23

Network Flows

Author: Arthur M. Hobbs, Department of Mathematics, Texas A&M University.

Prerequisites: The prerequisites for this chapter are graphs and trees. See Sections 9.1 and 10.1 of *Discrete Mathematics and Its Applications*.

Introduction

In this chapter we solve three very different problems.

Example 1 Joe the plumber has made an interesting offer. He says he has lots of short pieces of varying gauges of copper pipe; they are nearly worthless to him, but for only 1/5 of the usual cost of installing a plumbing connection under your house, he will use a bunch of T- and Y-joints he picked up at a distress sale and these small pipes to build the network shown in Figure 1. He claims that it will deliver three gallons per minute at maximum flow. He has a good reputation, so you are sure the network he builds will not leak and will cost what he promises, but he is no mathematician. Will the network really deliver as much water per minute as he claims? □

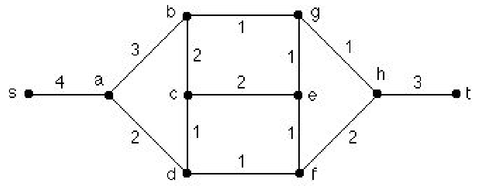


Figure 1. A plumber's nightmare.

Example 2 We want to block access to the sea from inland town s on river R. We can do this by dropping mines in the river, but because the river spreads out in a wide delta with several outlets, the number of mines required depends on where we drop them. The number of mines required in a channel ranges from a high of 20 mines in R to a low of 1 in some channels, as shown in Figure 2. In that figure, each channel is shown with a number indicating how many mines will block it. What is the smallest number of mines needed to block off s's access to the sea, and where should the mines be placed? □

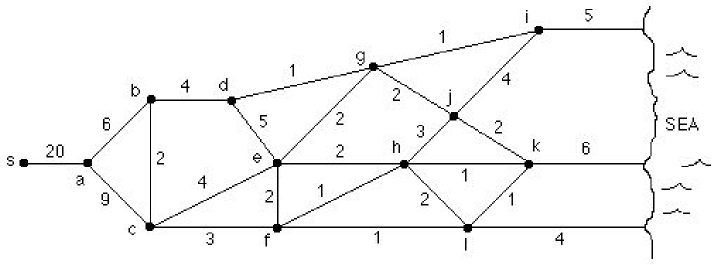


Figure 2. Delta system of river R and numbers of mines needed to close channels.

Example 3 At Major University, Professor Johnson is asked to hire graders for 100 sections spread among 30 different courses. Each grader may work for one, two, or three sections, with the upper bound being the grader's choice, but the number actually assigned being Professor Johnson's choice. Professor Johnson contacts the potential graders, learns from each both his choice of number of sections and which courses he is competent to grade, and makes a table showing this information, together with the number of sections of each course being offered. Because the real problem is too large to use as an example here, Table 1 gives a smaller example. How should the assignment of graders be made? □

In this chapter, we begin with Example 1, solve Example 2 on the way to solving Example 1, and then solve Example 3 by using the theory developed. Many more kinds of problems are solved using the methods of this chapter in the book *Flows in Networks* by Ford and Fulkerson [4].

	Course 1	Course 2	Course 3	Course 4	Max # Sec. Wanted
Student 1	yes	yes	no	no	3
Student 2	yes	no	yes	yes	2
Student 3	no	yes	yes	yes	3
# Sec. Needed	3	1	1	2	

Table 1. Graders and courses.

Flow Graphs

In Example 1, we have a network of pipes that can be modeled by a graph with weights on the edges. Here, the T- and Y-joints and the inlet and outlet are represented by vertices, the pipes are represented by edges, and the weight on each edge is the capacity of the corresponding pipe in gallons per minute. Moreover, in this example we have water flowing from vertex s to vertex t as labeled in Figure 1. For a general solution to the problem, let us use the following terminology.

Definition 1 Let G be a graph in which there are two designated vertices, one the *source* of all flow, and the other the *sink*, or recipient of all flow. At every other vertex, the amount of flow into the vertex equals the amount of flow out of the vertex. The flows are limited by weights, or *capacities*, on the edges. The edges may be undirected or directed. We designate the capacity of an edge e by $c(e)$. We will call a graph with capacities on the edges, a source s, and a sink t, a *capacitated* s,t*-graph*. □

Because we are searching for flows, we will show the flow through each edge of a capacitated s,t-graph as another number on the edge. To prevent confusion, we will designate the capacity of an edge and the amount of flow in it by a pair of numbers in parentheses on the edge, the capacity being the first number of the pair.

Example 4 Find a flow in the graph of Figure 3.

Solution: The path $p = s, b, a, t$ extends from s to t, and seen as a sequence of pipes, the largest amount of flow that could travel along it is the minimum of the capacities of the pipes comprising it. This minimum is 2, which is $c(s,b)$

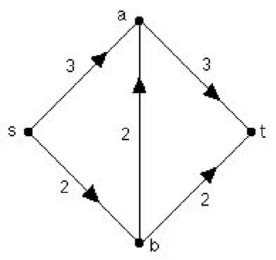


Figure 3. A small capacitated s,t-graph.

and also $c(b, a)$. Thus we put number pairs on each of the edges, the second entry being 2 for each edge in the path and 0 for the other two edges. The result is shown in Figure 4. □

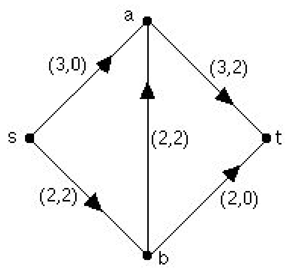


Figure 4. Graph of Figure 3 with flow along path s,b,a,t.

There are two ways we can view a flow, and Example 4 illustrates them both. One view is to trace out the path from the source to the sink of one or more units of flow. In the example, path p is such a path. The other view is to measure the total flow in each edge of the graph. This view is shown in the example by our placing the amount of flow along each edge. Since there is actually only one flow, namely the orderly procession of fluid from the source to the sink through the network, these two views must be equivalent.

When solving the problem of finding maximum flows through the graph, the second view is preferable for two reasons. If we are searching a very large network by hand, it may well be impossible for us to find a best set of paths from the source to the sink, especially after several paths for flow have already been found. Searching for such paths is very like searching through a maze, since flow in an edge limits, perhaps to 0, the additional flow that may pass through that edge. For the second reason, we need only realize that problems of this sort are likely to be programmed on a computer, and computers are much better at examining local situations than global ones.

However, using the first view, we can detect rules that must be satisfied by a flow described as in the second view. To state these rules easily, we need to define several terms.

Let A be a subset of V in directed graph $G = (V, E)$, and let $B = V - A$. Let $c(A, B)$ be the sum of the capacities of the edges directed in G from vertices in A to vertices in B, and let $c(B, A)$ be the sum of the capacities of the edges directed in G from vertices in B to vertices in A. Similarly, let $f(A, B)$ be the amount of flow from A to B, i.e., the sum of the flows in the edges directed from vertices in A to vertices in B. Let $f(B, A)$ be the amount of flow from B to A. Then the **net flow** $F(A)$ from A is defined by

$$F(A) = f(A, B) - f(B, A).$$

For example, in Figure 4, if $A = \{s, b\}$, then $f(A, B) = 2$ and $f(B, A) = 0$. Hence $F(A) = 2$. Similarly, $F(\{b\}) = 2 - 2 = 0$ and $F(\{s, t\}) = 2 - 2 = 0$, while $F(\{b, t\}) = 2 - 2 - 2 = -2$.

Note that $F(\{s\})$ is the total number of units of flow moving from the source to the sink in the graph. Our objective is to find a flow for which $F(\{s\})$ is maximum.

Since every unit of flow that enters a vertex other than the source or sink must go out of that vertex, we have the following theorem.

Theorem 1 Suppose G is a directed capacitated s, t-flow graph, and suppose $A \subseteq V(G)$.

1. If $s \in A$ and $t \notin A$, then $F(A) = F(\{s\})$.
2. If $t \in A$ and $s \notin A$, then $F(A) = -F(\{s\})$.
3. If $A \cap \{s, t\} = \emptyset$ or if $\{s, t\} \subseteq A$, then $F(A) = 0$.

Proof: These facts are evident because all of the material flowing goes out of the source, the material all goes into the sink, and none is lost or gained at any other vertex. ■

Although the definitions and theorem are given for directed graphs only, we can include undirected edges as follows. Although material can flow in either direction along an undirected edge, as a practical matter it will flow in only one direction (although we may not know in advance in which direction it will flow in a particular edge). Thus each undirected edge $e = \{a, b\}$ can be regarded as a pair of directed edges (a, b) and (b, a) joining the ends of e. Since the flow could go either way, we assign the capacity $c(e)$ of the edge e to both directed edges as $c(a, b)$ and $c(b, a)$. Further, using this device, we can designate the flow on the edge by $f(a, b)$ or $f(b, a)$ without ambiguity.

If, in the midst of an analysis, we discover we show flow going in both directions, it is clear that we can cancel the circulation in the two edges, leaving only that part of the flow which is actually passing through the pair of edges. For example, in Figure 5(a), we see a flow of 2 units from s to t, but $f(a, b) = 4$ while $f(b, a) = 2$. In Figure 5(b), the 2-unit circulation around the circuit a, b, a

has been eliminated (the 2 units along edge (b, a) have canceled 2 of the units along edge (a, b)) leaving the simpler, but equally accurate, picture of 2 units flowing from s to t through the path s, a, b, t. In general, if both (a, b) and (b, a) carry non-zero flow, we treat the combined flow by placing a flow of 0 on the smaller flow edge and replacing the flow on the other one by $|f(a, b) - f(b, a)|$.

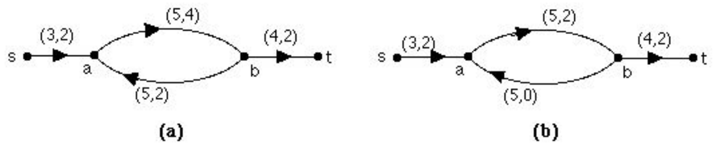


Figure 5. Flow graph illustrating cancellation of flow in a pair of oppositely directed edges.

Increasing the Flow

Example 5 Increase the flow in the graph of Figure 4.

Solution: In Figure 4 it is easy to see that the flow shown is not the largest possible. In fact, consider the path s, a, t. Since $c(s, a) = 3$ while $f(s, a) = 0$, we could get three more units of flow to vertex a by using the edge (s, a). Then one of those units could go on to t through (a, t) since $c(a, t) = 3$ while $f(a, t) = 2$, or $c(a, t) - f(a, t) = 1$. Thus we can get $\min(3, 3 - 2) = 1$ unit of flow through s, a, t, producing the flow shown in Figure 6(a). □

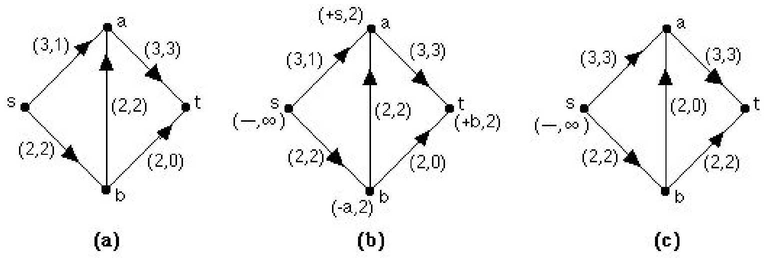


Figure 6. Steps in increasing flow.

In general, if we can find a sequence $p = s, x_1, x_2, \ldots, x_n, t$ from the source s to the sink t in which every directed edge is directed in the same direction as the sequence (so that the sequence describes a path) and in which every edge has capacity exceeding the flow already in the edge, then we can increase the flow from s to t by simply adding flow to the path p. In such a case, the total amount of flow that can be added cannot exceed the additional amount that can be forced through any one of the edges in p, so the total flow increase by using such a path p is

$$\min_{e \in E(p)} (c(e) - f(e)).$$

But what if no such sequence of vertices exists?

Example 6 Increase the flow in the graph of Figure 6(a).

Solution: It is not obvious that the flow in this graph can be increased. However, we note that we could get $c(s,a) - f(s,a) = 2$ more units of flow from s to a, and if we had 2 more available at b, we could move them on through (b,t) to t. But let us rearrange the flow. If we erase the two units of flow in edge (b,a), then the two flowing through (s,b) become available at b to go out through (b,t). Further, the two additional units available at a can replace the two that used to go through (b,a) to a, thus allowing the flow of 2 units through (a,t) to continue. Thus we arrive at the flow shown in Figure 6(c).

An intermediate step can be interposed between Figures 6(a) and 6(c). This step is shown in Figure 6(b), where the flow is the same as that in Figure 6(a). What is different is a record at each vertex of a possibility of increasing flow. We may suppose that an infinite supply is always available at s, so we have labeled s with $(-, \infty)$, where the "$-$" indicates only that all flow begins at s. At vertex a, we see the label $(+s, 2)$, which signifies that two units of flow could come through edge (s,a) because $c(s,a) - f(s,a) = 2$. At vertex b is the label $(-a, 2)$, showing that 2 units of flow could come to b by canceling the flow on edge (b,a). The operation of cancellation is shown by the "$-$" attached to a in the label. Finally, t has the label $(+b, 2)$, showing that 2 units of flow are available at t, coming from b. □

Let us look at the rearrangement of Example 6 another way. Consider the "path" s, a, b, t in Figure 6(b). We increased the flow on edges (s,a) and (b,t) by two units, and we decreased the flow on edge (b,a) by the same two units. These operations are signaled by the signs attached to the first label on each vertex as described in the example. The result was the increase of flow in the graph by two units.

Since s, a, b, t is not a path in Figure 6(a) (the edge (b,a) is directed against the order of the sequence), let us give such a sequence a name of its own.

Definition 2 A *chain* from x_0 to x_n in a directed graph G is a subgraph P of G whose vertices can be placed in a sequence $x_0, x_1, \ldots, x_n$ such that, for each $i \in \{0, 1, \ldots, n-1\}$, either $(x_i, x_{i+1}) \in E(P)$ or $(x_{i+1}, x_i) \in E(P)$ and no other edges are in $E(P)$. □

Figure 7 shows an example of a chain from x_0 to x_5.

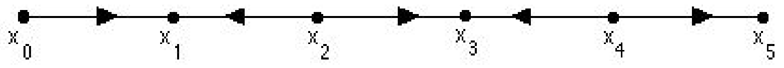


Figure 7. A chain from x_0 to x_5.

Example 6 illustrates the following general principle.

Theorem 2 Suppose P with vertex sequence $x_0, x_1, \ldots, x_n$ is a chain from the source $s = x_0$ to the sink $t = x_n$ in a capacitated s, t-graph G, and suppose, for each i, either $(x_i, x_{i+1}) \in E(P)$ and $c(x_i, x_{i+1}) - f(x_i, x_{i+1}) > 0$, or $(x_{i+1}, x_i) \in E(P)$ and $f(x_{i+1}, x_i) > 0$. Let x be the smallest among the values $c(x_i, x_{i+1}) - f(x_i, x_{i+1})$ on edges $(x_i, x_{i+1}) \in E(P)$ and $f(x_{i+1}, x_i)$ on edges $(x_{i+1}, x_i) \in E(P)$. Then increasing the flow by x on the edges (x_i, x_{i+1}) and decreasing it by x on the edges (x_{i+1}, x_i) of P increases $F(\{s\})$ by x. ■

We call a chain like that described in Theorem 2 an **augmenting chain** in the capacitated s, t-flow graph.

The labels on the vertices are used to find augmenting chains. Let us visualize ourselves as exploring the graph, starting at vertex s with infinitely many units of flow available to us. As we move through the graph, searching for t, we keep a record of the amounts of new flow that can reach each vertex. This record is the set of vertex labels we show in Figure 6(b).

The labels are governed by two considerations.

1. If x units of flow can reach vertex m, and if edge (m, n) exists, and if $c(m, n) - f(m, n) > 0$, then $y = \min(x, c(m, n) - f(m, n))$ units can be gotten to n by following the chain from s to m already found and then pushing as much flow as possible through (m, n). (This amount of flow is the smaller of the amount available and the amount that can go through the edge.) This is signaled by placing the label $(+m, y)$ at vertex n.

2. If x units of flow can reach vertex m, if edge (n, m) exists, and if $f(n, m) > 0$, then $y' = \min(x, f(n, m))$ units become available at n by canceling y' units of flow from edge (n, m). The effect of the cancellation is to feed the $y' \leq x$ units of flow needed at m after the cancellation by using y' of the x units of new flow available at m, while using the y' units that were flowing through (n, m) as the source of the new y' units available at n.

When we label vertex n by using a label on vertex m and either edge (m, n) or edge (n, m), we say we **label** n **from** m and that we are **labeling across** edge (m, n) or (n, m).

We interpret the labels backwards to describe the augmenting chain. The first label on each vertex names the vertex preceding it on the chain from s found during the labeling process. For example, consider the labels on the flow graph shown in Figure 8(a). There (ignoring signs) the first label on t is a, so the augmenting chain vertex sequence ends with a, t. The first label on a is b, so the augmenting chain vertex sequence ends with b, a, t. Finally, the first label on b is s, so the augmenting chain vertex sequence is s, b, a, t.

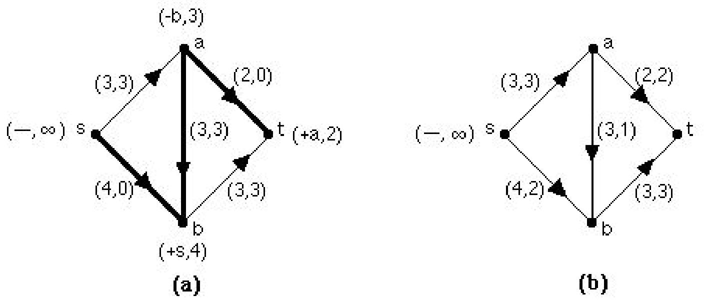


Figure 8. Showing an augmenting chain being found using labels and being used to increase flow.

When a vertex n is labeled from vertex m, the second label is always the minimum of the second label on m and the amount $c(m,n)-f(m,n)$ or $f(n,m)$ which can be gotten to n from m through the edge. Since the second label on s is ∞, the result is that the second label always gives the largest amount that can get to the vertex labeled. Thus when t is labeled, the second label of t states the increase of flow given by the augmenting chain specified by the labels. For example, in Figure 8(a) the augmenting chain has vertex sequence s, b, a, t, and the amounts of flow that can be forced through the edges of the chain are $4, 3$, and 2, successively, with a minimum of 2. Because of the way second labels are chosen, the second label on t is this amount 2.

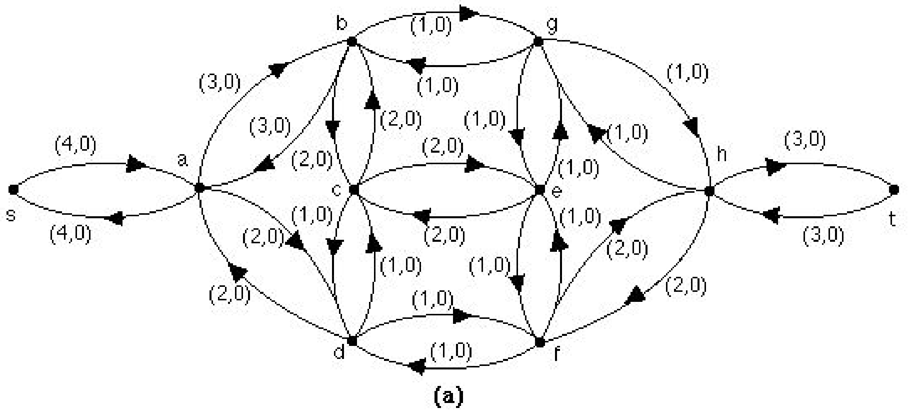
Example 7 Find an augmenting chain in the graph shown in Figure 8(a), in which there is a preexisting flow of 3 units.

Solution: We label s with $(-,\infty)$, and from there we label b with $(+s, 4)$. From b we can label a by canceling flow in (a, b), so a receives the label $(-b, 3)$. From a we can label t with label $(+a, 2)$. These labels are shown in Figure 8(a). Reading first labels backwards, we find the augmenting chain is s, b, a, t as discussed before, and the chain will increase the flow by 2 as the second label on t shows. The resulting flow is shown in Figure 8(b), where 3 units of flow go out of s to a and split there into 2 units that flow directly to t and one that goes to t through b. In addition, 2 units of flow go from s through b to t, making a total of 5 units of flow shown in Figure 8(b). □

While we are labeling we can keep track of what is possible, without concerning ourselves whether there is actually a chain available from s to t along which new flow can go. If we can label t, then the flow can be increased; we will show that if we cannot label t, then the flow cannot be increased.

We are also under no obligation to watch for a chain of labels. Since we always label an unlabeled vertex from a labeled vertex, s being always labeled with $(-, \infty)$, we automatically form a tree (disregarding the directions of edges). Thus for each labeled vertex x, our labels describe a unique chain from s to x. An example of such a tree is shown by boldface edges in Figure 8(a). As shown in that figure, if we mark on the graph the edges used in labeling vertices, we find the augmenting chain with vertices s, b, a, t shown in bold lines in Figure 8(a).

Example 8 Find a flow in the graph of Figure 1.



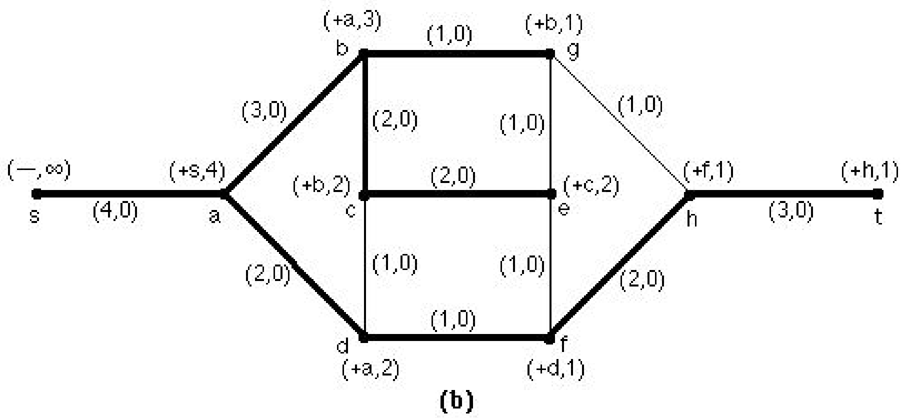


Figure 9. (a) Figure 1 converted to a directed capacitated s,t-graph. (b) The first labeling and the associated tree for Example 1.

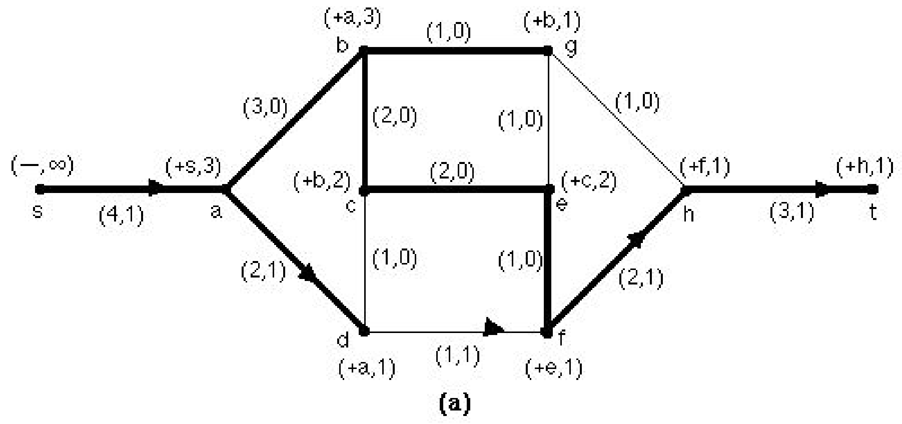
Solution: In Figure 9(a), the graph of Figure 1 is reproduced, except that each undirected edge has been replaced by two directed edges, one in each direction, each with the same capacity as the undirected edge. Also, the capacities have been written as the first entries of number pairs, with 0 for flow as the second number.* Starting with the label $(-, \infty)$ at s, we label across edges as described above. First a is labeled with $(+s, 4)$, and working from a we label b with $(+a, \min(4, 3)) = (+a, 3)$ and d with $(+a, \min(4, 2)) = (+a, 2)$, but we do not label s from a because s already has a label. Labeling from b, we do not need to label a, but we label c with $(+b, \min(3, 2)) = (+b, 2)$ and g with $(+b, \min(3, 1)) = (+b, 1)$. Then we label from c, placing $(+c, \min(2, 2)) = (+c, 2)$ on vertex e, but we place no label on d from c, since d already has a label. Next we label from d, placing label $(+d, \min(2, 1)) = (+d, 1)$ on vertex f. As before, we do not label a from d. Going on to vertex e, we label nothing from there, since all of its neighbors are already labeled. Next we label from f, placing label $(+f, \min(1, 2)) = (+f, 1)$ on vertex h. Since all neighbors of g are already labeled, we go on to h, from which we label t with $(+h, \min(1, 3)) = (+h, 1)$. Since t is labeled, we are done.

We use the labels to determine the chain from s to t that was found, along which a single unit of flow can flow (because the second label on t is 1). The first label on t is $+h$, showing that edge (h, t) is the last edge of the chain from s to t. The first label on h is $+f$, showing the next-to-the-last edge of the chain is (f, h). We continue reading backwards, using the first labels, obtaining the sequence "t, h, f, d, a, s" which reverses to give the chain s, a, d, f, h, t from s to t along which one unit of flow is added. The tree of edges used in our labeling is shown by bold lines in Figure 9(b), and Figure 10(a) shows the network with the flow we found. □

To describe a procedure that can be programmed, we need a rule for deciding which edge to label across next. Many such rules are possible; the one we have adopted here and used in Example 8 is straightforward: Label from the alphabetically earliest vertex x which has a label and which meets an edge that has not yet been considered from vertex x, and label all possible vertices from x before going on to the next vertex. This rule is called the **lexicographic ordering rule**.

Once we have increased the flow, the labels we placed on the vertices other than s are wrong for the new combination of graph and flow, so we erase the vertex labels and start again from s. Again we explore the graph, searching for a chain from s to t along which we can increase the flow.

* In Figures 9(b) through 12, to simplify the figures we have shown only the directed edges that contain flow when flow is present and the undirected edges when it is not.



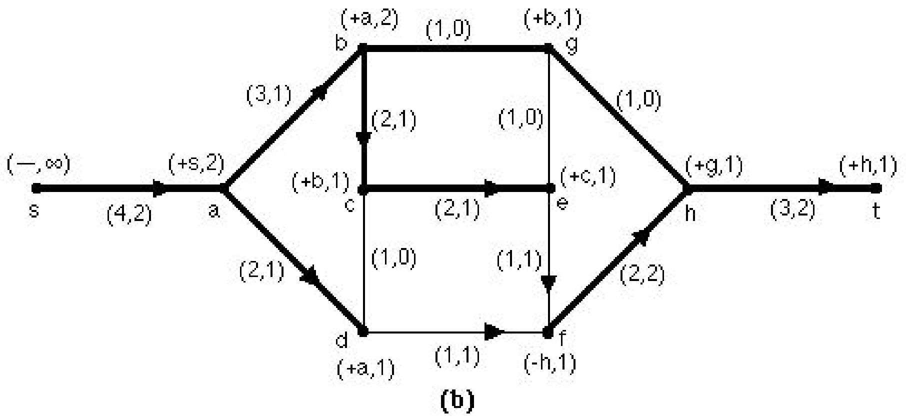


Figure 10. (a) Second labeling, associated tree, and first flow for Example 1. (b) Third labeling, associated tree, and second flow for Example 1.

Example 9 Increase the flow found in Example 8 as much as possible.

Solution: Labeling as before in the graph of Figure 10(a), we obtain the vertex labels shown in that figure. (Again the associated spanning tree is shown with bold lines.) Notice there that vertex f could not be labeled from d because $c(d,f) - f(d,f) = 1 - 1 = 0$. Hence f was labeled from e.

This time the augmenting chain is s, a, b, c, e, f, h, t, with one more unit of flow available. The resulting flow is shown in Figure 10(b). Again the labels are erased and new ones found. This time, since both $c(d,f) - f(d,f) = 0$ and $c(e,f) - f(e,f) = 1 - 1 = 0$, f has no label when labeling from e is completed. Hence we do not try then to label from f, but rather label from g. Later f

does receive a label; indeed, its label shows that flow should be canceled to reach f. In a more complex graph, it would then be reasonable to label from f. In this graph, however, we reach t without using f again. The tree of edges used is shown in Figure 10(b) with bold edges, and the augmenting chain is s, a, b, g, h, t.

The resulting flow is shown in Figure 11. We label again, as shown in Figure 11, and we obtain the tree of edges used, again shown there. However, the tree does not contain t, so we have not found an augmenting chain. Does this mean there is no augmenting chain? □

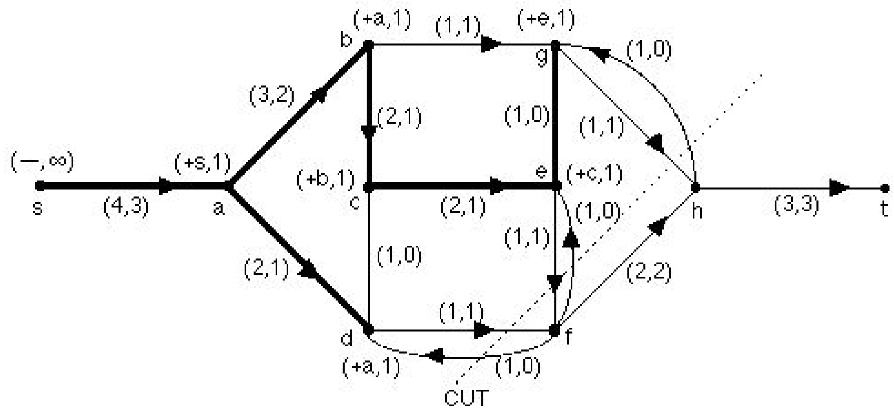


Figure 11. Fourth labeling, associated tree, and a minimum cut.

For the answer to that question, we must bring up machinery appropriate to Example 2.

Definition 3 Given a directed graph G with capacities on the edges, and given a nonempty subset A of its vertices such that $V(G) - A$ is also nonempty, the set of all edges of G directed from vertices in A toward vertices in $V(G) - A$ is called the *cut* $(A, V(G) - A)$. The *capacity* $c(A, V(G) - A)$ of this cut is the sum of the capacities of the edges in it. If $s \in A$ and $t \in V(G) - A$, then we call the cut an s, t-*cut*. □

Our objective in Example 2 is to find a cut $(A, V(G) - A)$ such that $c(A, V(G) - A)$ is minimum. In Theorem 3, we begin to tie the maximum flow and the minimum cut together, and in Theorem 4 we will complete the connection.

Example 10 Discuss the cuts in the graph of Figure 11.

Solution: In Figure 11, the set of vertices in the tree is $A = \{s, a, b, c, d, e, g\}$. The set of edges directed from A to $V(G)-A = \{f, h, t\}$ is the cut $\{(d, f), (e, f), (g, h)\}$. Notice that on each of these edges the flow equals the capacity and that on the return edges (f, d), (f, e), (h, g) in $(V(G) - A, A)$ the flow is zero. The capacity of the cut is $c(d, f) + c(e, f) + c(g, h) = 1 + 1 + 1 = 3$. Notice also that this cut is an s, t-cut, and that the capacity of the cut is exactly the same as the total flow $F(A) = F(\{s\}) = 3$. □

Theorem 3 The maximum flow from vertex s to vertex $t \neq s$ in a directed graph G with capacities on its edges is less than or equal to the capacity of any cut $(A, V(G) - A)$ having $s \in A$ and $t \notin A$.

Proof: Consider any s, t-cut $(A, V(G) - A)$ of graph G. Since any units of flow from s to t must pass through an edge of this cut, it follows immediately that $F(\{s\}) \leq c(A, V(G) - A)$. ■

In Example 10, several facts are evident. First, since s has a label and t does not, any flow from s to t must pass through one or another of the three edges in the cut $(A, V(G)-A)$. Second, since the return edges in $(V(G)-A, A)$ are empty, any flow passing through the three edges directed away from A must continue on toward t. Third, with the edges directed away from A full and the edge directed toward A empty, there is no conceivable way that the flow we have found could be increased. In other words, we have found a maximum flow in this graph from s to t, and its amount is equal to the sum of the capacities of the edges in this cut separating s from t.

Returning to the graph shown in Figure 6(c), we see another example of this situation. Vertex s is labeled $(-, \infty)$ as usual. Since $f(s, a) = c(s, a)$ and $f(s, b) = c(s, b)$, the tree of labeled vertices includes only vertex s, so $F(\{s\}) = c(\{s\}, \{a, b, t\}) = 5$.

Notice that there are no edges directed toward $\{s\}$, and both of the edges directed out of $\{s\}$ in Figure 6(c) are full. Thus, it is clear that the flow to vertex t cannot be increased from its value of 5 units shown in Figure 6(c), and the maximum flow is equal to the sum of the capacities of the edges in the set $\{(s, a), (s, b)\}$. (We will generalize these observations in Theorem 4, following Algorithm 1.)

The observations made in considering Examples 9 and 10 motivate Algorithm 1.

ALGORITHM 1 Max-flow min-cut algorithm.

```
procedure Max-flow min-cut($G$ := directed graph with source
    $s$, sink $t$, and capacities on all edges)
label $s$ with $(-, \infty)$
$stop := 0$
while $stop = 0$
begin
  while $t$ has no label
  begin
    $m$ := labeled vertex not previously examined, chosen by
      using an ordering rule
    $x$ := the second label on $m$
    $e$ := edge incident with $m$ and not previously examined
      while at $m$
    if $e = (m, n)$ and if $n$ is not labeled and if $c(e) - f(e) > 0$
      then label $n$ with $(+m, y)$ where $y = \min(c(e) - f(e), x)$
    if $e = (n, m)$ and if $n$ is not labeled and if $f(e) > 0$ then
      label $n$ with $(-m, y')$ where $y' = \min(f(e), x)$
    if no such vertex $m$ and edge $e$ exist then $stop := 1$
  end {vertex $t$ has been labeled}
  if $stop = 0$ then $p := x_0, x_1, \ldots, x_n$, an augmenting chain
    with $s = x_0$ and $x_n = t$
  {$p$ is found using the labels as described in the text}
  if $stop = 0$ then $v$ := the value of the second label on $t$
  $i := 0$
  while $i < n$ and $stop = 0$
  begin
    if the first label on $x_{i+1}$ is $+x_i$ then replace the second
      label $r$ on edge $(x_i, x_{i+1})$ with $r + v$
    if the first label on $x_{i+1}$ is $-x_i$ then replace the second
      label $r'$ on edge $(x_{i+1}, x_i)$ with $r' - v$
    {$r' \geq v$ by the use of minimum as we progress toward $t$}
    $i := i + 1$
  end
  erase all vertex labels except the one on $s$
end {No more labeling is possible. By Theorem 4, the cut
from the set of labeled vertices to the set of unlabeled vertices
is a minimum cut and the flow $F(\{s\})$ is a maximum flow.}
```

Example 11 Solve the problem of Example 2.

Solution: Represent the river system as a graph by assigning a vertex to the town s, to the ocean t, and to each of the intersections of the channels. Join two vertices with an undirected edge if a river channel joins the corresponding points in the river system, and give the edge a capacity equal to the number of mines it takes to block the corresponding channel. The resulting graph is shown in Figure 12. Applying Algorithm 1 with the lexicographic ordering (see Exercise 7), we obtain a maximum flow of 7 with a minimum cut consisting of edges (d, g), (e, g), (e, h), (f, h), and (f, l). Thus the solution to the problem of Example 2 is to mine the channels corresponding to these five edges with a total of 7 mines. □

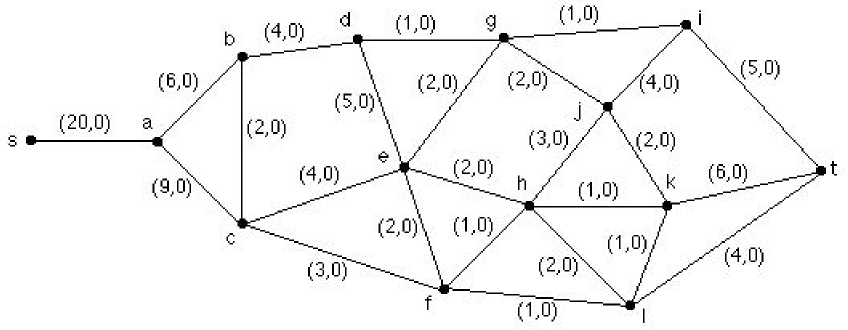


Figure 12. The river delta of Example 2 converted to a capacitated s,t-graph.

Examples 9 and 11 point the way to the following theorem, which shows that Algorithm 1 solves our problem.

Theorem 4 The Max-flow Min-cut Theorem The maximum s, t-flow in a capacitated s, t-graph is equal to the capacity of a minimum s, t-cut. Further, Algorithm 1 finds such a maximum flow and minimum cut.

Proof: We already know from Theorem 3 that the maximum flow amount

$$F(\{s\}) \leq c(A', V(G) - A')$$

for any minimum capacity s, t-cut $(A', V(G) - A')$. It will therefore suffice to find a flow f and an s, t-cut $(A, V(G) - A)$ such that $F(A) = c(A, V(G) - A)$ for the flow f.

Let Algorithm 1 run to completion, producing a flow f. Let A be the set of vertices labeled in the last pass through the algorithm. Since s always has a label, and since t could not be labeled on the last pass, $(A, V(G) - A)$ is an s, t-cut, and $F(A)$ is the amount of flow going from s to t as described by f.

Consider an edge (a, b) with $a \in A$ and $b \in V(G) - A$. Since a is labeled and b is not, we must have $f(a, b) = c(a, b)$ by the algorithm. Next, consider an edge (b', a') with $a' \in A$ and $b' \in V(G) - A$. Again, we note that a' has a label and b' does not; hence by the algorithm, $f(b', a') = 0$. Thus

$$F(A) = c(A, V(G) - A). \tag{1}$$

Since the maximum flow cannot be larger than $c(A, V(G) - A)$, $F(A)$ must be a maximum flow. The equality in (1) completes the proof of this theorem. ■

Complexity

Given a capacitated s, t-graph G, let a be the largest edge capacity, let e be the number of edges of G, and let v be the number of vertices of G. In each search for an augmenting chain in the first half of Algorithm 1, we may have to examine nearly all of the edges, each from both ends, so each pass through that half of the algorithm takes $O(e)$ steps. Since each augmenting chain increases the flow by at least one unit, there can be no more than $a + 1$ searches for an augmenting chain. The second half of Algorithm 1 requires only $O(p)$ steps, where p is the number of edges in the augmenting chain. Thus the complexity of Algorithm is governed by the first half of the algorithm and is $O(ae)$.

Sometimes the capacity a is many orders of magnitude larger than the number e of edges or the number v of vertices. In such a case, we would like a measure of the complexity of the algorithm that does not depend on a. Edmonds and Karp [2] have shown that, if vertices are scanned in the same order in which they receive labels, instead of using the lexicographic ordering, then the complexity is $O(v^5)$.

Assignment of Graders

Example 12 Set up Example 3 as a flow graph problem.

Solution: Referring to the table given in Example 3, represent Students $1, 2,$ and 3 by vertices S_1, S_2, and S_3, and represent Courses $1, 2, 3$, and 4 by vertices C_1, C_2, C_3, and C_4, respectively. Let s and t be two additional vertices. Join vertex s to S_1, S_2, and S_3 by directed edges, and assign capacity $c(s, S_i)$ equal to the number of sections Student i is willing to grade. Join each of vertices C_1, C_2, C_3, and C_4 to vertex t by directed edges, and assign capacity $c(C_i, t)$ equal to the number of sections of Course i being offered. For each i and j, add edge (S_i, C_j) if Student i is qualified to grade for Course j (i.e., if a "yes" appears in the row for Student i and the column for Course j of Table 1). For each such edge, let $c(S_i, C_j) = \infty$. The problem then becomes one of finding a maximum flow in the graph of Figure 13. $\square$

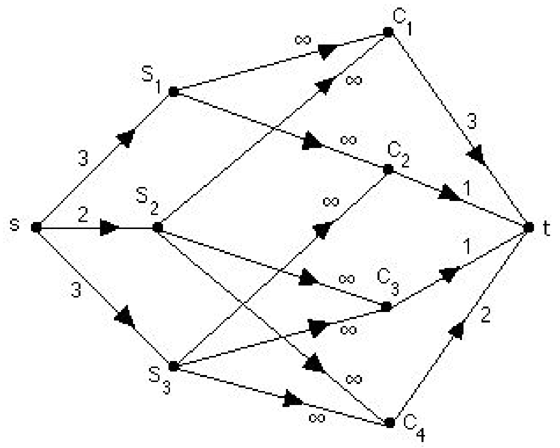


Figure 13. The flow graph for the grader assignment problem (Example 3).

By Theorem 4, the maximum flow in Figure 13 equals the capacity of the minimum cut. Let us look at a maximum flow as perceived under View 1 discussed early in this chapter. Each unit of flow passes from s to a vertex S_i, thence to a vertex C_j, and finally to vertex t. We may regard the unit as specifying an assignment of Student i to a section of Course j. For a fixed value of i, the number of units of flow passing through S_i cannot exceed $c(s, S_i)$, which is the maximum number of sections for which Student i wants to grade, so the assignment view of the units of flow cannot assign a student to more sections than he desires. Similarly, for a fixed value of j, the number of units passing through C_j cannot exceed $c(C_j, t)$, which is the number of sections needing graders. Thus no course will be assigned too many graders. Ideally, $(V(G) - \{t\}, \{t\})$ will turn out to be a minimum cut. Then by Theorem 4, the maximum flow will equal to capacity of that cut, and every section of every course will be assigned a grader.

Example 13 Solve Example 3.

Solution: Applying Algorithm 1 with lexicographic ordering to the graph of Figure 13, we find the flow shown in Figure 14, where the last labels and the associated tree (in bold edges) are also shown. From Figure 14, we see that $F(\{s\}) = c(V(G) - \{t\}, \{t\})$ as we hoped, so every section will get a grader. Further, interpreting the flow, we are told to assign Student 1 to all three sections of Course 1, Student 2 to the one section of Course 3 and one of the sections of Course 4, and Student 3 to the one section of Course 2 and to the other section of Course 4. □

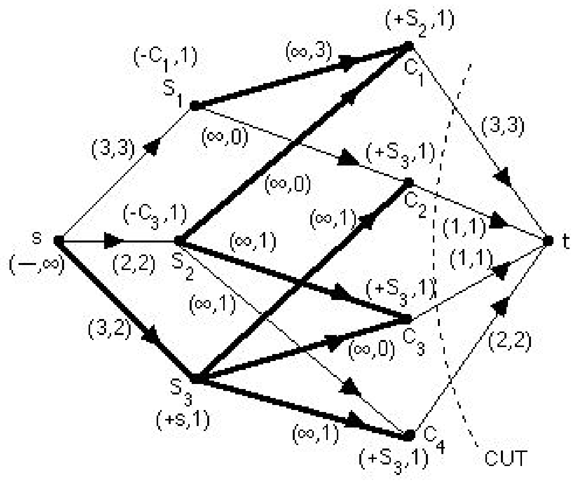


Figure 14. Maximum flow, associated tree, and cut for the grader assignment problem (Example 3).

The generalization of Examples 3, 12, and 13 to other assignments is straightforward. The possibilities in such assignments are that the students become fully assigned without finding graders for every section, or that, as in the case of Example 3, the sections receive enough graders but some students do not work as much as they want, or that every student gets enough work and every section receives a grader, or that some students do not get enough work while some sections are not graded. The last case is illustrated in one of the examples of Exercise 8. In every case, a maximum flow obtained by using Algorithm 1 will determine an assignment of as many graders to as many sections as possible.

Historical Note

The 1950s were exciting years at the RAND Corporation, which had been set up after World War II to provide the government with well-founded scientific advice. Among the workers there were Lester Ford, Jr. and Ray Fulkerson, developers of the theory presented in this chapter, George Dantzig [1], one of the most important developers of linear programming, and Merrill Flood and Selmer Johnson, who worked with Fulkerson and Dantzig on the traveling salesman problem (see the "Traveling Salesman Problem" chapter in this book).

The problems which led to the theory that has been presented in this chapter were posed by the Air Force to Ford and Fulkerson in 1955 [4] as a problem in railway traffic flow. Their first work on it involved linear programming, but as they refined their understanding of the problem and available methods, they hit on the flow algorithm and labeling scheme we have presented here. Their development of the work was so fast that the entire theory of this chapter was published by 1957 [3].

Suggested Readings

1. G. Dantzig, *Linear Programming and Extensions*, Princeton University Press, Princeton, N. J., 1998.

2. J. Edmonds and R. Karp, "Theoretical improvements in algorithmic efficiency for network flow problems", *J. ACM*, Vol. 19, 1972, pp. 248–264.

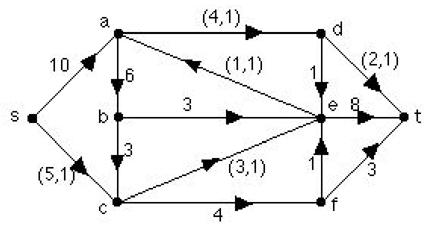
3. L. Ford, Jr. and D. Fulkerson, "A simple algorithm for finding maximal network flows and an application to the Hitchcock problem", *Canadian J. Math.*, Vol. 9, 1957, pp. 210–218.

4. L. Ford, Jr., and D. Fulkerson, *Flows in Networks*, Princeton University Press, Princeton, N. J., 1962.

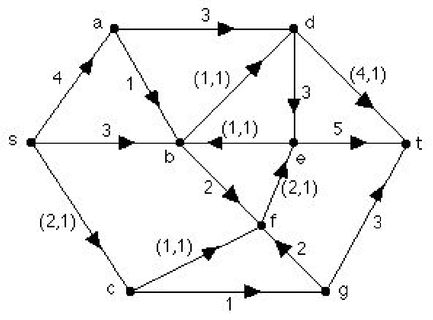
Exercises

In Exercises 1 and 2, one unit of flow is shown in the graph. Starting with that flow, use Algorithm 1 with the lexicographic ordering to find a maximum flow and a minimum cut in the graph.

1.



2.

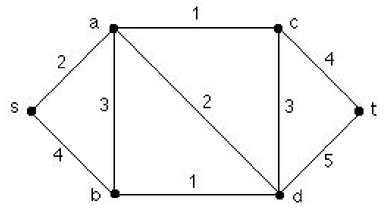


3. Using the graph and capacities of Exercise 1, find a maximum flow and minimum cut using Algorithm 1, but this time use the Edmonds and Karp ordering in which vertices are scanned in the same order in which they receive labels, instead of using the lexicographic ordering.

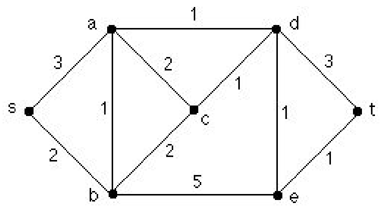
4. Using the graph and capacities of Exercise 2, find a maximum flow and minimum cut using Algorithm 1, but this time use the Edmonds and Karp ordering in which vertices are scanned in the same order in which they receive labels, instead of using the lexicographic ordering.

In Exercises 5 and 6, find a maximum flow and a minimum cut in the undirected graph by using Algorithm 1 with the lexicographic ordering.

5.



6.



7. Apply Algorithm 1 with the lexicographic ordering to the graph shown in Figure 12.

8. For each of the following tables, find a maximal assignment of graders to sections of courses, as in Example 3.

(a)

	Course 1	**Course 2**	**Course 3**	**Max. # Sec. Wanted**
Student 1	yes	no	no	1
Student 2	yes	yes	no	1
Student 3	no	yes	yes	3
Student 4	yes	no	yes	1
# Sec. Needed	4	1	1	

(b)

	Course 1	**Course 2**	**Course 3**	**Course 4**	**Max. # Sec. Wanted**
Student 1	yes	yes	yes	yes	1
Student 2	yes	yes	no	no	2
Student 3	no	yes	yes	no	1
Student 4	no	yes	yes	yes	3
# Sec. Needed	2	2	2	1	

⋆⋆**9.** An undirected graph G is **bipartite** if $V(G)$ is the disjoint union of two nonempty sets V_1 and V_2 such that every edge of G joins a vertex of V_1 with a vertex of V_2. A **matching** in graph G is a set M of edges of G such that no two of the edges in M meet the same vertex of G. A **covering** of graph G is a set C of vertices of G such that every edge of G has at least one end in C. Prove König's Theorem: If G is an undirected bipartite graph, then the maximum number of edges possible in a matching in G equals the minimum number of vertices possible in a covering of G. *Hint*: use a source and sink connected to different subsets of $V(G)$ by edges of capacity 1.

⋆⋆**10.** (This problem requires information from the beginning of the chapter "Network Survivability".) Recall from that chapter that a block is a graph without cut vertices and that a graph is 2-connected if it is a block with at least two edges. Show that an undirected graph G with at least three vertices is 2-connected if and only if, for any two distinct vertices v and w of G, there are two simple paths joining v and w which have only the vertices v and w in common. Note: This is the 2- connected case of Menger's theorem. *Hint*: Replace each vertex x of G other than v and w by two new vertices x_1 and x_2 and a directed edge (x_1, x_2). For each edge $\{x, y\}$ of G, add edges (x_2, y_1) and (y_2, x_1), treating v and w suitably. Assign appropriate capacities and use Algorithm 1 and Theorem 4.

11. In the new factory of the Sampson Manufacturing Company, the workers are to be assigned to machines. One worker will use just one machine on the job. Each worker has stated in his job application which types of machines he is competent to operate, and the company knows how many of each type of machine they have to be manned. Describe how the problem of making the assignments can be solved by using a flow graph.

12. The Computer Information Company (CIC) sells information to computer owners who call in by modem. Due to the quality of their service, their telephone lines are very busy. But it turns out that their customers are not always able to get through to them, even when the company has idle lines coming in. It is obvious that the problem lies with the telephone network, but where is the bottleneck? Investigating the problem, the company finds that any call coming out of a switching center owned by the telephone company can reach the next switching center; the problem seems to be that some switching centers do not have enough capacity to handle all of the calls for CIC that come to them. Data from the telephone company tells CIC how many calls are coming into each switching center from local users of CIC and what the pass-through capacity of each switching center is. Many calls pass through several switching centers on their way to the CIC office. How can CIC determine which switching centers are blocking their calls (thus helping CIC management decide where they might build a subsidiary CIC station to reduce the pass-through load on the inadequate switches)? *Hint*: Try a flow from CIC to the users, connect a single sink to the switches with edges whose capacities are the number of local users attached to the switches, and replace switches in the telephone network by weighted directed edges.

Computer Projects

1. Write a computer program to find a maximum flow and a minimum cut in a directed capacitated s, t-graph.

2. Write a computer program to find a maximum matching in a bipartite graph. (See Exercise 9 and its solution for the necessary ideas and definitions.)