Maximum Flow Problems

MA428, Dr Katerina Papadaki

I. Introduction

In this lecture we will cover the following:

1) Problem definition
2) Transforming problems into the max flow problem
3) LP Formulation
4) Properties of Max Flow problems and the Max-Flow Min-Cut theorem
5) The Augmenting Path Algorithm
6) Flows with Lower Bounds
7) Applications

This lecture covers material from chapter 6 of AMO and refers to material in chapter 7 of AMO, which can be found in the following pages 169-196; 198-201; 205; 213-14.

II. Problem definition

A. Preliminaries

Consider a network $G = (N, A)$ made up of $n = |N|$ nodes and $m = |A|$ directed edges or arcs. We assume that there is at most one arc between any pair of nodes in any one direction (i.e., no parallel arcs) and there are no self-loops.

We are concerned with transporting a single commodity from *source node* $s \in N$ to *sink node* $t \in N$. On each arc (i, j) we can transport an amount of the commodity that does not exceed the *capacity* u_{ij} associated with that arc. If arc (i, j) is not capacity constrained, then we let

K. Papadaki is with the Department of Management, Operational Research Group at London School of Economics, United Kingdom

E-mail: `k.p.papadaki@lse.ac.uk`

$u_{ij} = \infty$. We call the amount of the commodity transported on arc (i, j) the *flow* of arc (i, j) and we denote it by x_{ij}.

Definition In a network $G = (N, A)$ with $s, t \in N$, the *maximum flow problem* consists of sending as much flow as possible from source node s to sink node t by using the arcs of the network without exceeding the capacity of any arc.

B. Assumptions

1) *Connectivity*: We assume that there is a directed path between s and t.
   In a general network there may not be a directed path between s and t. However, unlike the shortest path problem, there is no need to check that such a path exists. We can deal with this problem by introducing, where it does not exist, a new arc (s, i) from source node s to all other nodes $i \in N \setminus \{s\}$ in G and define the capacity of all these arcs to be zero: $u_{si} = 0$. If the max flow turns out to be zero, we know that there is no $s - t$ path.
2) *No Unbounded solutions*: We assume that the network does not have an $s{-}t$ path composed only of arcs with infinite capacity.
   We can always look for such a path using the search algorithm (where admissible arcs are arcs with infinite capacity).
3) *Integral Capacities*: We assume that the arc capacities are integers.
   If arc capacities are not integer but rational numbers, then we can always multiply the with a constant to make them integer. However, if they are irrational numbers then the termination of some algorithms is problematic.

III. Transforming Problems into the Maximum Flow Problem

We can transform some similar problems into maximum flow problems. Some examples are mentioned below:

A. Problems with Node Capacities

Suppose some nodes in N also have a capacity constraint. If node j has a capacity of u_j, we split the node j into two nodes j_1 and j_2: the receiving and departing ends of node j. All the arcs into j, $(k, j) \in A$, in G are relabeled as (k, j_1), while the arcs out of j, $(j, k) \in A$, are relabeled (j_2, k) in the transformed network. We also add one extra arc (j_1, j_2) in the new network with capacity u_j. The max flow problem in the new network will take care of the node capacity restrictions. An example in shown in Figure 1.

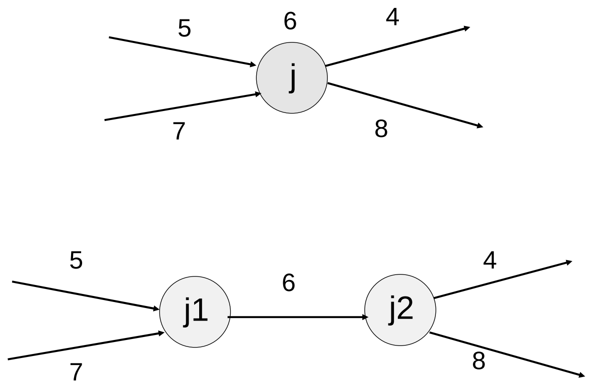

Fig. 1. Example of how to transform node capacity of node j to arc capacity of arc $(j1, j2)$.

B. Problems with Several Sources and/or Destinations

Suppose there were p source nodes $\{1, 2, \ldots, p\}$, each with maximum amounts a_r available to send to t, for $r = 1, 2, \ldots, p$. We can introduce a super source node s^* and arcs (s^*, r) with capacity $u_{s^*r} = a_r$ connecting the super source node s^* to all other source nodes r, $r \in \{1, 2, \ldots p\}$. The max flow problem between s^* and t solves the original problem. Similar changes can take care of multiple destinations.

C. Problems with Multiple arcs between two nodes

When we have multiple parallel arcs between two nodes, we can delete all but one arc and let its capacity equal the sum of capacities on all the parallel arcs. For an example see Figure 2.

D. Other Transformations

Slightly more complex transformations allow us to solve, among others, max flow problems with:

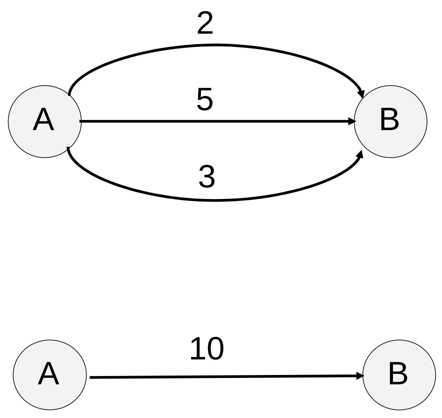

Fig. 2. We can combine multiple arcs to one arc with capacity the sum of the multiple arcs.

1) Positive lower bounds as well as upper bounds on arcs and
2) Demands at destination nodes and supplies at source nodes that have to be met exactly.

We investigate the above in section VII.

IV. LP Formulation

The following LP solves the max flow problem. Let x_{ij} = flow on arc (i, j), we let x denote the vector of all flows. Let the variable V represent the amount of flow leaving the source node s and reaching the sink node t. Then the maximum flow problem can be formulated as follows:

$$\max_{x_{ij}} \quad V$$

$$\text{s.t.} \sum_{j:(i,j)\in A} x_{ij} - \sum_{k:(k,i)\in A} x_{ki} = \begin{cases} V & \text{for } i = s \\ 0 & \text{for all } i \in N \setminus \{s, t\} \\ -V & \text{for } i = t \end{cases} \tag{1}$$

$$0 \leq x_{ij} \leq u_{ij} \quad \text{for all } (i, j) \in A$$

In this formulation there is a variable for every arc, representing the flow on the arc. There is an equality constraint corresponding to each node $i \in N$. For nodes s and t the relevant constraints imply that the net flow going out of s and the net flow coming into t must equal V. For all the remaining nodes, the equality constraints imply that the net flow on any intermediate nodes must be zero, i.e. what comes in must go out. These equality constraints are called *flow conservation* constraints.

Let A be the constraint matrix for the flow conservation constraints. Note that each column matrix A corresponds to an arc (i, j). Further, this column has exactly two non-zeroes entries one $+1$ in the row for node i (representing an outflow from node i) and one -1 in the row for node j (representing an inflow into node j). As we will see in later lectures, problems with such constraint matrices have very desirable properties.

In this lecture we will show that if the capacities are all integral then there exists an integer solution x to the LP defined in (1), and the optimal value is integer.

V. Properties of Max Flow problems and the Max-Flow Min-Cut theorem

A. *Preliminaries*

We first need to introduce some notation and definitions. Let $G = (N, A)$ be a network with source node s and sink node t.

Similar to our previous definition for undirected graphs, a *cut* $[S, \bar{S}]$ in a network $G = (N, A)$ is the set of arcs $(i, j) \in A$ such that either $i \in S$ and $j \in \bar{S}$, or $i \in \bar{S}$ and $j \in S$, where $S \subseteq N$ and $\bar{S} = N \setminus S$. It is the set of arcs that cross from S to $\bar{S}$ or from $\bar{S}$ to S, where $S, \bar{S}$ form a partition of the node set N.

Definition An $s - t$ *cut* in G is a cut such that $s \in S$ and $t \in \bar{S}$.

We distinguish between arcs in an $s - t$ cut with respect to their direction.

Definition In an $s - t$ cut $[S, \bar{S}]$, we call *forward arcs* to be the arcs in the $s - t$ cut whose direction is from S to $\bar{S}$, and we call *backward arcs* to be the arcs in the $s - t$ cut whose direction is from $\bar{S}$ to S. We denote the set of forward arcs by $(S, \bar{S})$ and the set of backward arcs by $(\bar{S}, S)$.

In a network with n nodes there are 2^{n-2} different $s - t$ cuts. For the network shown in Figure 3, the $s - t$ cuts are shown in the first column of Table I.

Definition We define the *capacity* of an $s - t$ cut $[S, \bar{S}]$ to be the sum of the capacities of its

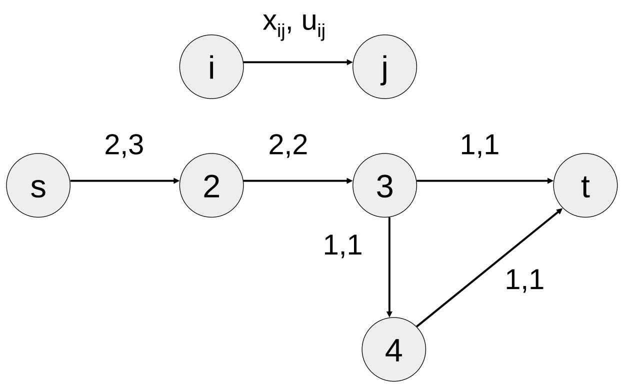

Fig. 3. The diagram shows the representation of a feasible flow. Each arc (i, j) has the ordered pair (x_{ij}, u_{ij}) associated with it, where x_{ij} is the flow through arc (i, j) and u_{ij} is the capacity of the arc (i.e. the upper bound on the flow of (i, j)).

forward arcs, and we denote it by $C(S, \bar{S})$:

$$C(S, \bar{S}) = \sum_{(i,j)\in(S,\bar{S})} u_{ij}$$

As you will see later we will be interested in $s - t$ cuts with minimum capacity.

Definition In a network G a *feasible flow* x of value V is a feasible solution to the problem defined in (1).

Thus a feasible flow x is a flow on each arc $(i, j) \in A$ that satisfies the flow conservation constraints, it is positive and within capacity. On each arc (i, j) on the diagram of a network we display the pair x_{ij}, u_{ij} which denotes the flow and capacity of each arc. Figure 3 shows a network with a feasible flow.

B. The weak max-flow min-cut theorem

We continue with some definitions.

Definition Given an $s-t$ cut $[S, \bar{S}]$ and a feasible flow x, we define the *net flow* of the cut with respect to x to be the total flow of the forward arcs minus the total flow on the backward arcs:

$$\text{Net flow of } s - t \text{ cut } [S, \bar{S}] = \sum_{(i,j)\in(S,\bar{S})} x_{ij} - \sum_{(i,j)\in(\bar{S},S)} x_{ij}$$

S	Forward Arcs (i,j)	x_{ij}, u_{ij}	Sum of Forward Flows	Backward Arcs (j,i)	x_{ji}, u_{ji}	Sum of Backward Flows	Net Flow	$C(S,\bar{S})$
s	$(s,2)$	$2,3$	2	–	–	0	2	3
$s,2$	$(2,3)$	$2,2$	2	–	–	0	2	2
$s,3$	$(s,2)$	$2,3$		$(2,3)$	$2,2$	2	2	5
	$(3,4)$	$1,1$						
	$(3,t)$	$1,1$	4					
$s,4$	$(s,2)$	$2,3$		$(3,4)$	$1,1$	1	2	4
	$(4,t)$	$1,1$	3					
$s,2,3$	$(3,4)$	$1,1$				0	2	2
	$(3,t)$	$1,1$	2	–	–			
$s,2,4$	$(2,3)$	$2,2$		$(3,4)$	$1,1$	1	2	3
	$(4,t)$	$1,1$	3					
$s,2,3,4$	$(3,t)$	$1,1$				0	2	2
	$(4,t)$	$1,1$	2	–	–			
$s,3,4$	$(s,2)$	$2,3$		$(2,3)$	$2,2$	2	2	5
	$(3,t)$	$1,1$						
	$(4,t)$	$1,1$	4					

TABLE I

CALCULATING THE NET FLOWS AND CAPACITIES OF EACH $s-t$ CUT FOR THE FEASIBLE FLOW SHOWN IN FIGURE 3.

The net flows and capacities of all the $s-t$ cuts of the network in Figure 3, are calculated in Table I. Note that the net flow from all the cuts is equal to $V = 2$ which is the flow in the network. Further, note that the net flow is always less than or equal to the capacity of the cut.

Now consider a feasible flow x with value V. What is the relationship between V and the net flow of any $s-t$ cut with respect to x. It turns out that they are equal i.e. the net flow in any $s-t$ cut is equal to V. To see this consider the following: Add the flow conservation constraints for all nodes $i \in S$. This gives us:

$$V = \sum_{i \in S} \left[\sum_{j:(i,j) \in A} x_{ij} - \sum_{k:(k,i) \in A} x_{ki} \right] \tag{2}$$

We can simplify the above expression by noting that whenever nodes p and q belong to S and $(p,q) \in A$, the variable x_{pq} in the first term within the brackets (for node $i = p$) cancels with $-x_{pq}$ that appears in the second term within the brackets (for node $i = q$). Further, note that if p and q both belong to $\bar{S}$ then x_{pq} does not appear in the above expression. Thus only when (p,q) is in the $s-t$ cut $[S,\bar{S}]$ does the term x_{pq} appear in the expression. Thus we can simplify

(2) to the form:

$$V = \sum_{(i,j)\in(S,\bar{S})} x_{ij} - \sum_{(i,j)\in(\bar{S},S)} x_{ij} = \text{Net flow of } [S,\bar{S}] \text{ cut w.r.t. } x \tag{3}$$

Thus we have shown that the net flow of an $s-t$ cut with respect to a feasible flow x is equal to the value of the flow V. The above is quite intuitive. It says that if there is a feasible flow of value V from s to t, then this flow must pass through every $s-t$ cut.

Now we can show the following proposition:

Proposition 5.1: The value V of any feasible flow x is less than or equal to the capacity of any $s-t$ cut $[S,\bar{S}]$ in the network G:

$$V \le C(S,\bar{S})$$

Proof: We use (3) to show this. We substitute $x_{ij} \le u_{ij}$ in the first sum of (3), and $x_{ij} \ge 0$ in the second sum of (3). This gives us:

$$V \le \sum_{(i,j)\in(S,\bar{S})} u_{ij} = C(S,\bar{S}) \tag{4}$$

■

Since proposition 5.1 holds for any feasible flow and any $s-t$ cut we have the following result:

Theorem 5.2: The maximum flow through a network is less than or equal to the capacity of the minimum capacity $s-t$ cut.

The above result is the *weak max-flow min-cut* theorem. The *max-flow min-cut* theorem states that the maximum flow of a network is equal to the capacity of the minimum capacity cut. To show this we will construct a feasible flow whose value is equal to the capacity of the minimum capacity cut.

C. *Residual Networks*

To proceed we need more definitions and notation.

Definition Given a network $G = (N,A)$ and a feasible flow x on G, the residual network $G(x) = (N,A')$ is defined as follows:

For each arc $(i,j) \in A$ in G with flow x_{ij} and capacity u_{ij} define the following two arcs in $G(x)$:

1) If $u_{ij} - x_{ij} > 0$ then create arc $(i, j) \in A'$ with capacity $u'_{ij} = u_{ij} - x_{ij}$; if $u_{ij} - x_{ij} = 0$ then omit arc.
2) If $x_{ij} > 0$ then create arc $(j, i) \in A'$ with capacity $u'_{ji} = x_{ij}$; if $x_{ij} = 0$ then omit arc.

We call the capacities of the residual network *residual capacities*. The residual network of Figure 3 is shown in Figure 4. The arc $(s, 2) \in A$ in G with $x_{s2} = 2$, $u_{s2} = 3$, is represented in $G(x)$ with two arcs. The first arc in $G(x)$ is $(s, 2)$ with residual capacity $u'_{s2} = u_{s2} - x_{s2} = 3 - 2 = 1$, and this denotes the remaining capacity if we wanted to increase our flow. The second arc in $G(x)$ is $(2, s)$ with residual capacity $u'_{2s} = x_{s2} = 2$, and this denotes the capacity already used and thus it is the amount that we can reduce the flow. In fact the capacity of arc $(2, s)$ in $G(x)$ shows that amount that we can increase the flow in opposite direction i.e. flow from 2 to s. By decreasing the flow from s to 2 we increase the flow from 2 to s. Thus the residual capacities are indeed capacities on how much we increase flow on both of these directions.

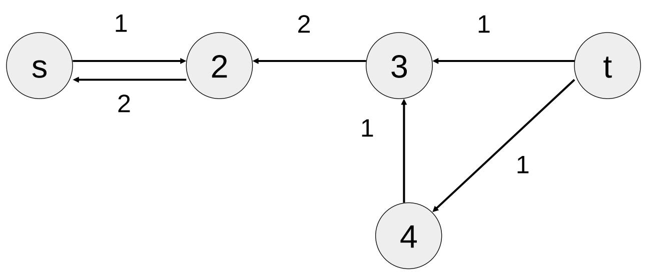

Fig. 4. The residual network that corresponds to the flow shown in Figure 3.

Note that each residual network $G(x)$ corresponds to a feasible flow x in the network G. Thus for different feasible flows we get different residual networks. Basically, the residual network helps us identify how to increase the flow x in the original network G.

There is an important property of the residual network that we state below:

Proposition 5.3: If there is no $s - t$ path in the residual network $G(x)$ that corresponds to feasible flow x with value V, then V is the maximum flow in G (i.e. x is optimal).

Before we prove the above proposition we consider how we check for an $s-t$ path in $G(x)$. We can use a search algorithm where we keep the marked nodes in S and the unmarked nodes in $\bar{S}$. The $LIST$ can start with $LIST = \{s\}$ and at each step we mark all the nodes that are reachable from s. This continues until node t is marked, in which case an $s-t$ path in $G(x)$ has been established, or until all the nodes that are reachable from s are marked but t is not reachable.

In the first case we have established a path in $G(x)$ from s to t, let's call it P, with positive residual capacities (since we omit arcs with zero residual capacities). These residual capacities indicate the amount of flow that we can increase on each arc. If we take the minimum of these capacities $\delta = \min_{(i,j)\in P} u'_{ij}$, then we know that we can increase the flow from s to t by the amount δ. Thus we augment the current flow by δ by using this $s-t$ path in $G(x)$. Then we construct a new residual network for the new flow and look again for an $s-t$ path. We repeat this until there is no $s-t$ path.

In the second case we terminate the search algorithm with the marked nodes in S and the unmarked nodes in $\bar{S}$ with $t \in \bar{S}$. Then we say that our search for an $s-t$ path in $G(x)$ terminated with the $s-t$ cut in G given by $[S,\bar{S}]$. We show the following result:

Lemma 5.4: If the search algorithm for an $s-t$ path in $G(x)$ terminates with an $s-t$ cut $[S,\bar{S}]$ in G, then all the forward arcs of this $s-t$ cut are saturated, $x_{ij} = u_{ij}$, and all backward arcs carry zero flow, $x_{ij} = 0$.

Proof: Suppose there exists a forward arc $(i,j) \in (S,\bar{S})$ that is not saturated: $x_{ij} < u_{ij}$. In the residual network this implies an arc (i,j) with $i \in S$ and $j \in \bar{S}$ with capacity $u_{ij} - x_{ij}$. This means that node j is reachable from s which contradicts the assumption that the search algorithm marked all the nodes that are reachable. A demonstration of a forward non-saturated arc in G is shown in Figure 5.

Suppose there exists a backward arc $(j,i) \in (\bar{S},S)$ that does not have zero flow: $x_{ji} > 0$. In the residual network this implies an arc (i,j) with $i \in S$ and $j \in \bar{S}$ with capacity x_{ji}. This means that node j is reachable from s which contradicts the assumption that the search algorithm marked all the nodes that are reachable. A demonstration of a backward non-zero flow arc in G is shown in Figure 6. ■

Now we are ready to prove proposition 5.3.

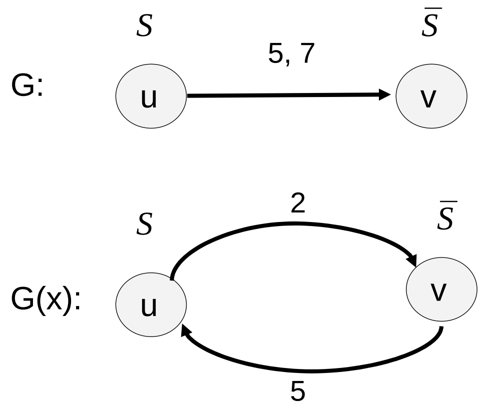

Fig. 5. Non-saturated forward arcs of cut $[S, \bar{S}]$.

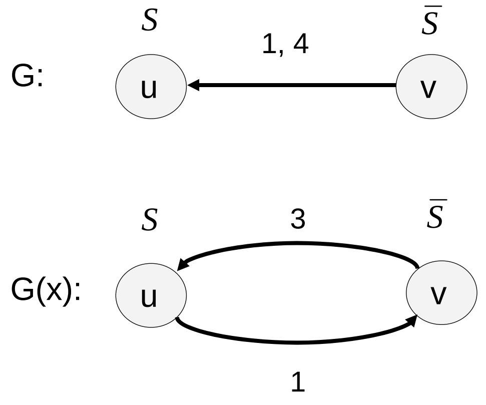

Fig. 6. Non-zero flow backward arcs of cut $[S, \bar{S}]$.

Proof: (of proposition 5.3)

If there is no $s - t$ path in $G(x)$ then we terminate with an $s - t$ cut $[S, \bar{S}]$ in G. Let x be the flow corresponding to residual network $G(x)$ with value V. Using equation (3) that we derived above we can write:

$$V = \sum_{(i,j) \in (S,\bar{S})} x_{ij} - \sum_{(i,j) \in (\bar{S},S)} x_{ij} \tag{5}$$

By lemma 5.4 we know that all the forward arcs in $(S, \bar{S})$ are saturated, i.e. $x_{ij} = u_{ij}$, and all the backward arcs in $(\bar{S}, S)$ have zero flow, i.e. $x_{ij} = 0$. Substituting these into (5) we get:

$$V = \sum_{(i,j) \in (S,\bar{S})} u_{ij} - 0 = C(S, \bar{S}),$$

which shows that the flow V is equal to the capacity of the $s-t$. From proposition 5.2 we know that V is the maximum possible flow since it can go no higher than the capacity of any cut. ■

D. *Max-Flow Min-Cut Theorem*

From the proof of proposition 5.3 we can see that the flow x with value V, whose corresponding residual network $G(x)$ has no $s-t$ path, is optimal. Further, the maximum flow V is equal to the capacity of the $s-t$ cut found when searching for an $s-t$ path in $G(x)$. From this and the weak max-flow min-cut theorem (theorem 5.2) we have:

Theorem 5.5: Max-Flow Min-Cut Theorem: The maximum flow in a network is equal to the capacity of the minimum capacity cut.

In section V-C we have devised a method for augmenting a feasible flow by finding an $s-t$ path in the residual network $G(x)$. If there is no augmenting path in $G(x)$ then the flow V is maximum (by proposition 5.3). If the flow V is maximum, then there cannot be an augmenting path because if there was we would be able to increase the flow. This is stated in the following theorem.

Theorem 5.6: Augmenting Path Theorem: A flow x^* is a maximum flow if and only if there is no augmenting path in the corresponding residual network $G(x^*)$.

The above two theorems are equivalent. Theorem 5.6 also suggests a method to find the maximum flow. We will elaborate on this in the next section.

It is important to note that the problem of finding the minimum capacity $s-t$ cut is the *dual* of the problem of finding the maximum flow. Thus the max-flow min-cut theorem is nothing else than the duality theorem for flow problems.

VI. Augmenting Path Algorithm

In this section we will describe the generic *augmenting path algorithm* also called *labeling algorithm*. We extend on ideas from the previous section.

A. Description of the Algorithm

In the previous section in our effort to prove the Max-Flow Min-Cut theorem, we have described methods to augment a feasible flow until we reach the capacity of the minimum cut.

Given a feasible flow x we look in the residual network $G(x)$ for an $s-t$ path. Such a path P would allow us to augment the flow x to $x+\delta$, where $\delta = \min_{(i,j)\in P} u'_{ij}$. Then we construct the residual network $G(x+\delta)$ and repeat until there is no augmenting path (or $s-t$ path in the residual network). To find out whether an $s-t$ path exists in $G(x)$ we use a search algorithm to identify nodes that are reachable from s. The algorithm terminates either in an $s-t$ path in $G(x)$ or an $s-t$ cut $[S,\bar{S}]$ is G. When the search algorithm terminates in an $s-t$ cut in the residual network we know that we have reached the maximum flow and that this $s-t$ cut is a minimum capacity cut.

The description of the algorithm is shown in Algorithm 1.

Algorithm 1 Augmenting Path Algorithm

```
begin
    Step 1: Let $x_{ij} = 0$ for all $(i,j) \in A$.
    Step 2: Use the search algorithm to find an $s-t$ path in $G(x)$.
            If there is an $s-t$ path $P$ in $G(x)$, then
                Let $\delta = \min_{(i,j)\in P} u'_{ij}$.
                Augment the flow $x$ by $\delta$ on the arcs in $G$ that correspond to arcs in $P$.
                Repeat Step 2.
            else
                Extract the $s-t$ cut in $G$.
                Go to Step 3.
            end
    Step 3: The flow $x$ is the maximum flow and the $s-t$ cut is the minimum capacity $s-t$ cut.
end;
```

Note that we need an initial feasible flow to start the algorithm and we can always start with the zero flow. Further, when $x = 0$, the residual graph $G(x)$ is identical to G.

B. An example

We consider the example of the network shown in Figure 7. As shown in Figure 7 we start with zero flow. The residual network with respect to zero flow is shown in Figure 8. As we can see the residual network for zero flow is identical to the original network.

Figure 9 shows 3 iterations of the algorithm. In Figure 9(a) we identify the augmenting path 1-3-4 in the residual network, which allows us to augment the flow by 4 units. In Figure 9(b) we update the residual network by incorporating this flow of 4 more units on the path 1-3-4. We could redraw the original network with this new flow and then reconstruct a residual network from scratch. However, we can do this by updating only the arcs $(1, 3)$ and $(3, 4)$ on the residual network.

Figure 9(c) shows the new augmenting path 1-2-3-4 which can augment the flow by 1 unit, and the updated residual network is shown in Figure 9(d). Again, Figure 9(e) shows another augmenting path 1-2-4 which can augment the flow by 1 unit, and the updated residual network is shown in Figure 9(f).

Now there is no augmenting path and the algorithm terminates with the $s-t$ cut $[S, \bar{S}]$, where $S = \{1\}$, $\bar{S} = \{2, 3, 4\}$. Now we know that the flow of $0+4+1+1 = 6$ is the maximum flow. The original network with the maximum flow is shown in Figure 10. Note that the capacity of this cut, which is the sum of the forward arcs $(1, 3)$ and $(1, 2)$, is $4 + 2 = 6$. This is indeed equal to the maximum flow.

Note that in this example we did not use any specific search algorithm to find the augmenting paths.

C. The Integrality Theorem

The following theorem follows from the augmenting path algorithm:

Theorem 6.1: Integrality Theorem: If all arc capacities are integer, then the maximum flow problem has an integer flow and the value of the maximum flow is integer.

Proof: In the augmenting path algorithm we start with zero flow and we augment it by taking the minimum of integer residual capacities. When the flow on each arc is updated, it is

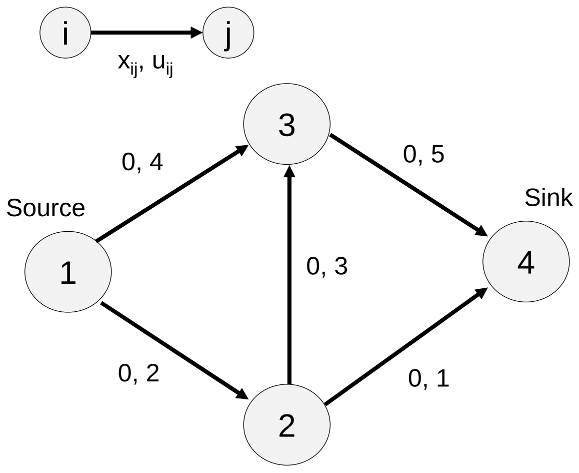

Fig. 7. The original network G with zero flow.

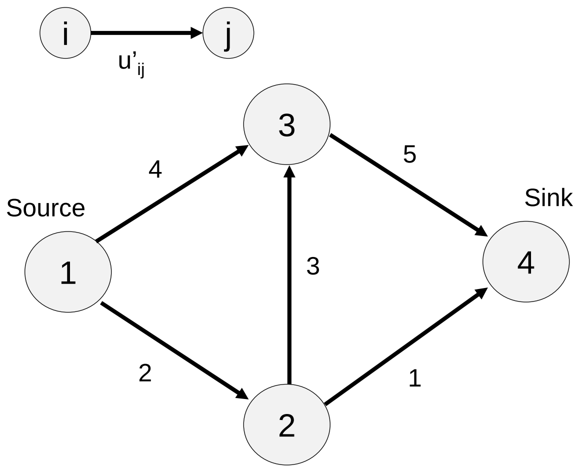

Fig. 8. The residual network of the network in Figure 7 for zero flow. Note that these are identical.

increased by an integer. Thus the augmenting path algorithm will result in an integer flow and also in an integer value for the maximum flow. ■

Note that this does not mean that all optimal flows x of a max flow problem will have to be integer. It just says that there exist integer solutions and in fact these are given to us by the augmenting path algorithm. The value V of the maximum flow of course must be integer.

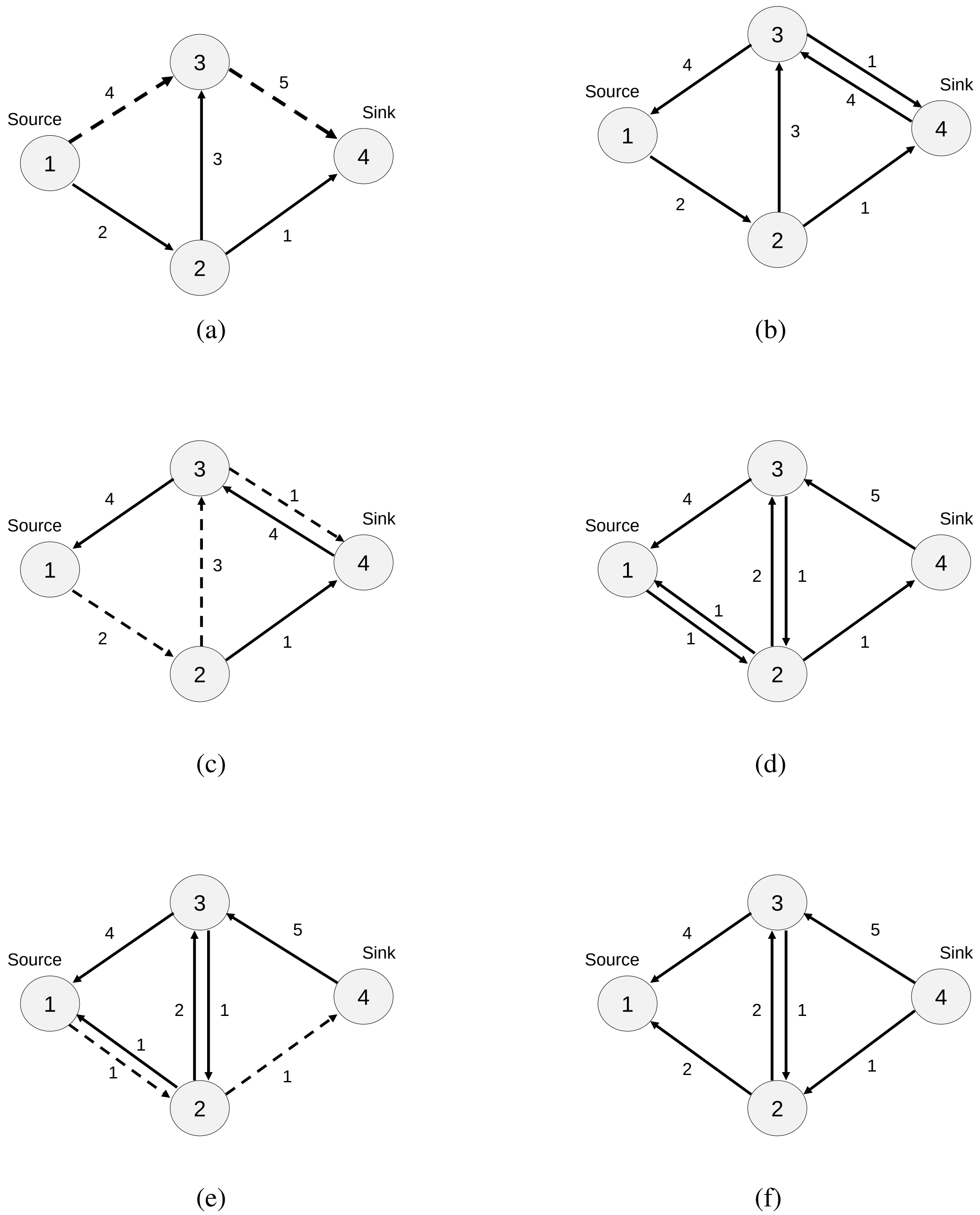

Fig. 9. Augmenting Path Algorithm on the residual network of Figure 7. (a) We pick augmenting path 1-3-4 and augment the flow by 4; (b) The residual network after the augmentation in (a); (c) We pick augmenting path 1-2-3-4 and augment the flow by 1; (d) The residual network after the augmentation in (c); (e) We pick augmenting path 1-2-4 and augment the flow by 1; (f) The residual network after the augmentation in (e). The minimum capacity cut is given by the partition $S = \{1\}$, $\bar{S} = \{2, 3, 4\}$.

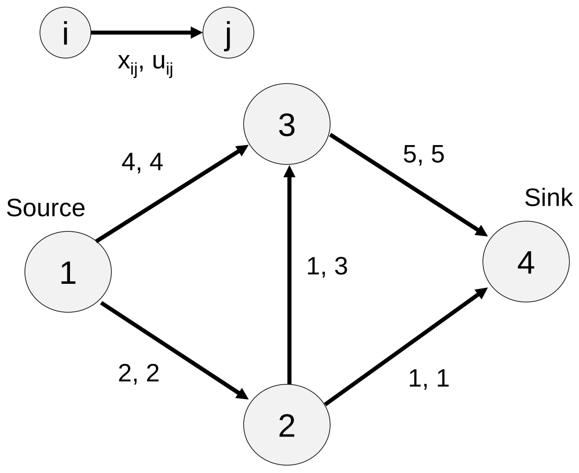

Fig. 10. The original network with the maximum flow of value 6. Note that the cut given by $S = \{1\}$, $\bar{S} = \{2, 3, 4\}$ is the minimum capacity cut and its capacity, which is the sum of the capacities of the forward arcs, is $4 + 2 = 6$.

D. Complexity

The generic augmenting path algorithm (also called *labeling algorithm*) is not a polynomial algorithm. It normally performs well but its worst case analysis could produce exponential number of steps.

The number of steps to find an augmenting path is at worst m since it might check all the arcs. Thus each augmenting step is $O(m)$. How many augmentations can the algorithm perform? Suppose that all arc capacities are integral and the maximum arc capacity is U. The capacity of the cut $[\{s\}, N \setminus \{s\}]$ is at most nU (in the worst case). If the augmenting path algorithm increases the flow by 1 unit at a time, it could perform nU augmentations. This gives the worse case complexity of $O(nmU)$.

If U is exponential in the number of nodes, for example $U = 2^n$, then this could be problem. Consider the network in Figure 11(a). If the algorithm alternates between paths $s - a - b - t$ and $s - b - a - t$ then it will perform 2×10^6 augmentations. Figures 11(b) and 11(c) show the residual networks after augmenting with respect to paths $s - a - b - t$ and then $s - b - a - t$.

There is a way to get around this problem and in fact Chapter 7 of AMO is dedicated to improved versions of the algorithm where the augmenting paths are picked carefully. Edmonds and Karp (1972) showed that if the shortest augmenting path is used (assigning cost of 1

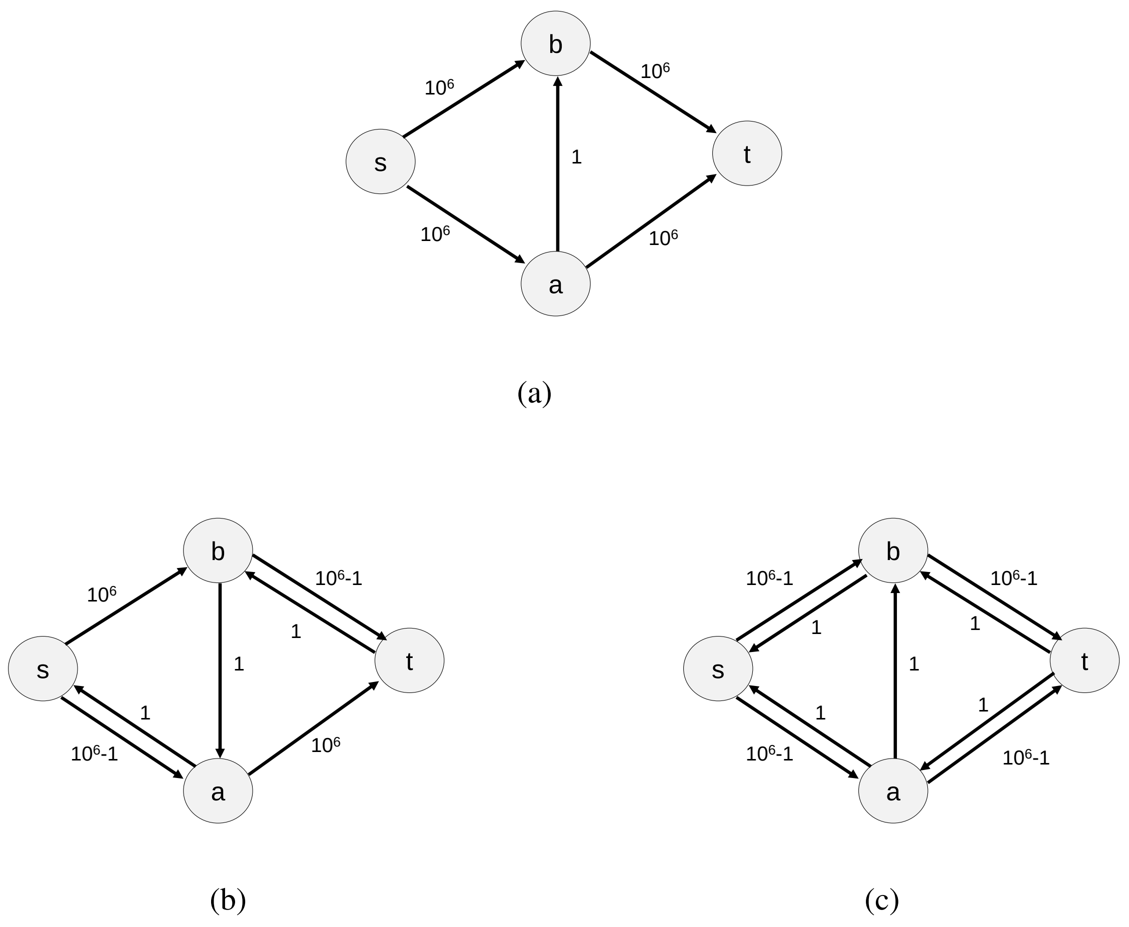

Fig. 11. Worst case scenario example for the generic augmenting path algorithm (or labeling algorithm). Alternating between paths $s-a-b-t$ and then $s-b-a-t$ would take the algorithm 2×10^6 augmentations to terminate.

on each arc) then at most $O(nm)$ augmenting paths will be used and thus the algorithm has complexity $O(nm^2)$. This is called the *shortest augmenting path algorithm*. There are also further improvements to this algorithm. We will not cover this in this lecture but for the interested readers see pages 213-214 of AMO.

Another drawback of the algorithm comes when the capacities are irrational numbers. Note that if the capacities are rational numbers we can multiply them all with constant to make them integer. However, when the capacities are irrational the algorithm might never terminate, even though max-flow min-cut theorem still holds even for irrational capacities.

VII. Flows with Lower Bounds

A. Problem Description

Suppose we also have lower bounds on the flow on each arc. For arc $(i, j) \in A$ let l_{ij} be the lower bound on the amount of flow that can go through (i, j). The max-flow problem formulation becomes:

$$\max_{x_{ij}} \quad V$$

$$\text{s.t.} \sum_{j:(i,j)\in A} x_{ij} - \sum_{k:(k,i)\in A} x_{ki} = \begin{cases} V & \text{for } i = s \\ 0 & \text{for all } i \in N \setminus \{s, t\} \\ -V & \text{for } i = t \end{cases} \tag{6}$$

$$l_{ij} \le x_{ij} \le u_{ij} \quad \text{for all } (i, j) \in A$$

All of the work we have done so far can be used to solve the maximum flow problem with lower bounds. The same theorems hold and the augmenting path algorithm can solve this problem. The only issue is that the augmenting path algorithm starts with the feasible flow of zero. Here, the zero flow is not feasible. Thus, the problem becomes one of finding a feasible flow. When we have a feasible flow then we can augment it in the same way as we did before.

B. Finding a feasible flow

To find a feasible flow of problem (6) we transform the problem several times. In the first transformation we transform it to a *circulation problem*. We add an arc (t, s) from the sink node t to the source node s with infinite capacity. Then we ask the question: Can we circulate a flow in the network that satisfies the constraints $l_{ij} \le x_{ij} \le u_{ij}$? The circulation problem can be expressed as follows:

Find x such that

$$\begin{aligned} &\sum_{j:(i,j)\in A} x_{ij} - \sum_{k:(k,i)\in A} x_{ki} = 0 \quad \text{for all } i \in N \\ &l_{ij} \le x_{ij} \le u_{ij} \quad \text{for all } (i, j) \in A \end{aligned} \tag{7}$$

Thus in the circulation problem the flow out of a node i must be equal to the flow into a node i and the flow must satisfy the lower and upper bound constraints. It is called the circulation problem because it will force a flow into s and also force a flow out of t. Note that zero flow is not a feasible solution.

Now we can make the next transformation. We substitute $x_{ij} = x'_{ij} + l_{ij}$ into (7). This gives us the following *supply and demand* problem:

$$\begin{aligned}
&\sum_{j:(i,j)\in A} x'_{ij} - \sum_{k:(k,i)\in A} x'_{ki} = b(i) \quad \text{for all } i \in N \\
&0 \le x'_{ij} \le u_{ij} - l_{ij} \quad \text{for all } (i,j) \in A \qquad (8) \\
&b(i) = -\sum_{j:(i,j)\in A} l_{ij} + \sum_{k:(k,i)\in A} l_{ki}
\end{aligned}$$

Note that we have eliminated the lower bound on the flow x'. Also, note that the right hand side of the conservation of flow constraints has changed. For each node i, the amount of flow out of i minus the amount of flow into i must equal $b(i)$. Thus, $b(i)$ can be interpreted as the amount of a commodity that i possess and will send out, i.e. the supply of node i. If it is negative it can be interpreted as the demand of node i. Thus our problem has become one of satisfying demands from supplies.

The term $b(i)$ is the sum of the lower bounds of arcs into i minus the sum of the lower bounds of arcs out of i. If we sum all the $b(i)$'s the terms will cancel and we get $\sum_{i\in N} b(i) = 0$ (verify this). Thus we know that there is as much demand as there is supply. Now our problem becomes to satisfy the demand at nodes with $b(i) < 0$ with the supply at nodes with $b(i) > 0$.

To do that we transform the problem into the following maximum flow problem:

1) Introduce a source s' and a sink t'.
2) If $b(i) > 0$ add arc (s', i) with capacity $b(i)$ (these are the supply nodes).
3) If $b(i) < 0$ add arc (i, t') with capacity $-b(i)$ (these are the demand nodes).
4) Find the maximum flow from s' to t' using the augmenting path algorithm (note that this could be zero).
5) If all the (s', i) arcs are saturated (i.e. have reached capacity) then all the (j, t') arcs will also be saturated, which means that supply has met demand and thus we have solved the supply and demand problem described in (8).

The above maximum flow problem sends as much flow as possible from s' to t'. All intermediate nodes (except s' and t') have to satisfy the flow conservation constraints for the maximum flow problem (i.e. flow-in = flow-out). Thus, if the flow on arc (s', i) is equal to its capacity $b(i)$, that means that node i will send $b(i)$ units of flow and thus satisfy its supply constraint. Further, if the flow on arc (j, t') is equal to $-b(j)$, that means that node j will receive $-b(j)$ units of flow and thus satisfy its demand constraint. Thus the above maximum flow problem solves the supply and demand problem if all the arcs (s', i) and (j, t') are saturated.

The solution of the above problem will give us the flows x' that are feasible for the supply and demand problem described in (8). By using $x = x' + l$ we have our feasible flow x for the original problem.

VIII. Applications

There are numerous applications of maximum flow problems. Some good examples can be found in AMO pages 169-177. Exercise 1 below is an example where the max flow problem can be used for scheduling.

Further, AMO describes applications of the max-flow min-cut theorem on combinatorial problems, where the theorem is used to show the duality between these problems. I ask you to look at one of these in exercise 4 below.

IX. Exercises

Solve the following exercises from AMO. Note that AMO refers to the generic augmenting path algorithm as the labeling algorithm.

1) 6.2
2) 6.12 (a) and (b)
3) 6.14
4) 6.46