3

MAXIMUM FLOWS IN NETWORKS

In Chapter 2, we discussed the shortest directed path problem. There, a network or simply a net is a weighted directed graph $G(V, E)$ in which each arc is associated with a real number called the length of the arc. The length may be positive, negative or zero. In the case of positive length, it may represent the physical length of a street or highway, or the energy absorption property of a system when it is changed from one state to another. On the other hand, a negative length may signify the release of energy when the system is transformed from one state to another. If we think of the network as an interconnection of pipelines or communication wires, the nodes represent the junctions of the pipelines or the communication centers and the arcs denote the pipelines or wires themselves. Then the nonnegative real number of an arc may represent the cross-sectional area of the corresponding pipeline or the message-carrying capacity of the corresponding wire. Such a problem comes up naturally in the study of transportation or communication networks. For this reason, we introduce a new concept of flows in a network and use nonnegative real numbers to indicate the maximum flow that can transfer through the arc. The nonnegative real number is called the arc capacity. In this chapter, we discuss various problems associated with the flows in a capacitated network.

3.1 FLOWS

A net is a directed graph $G(V, E)$ in which each arc (i, j) is assigned a nonnegative weight $c(i, j)$ called the *capacity* of the arc. The capacities of arcs of $G(V, E)$ can be considered as a function c from the arc set E to the nonnegative reals. Sometimes, it is convenient to allow infinite arc capacities. Also, to each arc $(i, j) \in E$ we assign a nonnegative weight $f(i, j)$ called the *flow* in arc (i, j). Thus, the flows in $G(V, E)$ are a function f from E to the nonnegative reals. More precisely, the net considered in this chapter is a *quadruplet* $G(V, E, c, f)$ with node set V, arc set E, capacity function c and flow function f. The set of flows associated with E denoted by $\{f(i, j)\}$ is called a *flow pattern.*

A flow pattern $\{f(i, j)\}$ is said to be *feasible* in $G(V, E, c, f)$ and of value f_{st} from node s to node t if it satisfies the following constraints: For each $i \in V$,

$$\sum_j f(i, j) - \sum_j f(j, i) = f_{st}, \qquad i = s \tag{3.1a}$$

$$= 0, \qquad i \neq s, t \tag{3.1b}$$

$$= -f_{st}, \qquad i = t \tag{3.1c}$$

$$c(i, j) \geqq f(i, j) \geqq 0, \qquad (i, j) \in E \tag{3.2}$$

Thus, to define a feasible flow pattern, two nodes s and t are distinguished. We call node s the *source* and node t the *sink*. All other nodes are referred to as the *intermediate nodes*. If the net flow out of node i is defined to be

$$\sum_j f(i, j) - \sum_j f(j, i) \tag{3.3}$$

then equation (3.1) states that the net flow out of the source s is f_{st}, the net flow into the sink t is f_{st} or the net flow out of the sink t is $-f_{st}$, and the net flow out of an intermediate node is zero. Equation (3.1b) is called a *conservation equation,* similar to Kirchhoff's current law equation for electrical networks. Our objective here is to determine a maximum possible feasible flow pattern, the value f_{st} of which is called the *maximum flow*.

We remark that since $f = 0$ and $f_{st} = 0$ are permitted, there is no question about the existence of a feasible flow pattern. While the arc set E is a subset of all the ordered pairs (i, j), $i \neq j$ and $i, j \in V$, with capacity function c nonnegative on E, we could assume the net G to be a complete directed graph by taking $c = 0$ for arcs not originally in E, or we could assume strict positivity of the capacity function c by deleting from E arcs having zero capacity. Notice that not all the equations of (3.1) are linearly independent. To see this, let $\boldsymbol{A}_a$ be the complete incidence matrix of G, and let $\boldsymbol{F}$ be the column vector of the arc flows of E, arranged in the same arc order as the columns of $\boldsymbol{A}_a$. Now row-partition the matrix $\boldsymbol{A}_a$ in accordance with the source s, intermediate nodes i and sink t:

$$\boldsymbol{A}_a = \begin{bmatrix} \boldsymbol{A}_s \\ \boldsymbol{A}_i \\ \boldsymbol{A}_t \end{bmatrix} \tag{3.4}$$

Equation (3.1) can be rewritten as

$$\mathbf{A}_a \boldsymbol{F} = \begin{bmatrix} \boldsymbol{A}_s \\ \boldsymbol{A}_i \\ \boldsymbol{A}_t \end{bmatrix} \boldsymbol{F} = \begin{bmatrix} f_{st} \\ \mathbf{0} \\ -f_{st} \end{bmatrix} \tag{3.5}$$

Observe that by adding all the equations of (3.5) we produce a row of zeros, implying that not all the equations are linearly independent. Therefore, we could omit any one of them without loss of generality. We prefer, however, to retain the one-to-one correspondence between the equations and nodes. Finally, we mention that by adding an arc (t, s) in G with capacity $c(t, s) \geqq f_{st}$, all equations in the resulting net can be taken as conservative equations.

We illustrate the above result by the following example.

Example 3.1

Consider the net $G(V, E, c, f)$ of Fig. 3.1, where node s is the source, t is the sink, and a, b, c and d are the intermediate nodes. The capacities of the arcs are given by

$$\begin{aligned} &c(s, a) = 2, \quad && c(a, s) = 1, \quad && c(s, b) = 3 \\ &c(a, b) = 3, && c(a, c) = 1, && c(b, c) = 1 \\ &c(b, d) = 4, && c(d, a) = 2, && c(d, c) = 1 \\ &c(c, t) = 2, && c(d, t) = 4, && c(t, d) = 1 \end{aligned} \tag{3.6}$$

These are given as the first members of the pairs of numbers written adjacent to the arcs of the net of Fig. 3.1. A feasible flow pattern given as the second members of the pairs of numbers written adjacent to the arcs

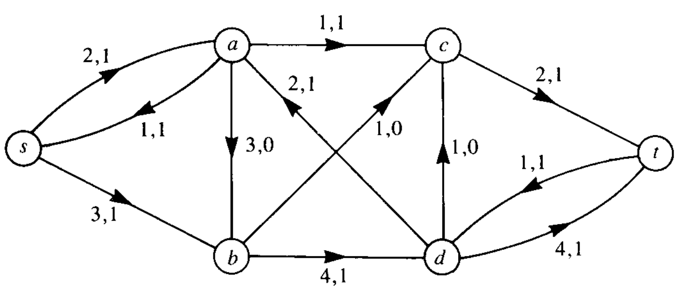

Fig. 3.1. A net $G(V, E, c, f)$ used to illustrate the flow problem.

of Fig. 3.1 is found to be

$$
\begin{aligned}
&f(s, a)=1, \qquad && f(a, s)=1, \qquad && f(s, b)=1 \\
&f(a, b)=0, && f(a, c)=1, && f(b, c)=0 \\
&f(b, d)=1, && f(d, a)=1, && f(d, c)=0 \\
&f(c, t)=1, && f(d, t)=1, && f(t, d)=1
\end{aligned}
\tag{3.7}
$$

The constraints corresponding to (3.5) are obtained from Fig. 3.1 as

$$
\begin{array}{c}
\\ s \\ a \\ b \\ c \\ d \\ t
\end{array}
\begin{array}{c}
\begin{array}{cccccccccccc}
(s,a) & (a,s) & (s,b) & (a,b) & (a,c) & (b,c) & (b,d) & (d,a) & (d,c) & (c,t) & (d,t) & (t,d)
\end{array} \\
\left[\begin{array}{cccccccccccc}
1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
-1 & 1 & 0 & 1 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & -1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 1 & 0 & 1 & -1 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 1
\end{array}\right]
\end{array}
\begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}
=
\begin{array}{c} s \\ a \\ b \\ c \\ d \\ t \end{array}
\left[\begin{array}{c} 1 \\ \hline 0 \\ 0 \\ 0 \\ 0 \\ \hline -1 \end{array}\right]
\tag{3.8}
$$

showing that the value f_{st} of this feasible flow pattern is 1.

3.2 *s–t* CUTS

A *cutset* of a directed graph or a net $G(V, E, c, f)$ is a subgraph consisting of a minimal collection of arcs whose removal reduces the rank of G by one. Intuitively, if we "cut" the arcs of a cutset, one of the components of G will be cut into two pieces. The name *cutset* has its origin in this interpretation. As examples of cutsets, consider the net $G(V, E, c, f)$ of Fig. 3.1. The subgraphs $(a, c)(d, a)(a, b)(s, b)$, $(a, c)(d, a)(b, c)(b, d)$, and $(c, t)(t, d)(d, t)$ are examples of cutsets. The broken lines of Fig. 3.2 show how these cutsets "cut" G. The subgraph

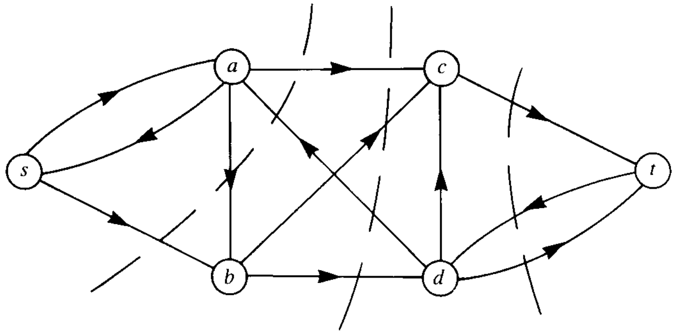

Fig. 3.2. A graph used to illustrate the concepts of a cut and a cutset.

$(a, c)(d, a)(a, b)(s, b)(c, t)(t, d)(d, t)$ is not a cutset because the removal of these arcs from G reduces the rank of G from 5 to 3, a reduction of two instead of one as required. This brings out the concept of a cut. A *cut* of G is either a cutset or an arc-disjoint union of cutsets of G. The subgraph $(a, c)(d, a)(a, b)(s, b)(c, t)(t, d)(d, t)$ is a cut but not a cutset. A cut can also be interpreted in another useful fashion. Let X be a nonempty proper subset of the node set V of G, and write $\bar{X} = V - X$. Then the set of arcs of G each of which is incident with one of its two endpoints in X and the other in $\bar{X}$ is a cut of G. In particular, if the removal of these arcs from G increases the number of components of G by one, then the cut is also a cutset. In Fig. 3.2, let $X = \{s, a\}$ and $\bar{X} = \{b, c, d, t\}$. Then the set of arcs (a, c), (a, b), (s, b) and (d, a) forms a cut of G. It is also a cutset of G. On the other hand, if we let $X = \{s, a, t\}$ and $\bar{X} = \{b, c, d\}$, the set of arcs (a, c), (d, a), (a, b), (s, b), (c, t), (t, d) and (d, t) forms a cut but not a cutset.

DEFINITION 3.1

***s–t* Cutset.** For two distinguished nodes s and t of a directed graph, an s–t *cutset* is a minimal set of arcs, the removal of which breaks all the directed paths from s to t in the directed graph.

The term minimal in the definition needs some clarification. A set S or a subgraph W is said to be *minimal* with respect to property P if no proper subset of S or no proper subgraph of W has property P.

To simplify the notation, we adopt the following conventions. Let X and Y be two subsets of the node set V of a net $G(V, E, c, f)$. We use the symbol (X, Y) to denote the set of all arcs (x, y) directed from $x \in X$ to $y \in Y$. For any function g from the arc set E to the reals, write

$$g(X, Y) = \sum_{(x,y)\in(X,Y)} g(x, y) \tag{3.9}$$

Likewise, when we deal with a function h from the node set V to the reals, we write

$$h(X) = \sum_{x \in X} h(x) \tag{3.10}$$

where $h(x)$ is the weight assigned to node x. For simplicity, we denote a set consisting only of one element by its single element. Thus, if X contains only a single node x, we write (x, Y), $g(x, Y)$ or $h(x)$ instead of $(\{x\}, Y)$, $g(\{x\}, Y)$ or $h(\{x\})$.

DEFINITION 3.2

***s–t* Cut.** For two distinguished nodes s and t of a directed graph $G(V, E)$, an *s–t cut* is the set of arcs $(X, \bar{X})$ of G with $s \in X$ and $t \in \bar{X}$, where X is a subset of V and $\bar{X} = V - X$.

We remark that an s–t cut may not be an s–t cutset. Consider, for example, the directed graph $G(V, E)$ of Fig. 3.3. Let $X = \{s, b\}$. Then

$$(X, \bar{X}) = \{(s, a), (b, a), (b, t)\} \tag{3.11}$$

where $\bar{X} = \{a, t\}$, is an s–t cut, but it is not an s–t cutset, because $\{(b, t)\}$ is a proper subset of $(X, \bar{X})$ and the removal of (b, t) from G will also break all directed paths from s to t. However, we can show that every s–t cutset is an s–t cut. To see this, let Q be an s–t cutset of a directed graph $G(V, E)$. Define the subset X of V recursively as follows:

(1) $s \in X$.
(2) If $x \in X$ and $(x, y) \in E - Q$, then $y \in X$.

Then $t \in \bar{X} = V - X$. We first show that every arc $(x, y) \in (X, \bar{X})$, $x \in X$ and $y \in \bar{X}$, is in Q. Suppose that (x, y) is not in Q. Then in $E - Q$ there is a directed path from s to y formed by a directed path from s to x followed by

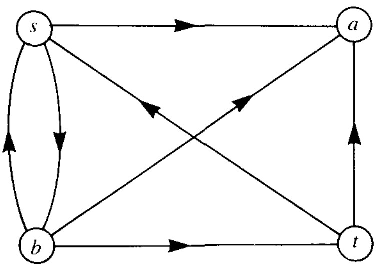

Fig. 3.3. An example of an s–t cut that is not an s–t cutset.

(x, y). This would imply that $y \in X$, a contradiction. Conversely, if there is an arc $(x, y) \in Q$ which is not in $(X, \bar{X})$, then Q is not an s–t cutset because the removal of $(X, \bar{X})$, a proper subset of Q, will also break all directed paths from s to t. Thus, every s–t cutset is an s–t cut.

In Fig. 3.3, (b, t) is an s–t cutset. Using the procedure outlined above yields the subset $X = \{s, a, b\}$. The corresponding s–t cut is found to be

$$(X, \bar{X}) = (\{s, a, b\}, t) = \{(b, t)\} \tag{3.12}$$

The *capacity* of an s–t cut $(X, \bar{X})$ in a net $G(V, E, c, f)$ is defined to be

$$c(X, \bar{X}) = \sum_{(x,y) \in (X, \bar{X})} c(x, y) \tag{3.13}$$

A minimum s–t cut $C_{\min}$ is an s–t cut with minimum capacity among all the s–t cuts:

$$c(C_{\min}) = \min_i \{c(X_i, \bar{X}_i)\} \tag{3.14}$$

where $(X_i, \bar{X}_i)$ is an s–t cut of G. Likewise, the *capacity* of an s–t cutset Q in G is defined by

$$c(Q) = \sum_{(x,y) \in Q} c(x, y) \tag{3.15}$$

A minimum s–t cutset $Q_{\min}$ is an s–t cutset with minimum capacity among all the s–t cutsets:

$$c(Q_{\min}) = \min_k \{c(Q_k)\} \tag{3.16}$$

where Q_k is an s–t cutset of G. We show that if all arc capacities are positive, then

$$c(C_{\min}) = \min_i \{c(X_i, \bar{X}_i)\} = \min_k \{c(Q_k)\} = c(Q_{\min}) \tag{3.17}$$

To justify (3.17), we use the fact that since every s–t cutset is an s–t cut, it is clear that

$$c(C_{\min}) \leqq c(Q_{\min}) \tag{3.18}$$

To complete the proof, we show that the reverse inequality is also true and hence the equality must hold. Since $C_{\min}$ is a minimum s–t cut and since all the arc capacities are positive by assumption, no proper subset of $C_{\min}$ can be an s–t cut; for, otherwise, the s–t cut formed by the proper subset would

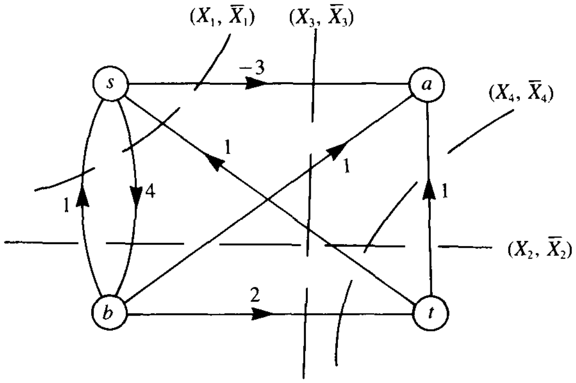

Fig. 3.4. An example showing that nonnegative arc capacities are necessary for identity (3.17) to hold.

have a smaller capacity than $C_{\min}$. This means that no proper subset of $C_{\min}$ can break all directed paths from s to t in G. Consequently, $C_{\min}$ is a minimal set of arcs, the removal of which breaks all the directed paths from s to t in G; and, by definition, it is also an s–t cutset. This implies

$$c(C_{\min}) \geqq c(Q_{\min}) \tag{3.19}$$

Combining this with (3.18) gives (3.17). The proof is completed.

To see that the hypothesis that all arc capacities be nonnegative is necessary for (3.17) to hold, consider the net $G(V, E, c, f)$ of Fig. 3.4. There are four s–t cuts, as follows:

$$(X_1, \bar{X}_1) = (s, \{a, b, t\}) = \{(s, a), (s, b)\} \tag{3.20a}$$

$$(X_2, \bar{X}_2) = (\{s, a\}, \{b, t\}) = \{(s, b)\} \tag{3.20b}$$

$$(X_3, \bar{X}_3) = (\{s, b\}, \{a, t\}) = \{(b, t), (b, a), (s, a)\} \tag{3.20c}$$

$$(X_4, \bar{X}_4) = (\{s, a, b\}, t) = \{(b, t)\} \tag{3.20d}$$

whose capacities are

$$c(X_1, \bar{X}_1) = c(s, a) + c(s, b) = -3 + 4 = 1 \tag{3.21a}$$

$$c(X_2, \bar{X}_2) = c(s, b) = 4 \tag{3.21b}$$

$$c(X_3, \bar{X}_3) = c(b, t) + c(b, a) + c(s, a) = 2 + 1 - 3 = 0 \tag{3.21c}$$

$$c(X_4, \bar{X}_4) = c(b, t) = 2 \tag{3.21d}$$

On the other hand, there are two s–t cutsets $Q_1 = \{(s, b)\}$ and $Q_2 = \{(b, t)\}$

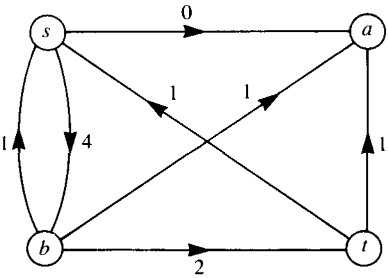

Fig. 3.5. An example showing that the zero arc capacity is permitted for identity (3.17) to hold.

whose capacities are $c(Q_1)=4$ and $c(Q_2)=2$. This gives $C_{\min}=0$ and $Q_{\min}=2$ and (3.17) fails to hold. The reason is that the net contains an arc (s, a) with negative arc capacity $c(s, a)=-3$. In passing, we mention that $Q_1=(X_2, \bar{X}_2)$ and $Q_2=(X_4, \bar{X}_4)$, confirming an earlier assertion that every s–t cutset is also an s–t cut.

With the exclusion of nets with negative arc capacity, (3.17) remains valid for nets with nonnegative arc capacities, zero included. Fig. 3.5 is a net containing a zero capacity arc (s, a). Since the nets of Figs. 3.4 and 3.5 are isomorphic, they possess the same s–t cuts and s–t cutsets. However, their capacities are different and are given below:

$$c(X_1, \bar{X}_1)=c(s, a)+c(s, b)=4, \qquad c(X_2, \bar{X}_2)=c(s, b)=4$$

$$c(X_3, \bar{X}_3)=c(b, t)+c(b, a)+c(s, a)=3, \qquad c(X_4, \bar{X}_4)=c(b, t)=2 \tag{3.22a}$$

$$c(Q_1)=c(s, b)=4, \qquad c(Q_2)=c(b, t)=2 \tag{3.22b}$$

showing that $C_{\min}=Q_{\min}=2$ and (3.17) holds. To justify this extension, we observe that if an s–t cut is not an s–t cutset, then a proper subset of the s–t cut is an s–t cutset. Since all arc capacities are nonnegative, the capacity of this s–t cut is at least as large as that of the proper subset. Hence, the s–t cutsets determine the minimum s–t cut.

Finally, we mention that a directed circuit L of G contains an equal number of arcs in common with the s–t cut $(X, \bar{X})$ and t–s cut $(\bar{X}, X)$. This follows from the observation that a directed circuit L and a cut $(X, \bar{X}) \cup (\bar{X}, X)$ have an even number of arcs in common, zero included, that in traversing along L the common arcs will alternately appear in $(X, \bar{X})$ and $(\bar{X}, X)$ because starting from a node x on L, eventually it has to return to x to complete the circuit.

Since an s–t cut $(X, \bar{X})$ breaks all directed paths from s to t, it is intuitively clear that the value f_{st} of a flow f in a net $G(V, E, c, f)$ cannot exceed the capacity of any s–t cut. Before we prove this, we rewrite the constraints (3.1) in terms of the simplified notation of (3.9): For each $i \in V$,

$$f(s, V) - f(V, s) = f_{st} \tag{3.23a}$$

$$f(i, V) - f(V, i) = 0, \qquad i \neq s, t \tag{3.23b}$$

$$f(t, V) - f(V, t) = -f_{st} \tag{3.23c}$$

These together with the capacity constraints

$$c(i, j) \geqq f(i, j) \geqq 0, \qquad (i, j) \in E \tag{3.24}$$

define a feasible flow pattern $\{f(i, j)\}$.

THEOREM 3.1

Let $(X, \bar{X})$ be an s–t cut of a net $G(V, E, c, f)$. Then the flow value f_{st} from s to t in G is given by

$$f_{st} = f(X, \bar{X}) - f(\bar{X}, X) \leqq c(X, \bar{X}) \tag{3.25}$$

Proof. Since f is a flow of G, it satisfies the equations (3.23). Summing these equations over $x \in X$ gives

$$f_{st} = \sum_{x \in X} [f(x, V) - f(V, x)] = f(X, V) - f(V, X) \tag{3.26}$$

Writing $V = X \cup \bar{X}$ in (3.26) and expanding the resulting terms yields

$$\begin{aligned} f_{st} &= f(X, X \cup \bar{X}) - f(X \cup \bar{X}, X) \\ &= f(X, X) + f(X, \bar{X}) - f(X, X) - f(\bar{X}, X) \\ &= f(X, \bar{X}) - f(\bar{X}, X) \end{aligned} \tag{3.27}$$

Since $f(\bar{X}, X) \geqq 0$ and $f(X, \bar{X}) \leqq c(X, \bar{X})$ by virtue of (3.24), (3.25) follows immediately. This completes the proof of the theorem.

The theorem states that the value of a flow from s to t is equal to the net flow across any s–t cut. In the net $G(V, E, c, f)$ of Fig. 3.1, which is redrawn in Fig. 3.6, the first member of a pair of numbers written adjacent to an arc denotes the arc capacity, and the second member denotes the flow in the arc. Three cuts are indicated in Fig. 3.6, the net flows of which are found to be

$$\begin{aligned} f(X_1, \bar{X}_1) - f(\bar{X}_1, X_1) &= f(a, c) + f(a, b) + f(s, b) - f(d, a) \\ &= 1 + 0 + 1 - 1 = 1 \leqq c(X_1, \bar{X}_1) = 1 + 3 + 3 = 7 \end{aligned} \tag{3.28a}$$

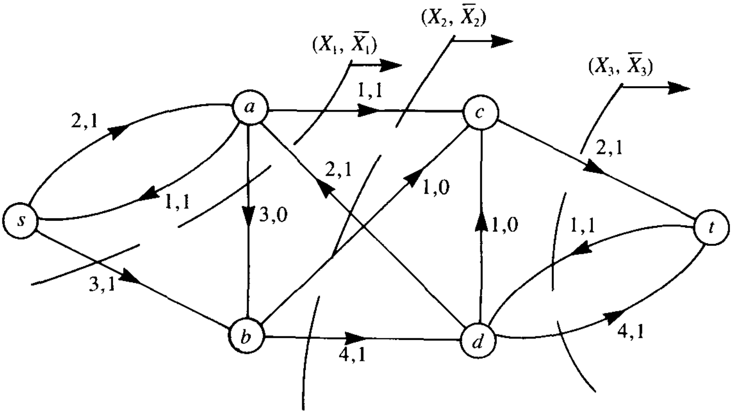

Fig. 3.6. A labeled net with a pair of numbers written adjacent to an arc, the first number denoting the arc capacity and the second number denoting the flow in the arc.

$$f(X_2, \bar{X}_2) - f(\bar{X}_2, X_2) = f(a, c) + f(b, c) + f(b, d) - f(d, a)$$
$$= 1 + 0 + 1 - 1 = 1 \leqq c(X_2, \bar{X}_2) = 1 + 1 + 4 = 6 \quad (3.28b)$$
$$f(X_3, \bar{X}_3) - f(\bar{X}_3, X_3) = f(c, t) + f(d, t) - f(t, d)$$
$$= 1 + 1 - 1 = 1 \leqq c(X_3, \bar{X}_3) = 2 + 4 = 6 \quad (3.28c)$$

The value of this flow from s to t is 1, being equal to the net flow across any one of the three s–t cuts.

3.3 MAXIMUM FLOW

A basic problem in flows is to find a feasible flow pattern that maximizes the value f_{st} of a flow in a given net $G(V, E, c, f)$. In this section, we state and prove the fundamental result concerning the maximum flow known as the max-flow min-cut theorem. The theorem was independently discovered by Ford and Fulkerson (1956) and Elias, Feinstein and Shannon (1956). Ford and Fulkerson (1956) gave a combinatorial proof which constructs a maximum flow and locates a minimum s–t cut. Their proof shows that there always exists a flow with value equal to the capacity of a minimum s–t cut. The proof will also serve as a basis for a number of other algorithms in this chapter. Elias, Feinstein and Shannon (1956) used an ingenious graph–theoretic approach by breaking a net into simpler nets until a solution for the simpler nets become evident. A third proof was given by Dantzig and Fulkerson (1956) using the Duality Theorem of linear programming.

THEOREM 3.2

(Max-flow min-cut theorem). The maximum flow value $f_{\max}$ from s to t in a net $G(V, E, c, f)$ is equal to the capacity of a minimum s–t cut, i.e.

$$f_{\max} = \max \{f_{st}\} = \min_i \{c(X_i, \bar{X}_i)\} \tag{3.29}$$

where $(X_i, \bar{X}_i)$ is an s–t cut, and $\max \{f_{st}\}$ is taken over all feasible flow patterns in G.

Proof. Let $(X, \bar{X})$ be an arbitrary s–t cut. Then from Theorem 3.1 the value f_{st} of any flow f is bounded above by the capacity of any s–t cut:

$$f_{st} \leqq f(X, \bar{X}) \leqq c(X, \bar{X}) \tag{3.30}$$

In particular, the maximum flow value $f_{\max}$ is bounded above by the capacity of a minimum s–t cut, or

$$f_{\max} = \max \{f_{st}\} \leqq \min_i \{c(X_i, \bar{X}_i)\} \tag{3.31}$$

Thus, to prove the theorem, it suffices to establish the existence of a flow f and an s–t cut $(X, \bar{X})$ such that the flow value equals the capacity of this s–t cut $(X, \bar{X})$ or

$$f(X, \bar{X}) = c(X, \bar{X}) \tag{3.32a}$$

$$f(\bar{X}, X) = 0 \tag{3.32b}$$

which is equivalent to stating that

$$\begin{aligned} f_{\max} = \max \{f_{st}\} &= f(X, \bar{X}) - f(\bar{X}, X) = f(X, \bar{X}) \\ &= c(X, \bar{X}) = \min_i \{c(X_i, \bar{X}_i)\} \end{aligned} \tag{3.33}$$

and the equality holds throughout (3.25). We now proceed to establish such a flow and to locate the desired s–t cut $(X, \bar{X})$.

Let f be a maximum flow in $G(V, E, c, f)$. Using f, define a subset X of the node set V recursively as follows:

(1) $s \in X$.
(2) If $x \in X$ and $f(x, y) < c(x, y)$, then $y \in X$.
(3) If $x \in X$ and $f(y, x) > 0$, then $y \in X$.

We assert that $t \in \bar{X} = V - X$; for, if not, there exists a path, not necessarily a directed path, between nodes s and t,

$$P_{st} = (s, i_2)(i_2, i_3)\ldots(i_{k-1}, t) \tag{3.34}$$

such that all forward arcs $(i_\alpha, i_{\alpha+1})$ of P_{st} are not saturated,

$$f(i_\alpha, i_{\alpha+1}) < c(i_\alpha, i_{\alpha+1}) \tag{3.35}$$

and all reverse arcs $(i_{\alpha+1}, i_\alpha)$ of P_{st} are not flowless,

$$f(i_{\alpha+1}, i_\alpha) > 0 \tag{3.36}$$

Here, an arc $(i_\alpha, i_{\alpha+1})$ of P_{st} is said to be a *forward arc* of P_{st} if in traversing from s to t on P_{st} the arc $(i_\alpha, i_{\alpha+1})$ is directed from i_α to $i_{\alpha+1}$. Otherwise, it is a *reverse arc* of P_{st}. Thus, an arc is a forward arc from s to t on P_{st} and becomes a reverse arc from t to s on P_{st}. We say that an arc (i, j) is *saturated* with respect to a flow f if $f(i, j) = c(i, j)$ and is *flowless* with respect to f if $f(i, j) = 0$. Thus, an arc that is both saturated and flowless can only have zero capacity. Let

$$w_1 = \min\{c(i_\alpha, i_{\alpha+1}) - f(i_\alpha, i_{\alpha+1})\} \tag{3.37}$$

taken over all forward arcs $(i_\alpha, i_{\alpha+1})$ of P_{st},

$$w_2 = \min\{f(i_{\alpha+1}, i_\alpha)\} \tag{3.38}$$

taken over all reverse arcs $(i_{\alpha+1}, i_\alpha)$ of P_{st}, and

$$w = \min(w_1, w_2) \tag{3.39}$$

We now define a new flow f^* based on the original flow f, as follows:

$$f^*(i_\alpha, i_{\alpha+1}) = f(i_\alpha, i_{\alpha+1}) + w \tag{3.40a}$$

for all forward arcs $(i_\alpha, i_{\alpha+1})$ of P_{st},

$$f^*(i_{\alpha+1}, i_\alpha) = f(i_{\alpha+1}, i_\alpha) - w \tag{3.40b}$$

for all reverse arcs $(i_{\alpha+1}, i_\alpha)$ of P_{st}, and

$$f^*(i, j) = f(i, j) \tag{3.40c}$$

for all other arcs (i, j) of G. It is straightforward to verify that the new flow function f^* thus defined is a feasible flow pattern having value $f_{st} + w$ from s to t, where f_{st} is the value of the original flow f. This shows that f is not a maximum flow, contrary to our assumption. Thus, t must be in $\bar{X}$, and $(X, \bar{X})$ is an s–t cut.

By construction, we see that

$$f(i, j) = c(i, j) \tag{3.41a}$$

for $(i, j) \in (X, \bar{X})$, $i \in X$ and $j \in \bar{X}$, and

$$f(j, i) = 0 \tag{3.41b}$$

for $(j, i) \in (\bar{X}, X)$, $j \in \bar{X}$ and $i \in X$; for otherwise j would be in X. Hence

$$f(X, \bar{X}) = c(X, \bar{X}) \tag{3.42a}$$

$$f(\bar{X}, X) = 0 \tag{3.42b}$$

This completes the proof of the theorem.

DEFINITION 3.3

Flow Augmenting Path. A path P_{st}, not necessarily a directed path, between nodes s and t in a given net $G(V, E, c, f)$ is said to be a *flow augmenting path* with respect to f if all forward arcs (i, j) of P_{st} are not saturated, $f(i, j) < c(i, j)$, and all reverse arcs (j, i) are not flowless, $f(j, i) > 0$, in traversing from s to t on P_{st}.

The path P_{st} defined in (3.34) is a flow augmenting path. Thus, a flow f is maximum if and only if there is no flow augmenting path with respect to f. The result is useful in that in order to increase the value of a flow, it suffices to look for flow augmenting paths. In a similar way, we see that an s–t cut $(X, \bar{X})$ is minimum if and only if for every maximum flow f, all arcs of $(X, \bar{X})$ are saturated and all arcs $(\bar{X}, X)$ are flowless.

Example 3.2

In the net $G(V, E, c, f)$ of Fig. 3.7, a feasible flow pattern is shown in the figure. The s–t cut

$$(X, \bar{X}) = (s, \{a, b, c, d, t\}) = \{(s, a), (s, b)\} \tag{3.43}$$

has minimum capacity

$$c(X, \bar{X}) = c(s, a) + c(s, b) = 2 + 3 = 5 \tag{3.44}$$

among all the s–t cuts. Thus, by Theorem 3.2 the maximum flow from s

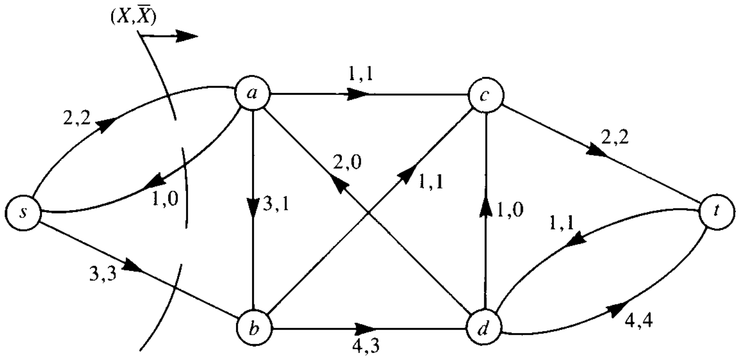

Fig. 3.7. A net showing a feasible flow pattern.

to t in G is equal to $c(X, \bar{X})$ or

$$f_{\max} = c(X, \bar{X}) = 5 \tag{3.45}$$

Observe that all arcs in $(X, \bar{X})$ are saturated and the arc in $(\bar{X}, X) = \{(a, s)\}$ is flowless. The s–t cut

$$(Y, \bar{Y}) = (\{s, a, b\}, \{c, d, t\}) = \{(a, c), (b, c), (b, d)\} \tag{3.46}$$

is not minimum because not all arcs in $(Y, \bar{Y})$ are saturated. Also, it is easy to check that there is no flow augmenting path in G.

We now state and prove other properties of the minimum s–t cuts.

THEOREM 3.3

Let $(X, \bar{X})$ and $(Y, \bar{Y})$ be two minimum s–t cuts in a net $G(V, E, c, f)$. Then $(X \cup Y, \overline{X \cup Y})$ and $(X \cap Y, \overline{X \cap Y})$ are also minimum s–t cuts of G, where $\overline{X \cup Y} = V - X \cup Y$ and $\overline{X \cap Y} = V - X \cap Y$.

Proof. If X is contained in Y, then $X \cup Y = Y$ and $X \cap Y = X$, giving

$$(X \cup Y, \overline{X \cup Y}) = (Y, \bar{Y}) \tag{3.47a}$$

$$(X \cap Y, \overline{X \cap Y}) = (X, \bar{X}) \tag{3.47b}$$

Likewise, if Y is contained in X, similar results are obtained, and the theorem is trivial. Thus, we assume that the intersections

$$\begin{aligned} A &= X \cap Y, \qquad & B &= \bar{X} \cap Y \\ C &= X \cap \bar{Y}, \qquad & D &= \bar{X} \cap \bar{Y} \end{aligned} \tag{3.48}$$

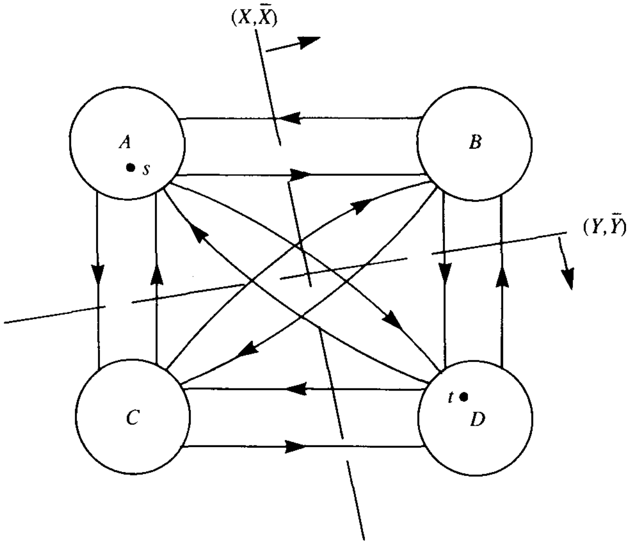

Fig. 3.8. A symbolic representation of the partition of the nodes of a net by the two minimum s–t cuts $(X, \bar{X})$ and $(Y, \bar{Y})$.

are not empty, as depicted in Fig. 3.8, where $s \in A$ and $t \in D$. Since $(X, \bar{X})$ and $(Y, \bar{Y})$ are minimum s–t cuts and $(A \cup B \cup C, D)$ and $(A, B \cup C \cup D)$ are s–t cuts, we must have the following inequalities:

$$c(A \cup B \cup C, D) + c(A, B \cup C \cup D) \geqq c(X, \bar{X}) + c(Y, \bar{Y}) \qquad (3.49)$$

each of which can be expanded as

$$c(A \cup B \cup C, D) = c(A, D) + c(B, D) + c(C, D) \qquad (3.50a)$$

$$c(A, B \cup C \cup D) = c(A, B) + c(A, C) + c(A, D) \qquad (3.50b)$$

$$c(X, \bar{X}) = c(A \cup C, B \cup D) = c(A, B) + c(A, D) + c(C, B) + c(C, D) \qquad (3.51a)$$

$$c(Y, \bar{Y}) = c(A \cup B, C \cup D) = c(A, C) + c(A, D) + c(B, C) + c(B, D) \qquad (3.51b)$$

Substituting these in (3.49) yields

$$c(C, B) + c(B, C) \leqq 0 \qquad (3.52)$$

Since $c(C, B)$ and $c(B, C)$ are both nonnegative, (3.52) can only be satisfied with the equality. Using this in (3.49) obtains

$$c(A \cup B \cup C, D) + c(A, B \cup C \cup D) = c(X, \bar{X}) + c(Y, \bar{Y}) \qquad (3.53)$$

which can be rewritten as

$$c(X \cup Y, \overline{X \cup Y}) + c(X \cap Y, \overline{X \cap Y}) = c(X, \bar{X}) + c(Y, \bar{Y}) \qquad (3.54)$$

Since $c(X, \bar{X})$ and $(Y, \bar{Y})$ are minimum s–t cuts, it is necessary that

$$c(X \cup Y, \overline{X \cup Y}) \geqq c(X, \bar{X}) = c(Y, \bar{Y}) \qquad (3.55a)$$

$$c(X \cap Y, \overline{X \cap Y}) \geqq c(X, \bar{X}) = c(Y, \bar{Y}) \qquad (3.55b)$$

These in conjunction with (3.54) shows that the equality must hold in (3.55), and $(X \cup Y, \overline{X \cup Y})$ and $(X \cap Y, \overline{X \cap Y})$ are minimum s–t cuts. This completes the proof of the theorem.

In words, Theorem 3.3 states that if there are two minimum s–t cuts that cross each other as in (3.48), then there are two more minimum s–t cuts that do not cross each other.

Example 3.3

A maximum feasible flow pattern is shown in Fig. 3.9 for the net $G(V, E, c, f)$. Two minimum s–t cuts are found to be

$$(X, \bar{X}) = (\{s, b, c\}, \{a, t\}) = \{(s, a), (s, t), (b, t)\} \qquad (3.56a)$$

$$(Y, \bar{Y}) = (\{s, a\}, \{b, c, t\}) = \{(a, t), (s, t), (s, c)\} \qquad (3.56b)$$

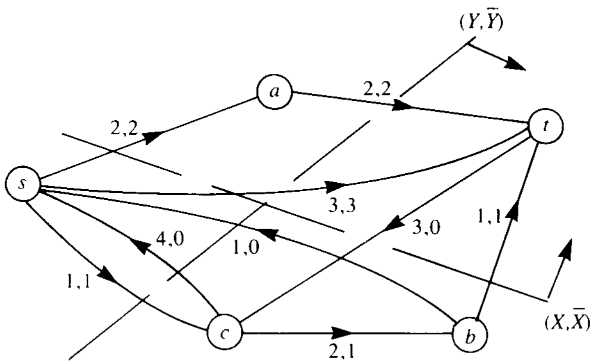

Fig. 3.9. A net with a maximum feasible flow pattern.

whose capacity is given by

$$c(X, \bar{X}) = c(Y, \bar{Y}) = 6 \tag{3.57}$$

Then according to Theorem 3.3 the s–t cuts

$$(X \cup Y, \overline{X \cup Y}) = (\{s, a, b, c\}, t) = \{(a, t), (s, t), (b, t)\} \tag{3.58a}$$

$$(X \cap Y, \overline{X \cap Y}) = (s, \{a, b, c, t\}) = \{(s, a), (s, t), (s, c)\} \tag{3.58b}$$

are also minimum. It is easy to confirm that the capacity of the s–t cuts in (3.58) is 6, being equal to the minimum s–t cut capacity.

3.4 FORD–FULKERSON ALGORITHM

The proof of the max-flow min-cut theorem given as Theorem 3.2 provides a simple and efficient algorithm for constructing a maximum flow and locating a minimum s–t cut in a given net $G(V, E, c, f)$. The Ford–Fulkerson algorithm is a procedure which systematically searches for a flow augmenting path, and then increases the flow along this path. It can be carried out by two routines: The Labeling Routine and the Augmentation Routine. The labeling routine searches for a flow augmenting path and the augmentation routine increases the flow along the flow augmenting path. The algorithm may start with zero flow, and terminates when the maximum flow is generated. With rare exception where the arc capacities are irrational, the algorithm terminates after a finite iteration.

The algorithm assigns labels to the nodes of a given net $G(V, E, c, f)$. The label assigned to a node is represented by an ordered triplet $(x, +, w)$ or $(x, -, w)$, where $x \in V$ and w is a positive number or ∞. During the labeling process, a node is always considered to be in one of the following three states:

(i) *Unlabeled node.* A node is *unlabeled* if it receives no label. At the beginning of the labeling process, every node is unlabeled.

(ii) *Labeled and unscanned node.* A node x is said to be *labeled and unscanned* if it has a label and if its neighboring nodes y, $(x, y) \in E$, have not all been labeled.

(iii) *Labeled and scanned node.* A node x is *labeled and scanned* if it has a label and if all of its neighboring nodes y, $(x, y) \in E$, have also been labeled.

We now describe Ford–Fulkerson's labeling routine and augmentation routine.

Labeling Routine

Step 1. Label s by $(s, +, \infty)$. Now s is labeled and unscanned and all other nodes are unlabeled and unscanned.

Step 2. Select any labeled and unscanned node x, and perform the following operations:

(a) For all nodes y, $(x, y) \in E$, that are unlabeled such that $f(x, y) < c(x, y)$, label y by $(x, +, w(y))$ where

$$w(y) = \min [w(x), c(x, y) - f(x, y)] \tag{3.59}$$

Then y is labeled and unscanned.

(b) For all nodes y, $(y, x) \in E$, that are unlabeled such that $f(y, x) > 0$, label y by $(x, -, w(y))$ where

$$w(y) = \min [w(x), f(y, x)] \tag{3.60}$$

Now change the label on x by circling the $+$ or $-$ entry. Then x is now labeled and scanned.

Step 3. Repeat Step 2 until either t is labeled or until no more labels can be assigned and t is unlabeled. In the former case, go to the Augmentation Routine; in the latter case, terminate.

Augmentation Routine

Step 1. Let $z = t$ and go to Step 2 below.

Step 2. If the label on z is $(q, +, w(z))$, increase the flow $f(q, z)$ by $w(t)$. If the label on z is $(q, -, w(z))$, decrease the flow $f(z, q)$ by $w(t)$.

Step 3. If $q = s$, discard all labels and return to Step 1 of the Labeling Routine. Otherwise, let $z = q$ and return to Step 2 of the Augmentation Routine.

We illustrate the Ford–Fulkerson algorithm by the following examples.

Example 3.4

Consider the net $G(V, E, c, f)$ of Fig. 3.1 which is redrawn in Fig. 3.10. The flow shown in the net G is not maximum and we shall apply the Ford–Fulkerson algorithm to generate a maximum flow, as follows:

First iteration

Labeling routine

Step 1. Label s by $(s, +, \infty)$.

Step 2. Node a is labeled by $(s, +, 1)$, and node b by $(s, +, 2)$. Circle the $+$ in $(s, +, \infty)$, giving $(s, \oplus, \infty)$.

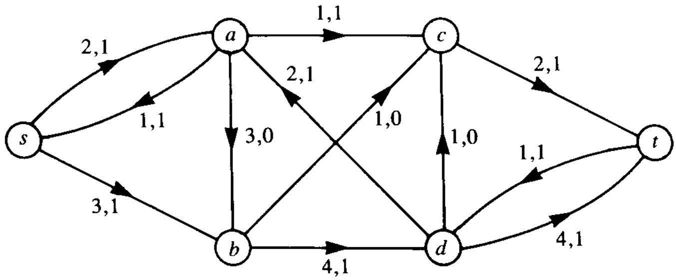

Fig. 3.10. A net used to illustrate the Ford–Fulkerson algorithm for the calculation of a maximum flow.

Step 3. Return to Step 2.

Step 2. Node c is labeled by $(b, +, 1)$ and change $(s, +, 2)$ to $(s, \oplus, 2)$. Node d is labeled by $(a, -, 1)$ and change $(s, +, 1)$ to $(s, \oplus, 1)$.

Step 3. Return to Step 2.

Step 2. Node t is labeled by $(c, +, 1)$, and change $(b, +, 1)$ to $(b, \oplus, 1)$.

Step 3. Go to Augmentation Routine.

Augmentation routine

Step 1. Let $z = t$.

Step 2. Increase the flow $f(c, t)$ by $w(t) = 1$.

Step 3. Set $z = c$.

Step 2. Increase the flow $f(b, c)$ by 1.

Step 3. Set $z = b$.

Step 2. Increase the flow $f(s, b)$ by 1.

Step 3. Discard all labels and return to Step 1 of the Labeling Routine.

The details of the first iteration are indicated in Fig. 3.11.

Second iteration

Labeling routine

Step 1. Label s by $(s, +, \infty)$.

Step 2. Node a is labeled by $(s, -, 1)$, and node b by $(s, +, 1)$. Change $(s, +, \infty)$ to $(s, \oplus, \infty)$.

Step 3. Return to Step 2.

Step 2. Node d is labeled by $(a, -, 1)$. Change $(s, -, 1)$ to $(s, \ominus, 1)$.

Step 3. Return to Step 2.

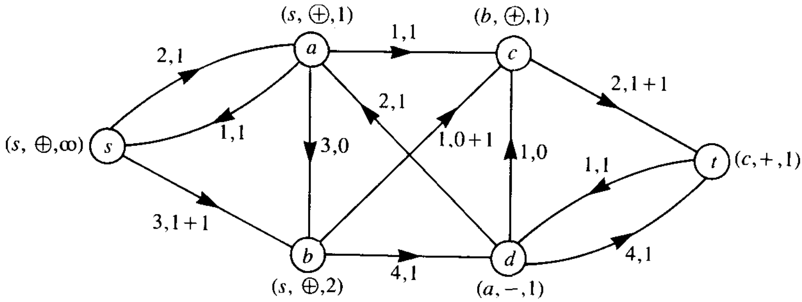

Fig. 3.11. The resulting net after the completion of the first iteration of the Ford–Fulkerson algorithm.

Step 2. Nodes c and t are labeled by $(d, +, 1)$. Change $(a, -, 1)$ to $(a, \ominus, 1)$.

Step 3. Go to Augmentation Routine.

Augmentation routine

Step 1. Let $z = t$.

Step 2. Increase the flow $f(d, t)$ by $w(t) = 1$.

Step 3. Set $z = d$.

Step 2. Decrease the flow $f(d, a)$ by 1.

Step 3. Set $z = a$.

Step 2. Decrease the flow $f(a, s)$ by 1.

Step 3. Discard all labels and return to Step 1 of the Labeling Routine.

The details of the second iteration are illustrated in Fig. 3.12.

The third, fourth and fifth iterations are shown in Figs. 3.13, 3.14 and 3.15, respectively. At the end of the fifth iteration, the algorithm terminates and the flow pattern shown in Fig. 3.15 is maximum. The

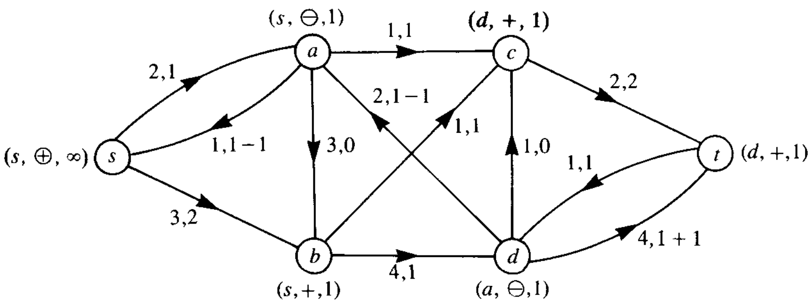

Fig. 3.12. The resulting net after the completion of the second iteration of the Ford–Fulkerson algorithm.

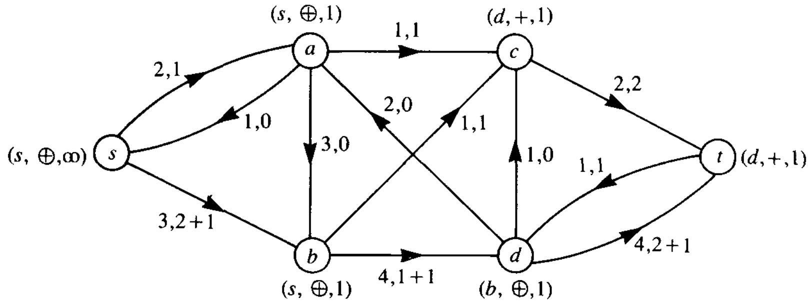

Fig. 3.13. The resulting net after the completion of the third iteration of the Ford–Fulkerson algorithm.

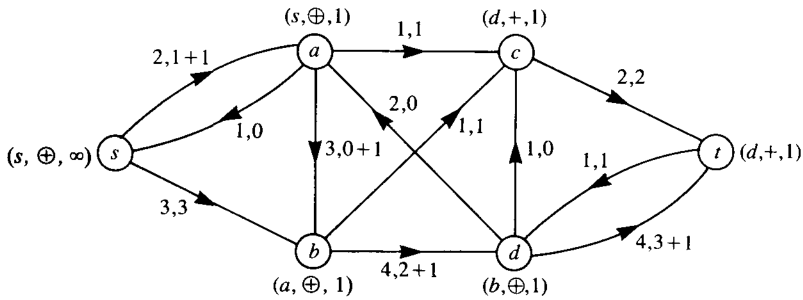

Fig. 3.14. The resulting net after the completion of the fourth iteration of the Ford–Fulkerson algorithm.

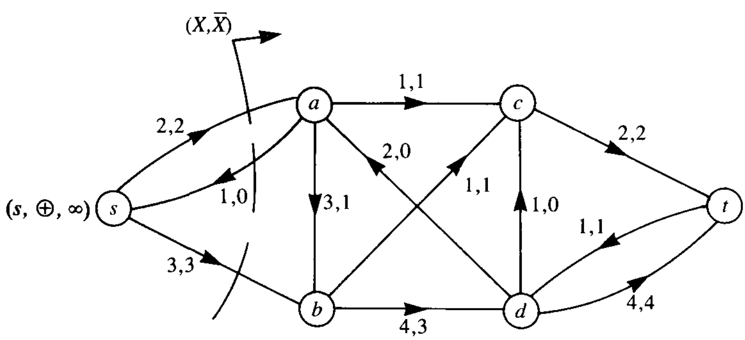

Fig. 3.15. The resulting maximum flow pattern after the application of the Ford–Fulkerson algorithm to the net of Fig. 3.10.

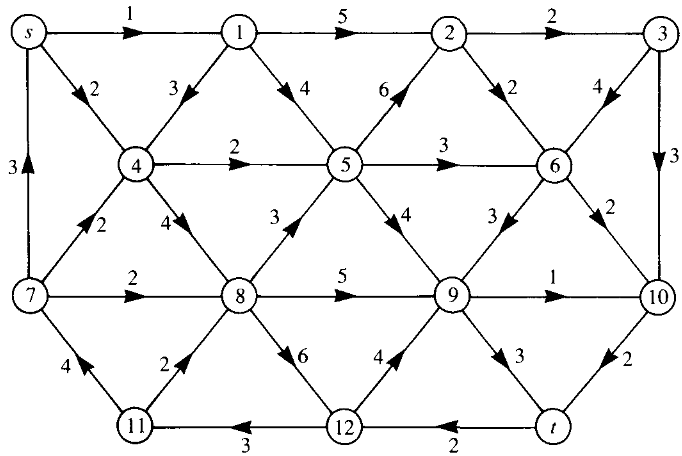

Fig. 3.16. A net used to illustrate the Ford–Fulkerson algorithm for the generation of a maximum flow pattern.

desired minimum s–t cut $(X, \bar{X})$ is found to be

$$(X, \bar{X}) = (s, \{a, b, c, d, t\}) = \{(s, a), (s, b)\} \tag{3.61}$$

the capacity of which is $c(X, \bar{X}) = c(s, a) + c(s, b) = 5$. The s–t cut $(X, \bar{X})$ is indicated in Fig. 3.15.

Example 3.5

We use the Ford–Fulkerson algorithm to generate a maximum flow in the net $G(V, E, c, f)$ of Fig. 3.16 from node s to node t. Initially, the flow f is set to zero. The algorithm proceeds as follows:

First iteration

Labeling routine. The routine finds an augmentation path

$$P_{st} = (s, 4)(4, 8)(8, 9)(9, t) \tag{3.62}$$

Augmentation routine. Since there is no reverse arc in P_{st}, the flow in each forward arc is increased by $w(t) = 2$.

The details of the first iteration are shown in Fig. 3.17.

Second iteration

Labeling routine. The routine finds an augmentation path

$$P_{st} = (s, 1)(1, 5)(5, 9)(9, t) \tag{3.63}$$

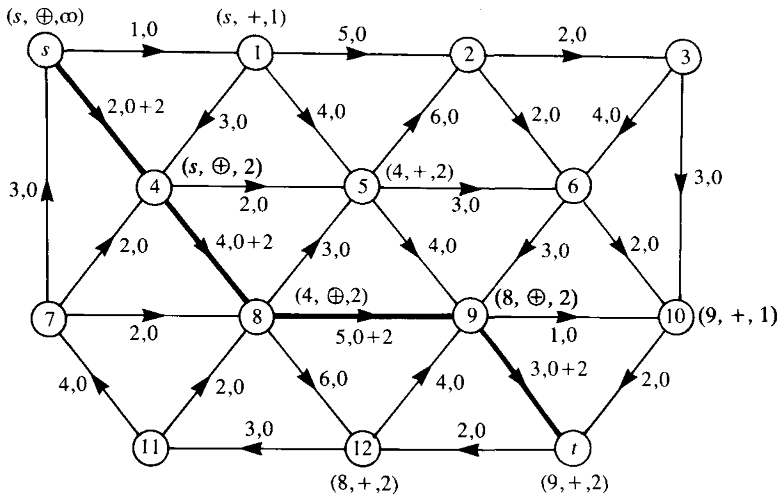

Fig. 3.17. The resulting net after the completion of the first iteration of the Ford–Fulkerson algorithm.

Augmentation routine. Since there is no reverse arc in P_{st}, the flow in each forward arc is increased by $w(t) = 1$.

The details of the second iteration are shown in Fig. 3.18.

Third iteration

Labeling routine. Node s is labeled by $(s, +, \infty)$, but no more labels can be assigned to other nodes. The program terminates.

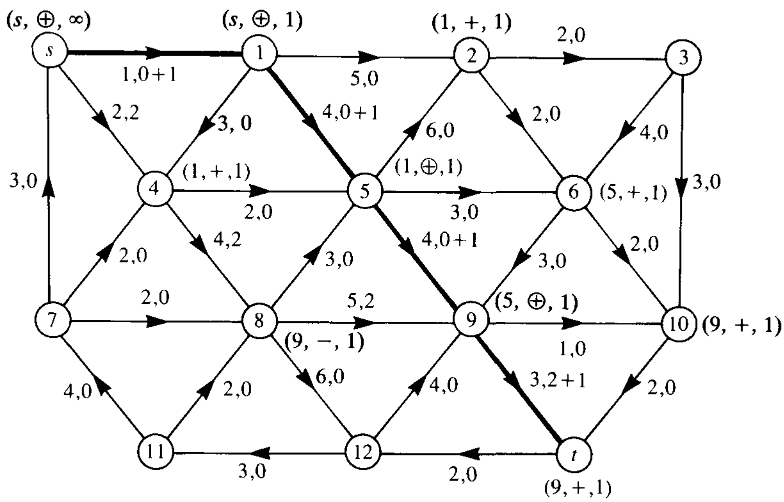

Fig. 3.18. The resulting net after the completion of the second iteration of the Ford–Fulkerson algorithm.

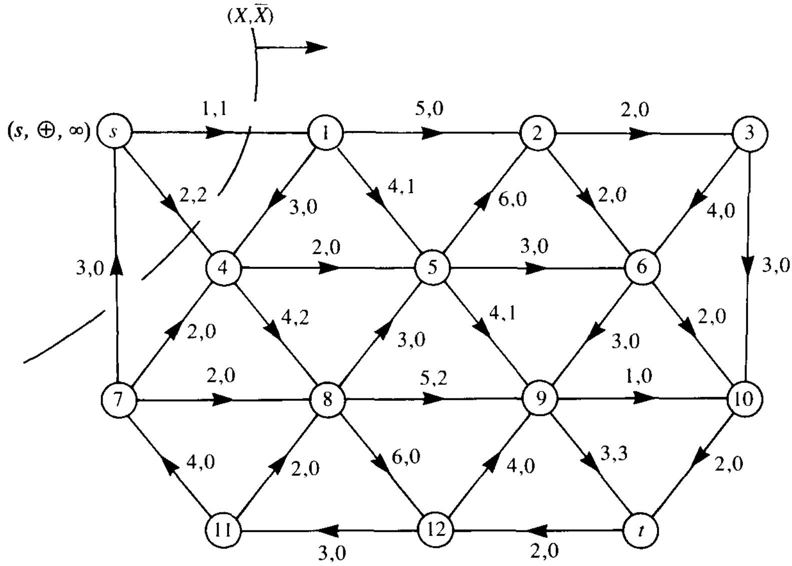

Fig. 3.19. The resulting maximum flow pattern after the application of the Ford–Fulkerson algorithm to the net of Fig. 3.16.

A maximum flow pattern is shown in Fig. 3.19. The desired minimum s–t cut $(X, \bar{X})$ shown in the figure is found to be

$$(X, \bar{X}) = (s, \bar{s}) = \{(s, 1), (s, 4)\} \tag{3.64}$$

whose capacity is $c(X, \bar{X}) = 3$.

3.4.1 Integrity Theorem

The Ford–Fulkerson algorithm is used not only to prove the max-flow min-cut theorem, but also to solve the maximum flow problem. The speed of the algorithm depends on the arc capacities in the net as well as on the numbers of nodes and arcs of the net. Indeed, in certain cases where the arc capacities are irrational, the algorithm might not converge at all. This will be elaborated in Section 3.4.2.

To ensure the termination of the algorithm, it will be necessary to assume that the capacity function c is integral valued. In the case of rational capacities for the arcs, they can always be converted to a problem with integral capacities by clearing fractions. Therefore, confining our attention to rational numbers is really no restriction for computational purposes. If the flow is integral and if the labeling routine identifies a flow augmenting path, then the maximum allowable flow change $w(t)$ of the augmentation routine, being the minimum of positive integers or the difference of two positive integers, is a positive integer, provided that the computation is initiated with an integral flow. For each flow augmenting path, the flow value is increased by at least one unit. Upon termination, a maximum integral flow has been generated. As a result, the Ford–Fulkerson algorithm must terminate in a finite number of iterations, provided of course that the capacity function c is integral valued. This leads to the following theorem.

THEOREM 3.4

(Integrity theorem) If the capacity function of a net is integral valued, then there exists a maximum integral valued flow in the net.

In the search for a flow augmenting path using the labeling routine, the first time we need to check at most $n-1$ nodes in an n-node net and the second time at most $n-2$ nodes, so there are at most $(n-1)+(n-2)+\ldots+1=n(n-1)/2$ nodes to be checked before a flow augmenting path is identified. If all arc capacities are integers, the maximum flow value f_{st} from s to t, being equal to the minimum s–t cut capacity, is also an integer. Since the flow value is increased by at least 1 unit per flow augmenting path, the Ford–Fulkerson algorithm requires at most $O(f_{st}n^2)$ steps. Since the flow value f_{st} is really an unknown quantity at the beginning, we do not have a bound in terms of the numbers of nodes and arcs of a net. In fact, it is possible to assign arc capacities to a very small net, so the algorithm will take a very long time to run. This will be shown by an example in the following section.

3.4.2 Irrational Arc Capacities

The Ford–Fulkerson algorithm is very popular because it is simple to understand and easy to implement. For many problems in practice, the algorithm terminates fairly quickly. However, in the case where the arc capacities are allowed to be irrational, it is possible to construct a net so that the algorithm not only will not terminate but also will converge to a wrong limit. In the following, we describe the steps of the algorithm for such a net due to Ford and Fulkerson (1962).

Before proceeding, we introduce a few terms that will be needed in the discussion. In a given net $G(V, E, c, f)$ with capacity function c and flow function f from s to t, the term $[c(x, y)-f(x, y)]$ is called the *residual capacity* of arc $(x, y) \in E$ with respect to f. Now consider the recursion

$$a_{k+2}=a_k-a_{k+1} \tag{3.65}$$

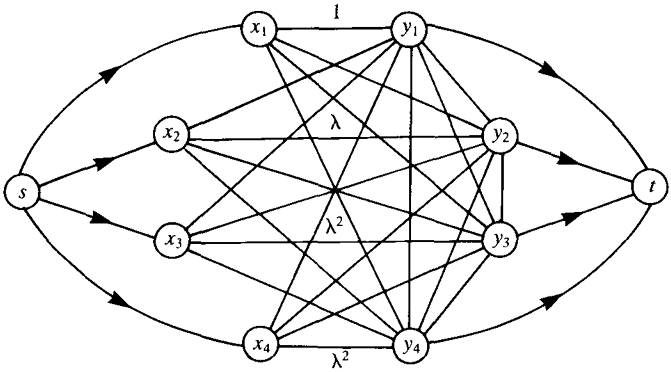

Fig. 3.20. The construction of a 10-node net, in which an undirected edge represents a pair of oppositely directed arcs, each having capacity equal to that of the undirected edge.

for all $k \geqq 0$, which has the solution

$$a_k = \lambda^k \tag{3.66}$$

where

$$\lambda = \frac{-1+\sqrt{5}}{2} < 1 \tag{3.67}$$

Thus, we have the following convergent series:

$$\sum_{k=0}^{\infty} \lambda^k = \frac{3+\sqrt{5}}{2} = S \tag{3.68}$$

We construct a 10-node net $G(V, E, c, f)$ as shown in Fig. 3.20, where an undirected edge represents a pair of oppositely directed arcs, each having capacity equal to that of the undirected edge. Four arcs (x_1, y_1), (x_2, y_2), (x_3, y_3) and (x_4, y_4) are distinguished, and are referred to as the *special arcs*. To every arc of E except the four special arcs we assign the capacity S. The special arcs are given the capacities

$$c(x_1, y_1) = 1 \tag{3.69a}$$

$$c(x_2, y_2) = \lambda \tag{3.69b}$$

$$c(x_3, y_3) = \lambda^2 \tag{3.69c}$$

$$c(x_4, y_4) = \lambda^2 \tag{3.69d}$$

For the first iteration using the labeling routine, we find the flow

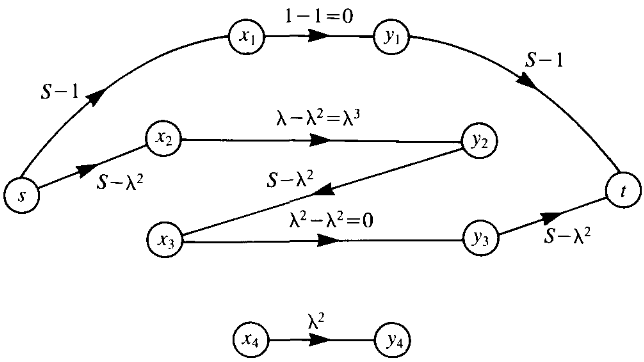

Fig. 3.21. A net showing the residual capacities of the arcs after augmenting the flow on the flow-augmenting path (3.70) by λ^2 units.

augmenting path $(s, x_1)(x_1, y_1)(y_1, t)$. We then increase the flow by 1 unit, the maximum permissible amount, along this path. The four special arcs will then have residual capacities 0, λ, λ^2, λ^2, respectively.

For the second iteration, we choose the flow augmenting path

$$P_{st1} = (s, x_2)(x_2, y_2)(y_2, x_3)(x_3, y_3)(y_3, t) \tag{3.70}$$

The only special arcs that are on this path are (x_2, y_2) and (x_3, y_3). Augmenting the flow on this path by λ^2, the maximum permissible amount, yields a flow pattern, the residual capacities of which are shown in Fig. 3.21. All other undisplayed arcs have residual capacity S. The residual capacities of the four special arcs now become 0, $\lambda - \lambda^2 = \lambda^3$, 0, λ^2, respectively, where we have used the fact that

$$\lambda^{k+2} = \lambda^k - \lambda^{k+1} \tag{3.71}$$

for all nonnegative integer k. Next, we choose the flow augmenting path

$$P_{st2} = (s, x_2)(x_2, y_2)(y_2, y_1)(x_1, y_1)(x_1, y_3)(x_3, y_3)(x_3, y_4)(y_4, t) \tag{3.72}$$

Observe that only the special arcs (x_1, y_1) and (x_3, y_3) are reverse arcs in this path, and all other arcs are forward arcs. Now augment the flow along this path by λ^3 units, again the maximum permissible amount. The residual capacities of the resulting arcs are shown in Fig. 3.22, where all the undisplayed arcs have residual capacity S except the arcs (y_2, x_3) and (y_3, t), which have residual capacity $S - \lambda^2$. The two augmentation steps together have increased the flow value by $\lambda^2 + \lambda^3 = \lambda$ units.

Observe that the residual capacities of the four special arcs are now changed to λ^3, 0, λ^3, λ^2. By relabeling the nodes of the special arcs, the special arcs have the residual capacities 0, λ^2, λ^3, λ^3, respectively.

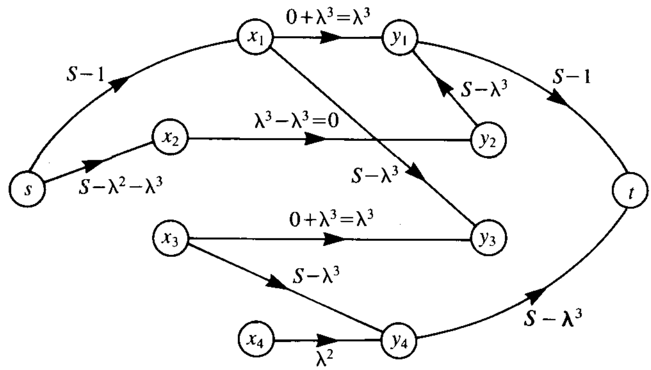

Fig. 3.22. A net showing the residual capacities of the arcs after augmenting the flow on the flow-augmenting path (3.72) by λ^3 units.

To complete our construction, we use the inductive process. Suppose that at the end of the kth iteration, the four special arcs (x_1', y_1'), (x_2', y_2'), (x_3', y_3'), (x_4', y_4'), after relabeling if necessary, have residual capacities

$$0,\ \lambda^k,\ \lambda^{k+1},\ \lambda^{k+1} \tag{3.73}$$

and the augmentation steps have increased the flow value by $\lambda^k + \lambda^{k+1} = \lambda^{k-1}$ units. We show that if the above procedure of using two flow augmentation steps is repeated, the special arcs of the resulting net will have residual capacities 0, λ^{k+1}, λ^{k+2}, λ^{k+2}, respectively. This is considered to be the $(k+1)$th iteration.

First, we choose the flow augmenting path

$$P'_{st1} = (s, x_2')(x_2', y_2')(y_2', x_3')(x_3', y_3')(y_3', t) \tag{3.74}$$

as shown in Fig. 3.23. Augment the flow along this path by the maximum amount of λ^{k+1} units. The resulting residual capacities of the four special arcs are as indicated in Fig. 3.23. Next, select the flow augmenting path

$$P'_{st2} = (s, x_2')(x_2', y_2')(y_2', y_1')(x_1', y_1')(x_1', y_3')(x_3', y_3')(x_3', y_4')(y_4', t) \tag{3.75}$$

As before, the special arcs (x_1', y_1') and (x_3', y_3') are reverse arcs in P'_{st2}, and all others are forward arcs. Augment the flow along this path by λ^{k+2} units. The residual capacities of the four special arcs in the resulting net are shown in Fig. 3.24, and are given by λ^{k+2}, 0, λ^{k+2}, λ^{k+1}, respectively. By relabeling the nodes of the special arcs, the residual capacities of the four special arcs at the end of the $(k+1)$th iteration are given by

$$0,\ \lambda^{k+1},\ \lambda^{k+2},\ \lambda^{k+2} \tag{3.76}$$

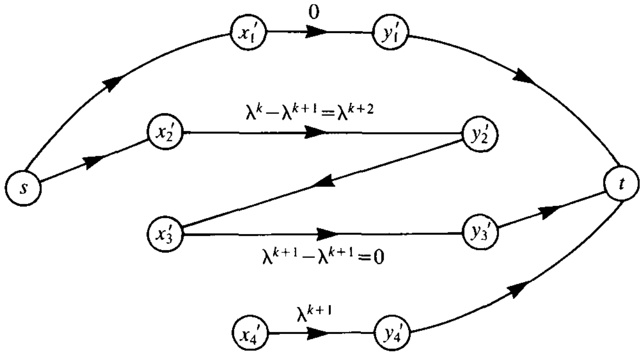

Fig. 3.23. A net showing the residual capacities of the four special arcs after augmenting the flow on the flow-augmenting path (3.74) by λ^{k+1} units.

The two augmentation steps together have increased the flow value by

$$\lambda^{k+1}+\lambda^{k+2}=\lambda^k \tag{3.77}$$

We now determine the flow value at the end of the $(k+1)$th iteration. Recall that at the end of the first iteration, the flow value is increased by 1 unit. At the end of the second iteration, the flow value is increased by λ units; at the end of the kth iteration, by λ^{k-1} units, and at the end of the $(k+1)$th iteration, by λ^k units. Therefore, the total flow value at the end of the $(k+1)$th iteration is given by

$$1+\lambda+\ldots+\lambda^{k-1}+\lambda^k=\sum_{j=0}^{k}\lambda^j \tag{3.78}$$

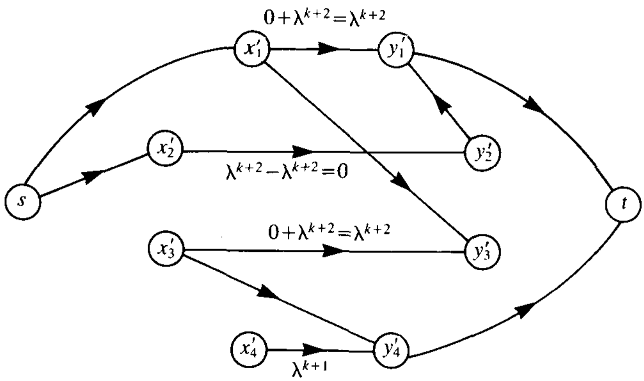

Fig. 3.24. A net showing the residual capacities of the four special arcs after augmenting the flow on the flow-augmenting path (3.75) by λ^{k+2} units.

showing from (3.68) that no non-special arc is ever required to carry more than

$$S = \sum_{u=0}^{\infty} \lambda^u \tag{3.79}$$

units of flow in repeating the inductive step. The process converges to a flow having value S, whereas the maximum flow value from s to t in the net of Fig. 3.20 is $4S$, being equal to the minimum s–t cut.

The conclusion is that if we allow irrational arc capacities, it is possible to construct a net with finite arc capacities, so that the Ford–Fulkerson algorithm will never terminate and the sequence of flow values will converge to a number that is not the maximum flow in the net. Because of this difficulty, many new algorithms have been proposed. They are guaranteed to terminate within a time bound that is independent of the arc capacities, and depends only on the numbers of nodes and arcs of the net.

We now turn our attention to the discussion of other algorithms. Many of these algorithms are based on the concept of the layered net, and are much more efficient than the Ford–Fulkerson algorithm.

3.5 LAYERED NETS

Given a net $G(V, E, c, f)$, we discuss how to induce a *layered net* $N(G)$ or simply N from G. An arc (i, j) of G is said to be *useful* from i to j if either (i, j) is not saturated, $f(i, j) < c(i, j)$, or $(j, i) \in E$ is not flowless, $f(j, i) > 0$. The layered net N will be constructed one layer at a time from the nodes of G, using the flow f as a guide. By definition, the source s is at the 0th layer. Then a node j is at layer 1 if there is a useful arc $(s, j) \in E$. We include arc (s, j) in N for each j and assign a capacity

$$c'(s, j) = c(s, j) - f(s, j) + f(j, s) \tag{3.80}$$

to (s, j). The set of all such j forms layer 1 nodes of N. To construct layer 2, we pick a node i in layer 1 and look for a useful arc $(i, u) \in E$, where u is not a node in layer 1 or $u \neq s$. We then include arc (i, u) in N for each i and u and assign a capacity

$$c'(i, u) = c(i, u) - f(i, u) + f(u, i) \tag{3.81}$$

to (i, u). The set of all such u forms layer 2 nodes of N. In general, a node y is in layer k if there is a useful arc $(x, y) \in E$, where x is a layer $k-1$ node and y does not belong to any of the previous layers $1, 2, \ldots, k-1$ or $y \neq s$. We then include arc (x, y) in N for each x and y and assign a capacity

$$c'(x, y) = c(x, y) - f(x, y) + f(y, x) \tag{3.82}$$

Note that the new capacity function c' represents the total residual or unused flow-carrying capacity of the arcs in both directions between their endpoints.

The layering process continues until we reach a layer where there is a useful arc (v, t) from some node v of this layer to the sink t, or else until no additional layers can be created by the above procedure. In the former case, we include arc (v, t) in N for each v and assign a capacity

$$c'(v, t) = c(v, t) - f(v, t) + f(t, v) \tag{3.83}$$

to (v, t), and the layering process is complete. In the latter case where no additional layers can be created but the sink has not been reached, the present flow function f in G is maximum and the maximum flow problem has been solved. Observe that not all the nodes of G need appear in N, and that in N all the arcs are directed from layer $k-1$ to layer k regardless of the direction of the useful arcs in G. In addition, even after an arc (x, y) is included in N, additional arcs (w, y) are included in N for all such layer $k-1$ nodes w. Finally, we remove all arcs that do not belong to any directed path from s to t in the resulting net. Fig. 3.25 shows the typical appearance of a layered net. In contrast to a general net, in a layered net N every directed path from the source s to some fixed node y in layer k contains exactly k arcs. These properties of layered nets are extremely useful in devising ways of finding their flows.

We summarize the above by presenting an algorithm for generating a layered net $N(V', E', c', f')$ from a given net $G(V, E, c, f)$, as follows: Let $U_i \subset V$ and $V_i \subset U_i$ for $i = 0, 1, 2, \ldots, k \leqq n-1$.

Step 1. $U_0 = \{s\}$. Set $i = 0$.

Step 2. $U_{i+1} = \{y \mid (x, y) \in E,\ x \in U_i,\ y \notin U_0 \cup U_1 \cup \cdots \cup U_i, \text{ and } c(x, y) - f(x, y) + f(y, x) > 0\}$.

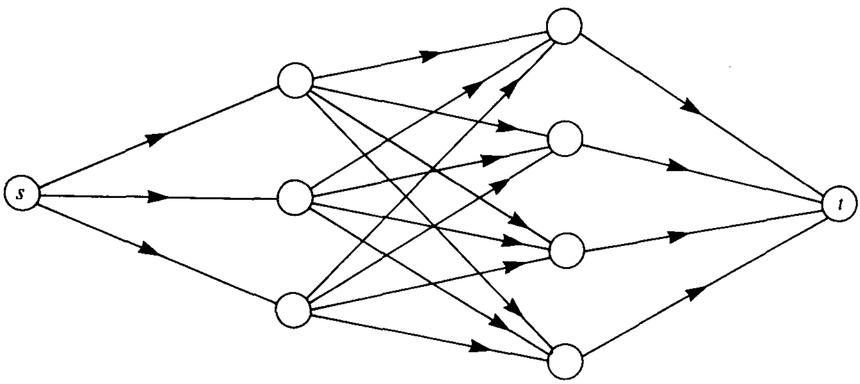

Fig. 3.25. The typical appearance of a layered net.