Suppose an oil company has a pipeline network as shown in Figure 1, where the arc labels indicate the maximum number of barrels that can be pumped through the pipeline per hour. The company wants to know the maximum number of barrels they can pump per hour from node $s$ to node $t$. This is an instance of the maximum flow problem, a fundamental optimisation problem that comes up over and over again in various applications ranging from engineering to supply chain management, but also in pure mathematics. In particular, understanding this problem is incredibly useful in a lot of situations where it is not at all obvious that anything is flowing through a network. In order to study the problem a bit more closely, let’s start by being a bit more precise with the basic concepts.

**Definition 1.** A flow network consists of a set $V$ of nodes together with a set $A$ of arcs, where an arc $a$ is just a pair $a = (u, v)$ of nodes, and we say that arc $a$ points from node $u$ to node $v$. In addition, there are two special nodes: a source node $s$ and a target node $t$, and each arc $a$ comes with a positive number $c_a$, called its capacity.

For the network shown in Figure 1 $V = \{s, t, 1, 2, 3\}$, $A = \{(s, 1), (1, 2), (2, t), (1, 3), (3, t), (1, 2)\}$, and, for instance the capacity of arc $(1, 2)$ is $c_{(1, 2)} = 3$.

**Definition 2.** For a given flow network, a feasible flow is an assignment of numbers $x_a$ to arc $a$, such that the following conditions are satisfied:

**Capacity constraints:** $0 \leq x_a \leq c_a$ for every arc $a \in A$, and

**Flow conservation:** For every node $i \not\in \{s, t\}$, the total flow into node $i$ equals the total flow out of node $i$.

Figure 2 illustrates a feasible flow for the network in Figure 1. The numbers in brackets are the capacities $c_a$, and the numbers in front of the brackets are the flows $x_a$. It is easy to check the capacity constraints: the number in front of the brackets is never larger than the number in brackets. For the flow conservation constraint, we can check the three nodes 1, 2 and 3:

- node 1: in-flow $x_{s1} = 2$, out-flow $x_{12} + x_{13} = 1 + 1 = 2$
- node 2: in-flow $x_{s2} + x_{12} = 0 + 1 = 1$, out-flow $x_{2t} = 1$
- node 3: in-flow $x_{13} = 1$, out-flow $x_{3t} = 1$

Note that in this example the total flow out of node $s$ equals the total flow into node $t$. This is no coincidence. A good way to see this is through the concept of a cut in the network.

E-mail address: tkalinow@une.edu.au.
Definition 3. A cut in a flow network is a node set $W \subseteq V$ with $s \in W$ and $t \notin W$. The capacity of a cut $W$ is the sum of the capacities of all arcs “leaving $W$”, that is, all arcs $(i, j)$ with $i \in W$ and $j \notin W$.

In Figure 3 this is illustrated for the cuts $W_1 = \{s, 1, 2\}$ and $W_2 = \{s, 2\}$ with capacities $c_{2t} + c_{13} = 2 + 4 = 6$ and $c_{s1} + c_{2t} = 2 + 2 = 4$, respectively. For a cut $W$ and a feasible flow $x = (x_a)_{a \in A}$, let $f(W, x)$ be the net flow out of $W$, that is the sum of the values $x_a$ over all arcs $a = (i, j)$ with $i \in W$ and $j \notin W$ minus the sum of the flows on the arcs $a = (i, j)$ with $i \notin W$ and $j \in W$. For instance, for the flow $x$ illustrated in Figure 2, and the cuts shown in Figure 3, we obtain

\[
f(W_1, x) = x_{2t} + x_{13} = 1 + 1 = 2,
\]

\[
f(W_2, x) = x_{s1} + x_{2t} - x_{12} = 2 + 1 - 1 = 2.
\]

It is not a coincidence that these two values are equal. In fact, it is not hard to see that $f(W, x)$ actually does not depend on the cut $W$, but only on the flow $x$.

Observation. If $x$ is a flow, and $W$ is a cut, then $f(W, x) = f(\{s\}, x)$. This number is called the value of the flow $x$.

A good way to verify this is by starting with $\{s\}$ and then adding the nodes of $W$ step by step. At the point when node $i$ is added to $W$, we gain the total flow from node $i$ to the complement of the current $W$, and we lose the total flow from nodes in the current $W$ into node $i$. It follows from the flow conservation condition for node $i$ that these two contributions cancel, and the addition of node $i$ does not change the value. Applying the above observation to the cuts $\{s\}$, and $V \setminus \{t\}$, we see that the net flow out of node $s$ is equal to the net flow into node $t$.

Given a flow network, the maximum flow problem asks for a feasible flow of maximum value. For instance, the flow $x$ illustrated in Figure 2 has value 2, and the oil company might be interested if they can do better.

Note that for every cut $W$, the value of a feasible flow is the flow out of $W$ minus the flow into $W$, and since the flow out of $W$ cannot exceed the capacity of $W$ (which is just the sum of the capacities of all arcs out of $W$), the capacity of every cut provides an upper bound for the maximum flow value. For our example network it is not hard to list all cuts and check their capacities (see Table 1). From the fact that there is a cut of capacity 3, we know that we cannot hope for a flow of value greater than 3, and in our example we can actually find a flow of value 3 by just looking at the picture (see Figure 4). We have a cut and a flow, and the
Table 1. Cuts and their capacities for the example network

<table>
<thead>
<tr>
<th>cut</th>
<th>capacity</th>
<th>cut</th>
<th>capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>{s}</td>
<td>5</td>
<td>{s, 1}</td>
<td>6</td>
</tr>
<tr>
<td>{s, 1}</td>
<td>10</td>
<td>{s, 1, 3}</td>
<td>7</td>
</tr>
<tr>
<td>{s, 2}</td>
<td>4</td>
<td>{s, 2, 3}</td>
<td>5</td>
</tr>
<tr>
<td>{s, 3}</td>
<td>6</td>
<td>{s, 1, 2, 3}</td>
<td>3</td>
</tr>
</tbody>
</table>

Figure 4. A maximum flow

fact that the capacity of the cut is equal to the value of the flow provides an optimality certificate for both of them: The cut must be a cut of minimal capacity, because any cut of smaller capacity would prohibit a flow of this value, and the flow must be a maximum flow because it is impossible to push more flow across the given cut. This is an instance of a very interesting phenomenon called duality which is at the bottom of a lot of beautiful theory and powerful algorithms developed over the last 100 years.

The following theorem is a cornerstone of mathematical optimisation. It tells us that we can always expect to end up in the nice situation that we have a flow and a cut of the same value, so that each of them certifies the optimality of the other one.

**Theorem (MaxFlow-MinCut).** For every flow network, the maximum flow value is equal to the minimum cut capacity.

This theorem can be proved by describing an algorithm that produces a flow $x$ and a cut $W$ such that the value of $x$ is equal to the capacity of $W$. This algorithm is based on the residual network.

**Definition 4.** Given a flow network with node set $V$, arc set $A$ and capacities $c_a$, together with a feasible flow $x = (x_a)_{a \in A}$, the residual network has the same node set $V$, and the arc set $A^R$ defined as follows:

- **Forward:** If $(i, j)$ is an arc in $A$ with $x_{ij} < c_{ij}$ then $A^R$ contains arc $(i, j)$ with capacity $c_{ij}^R = c_{ij} - x_{ij}$.
- **Backward:** If $(i, j)$ is an arc in $A$ with $x_{ij} \neq 0$ (and necessarily $x_{ij} > 0$) then $A^R$ contains the reverse arc $(j, i)$ with capacity $c_{ji}^R = x_{ij}$.

Intuitively, a forward arc $(i, j)$ in the residual network means that we have the option to increase the flow on arc $(i, j)$ by $c_{ij}^R$ units, and a backward arc $(j, i)$ means that we have the option to decrease the flow on arc $(i, j)$ by $c_{ji}^R$ units. In particular, if the current flow on arc $(i, j)$ is strictly between 0 and the capacity of the arc, $0 < x_{ij} < c_{ij}$, then both the forward arc $(i, j)$ and the backward arc $(j, i)$ are present in the residual network (with residual capacities $c_{ij}^R = c_{ij} - x_{ij}$ and $c_{ji}^R = x_{ij}$). As an example, the flow shown in Figure 5 yields the residual network shown in Figure 6. A path from $s$ to $t$ in the residual network is called an augmenting path. If there is an augmenting path then we can increase the value of the current flow by “pushing flow along the augmenting path” which means by increasing the flow on the arcs corresponding to forward arcs, and by decreasing the flow on the arcs corresponding to backward arcs. For instance the auxiliary network in Figure 8 contains the augmenting path indicated in Figure 7 and pushing 1 unit of flow along this path yields the new flow shown in Figure 9, which we already know is optimal because its value is 3. But if we don’t know that the flow is optimal, we can simply construct the corresponding residual network which is shown in Figure 9. From the fact that there is no augmenting path (that is, no path from $s$ to $t$), we can conclude that the current flow
is optimal. In fact, the residual network even provides a cut that certifies the optimality. Let \( W \) be the set of all nodes that are reachable from node \( s \) in the residual network (in Figure 9 \( W = \{s, 1, 2, 3\} \) which in view of Table 1 shouldn’t come as a surprise). Then \( s \in W \), and since there is no augmenting path \( t \not\in W \), hence \( W \) is a cut. We make the following two observations:

1. Every arc \( a = (i, j) \) with \( i \in W \) and \( j \not\in W \) is at capacity, \( x_a = c_a \), because otherwise the arc \( (i, j) \) would be in \( A^R \), and \( j \) would be reachable, that is \( j \in W \).

2. Every arc \( a = (i, j) \) with \( i \not\in W \) and \( j \in W \) has zero flow, \( x_a = 0 \) because otherwise the arc \( (j, i) \) would be in \( A^R \), and \( i \) would be reachable, that is \( i \in W \).

Taking these two observations together we conclude that \( f(W, x) \) is equal to the capacity of \( W \), and therefore \( W \) is a minimum cut, and \( x \) is a maximum flow.

As a consequence of the above discussion, the maximum flow problem can be solved as described in Algorithm 1. Note that this description does not specify all the detail, and in particular for the choice of the augmenting path in the first line of the while loop, there are many different possible ways of doing it, and in practice this choice can make a huge difference in terms of the time needed to find the maximum flow. There is one more consequence that tends to be very useful in applications: Provided all the capacities are integer, then
Algorithm 1 The augmenting path algorithm

**Input:** flow network \((V, A, s, t)\) with capacities \(c_a\)

Initialise the flow by setting \(x_a = 0\) for all arcs \(a\)

Construct the residual network

**while** there exists an augmenting path in the residual network **do**

Choose an augmenting path \(P\)

Let \(u\) be the minimum residual capacity along the path \(P\)

Update the flow \(x\) by pushing \(u\) units of flow along \(P\)

Update the residual network

Let \(W\) be the set of nodes that are reachable in the residual network

**Output:** maximum flow \(x\) and minimum cut \(W\)

in each iteration an integer amount of flow is pushed along the augmenting path and the residual capacities stay integer. In particular, the resulting maximal flow has integer values \(x_a\) for every arc \(a\), a property that cannot be taken for granted in more general optimisation problems.

**Some historical remarks**

(1) The first augmenting path algorithm was published by Ford and Fulkerson in 1956. Their augmentation rule is guaranteed to terminate if capacities are rational numbers. It is possible to construct instances with irrational capacities where the algorithm forever keeps augmenting smaller and smaller amounts of flow. The worst case run time of the algorithm depends on the capacities.

(2) The algorithm of Edmonds and Karp from 1972 (independently by Dinic in 1970) chooses a shortest augmenting path. It always finds a maximum flow and its run time can be bounded in terms of the number of nodes and arcs only (independent of the capacities).

(3) Other methods have been developed (push-relabel algorithms, blocking flow algorithms) and a lot of effort has been spent on efficient implementations and clever data structures.

(4) The latest and greatest is an algorithm found by James Orlin in 2012 whose run time can be bounded by the product of the number of nodes, the number of arcs and a constant.

**Problems**

(1) Apply the augmenting path algorithm to find a maximum flow in the network shown in Figure 10.

(2) At a medial clinic 169 patients arrived in need of emergency treatment. Each of them requires a transfusion of one unit of whole blood. The clinic has supplies of 170 units of whole blood. The number of units of blood available in each of the four major blood groups and the distribution of patients among the groups is summarised in the table below.
Blood type  | A     | B     | O     | AB    |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Supply</td>
<td>46</td>
<td>34</td>
<td>45</td>
<td>45</td>
</tr>
<tr>
<td>Demand</td>
<td>39</td>
<td>38</td>
<td>42</td>
<td>50</td>
</tr>
</tbody>
</table>

Type A patients can only receive type A or O; type B patients can receive only type B or O; type O patients can receive only type O; and type AB patients can receive any of the four types.

(a) Formulate the problem of distributing the blood to the maximum number of patients as a maximum flow problem. Draw the network and label each arc by its capacity. **[Hint: Your network should have 10 nodes: a source node s, a node for each of the four blood types (named 1 to 4), a node for each blood type (named 5 to 8), and a target node t.]**

(b) Consider the following distribution:
- All 39 patient of type A receive type A.
- All 38 patients of type B receive type O.
- 7 patients of type O receive type O.
- The patients of type AB are divided as follows: 7 receive type A, 34 receive type B, and 9 receive type AB.

Illustrate the corresponding flow by drawing a network with the same nodes as in part (a), but containing only the arcs with nonzero flow. Label each arc with its flow value. What is the value of this flow?

(c) For the flow in part (b), find an augmenting path with residual capacity 34.

(d) Using the result from part (c), determine a minimum cut in the network, i.e., list the vertices on the source side of the cut.

(e) Use the minimum cut to deduce a rigorous and concise explanation of why not all of the patients can receive blood from the available supply.

(3) Let $X$ be a set of employees, and let $Y$ be a set of jobs. For every job $y \in Y$ let $X(y) \subseteq X$ be the set of employees that are qualified to do job $y$. For a set of jobs $\{y_1, y_2, \ldots, y_k\} \subseteq Y$, let $X(\{y_1, y_2, \ldots, y_k\})$ be the set of employees that can do at least one of the jobs in $\{y_1, y_2, \ldots, y_k\}$, that is,

$$X(\{y_1, y_2, \ldots, y_k\}) = X(y_1) \cup X(y_2) \cup \cdots \cup X(y_k).$$

We are interested in the question if it is possible to assign to every job an employee who can do it, such that every employee is assigned at most one job. Prove Hall’s theorem: It is is possible to assign all the jobs to employees if and only if $|X(\{y_1, y_2, \ldots, y_k\})| \geq k$ for every job set $\{y_1, y_2, \ldots, y_k\} \subseteq Y$.

(4) A bipartite graph consists of two disjoint sets $X$ and $Y$, whose elements are called vertices, and a set of pairs $(x, y)$, called edges, with $x \in X$ and $y \in Y$. A set $C \subseteq X \cup Y$ is called a vertex cover if every edge contains at least one vertex from $C$. A set $M$ of edges is called a matching if no vertex is contained in more than one edge from $M$.

(a) In the bipartite graph shown in Figure 11 find a matching $M$ of maximum size and a vertex cover $C$ of minimum size.

(b) Prove König’s theorem: The maximum size of a matching is equal to the minimum size of a vertex cover.