stable set, then (14.46) is trivial. If C_Π is not a stable set, take $u, v \in C_\Pi$ with $(v, u) \in A$. By the minimality of C_Π, the path P_u in Π ending at u consists of more than u alone, since otherwise we could extend the path ending at v by u. So P_u has a one but last vertex, w say.

Let $\widetilde{\Pi}$ be obtained from Π by deleting u from P_u. So $\widetilde{\Pi}$ is a partition of $V \setminus \{u\}$ into directed paths. By induction, there is a partition $\widetilde{\Pi}'$ of $V \setminus \{u\}$ into directed paths such that $C_{\widetilde{\Pi}'} \subseteq C_{\widetilde{\Pi}}$ and such that $|C_{\widetilde{\Pi}'}| \leq \alpha(V \setminus \{u\})$.

If one of the paths in $\widetilde{\Pi}'$ ends at w, we can extend it with u, and obtain a partition Π' of V as required. If none of the paths in $\widetilde{\Pi}'$ end at w, but one of the paths ends at v, we can extend it with u, again obtaining a partition Π' as required. If none of the paths in $\widetilde{\Pi}'$ end at v or w, then augmenting $\widetilde{\Pi}'$ by a path consisting of u alone, gives a partition Π' of V with $C_{\Pi'} \subset C_\Pi$, contradicting the minimality of C_Π. ∎

Theorem 14.14 gives no min-max relation, as is shown by a directed circuit of length $2k$: the vertices can be covered by one directed path, while there exist k pairwise nonadjacent vertices.

A consequence of Theorem 14.14 is Dilworth's decomposition theorem: for any partially ordered set $(S, \leq)$ take $V := S$ and $A := \{(x, y) \mid x < y\}$. Another consequence is the graph-theoretical result of Rédei [1934] that each tournament has a Hamiltonian path:

Corollary 14.14a (Rédei's theorem)**.** *Each tournament has a Hamiltonian path.*

Proof. This is the special case where $\alpha(D) = 1$ in the Gallai-Milgram theorem. ∎

Berge [1982b] posed the following conjecture generalizing the Gallai-Milgram theorem. Let $D = (V, A)$ be a digraph and let $k \in \mathbb{Z}_+$. Then for each path collection $\mathcal{P}$ partitioning V and minimizing

(14.47) $$\sum_{P\in\mathcal{P}} \min\{|VP|, k\},$$

there exist disjoint stable sets $C_1, \ldots, C_k$ in D such that each $P \in \mathcal{P}$ intersects $\min\{|VP|, k\}$ of them. This was proved by Saks [1986] for acyclic graphs. For extensions and related results, see Linial [1978,1981], Saks [1986], and Thomassé [2001].

14.8b. Partially ordered sets and distributive lattices

There is a strong relation between the class of partially ordered sets and the class of partially ordered sets of a special type, the *distributive lattices*. It formed the original motivation for Dilworth to study minimum chain partitions of partially ordered sets.

Let $(S, \leq)$ be a partially ordered set and let $a, b \in S$. Then an element $c \in S$ is called the *meet* of a and b if for each $s \in S$:

(14.48) $s \leq c$ if and only if $s \leq a$ and $s \leq b$.

Note that if the meet of a and b exists, it is unique. Similarly, c is called the *join* of a and b if for each $s \in S$:

(14.49) $s \geq c$ if and only if $s \geq a$ and $s \geq b$.

Again, if the join of a and b exists, it is unique.

S is called a *lattice* if each pair s, t of elements of S has a meet and a join; they are denoted by $s \wedge t$ and $s \vee t$, respectively. The lattice is called *distributive* if

(14.50) $s \wedge (t \vee u) = (s \wedge t) \vee (s \wedge u)$ and $s \vee (t \wedge u) = (s \vee t) \wedge (s \vee u)$

for all $s, t, u \in S$. (In fact it suffices to require only one of the two equalities.)

Each partially ordered set $(S, \leq)$ gives a distributive lattice in the following way. Call a subset $I \subseteq S$ a *lower ideal*, or just an *ideal*, if $t \in I$ and $s \leq t$ implies that $s \in I$. Let $\mathcal{I}_S$ be the collection of ideals in S. Then $(\mathcal{I}_S, \subseteq)$ is a distributive lattice. This follows directly from the fact that for $I, J \in \mathcal{I}_S$ one has $I \wedge J = I \cap J$ and $I \vee J = I \cup J$, and hence (14.50) is elementary set theory. Thus

(14.51) $(S, \leq) \to (\mathcal{I}_S, \subseteq)$

associates a distributive lattice with any partially ordered set.

In fact, each finite distributive lattice can be obtained in this way; that is, we can reverse (14.51). For any lattice $(L, \leq)$, call an element $u \in L$ *join-irreducible* if there exist no $s, t \in L$ with $s \neq u$, $t \neq u$, and $u = s \vee t$. Let J_L be the set of join-irreducible elements in L. Trivially,

(14.52) $(L, \leq) \to (J_L, \leq)$

associates a partially ordered set with any distributive lattice.

Theorem 14.15. *Functions* (14.51) *and* (14.52) *are inverse to each other.*

Proof. First, let $(S \leq)$ be a partially ordered set. For each $s \in S$, let $I_s := \{t \in S \mid t \leq s\}$. Then an element I of $\mathcal{I}_S$ is join-irreducible if and only if there exists an $s \in S$ with $I = I_s$. Moreover, $s \leq t$ if and only if $I_s \subseteq I_t$. Thus we have an isomorphism of $(S, \leq)$ and $(J_{\mathcal{I}_S}, \subseteq)$.

Conversely, let $(L, \leq)$ be a distributive lattice. For each $s \in L$, let $J_s := \{t \in J_L \mid t \leq s\}$. This gives a one-to-one relation between elements of L and ideals in J_L. Indeed, if I is an ideal in J_L, let $s := \bigvee I$. Then $I = J_s$. Clearly, $I \subseteq J_s$, since $t \leq s$ for each $t \in I$. Conversely, let $u \in J_s$. Then, as L is distributive,

$$u = u \wedge s = u \wedge \bigvee_{t \in I} t = \bigvee_{t \in I} (u \wedge t). \tag{14.53}$$

Since u is join-irreducible, $u = u \wedge t$ for some $t \in I$. Hence $u \leq t$, and therefore $u \in I$.

Moreover, for any $s, t \in L$ one has: $s \leq t$ if and only if $J_s \subseteq J_t$. So we have an isomorphism of $(L, \leq)$ and $(\mathcal{I}_{J_L}, \subseteq)$. ∎

There is moreover a one-to-one relation between ideals I in a partially ordered set $(S, \leq)$ and antichains A in S, given by:

$$A = I^{\max} \text{ and } I = A^{\downarrow}. \tag{14.54}$$

Here, for any $Y \subseteq S$, $Y^{\max}$ denotes the set of maximal elements of Y and

$$Y^{\downarrow} := \{s \in S \mid \exists t \in Y : s \leq t\}. \tag{14.55}$$

For each d, the set $\mathbb{Z}^d$ is a distributive lattice, under the usual order: $x \leq y$ if and only if $x_i \leq y_i$ for each $i = 1, \ldots, d$. Any finite distributive lattice $(L, \leq)$ is a sublattice of $\mathbb{Z}^d$ for some d (as will follow from the next theorem). That is, there is an injection $\phi : L \to \mathbb{Z}^d$ such that $\phi(s \wedge t) = \phi(s) \wedge \phi(t)$ and $\phi(s \vee t) = \phi(s) \vee \phi(t)$ for all $s, t \in L$.

Let $d(L)$ be the minimum number d for which L is (isomorphic to) a sublattice $\mathbb{Z}^d$. As Dilworth [1950] showed, the number $d(L)$ can be characterized with Dilworth's decomposition theorem.

To this end, an element s of a partially ordered set $(S, \leq)$ is said to *cover* $t \in S$ if $s > t$ and there is no $u \in S$ with $s > u > t$. For any $s \in S$, let cover(s) be the number of elements covered by s.

Then the following result of Dilworth [1950] can be derived from Dilworth's decomposition theorem:

Theorem 14.16. *Let L be a finite distributive lattice. Then*

$$d(L) = \max_{s \in L} \text{cover}(s). \tag{14.56}$$

Proof. We first show that $d(L) \geq \text{cover}(s)$ for each $s \in L$. Let $d := d(L)$, let $L \subset \mathbb{Z}^d$, and choose $s \in L$. Let Y be the set of elements covered by s. For each $t \in Y$, let $U_t := \{i \mid t_i < s_i\}$. Now for all $t, u \in Y$ with $t \neq u$ one has $t \vee u = s$; hence $U_t \cap U_u = \emptyset$. As $U_t \neq \emptyset$ for all $t \in Y$, we have $|Y| \leq d$.

So $d(L) \geq \max_{s \in L} \text{cover}(s)$. To see equality, by Theorem 14.15 we may assume that $L = \mathcal{I}_S$ (the set of ideals in S) for some partially ordered set $(S, \leq)$, ordered by inclusion. For any ideal I in S, cover(I) is the number of inclusionwise maximal ideals $J \subset I$. Each such ideal J is equal to $I \setminus \{t\}$ for some $t \in I^{\max}$. So cover$(I) = |I^{\max}|$. Hence $\max_{s \in L} \text{cover}(s)$ is equal to the maximum antichain size in S. Let this be d, say.

By Dilworth's decomposition theorem, S can be covered by d chains, $C_1, \ldots, C_d$ say. For each j, the collection $\mathcal{I}_{C_j}$ of ideals in C_j is again a chain (ordered by

inclusion). Now $I \to (I \cap C_1, \ldots, I \cap C_d)$ embeds $\mathcal{I}_S$ into the product $\mathcal{I}_{C_1} \times \cdots \times \mathcal{I}_{C_d}$. Therefore $d(\mathcal{I}_S) \leq d$. ∎

The following was noted by Dilworth [1960]. The relations (14.54) give the following partial order on the collection $\mathcal{A}_S$ of antichains in a partially ordered set $(S, \leq)$:

(14.57) $A \preceq B$ if and only if $A^{\downarrow} \subseteq B^{\downarrow}$

for $A, B \in \mathcal{A}_S$. As $(\mathcal{I}_S, \subseteq)$ is a lattice, also $(\mathcal{A}_S, \preceq)$ is a lattice.

Theorem 14.17. *Let $(S, \leq)$ be a partially ordered set and let A and B be maximum-size antichains. Then also $A \wedge B$ and $A \vee B$ are maximum-size antichains.*

Proof. One has $|A \wedge B| + |A \vee B| \geq |A| + |B|$. Indeed, $A \cup B \subseteq (A \wedge B) \cup (A \vee B)$ and $A \cap B \subseteq (A \wedge B) \cap (A \vee B)$. So

$$|A \wedge B| + |A \vee B| = |(A \wedge B) \cup (A \vee B)| + |(A \wedge B) \cap (A \vee B)| \geq |A \cup B| + |A \cap B| = |A| + |B|. \tag{14.58}$$

As $|A \wedge B| \leq |A| = |B|$ and $|A \vee B| \leq |A| = |B|$, we have $|A \wedge B| = |A \vee B| = |A| = |B|$. ∎

In terms of distributive lattices this gives:

Corollary 14.17a. *Let L be a finite distributive lattice. Then the elements s maximizing* cover(s) *form a sublattice of L.*

Proof. We can represent L as the set $\mathcal{I}_S$ of ideals in a partially ordered set $(S, \leq)$. As one has cover$(I) = |I^{\max}|$ for any $I \in \mathcal{I}_S$, the result follows from Theorem 14.17. ∎

Theorem 14.17 led Dilworth [1960] to derive an alternative proof of Dilworth's decomposition theorem. For another proof and application of Theorem 14.17, see Freese [1974]. Theorem 14.17 was extended to maximum-size unions of k antichains by Greene and Kleitman [1976].

14.8c. Maximal chains

Let $(S, \leq)$ be a partially ordered set. Call a chain *maximal* if it is contained in no other chain. As 'complementary' to Dilworth's decomposition theorem, Greene and Kleitman [1976] observed:

Theorem 14.18. *The maximum number of disjoint maximal chains is equal to the minimum size of a set intersecting all maximal chains.*

Proof. Define a digraph $D = (S, A)$ where A consists of all pairs (s, t) where t covers s. Let U and W be the sets of minimal and maximal elements of S, respectively. So maximal chains correspond to $U - W$ paths, and the theorem follows from Menger's theorem. ∎

14.8d. Further notes

Related results were given by Bogart [1970], Saks [1986], and Behrendt [1988], extensions and algorithms by Cameron and Edmonds [1979], Frank [1980a], Linial [1981], Cameron [1982,1985,1986], and Hoffman [1983], surveys by Greene [1974b], Hoffman [1982], West [1982], and Bogart, Greene, and Kung [1990], and an introduction to and historical account of Dilworth's decomposition theorem by Dilworth [1990].

Fleiner [1997] proved the following conjecture of A. Frank: Let $(S, \leq)$ be a partially ordered set and let M be a perfect matching on S such that for any two $uu', vv' \in M$ with $u \leq v$, one has $v' \leq u'$. (This is called a *symmetric partially ordered set*.) Call a subset C of M a *symmetric chain* if S has a chain intersecting each edge in C. Then the minimum number of symmetric chains covering M is equal to the maximum value of

$$\sum_{i=1}^{k} \lceil \tfrac{1}{2}|X_i| \rceil, \tag{14.59}$$

where $X_1, \ldots, X_k$ are disjoint subsets of M (for some k) with the properties that (i) no X_i contains a symmetric chain of size 3, and (ii) if $i \neq j$, then there exist no $x \in e \in X_i$ and $y \in f \in X_j$ with $x \leq y$.

Chapter 15

Connectivity and Gomory-Hu trees

Since a minimum $s-t$ cut can be found in polynomial time, also the connectivity of a graph can be determined in polynomial time, just by checking all pairs s, t. However, there are more economical methods, which we discuss in this chapter.
A finer description of the edge-connectivity of a graph G is given by the function $r_G(s,t)$, defined as the minimum size of a cut separating s and t. A concise description of the corresponding minimum-size cuts is given by the Gomory-Hu tree — see Section 15.4.

15.1. Vertex-, edge-, and arc-connectivity

For any undirected graph $G = (V, E)$, the *vertex-connectivity*, or just *connectivity*, of G is the minimum size of a subset U of V for which $G - U$ is not connected. A subset U of V attaining the minimum is called a *minimum vertex-cut*. If no such U exists (namely, if G is complete), then the (vertex-)connectivity is ∞.

The connectivity of G is denoted by $\kappa(G)$. If $\kappa(G) \geq k$, G is called *k-vertex-connected*, or just *k-connected*.

The following direct consequence of Menger's theorem was formulated by Whitney [1932a]:

Theorem 15.1. *An undirected graph $G = (V, E)$ is k-connected if and only if there exist k internally vertex-disjoint paths between any two nonadjacent vertices s and t.*

Proof. Directly from the vertex-disjoint version of Menger's theorem (Corollary 9.1a). ∎

Similarly, the *edge-connectivity* of G is the minimum size of a subset C of E for which $G - C$ is not connected. So it is the minimum size of any cut $\delta(U)$ with $\emptyset \neq U \neq V$. A cut C attaining the minimum size is called a

minimum cut. The edge-connectivity of G is denoted by $\lambda(G)$. If $\lambda(G) \geq k$, G is called *k-edge connected*. Then:

Theorem 15.2. *An undirected graph $G = (V, E)$ is k-edge-connected if and only if there exist k edge-disjoint paths between any two vertices s and t.*

Proof. Directly from the edge-disjoint version of Menger's theorem (Corollary 9.1b). ∎

Similar terminology and characterizations apply to digraphs. For any digraph $D = (V, A)$ the *vertex-connectivity*, or just *connectivity*, of D is the minimum size of a subset U of V for which $D - U$ is not strongly connected. A set U attaining the minimum is called a *minimum vertex-cut*. If no such U exists (that is, if each pair (u, v) of vertices is an arc), then the (vertex-)connectivity of D is ∞.

The connectivity of D is denoted by $\kappa(D)$. If $\kappa(D) \geq k$, D is called *k-vertex-connected*, or just *k-connected*. Now one has:

Theorem 15.3. *A digraph $D = (V, A)$ is k-connected if and only if there exist k internally vertex-disjoint $s - t$ paths for any $s, t \in V$ for which there is no arc from s to t.*

Proof. Directly from the vertex-disjoint version of Menger's theorem (Corollary 9.1a). ∎

Finally, the *arc-connectivity* of D is the minimum size of a subset C of A for which $D - C$ is not strongly connected. That is, it is the minimum size of any cut $\delta^{\text{out}}(U)$ with $\emptyset \neq U \neq V$. Any cut attaining the minimum size is called a *minimum cut*. The arc-connectivity of D is denoted by $\lambda(D)$. If $\lambda(D) \geq k$, D is called *k-arc-connected* or *strongly k-connected*. (So D is 1-arc-connected if and only if D is strongly connected.) Then:

Theorem 15.4. *A digraph $D = (V, A)$ is k-arc-connected if and only if there exist k arc-disjoint paths between any two vertices s and t.*

Proof. Directly from the arc-disjoint version of Menger's theorem (Corollary 9.1b). ∎

Shiloach [1979a] observed that Edmonds' disjoint arborescences theorem (to be discussed in Chapter 53) implies a stronger characterization of k-arc-connectivity: a digraph $D = (V, A)$ is k-arc-connected if and only if for all $s_1, t_1, \ldots, s_k, t_k \in V$ there exist arc-disjoint paths $P_1, \ldots, P_k$, where P_i runs from s_i to t_i $(i = 1, \ldots, k)$ — see Corollary 53.1d.

A similar characterization does not hold for the undirected case, as is shown by the 2-edge-connected graph in Figure 15.1.

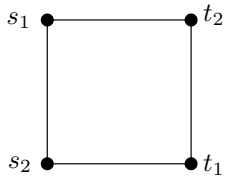


Figure 15.1

15.2. Vertex-connectivity algorithmically

It is clear that the vertex-connectivity of a directed or undirected graph can be determined in polynomial time, just by finding a minimum-size $s-t$ vertex-cut for each pair s,t of vertices. Since by Corollary 9.7a, a minimum-size $s-t$ vertex-cut can be found in $O(n^{1/2}m)$ time, this yields an $O(n^{5/2}m)$ algorithm. In fact, Podderyugin [1973] (for undirected graphs) and Even and Tarjan [1975] observed that one need not consider every pair of vertices:

Theorem 15.5. *A minimum-size vertex-cut in a digraph D can be found in $O(\kappa(D)n^{3/2}m)$ time.*

Proof. Let $D=(V,A)$ be a digraph. We may assume that D is simple. Order V arbitrarily as $v_1,\dots,v_n$. For $i=1,2,\dots$, determine, for each $v\in V$, a minimum v_i-v vertex-cut $C_{v_i,v}$ and a minimum $v-v_i$ vertex-cut C_{v,v_i}. (This takes $O(n^{3/2}m)$ time by Corollary 9.7a.) At any moment, let c be the minimum size of the cuts found so far. We stop if $i>c+1$. Then c is the vertex-connectivity of D.

Indeed let C be a minimum-size vertex-cut. Then for $i:=\kappa(D)+2$, there is a $j<i$ with $v_j\notin C$. Hence there is a vertex v such that C is a v_j-v or a $v-v_j$ vertex-cut. Assume without loss of generality that C is a v_j-v vertex-cut. Then $c\le|C_{v_j,v}|=|C|=\kappa(D)$. ∎

Since $\kappa(D)\le m/n$ if D is not complete, this implies:

Corollary 15.5a. *A minimum-size vertex-cut in a digraph can be found in $O(n^{1/2}m^2)$ time.*

Proof. Immediately from Theorem 15.5, since if D is not complete, $\kappa(D)$ is at most the minimum outdegree of D, and hence $\kappa(D)\le m/n$. ∎

If we want to test the k-connectivity for some fixed k, we may use the following result given by Even [1975]:

Theorem 15.6. *Given a digraph D and an integer k, one can decide in $O((k+\sqrt{n})k\sqrt{n}m)$ time if D is k-connected, and if not, find a minimum cut.*

Proof. Let $V = \{1, \dots, n\}$. Determine

(15.1)
(i) for all $i, j \in \{1, \dots, k\}$ with $(i, j) \notin A$, a minimum-size $i - j$ vertex-cut if it has size less than k;
(ii) for each $i = k+1, \dots, n$ a minimum-size $\{1, \dots, i-1\} - i$ vertex-cut if it has size less than k, and a minimum-size $i - \{1, \dots, i-1\}$ vertex-cut if it has size less than k.

We claim that if we find any vertex-cut, the smallest among them is a minimum-size vertex-cut. If we find no vertex-cuts, then D is k-connected.

To see this, let U be a minimum-size vertex-cut with $|U| < k$. Suppose that each vertex-cut found has size $> |U|$. Then for all distinct $i, j \in \{1, \dots, k\} \setminus U$ there is an $i-j$ path avoiding U. So $\{1, \dots, k\} \setminus U$ is contained in some strong component K of $D - U$. As $D - U$ is not strongly connected, $D - U$ has a vertex not in K. Let i be the smallest index $i \notin K \cup U$. As there exist $|U|+1$ disjoint $\{1, \dots, i-1\} - i$ paths, $D - U$ has a $j - i$ path for some $j < i$; then $j \in K$. Similarly, $D - U$ contains an $i - j'$ path for some $j' \in K$. This contradicts the fact that $i \notin K$ and K is a strong component.

This implies the theorem. Indeed, by Corollaries 9.3a and 9.7a one can find a vertex-cut as in (15.1)(i) or (ii) in time $O(\min\{k, \sqrt{n}\}m)$ time. So in total it takes $O((k^2 + n)\min\{k, \sqrt{n}\}m) = O((k + \sqrt{n})k\sqrt{n}\,m)$ time. ∎

This implies:

Corollary 15.6a. *A minimum-size vertex-cut in a digraph can be found in time* $O(\max\{\frac{m^3}{n\sqrt{n}}, m^2\})$.

Proof. From Theorem 15.6 by taking $k := \lfloor m/n \rfloor$. ∎

Matula [1987] showed that Theorem 15.6 implies the following result of Galil [1980b], where $\kappa(D)$ is the vertex-connectivity of D:

Corollary 15.6b. *A minimum-size vertex-cut in a digraph D can be found in* $O((\kappa(D) + \sqrt{n})\kappa(D)\sqrt{n}\,m)$ *time.*

Proof. For $k = 2, 2^2, 2^3, \dots$ test with the algorithm of Theorem 15.6 if D is k-connected. Stop if D is not k-connected; then the algorithm gives a minimum-size vertex-cut. Let l be such that $k = 2^l$. As $2^l \leq 2\kappa(D)$, this takes time

(15.2)
$$O(\sum_{i=1}^{l}((2^i + \sqrt{n})2^i\sqrt{n}\,m)) = O((4^{l+1} + 2^{l+1}\sqrt{n})\sqrt{n}\,m)$$
$$= O((\kappa(D)^2 + \kappa(D)\sqrt{n})\sqrt{n}\,m).$$ ∎

Since vertex-cuts in an undirected graph G are equal to vertex-cuts in the digraph obtained from G by replacing each edge by two oppositely oriented

arcs, the results above immediately imply similar results for vertex-cuts and vertex-connectivity in undirected graphs.

15.2a. Complexity survey for vertex-connectivity

Complexity survey for vertex-connectivity (in directed graphs, unless stated otherwise; again, $*$ indicates an asymptotically best bound in the table):

	$O(n^2 \cdot \mathrm{VC}(n,m))$	(trivial)
	$O(kn \cdot \mathrm{VC}_k(n,m))$	Kleitman [1969]
	$O(\kappa n \cdot \mathrm{VC}(n,m))$	Podderyugin [1973], Even and Tarjan [1975]
	$O((k^2+n) \cdot \mathrm{VC}_k(n,m))$	Even [1975] (cf. Esfahanian and Hakimi [1984])
	$O((\kappa^2+n) \cdot \mathrm{VC}_\kappa(n,m))$	Galil [1980b] (cf. Matula [1987])
	$O((\kappa+\sqrt{n})\kappa^2 n^{3/2})$	*undirected* Nagamochi and Ibaraki [1992a], Cheriyan and Thurimella [1991]
	$O((\kappa+\sqrt{n})\kappa^2 n^{3/2}\log_n(n^2/m))$	*undirected* Feder and Motwani [1991, 1995]
	$O((\kappa^3+n)m)$	Henzinger, Rao, and Gabow [1996, 2000]
	$O(\kappa nm)$	Henzinger, Rao, and Gabow [1996, 2000]
	$O((\kappa^3+n)\kappa n)$	*undirected* Henzinger, Rao, and Gabow [1996,2000]
	$O(\kappa^2 n^2)$	*undirected* Henzinger, Rao, and Gabow [1996,2000]
$*$	$O((\kappa^{5/2}+n)m)$	Gabow [2000b]
$*$	$O((\kappa+n^{1/4})n^{3/4}m)$	Gabow [2000b]
$*$	$O((\kappa^{5/2}+n)\kappa n)$	*undirected* Gabow [2000b]
$*$	$O((n^{1/4}+\kappa)\kappa n^{7/4})$	*undirected* Gabow [2000b]

Here κ denotes the vertex-connectivity of the graph. Note that $\kappa \leq m/n$. If k is involved, the time bound is for determining $\min\{\kappa, k\}$. By $\mathrm{VC}(n,m)$ we denote the time needed to find the minimum size of an $s-t$ vertex-cut for fixed s,t. Moreover, $\mathrm{VC}_k(n,m)$ denotes the time needed to find the minimum size of an $s-t$ vertex-cut if this size is less than k. We refer to Sections 9.5 and 9.6a for bounds on $\mathrm{VC}(n,m)$ and $\mathrm{VC}_k(n,m)$. Note that $\mathrm{VC}_k(n,m) = O(\min\{k,\sqrt{n}\}m)$.

By the observation of Matula [1987] (cf. Corollary 15.6b above), if $\min\{k,\kappa\}$ can be determined in time $O(k^\alpha f(n,m))$ (for some $\alpha \geq 1$), then κ can be determined in time $O(\kappa^\alpha f(n,m))$.

15.2b. Finding the 2-connected components

In this section we show the result due to Paton [1971], Tarjan [1972], and Dinits, Zajtsev, and Karzanov [1974] (cf. Hopcroft and Tarjan [1973a]) that the 2-vertex-connected components of an undirected graph can be found in linear time. Hence, the 2-connectivity of an undirected graph can be tested in linear time.

Let $G = (V, E)$ be an undirected graph and let $k \in \mathbb{Z}_+$. A *k-connected component* is an inclusionwise maximal subset U of V for which $G[U]$ is k-connected. A *block* is a 2-connected component U with $|U| \geq 2$.

We note

(15.3) if U and W are two different k-connected components, then $|U \cap W| < k$.

Indeed, as $G[U \cup W]$ is not k-connected, there is a subset C of $U \cup W$ with $G[(U \cup W) \setminus C]$ disconnected and $|C| < k$. As $(U \cup W) \setminus C = (U \setminus C) \cup (W \setminus C)$ and as $G[U \setminus C]$ and $G[W \setminus C]$ are connected, it follows that $(U \setminus C) \cap (W \setminus C) = \emptyset$. Hence $C \supseteq U \cap W$, and therefore $|U \cap W| < k$.

(15.3) implies that each edge of G is contained in a unique 2-connected component. So the 2-connected components partition the edge set. One may show:

(15.4) edges e and e' are contained in the same 2-connected component if and only if G has a circuit C containing both e and e'.

Indeed, if C exists, it forms a 2-connected subgraph of G, and hence e and e' are contained in some 2-connected component. Conversely, if $e = uv$ and $e' = u'v'$ are contained in a 2-connected component H, by Menger's theorem H has two vertex-disjoint $\{u, v\} - \{u', v'\}$ paths; with e and e' these paths form a circuit C as required.

Theorem 15.7. *The collection of blocks of a graph $G = (V, E)$ can be identified in linear time.*

Proof. By Corollary 6.6a, we may assume that G is connected. Choose $s \in V$ arbitrarily. Apply depth-first search starting at s. If we orient the final tree to become a rooted tree (V, T) with root s, all further edges of G connect two vertices u, v such that T has a directed $u - v$ path P. For each such edge, make an arc (v, u'), where u' is the second vertex of P. The arcs in T and these new arcs form a directed graph denoted by $D = (V, A)$.

By adapting the depth-first search, we can find A is linear time. Indeed, while scanning s, we keep a directed path Q in T formed by the vertices whose scanning has begun but is not yet finished. If we scan v and meet an edge uv with u on Q, we can find u' and construct the arc (v, u').

Since no arc of D enters s, $\{s\}$ is a strong component. For any strong component K of D one has:

(15.5) the subgraph of T induced by K is a subtree.

Indeed, for any arc $(u, v) \in A \setminus T$ spanned by K, the $v - u$ path in T is contained in K, since it forms a directed circuit with (u, v) and since K is a strong component. This proves (15.5).

(15.5) implies that for each strong component K of D with $K \neq \{s\}$, there is a unique arc of T entering K; let u_K be its tail and define $K' := K \cup \{u_K\}$. We finally show

(15.6) $\{K' \mid K$ strong component of D, $K \neq \{s\}\}$ is equal to the collection of blocks of G.

This proves the theorem (using Theorem 6.6).

Let (t, u, v) be a directed path in T. Then

(15.7) tu and uv are contained in the same block of G if and only if D has a directed $v - u$ path.

To see this, let W be the set of vertices reachable in (V, T) from v. Then:

(15.8) D has a directed $v - u$ path $\iff$ D has an arc leaving W $\iff$ G has an edge leaving W and not containing u $\iff$ G has a $v - t$ path not traversing u $\iff$ tu and uv are contained in a circuit of G $\iff$ tu and uv are contained in the same block of G.

This proves (15.7).

(15.7) implies that for each strong component $K \subseteq V \setminus \{s\}$ of D, the set K' is contained in a block of G. Conversely, let B be a block of G. Then B induces a subtree of T. Otherwise, B contains two vertices u and v such that the undirected $u - v$ path P in T has length at least two and such that no internal vertex of P belongs to B. Let Q be a $u - v$ path in B. Then P and Q form a circuit, hence a 2-connected graph. So P is contained in B, a contradiction.

So B induces a subtree of T. Let u be its root. As $B \setminus \{u\}$ induces a connected subgraph of G, there is a unique arc in T entering $B \setminus \{u\}$. So also $B \setminus \{u\}$ induces a subtree of T. Then (15.7) implies that $B \setminus \{u\}$ is contained in a strong component of D. ∎

Corollary 15.7a. *The 2-connectivity of an undirected graph can be tested in linear time.*

Proof. A graph (V, E) is 2-connected if and only if V is a 2-connected component. So Theorem 15.7 implies the result. ∎

Hopcroft and Tarjan [1973b] gave a linear-time algorithm to test 3-connectivity of an undirected graph; more generally, to decompose an undirected graph into 3-connected components (cf. Miller and Ramachandran [1987,1992]). Kanevsky and Ramachandran [1987,1991] gave an $O(n^2)$ algorithm to test 4-connectivity of an undirected graph.

Finding all 2- and 3-vertex-cuts of an undirected graph has been investigated by Tarjan [1972], Hopcroft and Tarjan [1973b], Kanevsky and Ramachandran [1987, 1991], Miller and Ramachandran [1987,1992], and Kanevsky [1990a]. Kanevsky [1990b] showed that for each fixed k, the number of vertex-cuts of size k in a k-vertex-connected graph is $O(n^2)$ (cf. Kanevsky [1993]). Related results can be found in Gusfield and Naor [1990], Cohen, Di Battista, Kanevsky, and Tamassia [1993], Gabow [1993b,1995c], and Cheriyan and Thurimella [1996b,1999].

15.3. Arc- and edge-connectivity algorithmically

Denote by $\mathrm{EC}(n, m)$ the time needed to find a minimum-size $s-t$ cut for any given pair of vertices s, t. Even and Tarjan [1975] (and Podderyugin [1973]

for undirected graphs) observed that one need not check all pairs of vertices to find a minimum cut:

Theorem 15.8. *A minimum-size cut in a digraph can be found in* $O(n \cdot \mathrm{EC}(n,m))$ *time.*

Proof. Choose $s \in V$. For each $t \neq s$, determine a minimum $s-t$ cut $C_{s,t}$ and a minimum $t-s$ cut $C_{t,s}$. The smallest among all these cuts is a minimum cut. ∎

Hence we have for the arc-connectivity:

Corollary 15.8a. *The arc-connectivity of a digraph can be determined in* $O(n \cdot \mathrm{EC}(n,m))$ *time.*

Proof. Directly from Theorem 15.8. ∎

As $\mathrm{EC}(n,m) = O(m^{3/2})$ by Corollary 9.6a, it follows that the arc-connectivity can be determined in time $O(nm^{3/2})$. Actually, also in time $O(m^2)$, since we need to apply the disjoint paths algorithm only until we have at most $k := \lfloor m/n \rfloor$ arc-disjoint paths, as the arc-connectivity is at most m/n (there is a $v \in V$ with $|\delta^{\mathrm{out}}(v)| \leq \lfloor m/n \rfloor$).

Moreover, again by Corollary 9.6a, $\mathrm{EC}(n,m) = O(n^{2/3}m)$ for simple digraphs, and hence the arc-connectivity of a simple directed graph can be determined in time $O(n^{5/3}m)$ (cf. Esfahanian and Hakimi [1984]).

Schnorr [1978b,1979] showed that in fact:

Theorem 15.9. *Given a digraph* D *and an integer* k*, one can decide in* $O(knm)$ *time if* D *is* k*-arc-connected, and if not, find a minimum cut.*

Proof. In Theorem 15.8 one needs to check only if there exist k arc-disjoint $s-t$ paths, and if not find a minimum-size $s-t$ cut. This can be done in time $O(km)$, as we saw in Corollary 9.3a. ∎

With a method of Matula [1987] this implies, where $\lambda(D)$ is the arc-connectivity of D:

Corollary 15.9a. *A minimum-size cut in a digraph* D *can be found in time* $O(\lambda(D)nm)$.

Proof. For $k = 2, 2^2, 2^3, \ldots$ test if D is k-arc-connected, until we find that D is not k-arc-connected, and have a minimum-size cut. With the method of Theorem 15.9 this takes time $O((2+2^2+2^3+\cdots+2^l)nm)$, with $2^l \leq 2\lambda(D)$. So $2+2^2+2^3+\cdots+2^l \leq 2^{l+1} \leq 4\lambda(D)$, and the result follows. ∎

For undirected graphs, Nagamochi and Ibaraki [1992b] showed that the edge-connectivity of an undirected graph can be determined in time $O(nm)$ (for simple graphs this bound is due to Podderyugin [1973]). We follow the shortened algorithm described by Frank [1994b] and Stoer and Wagner [1994, 1997].

Theorem 15.10. *Given an undirected graph G, a minimum cut in G can be found in time $O(nm)$.*

Proof. Let $G = (V, E)$ be a graph. For $U \subseteq V$ and $v \in V \setminus U$, let $d(U, v)$ denote the number of edges connecting U and v. Let $r(u, v)$ denote the minimum capacity of a $u - v$ cut.

Call an ordering $v_1, \ldots, v_n$ of the vertices of G a *legal order* for G if $d(\{v_1, \ldots, v_{i-1}\}, v_i) \geq d(\{v_1, \ldots, v_{i-1}\}, v_j)$ for all i, j with $1 \leq i < j \leq n$. Then:

(15.9) If $v_1, \ldots, v_n$ is a legal order for $G = (V, E)$, then $r(v_{n-1}, v_n) = d(v_n)$.

To see this, let C be any $v_{n-1} - v_n$ cut. Define $u_0 := v_1$. For $i = 1, \ldots, n-1$, define $u_i := v_j$, where j is the smallest index such that $j > i$ and C is a $v_i - v_j$ cut. Note that for each $i = 1, \ldots, n-1$ one has

$$d(\{v_1, \ldots, v_{i-1}\}, u_i) \leq d(\{v_1, \ldots, v_{i-1}\}, u_{i-1}), \tag{15.10}$$

since if $u_{i-1} \neq u_i$, then $u_{i-1} = v_i$, in which case (15.10) follows from the legality of the order.

Then we have

$$\begin{aligned} d(C) &\geq \sum_{i=1}^{n-1} d(v_i, u_i) \\ &= \sum_{i=1}^{n-1} (d(\{v_1, \ldots, v_i\}, u_i) - d(\{v_1, \ldots, v_{i-1}\}, u_i)) \\ &\geq \sum_{i=1}^{n-1} (d(\{v_1, \ldots, v_i\}, u_i) - d(\{v_1, \ldots, v_{i-1}\}, u_{i-1})) \\ &= d(\{v_1, \ldots, v_{n-1}\}, v_n) = d(v_n), \end{aligned} \tag{15.11}$$

showing (15.9).

Next one has that a legal order for a given graph G can be found in time $O(m)$. Indeed, one can find $v_1, v_2, v_3, \ldots$ successively: if $v_1, \ldots, v_{i-1}$ have been found, we need to find a $v \in V \setminus \{v_1, \ldots, v_{i-1}\}$ maximizing $d(\{v_1, \ldots, v_{i-1}\}, v)$. With a 'bucket' data structure this can be done in $O(m)$ time.[23]

[23] Suppose that we have a set V and a function $\phi : V \to \mathbb{Z}$. We can select and delete a v maximizing $\phi(v)$ in $O(1)$ time and to reset $\phi(v)$ from k to k' in $O(|k' - k|)$ time.

Concluding, in any given graph $G = (V, E)$ with $|V| \geq 2$, we can find two vertices s and t with $r(s,t) = d(s)$ in time $O(m)$. Identify s and t, and find recursively a minimum cut C in the new graph. Then $\delta(s)$ is a minimum cut separating s and t, and C is a minimum cut not separating s and t. Hence, the smallest of the two is a minimum cut. ∎

(Another correctness proof was given by Fujishige [1998].)

Theorem 15.10 extends to capacitated graphs (Nagamochi and Ibaraki [1992b]):

Theorem 15.11. *Given an undirected graph $G = (V, E)$ and a capacity function $c : E \to \mathbb{Q}_+$, a minimum-capacity cut can be found in time $O(n(m + n \log n))$.*

Proof. This can be shown in the same way as Theorem 15.10, using Fibonacci heaps for finding a legal order. ∎

15.3a. Complexity survey for arc- and edge-connectivity

Survey for arc-complexity (in uncapacitated directed graphs, unless stated otherwise):

	$O(n \cdot \mathrm{EC}(n,m))$	(trivial)
	$O(nm)$	*simple undirected* Podderyugin [1973]
	$O(n \cdot \mathrm{EC}_l(n,m))$	Schnorr [1978b,1979]
	$O(\lambda^3 n^2 + \lambda m)$	Timofeev [1982]
	$O(\lambda n^2)$	*simple undirected* Karzanov and Timofeev [1986], Matula [1987]
	$O(n \cdot \mathrm{EC}_\lambda(n,m))$	Matula [1987]
	$O(nm)$	*simple* Mansour and Schieber [1989]
	$O(\lambda^2 n^2)$	*simple* Mansour and Schieber [1989]
*	$O(n \frac{\log \delta}{\delta} \cdot \mathrm{EC}(n,m))$	*simple* N. Alon, 1988 (cf. Mansour and Schieber [1989])
	$O(nm \log_n(n^2/m))$	*undirected* Feder and Motwani [1991,1995]

≫

To this end, partition V into classes V_j, where $V_j := \{v \in V \mid \phi(v) = j\}$. Each nonempty V_j is ordered as a doubly linked list, and the nonempty V_j among them are ordered as a doubly linked list L, in increasing order of j. Then in $O(1)$ time we can choose the largest j for which V_j is nonempty, choose $v \in V_j$, delete v from V_j, and possibly delete V_j from L (if V_j has become empty). Resetting $\phi(v)$ from k to k' can be done by finding or creating $V_{k'}$ in L, which takes $O(|k' - k|)$ time.

Having this data structure, throughout let $U := V \setminus \{v_1, \ldots, v_i\}$, and for each $v \in U$ let $\phi(v) := d(\{v_1, \ldots, v_i\}, v)$. If v_{i+1} has been found, we must delete v_{i+1} from U, and reset, for each neighbour v of v_{i+1} in U, $\phi(v)$ to $d(\{v_1, \ldots, v_{i+1}\}, v)$. This gives an $O(m)$-time algorithm.

continued

	$O(nm)$	*undirected* Nagamochi and Ibaraki [1992b]
	$O(m+\lambda^2 n^2)$	*undirected* Nagamochi and Ibaraki [1992b]
	$O(m+\tilde{m}n+n^2\log n)$	*undirected* Nagamochi and Ibaraki [1992b]
*	$O(n(m+n\log n))$	*capacitated undirected* Nagamochi and Ibaraki [1992b]
*	$O(nm\log(n^2/m))$	*capacitated* Hao and Orlin [1992,1994]
*	$O(\lambda m\log(n^2/m))$	Gabow [1991a,1995a]
*	$O(m+\lambda^2 n\log(n/\lambda))$	*undirected* Gabow [1991a,1995a]

Here λ denotes the arc- or edge-connectivity of the graph, $\tilde{m}$ the number of parallel classes of edges, and δ the minimum (out-)degree. Note that $\lambda \leq \delta \leq 2m/n$. If l is involved, the time bound is for determining $\min\{\lambda, l\}$.

$\mathrm{EC}(n,m)$ denotes the time needed to find the minimum size of an $s-t$ cut, for fixed s,t. Moreover, $\mathrm{EC}_l(n,m)$ denotes the time needed to find the minimum size of an $s-t$ cut (for fixed s,t) if this size is less than l. We refer to Sections 9.4 and 9.6a for bounds on $\mathrm{EC}(n,m)$ and $\mathrm{EC}_l(n,m)$.

By the observation of Matula [1987] (cf. Corollary 15.9a above), if $\min\{\lambda, l\}$ can be determined in time $O(l^\alpha f(n,m))$ (for some $\alpha \geq 1$), then λ can be determined in time $O(\lambda^\alpha f(n,m))$.

Matula [1993] gave a linear-time $2+\varepsilon$-approximative algorithm determining the edge-connectivity. (Related work was done by Henzinger [1997].)

Galil and Italiano [1991] described a linear-time method to make from a graph G a graph $\phi_k(G)$, with $m+(k-2)n$ vertices and $(2k-3)m$ edges such that: G is k-edge-connected $\iff$ $\phi_k(G)$ is k-vertex-connected. This implies, for instance, that 3-edge-connectivity can be tested in linear time (as Hopcroft and Tarjan [1973b] showed that 3-vertex-connectivity can be tested in linear time). Related work was reported by Esfahanian and Hakimi [1984] and Padberg and Rinaldi [1990a].

Karger and Stein [1993,1996] gave a randomized minimum cut algorithm for undirected graphs, with running time $O(n^2\log^3 n)$. Karger [1996,2000] gave an improvement to $O(m\log^3 n)$.

Nagamochi, Ono, and Ibaraki [1994] report on computational experiments with the Nagamochi-Ibaraki algorithm. An experimental study of several minimum cut algorithms was presented by Chekuri, Goldberg, Karger, Levine, and Stein [1997].

15.3b. Finding the 2-edge-connected components

Let $G=(V,E)$ be an undirected graph and let $k \in \mathbb{Z}_+$. Consider the relation $\sim$ on V defined by:

(15.12) $\quad u \sim v \iff G$ has k edge-disjoint $u-v$ paths.

Then $\sim$ is an equivalence relation. This can be seen with Menger's theorem. If $u \sim v$ and $v \sim w$, then $u \sim w$; otherwise, there is a $u-w$ cut C of size less than k. Then C is also a $u-v$ cut or a $v-w$ cut, contradicting the fact that $u \sim v$ and $v \sim w$.

The equivalence classes are called the *k-edge-connected components* of G. So the 1-edge connected components of G coincide with the components of G, and can be found in linear time by Corollary 6.6a. Also for $k = 2$, the k-edge-connected components can be found in linear time (Karzanov [1970]; we follow the proof of Tarjan [1972]):

Theorem 15.12. *Given an undirected graph $G = (V, E)$, its 2-edge-connected components can be found in linear time.*

Proof. We may assume that G is connected, since by Corollary 6.6a, the components of G can be found in linear time.

Choose $s \in V$ arbitrarily, and consider a depth-first search tree T starting at s. Orient each edge in T away from s. For each remaining edge $e = uv$, there is a directed path in T that connects u and v. Let the path run from u to v. Then orient e from v to u. This gives the orientation D of G.

Then any edge not in T belongs to a directed circuit in D. Moreover, any edge in T that is not a cut edge, belongs to a directed circuit in D. Then the 2-edge-connected components of G coincide with the strong components of D. By Theorem 6.6, these components can be found in linear time. ∎

More on finding 2-edge-connected components can be found in Gabow [2000a].

15.4. Gomory-Hu trees

In previous sections of this chapter we have considered the problem of determining a minimum cut in a graph, where the minimum is taken over all pairs s, t. The *all-pairs minimum-size cut problem* asks for a minimum $s - t$ cut for all pairs of vertices s, t. Clearly, this can be solved in time $O(n^2\tau)$, where τ is the time needed for finding a minimum $s - t$ cut for any given s, t.

Gomory and Hu [1961] showed that for *undirected* graphs it can be done faster, and that there is a concise structure, the Gomory-Hu tree, to represent all minimum cuts. Similarly for the capacitated case.

Fix an undirected graph $G = (V, E)$ and a capacity function $c : E \to \mathbb{R}_+$. A *Gomory-Hu tree* (*for* G and c) is a tree $T = (V, F)$ such that for each edge $e = st$ of T, $\delta(U)$ is a minimum-capacity $s - t$ cut of G, where U is any of the two components of $T - e$. (Note that it is not required that T is a subgraph of G.)

Gomory and Hu [1961] showed that for each G, c there indeed exists a Gomory-Hu tree, and that it can be found by $n - 1$ minimum-cut computations.

For distinct $s, t \in V$, define $r(s, t)$ as the minimum capacity of an $s - t$ cut. The following triangle inequality holds:

$$r(u, w) \geq \min\{r(u, v), r(v, w)\} \tag{15.13}$$

for all distinct $u, v, w \in G$. Now a Gomory-Hu tree indeed describes concisely minimum-capacity $s - t$ cuts for all s, t:

Theorem 15.13. *Let $T = (V, F)$ be a Gomory-Hu tree. Consider any $s, t \in V$, the $s-t$ path P in T, an edge $e = uv$ on P with $r(u, v)$ minimum, and any component K of $T-e$. Then $r(s, t) = r(u, v)$ and $\delta(K)$ is a minimum-capacity $s - t$ cut.*

Proof. Inductively, (15.13) gives $r(s, t) \geq r(u, v)$. Moreover, $\delta(K)$ is an $s - t$ cut, and hence $r(s, t) \leq c(\delta(K)) = r(u, v)$. ∎

To show that a Gomory-Hu tree does exist, we first prove:

Lemma 15.14α. *Let $s, t \in V$, let $\delta(U)$ be a minimum-capacity $s - t$ cut in G, and let $u, v \in U$ with $u \neq v$. Then there exists a minimum-capacity $u - v$ cut $\delta(W)$ with $W \subseteq U$.*

Proof. Consider a minimum-capacity $u - v$ cut $\delta(X)$. By symmetry we may assume that $s \in U$ (otherwise interchange s and t), $t \notin U$, $s \in X$ (otherwise replace X by $V \setminus X$), $u \in X$ (otherwise interchange u and v), and $v \notin X$. So one of the diagrams of Figure 15.2 applies.

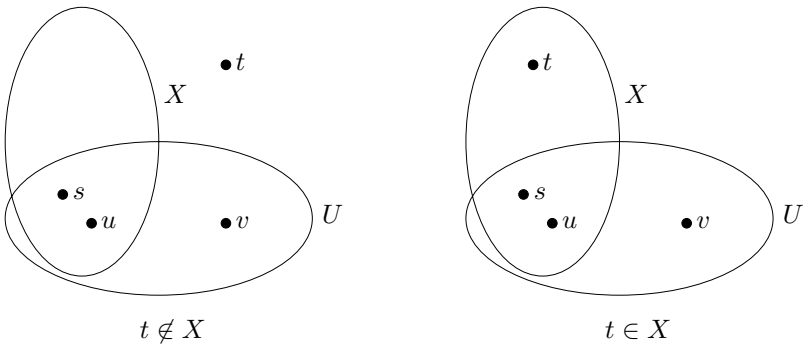


Figure 15.2

In particular, $\delta(U \cap X)$ and $\delta(U \setminus X)$ are $u-v$ cuts. If $t \notin X$, then $\delta(U \cup X)$ is an $s - t$ cut. As

$$c(\delta(U \cap X)) + c(\delta(U \cup X)) \leq c(\delta(U)) + c(\delta(X)) \tag{15.14}$$

and

$$c(\delta(U \cup X)) \geq c(\delta(U)), \tag{15.15}$$

we have $c(\delta(U \cap X)) \leq c(\delta(X))$. So $\delta(U \cap X)$ is a minimum-capacity $u - v$ cut.

If $t \in X$, then $\delta(X \setminus U)$ is an $s - t$ cut. As

$$c(\delta(U \setminus X)) + c(\delta(X \setminus U)) \leq c(\delta(U)) + c(\delta(X)) \tag{15.16}$$

and

(15.17) $c(\delta(X \setminus U)) \geq c(\delta(U))$,

we have $c(\delta(U \setminus X)) \leq c(\delta(X))$. So $\delta(U \setminus X)$ is a minimum-capacity $u - v$ cut. ∎

This lemma is used in proving the existence of Gomory-Hu trees:

Theorem 15.14. *For each graph $G = (V, E)$ and each capacity function $c : E \to \mathbb{R}_+$ there exists a Gomory-Hu tree.*

Proof. Define a *Gomory-Hu tree for* a set $R \subseteq V$ to be a pair of a tree (R, T) and a partition $(C_r \mid r \in R)$ of V such that:

(15.18) (i) $r \in C_r$ for each $r \in R$,
(ii) $\delta(U)$ is a minimum-capacity $s-t$ cut for each edge $e = st \in T$, where $U := \bigcup_{u \in K} C_u$ and K is a component of $T - e$.

We show by induction on $|R|$ that for each nonempty $R \subseteq V$ there exists a Gomory-Hu tree for R. Then for $R = V$ we have a Gomory-Hu tree.

If $|R| = 1$, (15.18) is trivial, so assume $|R| \geq 2$. Let $\delta(W)$ be a minimum-capacity cut separating at least one pair of vertices in R. Contract $V \setminus W$ to one vertex, v' say, giving graph G'. Let $R' := R \cap W$. By induction, G' has a Gomory-Hu tree (R', T'), $(C'_r \mid r \in R')$ for R'.

Similarly, contract W to one vertex, v'' say, giving graph G''. Let $R'' := R \setminus W$. By induction, G'' has a Gomory-Hu tree (R'', T''), $(C''_r \mid r \in R'')$ for R''.

Now let $r' \in R'$ be such that $v' \in C'_{r'}$. Similarly, let $r'' \in R''$ be such that $v'' \in C''_{r''}$. Let $T := T' \cup T'' \cup \{r'r''\}$, Let $C_{r'} := C'_{r'} \setminus \{v'\}$ and let $C_r := C'_r$ for all other $r \in R'$. Similarly, let $C_{r''} := C''_{r''} \setminus \{v''\}$ and let $C_r := C''_r$ for all other $r \in R''$.

Now (R, T) and the C_r form a Gomory-Hu tree for R. Indeed, for any $e \in T$ with $e \neq r'r''$, (15.18) follows from Lemma 15.14α. If $e = r'r''$, then $U = W$ and $\delta(W)$ is a minimum-capacity $r' - r''$ cut (as it is minimum-capacity over all cuts separating at least one pair of vertices in R). ∎

The method can be sharpened to give the following algorithmic result:

Theorem 15.15. *A Gomory-Hu tree can be found by $n - 1$ applications of a minimum-capacity cut algorithm.*

Proof. In the proof of Theorem 15.14, it suffices to take for $\delta(W)$ just a minimum-capacity $s - t$ cut for at least one pair $s, t \in R$. Then $\delta(W)$ is also a minimum-capacity $r' - r''$ cut. For suppose that there exists an $r' - r''$ cut $\delta(X)$ of smaller capacity. We may assume that $s \in W$ and $t \notin W$. As $\delta(W)$ is a minimum-capacity $s - t$ cut, $\delta(X)$ is not an $s - t$ cut. So it should separate

s and r' or t and r''. By symmetry, we may assume that it separates s and r'. Then it also as is a $u-v$ cut for some edge uv on the $s-r'$ path in T'. Let uv determine cut $\delta(U)$. This cut is an $s-t$ cut, and hence $c(\delta(U)) \geq c(\delta(W))$. On the other hand, $c(\delta(U)) \leq c(\delta(X))$, as $\delta(U)$ is a minimum-capacity $u-v$ cut. This contradicts our assumption that $c(\delta(X)) < c(\delta(W))$. ▮

This implies for the running time:

Corollary 15.15a. *A Gomory-Hu tree can be found in time $O(n\tau)$ time, if for any $s,t \in V$ a minimum-capacity $s-t$ cut can be found in time τ.*

Proof. Directly from Theorem 15.15. ▮

Notes. The method gives an $O(m^2)$ method to find a Gomory-Hu tree for the capacity function $c = \mathbf{1}$, since $O(m^2) = O(\sum_v d(v)m)$, and for each new vertex v a minimum cut can be found in time $O(d(v)m)$. Hao and Orlin [1992,1994] gave an $O(n^3)$-time method to find, for given graph $G = (V, E)$ and $s \in V$, all minimum-size $s-t$ cuts for all $t \neq s$ (with push-relabel). Shiloach [1979b] gave an $O(n^2m)$ algorithm to find a maximum number of edge-disjoint paths between all pairs of vertices in an undirected graph. Ahuja, Magnanti, and Orlin [1993] showed that the best directed all-pairs cut algorithm takes $\Omega(n^2)$ max-flow iterations.

For planar graphs, Hartvigsen and Mardon [1994] gave an $(n^2 \log n + m)$ algorithm to find a Gomory-Hu tree (they observed that this bound can be derived also from Frederickson [1987b]). This improves a result of Shiloach [1980a], who gave an $O(n^2(\log n)^2)$-time algorithm to find minimum-size cuts between all pairs of vertices in a planar graph.

Theorem 15.13 implies that a Gomory-Hu tree for a graph $G = (V, E)$ is a maximum-weight spanning tree in the complete graph on V, for weight function $r(u, v)$. However, not every maximum-weight spanning tree is a Gomory-Hu tree (for $G = K_{1,2}$, $c = \mathbf{1}$, only G itself is a Gomory-Hu tree, but all spanning trees on $VK_{1,2}$ have the same weight).

More on Gomory-Hu trees can be found in Elmaghraby [1964], Hu and Shing [1983], Agarwal, Mittal, and Sharma [1984], Granot and Hassin [1986], Hassin [1988], Chen [1990], Gusfield [1990], Hartvigsen and Margot [1995], Talluri [1996], Goldberg and Tsioutsiouliklis [1999,2001], and Hartvigsen [2001b]. Generalizations were given by Cheng and Hu [1990,1991,1992] and Hartvigsen [1995] (to matroids).

15.4a. Minimum-requirement spanning tree

Hu [1974] gave the following additional application of Gomory-Hu trees. Let $G = (V, E)$ be an undirected graph and let $r : E \to \mathbb{R}_+$ be a 'requirement' function (say, the number of telephone calls to be made between the end vertices of e).

We want to find a tree T on V minimizing

$$\sum_{e \in E} r(e)\mathrm{dist}_T(e), \tag{15.19}$$

where $\mathrm{dist}_T(e)$ denotes the distance in T between the end vertices of e.

Now any Gomory-Hu tree T for G and capacity function r indeed minimizes (15.19). To see this, let for any edge f of T, $R_T(f)$ be equal to the requirement (= capacity) of the cut determined by the two components of $T-f$. Then (15.19) is equal to

$$\sum_{f\in T} R_T(f). \tag{15.20}$$

Now T minimizes (15.20), as was shown by Adolphson and Hu [1973]. For let T' be any other spanning tree on V. Then for each $f=st\in T$ and each edge f' on the $s-t$ path in T' one has

$$R_{T'}(f')\geq R_T(f), \tag{15.21}$$

since the components of $T-f$ determine a minimum-capacity $s-t$ cut, and since the components of $T'-f'$ determine an $s-t$ cut. Since T and T' are spanning trees, there exists a one-to-one function $\phi: T\to T'$ such that for each $f=st\in T$, $\phi(f)$ is an edge on the $s-t$ path in T'.

To see this, let u be an end vertex of T. Let $f=uv$ be the edge of T incident with u, and define $\phi(f)$ to be the first edge of the $u-v$ path in T'. Delete f and contract $\phi(f)$. Then induction gives the required function.

So (15.21) implies that (15.20) is not decreased by replacing T by T'. Hence T minimizes (15.20), and therefore also (15.19).

15.5. Further results and notes

15.5a. Ear-decomposition of undirected graphs

In Section 6.5c we characterized the strongly connected digraphs as those digraphs having an ear-decomposition. We now consider the undirected case, and we will see a correspondence between ear-decompositions and 2-(edge-)connected graphs.

Let $G=(V,E)$ be an undirected graph. An *ear* of G is a path or circuit P in G, of length ≥ 1, such that all internal vertices of P have degree 2 in G. The path may consist of a single edge — so any edge of G is an ear. A *proper ear* is an ear that is a path, that is, has two different ends.

If I is the set of internal vertices of an ear P, we say that G arises from $G-I$ by *adding ear*. An *ear-decomposition* of G is a series of graphs $G_0, G_1, \ldots, G_k$, where $G_0=K_1$, $G_k=G$, and G_i arises from G_{i-1} by adding an ear ($i=1,\ldots,k$). If $G_0=K_2$ and G_i arises from G_{i-1} by adding a proper ear, it is a *proper ear-decomposition*.

Graphs with a *proper* ear-decomposition were characterized by Whitney [1932b]:

Theorem 15.16. *A graph $G=(V,E)$ with $|V|\geq 2$ has a proper ear-decomposition if and only if G is 2-vertex-connected.*

Proof. Necessity follows from the facts that K_2 is 2-vertex-connected and that 2-vertex-connectivity is maintained under adding proper ears. To see sufficiency, let G be 2-vertex-connected, and let $G'=(V',E')$ be a subgraph of G that has a

proper ear-decomposition, with $|E'|$ as large as possible. Suppose that $E' \neq E$, and let $e = uv$ be an edge in $E \setminus E'$ incident with V'; say $u \in V'$. By the 2-connectivity of G, there is a path from v to V' avoiding u. Let P be a shortest such path. Then path e, P is a proper ear that can be added to G', contradicting the maximality of $|E'|$. ∎

Similarly, graphs having an ear-decomposition are characterized by being 2-edge-connected (this is implicit in Robbins [1939]):

Theorem 15.17. *A graph $G = (V, E)$ has an ear-decomposition if and only if G is 2-edge-connected.*

Proof. Necessity follows from the facts that K_1 is 2-edge-connected and that 2-edge-connectivity is maintained under adding ears. To see sufficiency, let G be 2-edge-connected, and let $G' = (V', E')$ be a subgraph of G that has an ear-decomposition, with $|E'|$ as large as possible. Suppose that $E' \neq E$, and let $e = uv$ be an edge in $E \setminus E'$ incident with V'; say $u \in V'$. Let C be a circuit in G traversing e. Let C start with $u, e, \ldots$. Let s be the first vertex in C, after e, that belongs to V'. Then subpath $P = u, e \ldots, w$ of C is an ear that can be added to G', contradicting the maximality of $|E'|$. ∎

15.5b. Further notes

Dinits, Karzanov, and Lomonosov [1976] showed that the set of all minimum-capacity cuts of an undirected graph (with positive capacities on the edges) has the "cactus structure", as follows. A *cactus* is a connected graph such that each edge belongs to at most one circuit. Let $G = (V, E)$ be a graph with a capacity function $c : E \to \mathbb{Z}_+$ such that the minimum cut capacity λ is positive. Then there exist a cactus K with $O(|V|)$ vertices and a function $\phi : V \to VK$ such that for each inclusionwise minimal cut $\delta_K(U)$ of K, the set $\delta_G(\phi^{-1}(U))$ is a cut of capacity λ, and such that each minimum-capacity cut in G can be obtained this way. Moreover, K is a tree when λ is odd. It follows that the number of minimum-capacity cuts is at most $\binom{n}{2}$ (and at most $n - 1$ when λ is odd), and that the vertices of G can be ordered as $v_1, \ldots, v_n$ so that each minimum-capacity cut is of the form $\delta(\{v_i, v_{i+1}, \ldots, v_j\})$ for some $i \leq j$. Related results can be found in Picard and Queyranne [1980], Karzanov and Timofeev [1986], Gabow [1991b,1993b,1995c], Gusfield and Naor [1993], Karger and Stein [1993,1996], Nagamochi, Nishimura, and Ibaraki [1994,1997], Benczúr [1995], Henzinger and Williamson [1996], Karger [1996,2000], Fleischer [1998a,1999b], Dinitz and Vainshtein [2000], and Nagamochi, Nakao, and Ibaraki [2000].

Gusfield and Naor [1990,1991] considered the analogue of the Gomory-Hu tree for *vertex*-cuts.

A theorem of Mader [1971] implies that each k-connected graph $G = (V, E)$ contains a k-connected spanning subgraph with $O(k|V|)$ edges — similarly for k-edge-connected. This was extended by Nagamochi and Ibaraki [1992a], showing that for each k, each graph $G = (V, E)$ has a subgraph $G_k = (V, E_k)$ such that $|E_k| = O(k|V|)$ and such that for all $s, t \in V$:

(15.22) (i) $\lambda_{G_k}(s, t) \geq \min\{\lambda_G(s, t), k\}$,

(ii) $\kappa_{G_k}(s,t) \geq \min\{\kappa_G(s,t), k\}$ if $st \notin E$.

Here $\lambda_H(s,t)$ ($\kappa_H(s,t)$, respectively) denotes the maximum number of edge-disjoint (internally vertex-disjoint, respectively) $s-t$ paths in H. They also gave a linear-time algorithm finding G_k. A shorter proof and a generalization was given by Frank, Ibaraki, and Nagamochi [1993].

Frank [1995] showed that the following is implied by the existence of a Gomory-Hu tree. Let $G=(V,E)$ be an undirected graph of minimum degree k. Then there exist two distinct vertices $s,t \in V$ connected by k edge-disjoint paths. This follows by taking for s a vertex of degree 1 in the Gomory-Hu tree, and for t its neighbour in this tree. Then $\delta_E(s)$ is a minimum-size cut separating s and t.

Tamir [1994] observed that the up hull P of the incidence vectors of the nontrivial cuts of an undirected graph $G=(V,E)$ can be described as follows. Let $D=(V,A)$ be the digraph with A being the set of all ordered pairs (u,v) for adjacent $u,v \in V$. Choose $r \in V$ arbitrarily. Then P is equal to the projection to x-space of the polyhedron in the variables $x \in \mathbb{R}^E$ and $y \in \mathbb{R}^A$ determined by:

(15.23)

(i)	$y(a) \geq 0$	for each $a \in A$,
(ii)	$y(B) \geq 1$	for each r-arborescence B,
(iii)	$x(e) = y(u,v) + y(v,u)$	for each edge $e = uv$ of G.

(Here an *r-arborescence* is a subset B of A such that (V,B) is a rooted tree rooted at r.) This can be shown with the help of the results to be discussed in Chapter 53. To see this, consider any $c \in \mathbb{R}^E_+$. Then the minimum value of $c^\mathsf{T}x$ over all x,y satisfying (15.23), is equal to the minimum value of $d^\mathsf{T}y$ over all x,y satisfying (15.23), where $d(u,v) := c(uv)$ for each $(u,v) \in A$. This is equal to the minimum value of $d^\mathsf{T}y$ over all y satisfying (i) and (ii) of (15.23). By Corollary 53.1f, below this is equal to the minimum d-weight of an r-cut in D, which is equal to the minimum c-weight of a nontrivial cut in G.

No explicit description in terms of linear inequalities is known for the up hull of the incidence vectors of nontrivial cuts. Alevras [1999] gave descriptions for small instances (up to seven vertices for undirected graphs and up to five vertices for directed graphs).

The minimum k-cut problem: 'find a partition of the vertex set of a graph into k nonempty classes such that the number of edges connecting different classes is minimized', is NP-complete if k is part of the input (there is an easy reduction from the maximum clique problem, as the problem is equivalent to maximizing the number of edges spanned by the classes in the partition). For fixed k however, it was shown to be polynomial-time solvable by Goldschmidt and Hochbaum [1988,1994]. If we prescribe certain vertices to belong to the classes, the problem is NP-complete even for $k=3$ (Dahlhaus, Johnson, Papadimitriou, Seymour, and Yannakakis [1992, 1994]). More on this problem can be found in Hochbaum and Shmoys [1985], Lee, Park, and Kim [1989], Chopra and Rao [1991], Cunningham [1991], He [1991], Saran and Vazirani [1991,1995], Garg, Vazirani, and Yannakakis [1994], Kapoor [1996], Burlet and Goldschmidt [1997], Kamidoi, Wakabayashi, and Yoshida [1997], Călinescu, Karloff, and Rabani [1998,2000], Hartvigsen [1998b], Karzanov [1998c], Cunningham and Tang [1999], Karger, Klein, Stein, Thorup, and Young [1999], Nagamochi and Ibaraki [1999a,2000], Nagamochi, Katayama, and Ibaraki [1999, 2000], Goemans and Williamson [2001], Naor and Rabani [2001], Zhao, Nagamochi, and Ibaraki [2001], and Ravi and Sinha [2002].

Surveys on connectivity are given by Even [1979], Mader [1979], Frank [1995], and Subramanian [1995] (edge-connectivity). For the decomposition of 3-connected graphs into 4-connected graphs, see Coullard, Gardner, and Wagner [1993].