Math 530 Spring 2022, Lecture 26: Networks and flows

website: `https://www.cip.ifi.lmu.de/~grinberg/t/22s`

1. Networks and flows

Today I will give an introduction to **network flows** and their optimization. This is a topic of great interest to logisticians, as even the simplest results have obvious applications to scheduling trains and trucks. It also has lots of purely mathematical consequences; in particular, we will use network flows to finally prove the Hall-König matching theorem (and thus the HMT, König's theorem, and their many consequences).

I will follow my notes [17s-lec16], which are a good place to look up the details of some proofs that I will only sketch. That said, I will be using multidigraphs instead of simple digraphs, so some adaptations will be necessary (since [17s-lec16] only works with simple digraphs). These adaptations are generally easy.

I will only cover the very basics of network flow optimization, leading to a proof of the max-flow-min-cut theorem (for integer-valued flows) and to a proof of the Hall-König matching theorem. For the deeper reaches of the theory, see [ForFul74] (a classical textbook written by the inventors of the subject), [Schrij17, Chapter 4] and [Schrij03, Part I].

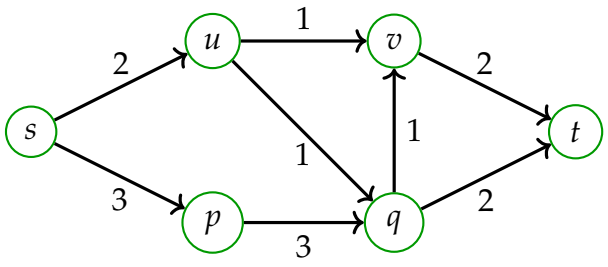
1.1. Definition

Recall that we use the notation $\mathbb{N} = \{0, 1, 2, \ldots\}$.

Definition 1.1.1. A **network** consists of

- a multidigraph $D = (V, A, \psi)$;
- two distinct vertices $s \in V$ and $t \in V$, called the **source** and the **sink**, respectively;
- a function $c : A \to \mathbb{N}$, called the **capacity function**.

Example 1.1.2. Here is an example of a network:



.

Here, the multidigraph D is the one we drew (it is a simple digraph, so we have not labeled its arcs); the vertices s and t are the vertices labeled s and t; the values of the function c on the arcs of D are written on top of these respective arcs (e.g., we have $c\left((s,p)\right) = 3$ and $c\left((u,q)\right) = 1$).

Remark 1.1.3. The digraph D in Example 1.1.2 has no cycles and satisfies $\deg^- s = \deg^+ t = 0$. This is not required in the definition of a network, although it is satisfied in many basic applications.

Also, all capacities $c\left(a\right)$ in Example 1.1.2 were positive. This, too, is not required; however, arcs with capacity 0 do not contribute anything useful to the situation, so they could just as well be absent.

Remark 1.1.4. The notion of "network" we just defined is just one of a myriad notions of "network" that can be found all over mathematics. Most of them can be regarded as graphs with "some extra structures"; apart from this, they don't have much in common.

Definition 1.1.5. Let N be a network consisting of a multidigraph $D = (V, A, \psi)$, a source $s \in V$, a sink $t \in V$ and a capacity function $c : A \to \mathbb{N}$. Then:

(a) For any arc $a \in A$, we call the number $c\left(a\right) \in \mathbb{N}$ the **capacity** of the arc a.

(b) For any subset S of V, we let $\overline{S}$ denote the subset $V \setminus S$ of V.

(c) If P and Q are two subsets of V, then $[P, Q]$ shall mean the set of all arcs of D whose source belongs to P and whose target belongs to Q. That is,

$$[P, Q] := \left\{a \in A \mid \psi\left(a\right) \in P \times Q\right\}.$$

(d) If P and Q are two subsets of V, and if $d : A \to \mathbb{N}$ is any function, then the number $d\left(P, Q\right) \in \mathbb{N}$ is defined by

$$d\left(P, Q\right) := \sum_{a \in [P,Q]} d\left(a\right).$$

(In particular, we can apply this to $d = c$, and then get $c(P,Q) = \sum_{a \in [P,Q]} c(a)$.)

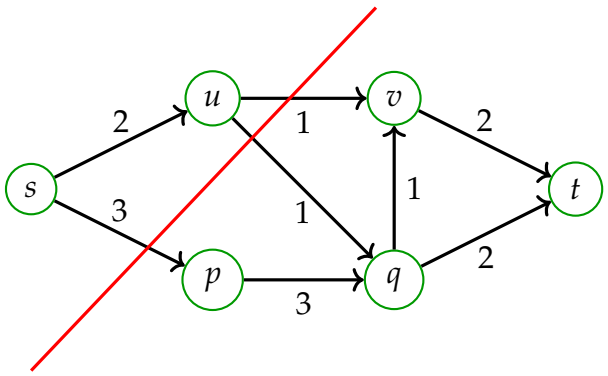
Example 1.1.6. Let us again consider the network from Example 1.1.2. For the subset $\{s,u\}$ of V, we have $\overline{\{s,u\}} = \{p,v,q,t\}$ and

$$\left[\{s,u\}, \overline{\{s,u\}}\right] = \{sp,\ uv,\ uq\}$$

(recall that our D is a simple digraph, so an arc is just a pair of two vertices) and

$$c\left(\{s,u\}, \overline{\{s,u\}}\right) = \sum_{a \in \left[\{s,u\}, \overline{\{s,u\}}\right]} c(a) = \underbrace{c(sp)}_{=3} + \underbrace{c(uv)}_{=1} + \underbrace{c(uq)}_{=1}$$
$$= 3 + 1 + 1 = 5.$$

We can make this visually clearer if we draw a "border" between the sets $\{s,u\}$ and $\overline{\{s,u\}}$:



.

Then, $\left[\{s,u\}, \overline{\{s,u\}}\right]$ is the set of all arcs that cross this border from $\{s,u\}$ to $\overline{\{s,u\}}$. (Of course, this visualization works only for sums of the form $d\left(P, \overline{P}\right)$, not for the more general case of $d(P,Q)$ where P and Q can have elements in common. But the $d\left(P, \overline{P}\right)$ are the most useful sums.)

Let us now define flows on a network:

Definition 1.1.7. Let N be a network consisting of a multidigraph $D = (V, A, \psi)$, a source $s \in V$, a sink $t \in V$ and a capacity function $c : A \to \mathbb{N}$.

A **flow** (on the network N) means a function $f : A \to \mathbb{N}$ with the following properties:

- We have $0 \leq f(a) \leq c(a)$ for each arc $a \in A$. This condition is called

the **capacity constraints** (we are using the plural form, since there is one constraint for each arc $a \in A$).

- For any vertex $v \in V \setminus \{s, t\}$, we have

$$f^-(v) = f^+(v),$$

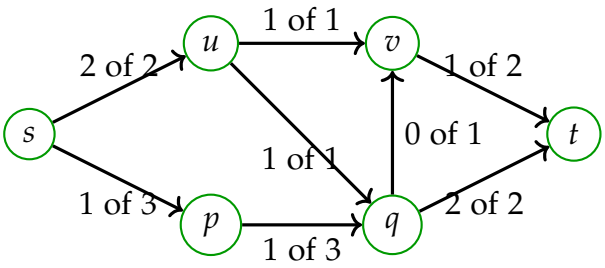
where we set

$$f^-(v) := \sum_{\substack{a \in A \text{ is an arc} \\ \text{with target } v}} f(a) \qquad \text{and} \qquad f^+(v) := \sum_{\substack{a \in A \text{ is an arc} \\ \text{with source } v}} f(a).$$

This is called the **conservation constraints**.

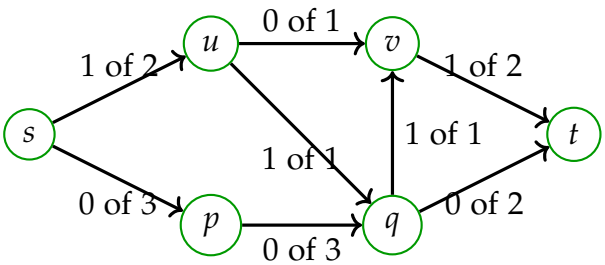
If $f : A \to \mathbb{N}$ is a flow and $a \in A$ is an arc, then the nonnegative integer $f(a)$ will be called the **arc flow** of f on a.

Example 1.1.8. To draw a flow f on a network N, we draw the network N, with one little tweak: Instead of writing the capacity $c(a)$ atop each arc $a \in A$, we write "$f(a)$ of $c(a)$" atop each arc $a \in A$. For example, here is a flow f on the network N from Example 1.1.2:



(so, for example, $f(su) = 2$, $f(pq) = 1$ and $f(qv) = 0$).

For another example, here is a different flow g on the same network N:



There are several intuitive ways to think of a network N and of a flow on it:

- We can visualize N as a collection of one-way roads: Each arc $a \in A$ is a one-way road, and its capacity $c(a)$ is how much traffic it can (maximally)

handle per hour. A flow f on N can then be understood as traffic flowing through these roads, where $f(a)$ is the amount of traffic that travels through the arc a in an hour. The conservation constraints say that the traffic out of a given vertex v equals the traffic into v unless v is one of s and t. (We imagine that traffic can arbitrarily materialize or dematerialize at s and t.)

- We can visualize N as a collection of pipes: Each arc $a \in A$ is a pipe, and its capacity $c(a)$ is how much water it can maximally transport in a second. A flow f on N can then be viewed as water flowing through the pipes, where $f(a)$ is the amount of water traveling through a pipe a in a second. The capacity constraints say that no pipe is over its capacity or carries a negative amount of water. The conservation constraints say that at every vertex v other than s and t, the amount of water coming in (that is, $f^-(v)$) equals the amount of water moving out (that is, $f^+(v)$); that is, there are no leaks and no water being injected into the system other than at s and t. This is why s is called the "source" and t is the "sink". A slightly counterintuitive aspect of this visualization is that each pipe has a direction, and water can only flow in that one direction (from source to target). That said, you can always model an undirected pipe by having two pipes of opposite directions.
- We can regard N as a money transfer scheme: Each vertex $v \in V$ is a bank account, and the goal is to transfer some money from s to t. All other vertices v act as middlemen. Each arc $a \in A$ corresponds to a possibility of transfer from its source to its target; the maximum amount that can be transferred on this arc is $c(a)$. A flow describes a way in which money is transferred such that each middleman vertex $v \in V \setminus \{s,t\}$ ends up receiving exactly as much money as it gives away.

Needless to say, these visualizations have been chosen for their intuitive grasp; the real-life applications of network flows are somewhat different.

Remark 1.1.9. Flows on a network N can be viewed as a generalization of paths on the underlying digraph D. Indeed, if $\mathbf{p}$ is a path from s to t on the digraph $D = (V, A, \psi)$ underlying a network N, then we can define a flow $f_{\mathbf{p}}$ on N as follows:

$$f_{\mathbf{p}}(a) = \begin{cases} 1, & \text{if } a \text{ is an arc of } \mathbf{p}; \\ 0, & \text{otherwise} \end{cases} \qquad \text{for each } a \in A,$$

provided that all arcs of $\mathbf{p}$ have capacity ≥ 1. An example of such a flow is the flow g in Example 1.1.8.

Next, we define certain numbers related to any flow on a network:

Definition 1.1.10. Let N be a network consisting of a multidigraph $D = (V, A, \psi)$, a source $s \in V$, a sink $t \in V$ and a capacity function $c : A \to \mathbb{N}$. Let $f : A \to \mathbb{N}$ be an arbitrary map (e.g., a flow on N). Then:

(a) For each vertex $v \in V$, we set

$$f^-(v) := \sum_{\substack{a \in A \text{ is an arc} \\ \text{with target } v}} f(a) \qquad \text{and} \qquad f^+(v) := \sum_{\substack{a \in A \text{ is an arc} \\ \text{with source } v}} f(a).$$

We call $f^-(v)$ the **inflow** of f into v, and we call $f^+(v)$ the **outflow** of f from v.

(b) We define the **value** of the map f to be the number $f^+(s) - f^-(s)$. This value is denoted by $|f|$.

Example 1.1.11. The flow f in Example 1.1.8 satisfies

$$\begin{aligned} f^+(s) &= 3, \qquad f^-(s) = 0, \\ f^+(u) &= f^-(u) = 2, \\ f^+(p) &= f^-(p) = 1, \\ f^+(v) &= f^-(v) = 1, \\ f^+(q) &= f^-(q) = 2,, \\ f^+(t) &= 0, \qquad f^-(t) = 3 \end{aligned}$$

and has value $|f| = 3$. The flow g in Example 1.1.8 has value $|g| = 1$. More generally, the flow $f_{\mathbf{p}}$ in Remark 1.1.9 always has value $|f_{\mathbf{p}}| = 1$.

Example 1.1.12. For any network N, we can define the **zero flow** on N. This is the flow $0_A : A \to \mathbb{N}$ that sends each arc $a \in A$ to 0. This flow has value $|0_A| = 0$.

Now we can state an important optimization problem, known as the **maximum flow problem**: Given a network N, how can we find a flow of maximum possible value?

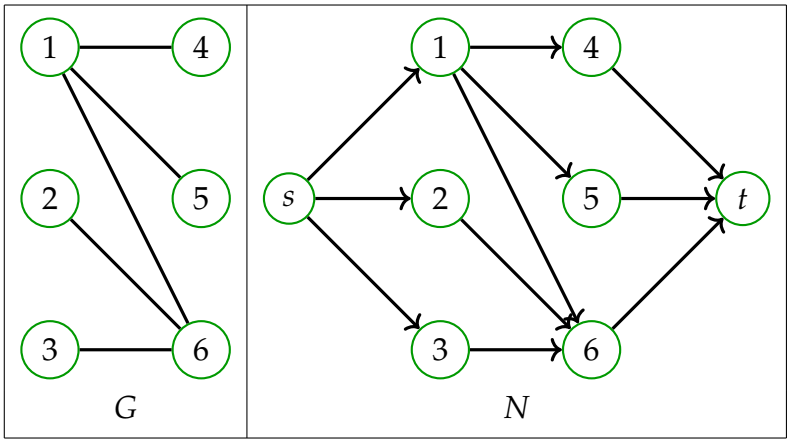
Example 1.1.13. Finding a maximum matching in a bipartite graph is a particular case of the maximum flow problem.

Indeed, let (G, X, Y) be a bipartite graph. Then, we can transform this graph into a network N as follows:

- Add two new vertices s and t.

- Turn each edge e of G into an arc $\overrightarrow{e}$ whose source is the X-endpoint of e (that is, the endpoint of e that belongs to X) and whose target is the Y-endpoint of e (that is, the endpoint of e that belongs to Y).
- Add an arc from s to each vertex in X.
- Add an arc from each vertex in Y to t.
- Assign to each arc the capacity 1.

Here is an example of a bipartite graph (G, X, Y) (as usual, drawn with the X-vertices on the left and with the Y-vertices on the right) and the corresponding network N:

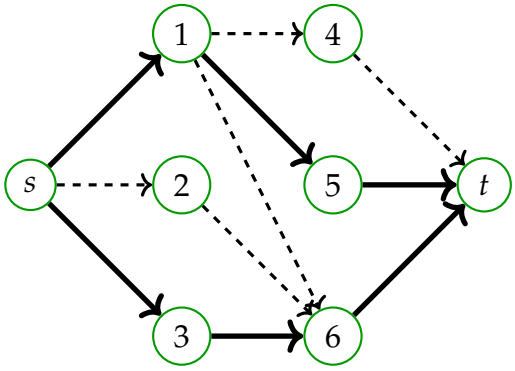


(we are not showing the capacities of the arcs, since they are all equal to 1).

The flows of this network N are in bijection with the matchings of G. Namely, if f is a flow on N, then the set

$$\left\{e \in \mathrm{E}(G) \mid f\left(\overrightarrow{e}\right) = 1\right\}$$

is a matching of G. Conversely, if M is a matching of G, then we obtain a flow f on N by assigning the arc flow 1 to all arcs of the form $\overrightarrow{e}$ where $e \in M$, as well as assigning the arc flow 1 to every new arc that joins s or t to a vertex matched in M. All other arcs are assigned the arc flow 0. For instance, in our above example, the matching $\{15,\ 36\}$ corresponds to the following flow:



,

where we are using the convention that an arc a with $f(a) = 0$ is drawn dashed whereas an arc a with $f(a) = 1$ is drawn boldfaced (thankfully, the only possibilities for $f(a)$ are 0 and 1, because all capacities are 1).

One nice property of this bijection is that if a flow f corresponds to a matching M, then $|f| = |M|$. Thus, finding a flow of maximum value means finding a matching of maximum size.

(See [17s-lec16, Proposition 1.36 till Proposition 1.40] for details and proofs; that said, the proofs are straightforward and you will probably "see" them just by starting at an example.)

1.2. Basic properties of flows

Before we approach the maximum flow problem, let us prove some simple observations about flows:

Proposition 1.2.1. Let N be a network consisting of a multidigraph $D = (V, A, \psi)$, a source $s \in V$, a sink $t \in V$ and a capacity function $c : A \to \mathbb{N}$. Let $f : A \to \mathbb{N}$ be a flow on N. Then,

$$\begin{aligned} |f| &= f^+(s) - f^-(s) \\ &= f^-(t) - f^+(t). \end{aligned}$$

Proof. We have

$$\sum_{v \in V} f^+(v) = \underbrace{\sum_{v \in V} \sum_{\substack{a \in A \text{ is an arc} \\ \text{with source } v}} f(a)}_{= \sum_{a \in A}} \qquad \left(\text{since } f^+(v) \text{ is defined as } \sum_{\substack{a \in A \text{ is an arc} \\ \text{with source } v}} f(a) \right)$$
$$= \sum_{a \in A} f(a)$$

(note that this is a generalization of the familiar fact that $\sum_{v \in V} \deg^+ v = |A|$). Similarly, $\sum_{v \in V} f^-(v) = \sum_{a \in A} f(a)$. Hence,

$$\sum_{v \in V} \left(f^-(v) - f^+(v) \right) = \underbrace{\sum_{v \in V} f^-(v)}_{= \sum_{a \in A} f(a)} - \underbrace{\sum_{v \in V} f^+(v)}_{= \sum_{a \in A} f(a)} = 0. \tag{1}$$

However, by the conservation constraints, we have $f^-(v) = f^+(v)$ for each $v \in V \setminus \{s, t\}$. In other words, $f^-(v) - f^+(v) = 0$ for each $v \in V \setminus \{s, t\}$. Thus,

in the sum $\sum_{v \in V} \left(f^-(v) - f^+(v)\right)$, all addends are 0 except for the addends for $v = s$ and for $v = t$. Hence, the sum boils down to these two addends:

$$\sum_{v \in V} \left(f^-(v) - f^+(v)\right) = \left(f^-(s) - f^+(s)\right) + \left(f^-(t) - f^+(t)\right).$$

Comparing this with (1), we obtain

$$\left(f^-(s) - f^+(s)\right) + \left(f^-(t) - f^+(t)\right) = 0,$$

so that

$$f^-(t) - f^+(t) = -\left(f^-(s) - f^+(s)\right) = f^+(s) - f^-(s) = |f|$$

(by the definition of $|f|$). This proves Proposition 1.2.1. □

Proposition 1.2.2. Let N be a network consisting of a multidigraph $D = (V, A, \psi)$, a source $s \in V$, a sink $t \in V$ and a capacity function $c : A \to \mathbb{N}$. Let $f : A \to \mathbb{N}$ be a flow on N. Let S be a subset of V. Then:

(a) We have

$$f\left(S, \overline{S}\right) - f\left(\overline{S}, S\right) = \sum_{v \in S} \left(f^+(v) - f^-(v)\right).$$

(Recall that we are using Definition 1.1.5 here, so that $f(P, Q)$ means $\sum_{a \in [P,Q]} f(a)$.)

(b) Assume that $s \in S$ and $t \notin S$. Then,

$$|f| = f\left(S, \overline{S}\right) - f\left(\overline{S}, S\right).$$

(c) Assume that $s \in S$ and $t \notin S$. Then,

$$|f| \leq c\left(S, \overline{S}\right).$$

(d) Assume that $s \in S$ and $t \notin S$. Then, $|f| = c\left(S, \overline{S}\right)$ if and only if

$$\left(f(a) = 0 \text{ for all } a \in \left[\overline{S}, S\right]\right)$$

and

$$\left(f(a) = c(a) \text{ for all } a \in \left[S, \overline{S}\right]\right).$$

Proof. Let me first make these claims intuitive in terms of the "money transfer scheme" model for our network. Consider S as a country. Then, $f\left(S, \overline{S}\right)$ is the "export" from this country S (that is, the total wealth exported from S), whereas

$f\left(\overline{S}, S\right)$ is the "import" into this country S (that is, the total wealth imported into S). Thus, part **(a)** of the proposition is saying that the "net export" of S (that is, the export from S minus the import into S) can be computed by summing the "outflow minus inflow" values of all accounts in S. This should match the intuition for exports and imports (in particularly, any transfers that happen within S should cancel out when we sum the "outflow minus inflow" values of all accounts in S). Part **(b)** says that if the country S contains the source s but not the sink t (that is, the goal of the network is to transfer money out of the country), then the total value transferred is actually the net export of S. Part **(c)** claims that this total value is no larger than the total "export capacity" $c\left(S, \overline{S}\right)$ (that is, the total capacity of the "export arcs" $a \in \left[S, \overline{S}\right]$). Part **(d)** says that if equality holds in this inequality (i.e., if the total value equals the total export capacity), then each "import arc" $a \in \left[\overline{S}, S\right]$ is unused (i.e., nothing is imported into S), whereas each "export arc" $a \in \left[S, \overline{S}\right]$ is used to its full capacity.

I hope this demystifies all claims of the proposition. But for the sake of completeness, here are rigorous proofs (though rather terse ones, since I assume you have seen enough manipulations of sum to fill in the details):

(a) This follows from

$$\begin{aligned}
\sum_{v \in S}\left(f^{+}(v)-f^{-}(v)\right) &= \underbrace{\sum_{v \in S} f^{+}(v)}_{\substack{=f(S,V) \\ \text{(why?)}}} - \underbrace{\sum_{v \in S} f^{-}(v)}_{\substack{=f(V,S) \\ \text{(why?)}}} \\
&= \underbrace{f(S,V)}_{\substack{=f(S,S)+f\left(S,\overline{S}\right) \\ \text{(since } V \text{ is the union of the} \\ \text{two disjoint sets } S \text{ and } \overline{S}\text{)}}} - \underbrace{f(V,S)}_{\substack{=f(S,S)+f\left(\overline{S},S\right) \\ \text{(since } V \text{ is the union of the} \\ \text{two disjoint sets } S \text{ and } \overline{S}\text{)}}} \\
&= f(S,S)+f\left(S,\overline{S}\right)-\left(f(S,S)+f\left(\overline{S},S\right)\right) = f\left(S,\overline{S}\right)-f\left(\overline{S},S\right).
\end{aligned}$$

(b) We have $S \setminus \{s\} \subseteq V \setminus \{s,t\}$ (since $t \notin S$). From part **(a)**, we obtain

$$\begin{aligned}
f\left(S,\overline{S}\right)-f\left(\overline{S},S\right) &= \sum_{v \in S}\left(f^{+}(v)-f^{-}(v)\right) \\
&= \underbrace{\left(f^{+}(s)-f^{-}(s)\right)}_{\substack{=|f| \\ \text{(by the definition of } |f|\text{)}}} + \sum_{v \in S \setminus \{s\}} \underbrace{\left(f^{+}(v)-f^{-}(v)\right)}_{\substack{=0 \\ \text{(by the conservation constraints,} \\ \text{since } v \in S \setminus \{s\} \subseteq V \setminus \{s,t\}\text{)}}} \qquad \left(\text{since } s \in S\right) \\
&= |f|.
\end{aligned}$$

This proves part **(b)**.

(c) The capacity constraints yield that $f(a) \leq c(a)$ for each arc $a \in A$. Summing up these inequalities over all $a \in \left[S, \overline{S}\right]$, we obtain $f\left(S,\overline{S}\right) \leq c\left(S,\overline{S}\right)$. The capacity constraints furthermore yield that $f(a) \geq 0$ for each arc $a \in A$. Summing up these inequalities over all $a \in \left[\overline{S}, S\right]$, we obtain $f\left(\overline{S},S\right) \geq 0$. Hence, part **(b)** yields

$$|f| = \underbrace{f\left(S,\overline{S}\right)}_{\leq c\left(S,\overline{S}\right)} - \underbrace{f\left(\overline{S},S\right)}_{\geq 0} \leq c\left(S,\overline{S}\right).$$

This proves part **(c)**.

(d) We must characterize the equality case in part **(c)**. However, recall the proof of part **(c)**: We obtained the inequality $|f| \leq c\left(S, \overline{S}\right)$ by summing up the inequalities $f(a) \leq c(a)$ over all arcs $a \in \left[S, \overline{S}\right]$ and subtracting the sum of the inequalities $f(a) \geq 0$ over all arcs $a \in \left[\overline{S}, S\right]$. Hence, in order for the inequality $|f| \leq c\left(S, \overline{S}\right)$ to become an equality, it is necessary and sufficient that all the inequalities involved – i.e., the inequalities $f(a) \leq c(a)$ for all arcs $a \in \left[S, \overline{S}\right]$ as well as the inequalities $f(a) \geq 0$ for all arcs $a \in \left[\overline{S}, S\right]$ – become equalities. In other words, it is necessary and sufficient that we have

$$\left(f(a) = 0 \text{ for all } a \in \left[\overline{S}, S\right]\right)$$

and

$$\left(f(a) = c(a) \text{ for all } a \in \left[S, \overline{S}\right]\right).$$

This proves Proposition 1.2.2 **(d)**. □

1.3. The max-flow-min-cut theorem

One more definition, before we get to the hero of this story:

Definition 1.3.1. Let N be a network consisting of a multidigraph $D = (V, A, \psi)$, a source $s \in V$, a sink $t \in V$ and a capacity function $c : A \to \mathbb{N}$. Then:

(a) A **cut** of N shall mean a subset of A that has the form $\left[S, \overline{S}\right]$, where S is a subset of V satisfying $s \in S$ and $t \notin S$.

(b) The **capacity** of a cut $\left[S, \overline{S}\right]$ is defined to be the number $c\left(S, \overline{S}\right) = \sum_{a \in \left[S, \overline{S}\right]} c(a)$.

Example 1.3.2. Let us again consider the network from Example 1.1.2. Then, $\left[\{s, u\}, \overline{\{s, u\}}\right] = \{sp,\ uv,\ uq\}$ is a cut of this network, and its capacity is $c\left(\{s, u\}, \overline{\{s, u\}}\right) = 5$.

Now, Proposition 1.2.2 **(c)** says that the value of any flow f can never be larger than the capacity of any cut $\left[S, \overline{S}\right]$. Thus, in particular, the maximum value of a flow is $\leq$ to the minimum capacity of a cut.

Furthermore, Proposition 1.2.2 **(d)** says that if this inequality is an equality – i.e., if the value of some flow f equals the capacity of some cut $\left[S, \overline{S}\right]$ –, then the flow f must use each arc that crosses the cut in the right direction (from S to $\overline{S}$) to its full capacity and must not use any of the arcs that cross the cut in the wrong direction (from $\overline{S}$ to S).

It turns out that this inequality actually **is** an equality for any maximum flow and any minimum cut:

Theorem 1.3.3 (max-flow-min-cut theorem)**.** Let N be a network consisting of a multidigraph $D = (V, A, \psi)$, a source $s \in V$, a sink $t \in V$ and a capacity function $c : A \to \mathbb{N}$. Then,

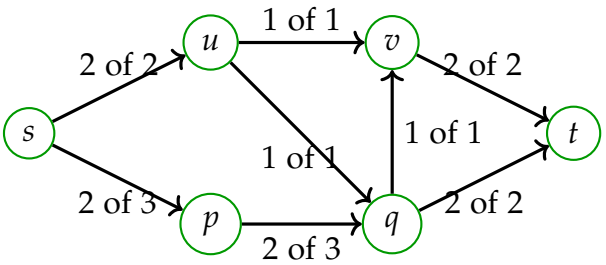
$$\max\left\{|f| \;\mid\; f \text{ is a flow}\right\} = \min\left\{c\left(S, \overline{S}\right) \;\mid\; S \subseteq V;\ s \in S;\ t \notin S\right\}.$$

In other words, the maximum value of a flow equals the minimum capacity of a cut.

We shall soon sketch a proof of this theorem that doubles as a fairly efficient (polynomial-time) algorithm for finding both a maximum flow (i.e., a flow of maximum value) and a minimum cut (i.e., a cut of minimum capacity). The algorithm is known as the **Ford-Fulkerson algorithm**, and is sufficiently fast to be useful in practice. The idea is to start by having f be the zero flow (i.e., the flow from Example 1.1.12), and then gradually increase its value $|f|$ by making changes to some of its arc flows $f(a)$.

Of course, we cannot unilaterally change the arc flow $f(a)$ on a single arc, since this will (usually) mess up the conservation constraints. Thus, if we change $f(a)$, then we will also have to change $f(b)$ for some other arcs $b \in A$ to make the result a flow again. One way to do this is to increase all arc flows $f(a)$ along some path from s to t. Here is an example of such an increase:

Example 1.3.4. Consider the flow f from Example 1.1.8. We can increase the arc flows $f(sp)$, $f(pq)$, $f(qv)$, $f(vt)$ of f on all the arcs of the path (s, p, q, v, t) (since neither of these arcs is used to its full capacity). As a result, we obtain the following flow h:

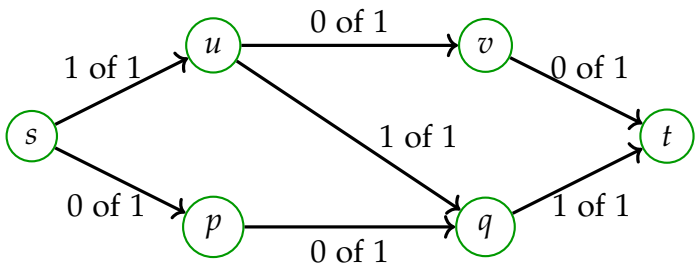


,

whose value $|h|$ is 4. It is easy to see that this is actually the maximum value of a flow on our network (since $|h| = 4$ equals the capacity $c\left(\overline{\{t\}}, \{t\}\right)$ of the cut $\left[\overline{\{t\}}, \{t\}\right]$, but Proposition 1.2.2 **(c)** tells us that the value of any flow is $\leq$ to the capacity of any cut).

However, simple increases like the one we just did are not always enough to find a maximum flow. They can leave us stuck at a "local maximum" – i.e., at a flow which does not have any more paths from s to t that can be used for any further increases (i.e., any path from s to t contains an arc that is already used to its full capacity), yet is not a maximum flow. Here is an example:

Example 1.3.5. Consider the following network and flow:

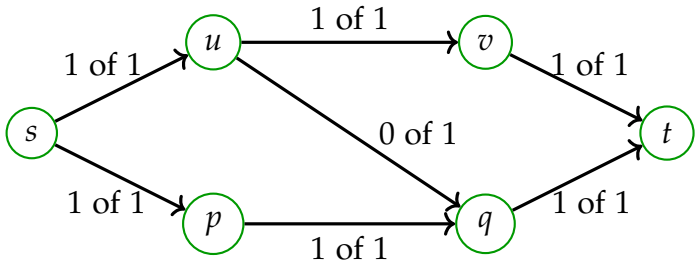


.

This flow is not maximum, but each path from s to t has at least one arc that is used to its full capacity. Thus, we cannot improve this flow by increasing all its arc flows on any given path from s to t.

The trick to get past this hurdle is to use a "zig-zag path" – i.e., not a literal path, but rather a sequence $(v_0, a_1, v_1, a_2, v_2, \ldots, a_k, v_k)$ of vertices and arcs that can use arcs both in the forward and backward directions (i.e., any $i \in \{1, 2, \ldots, k\}$ has to satisfy either $\psi(a_i) = (v_{i-1}, v_i)$ or $\psi(a_i) = (v_i, v_{i-1})$). Instead of increasing the flow on all arcs of this "path", we do something slightly subtler: On the forward arcs, we increase the flow; on the backward arcs, we decrease it (all by the same amount). This, too, preserves the conservation constraints (think about why; we will soon see a rigorous proof), so it is a valid way of increasing the value of a flow. Here is an example:

Example 1.3.6. Consider the flow in Example 1.3.5. The underlying digraph has a "zig-zag path" (s, p, q, u, v, t), which uses the arc uq in the backward direction. We can decrease the arc flows of f on all forward arcs sp, pq, uv and vt of this "zig-zag path", and decrease it on the backward arc uq. As a result, we obtain the flow



.

This new flow has value 2, and can easily be seen to be a maximum flow.

Good news: Allowing ourselves to use "zig-zag paths" like this (rather than literal paths only), we never get stuck at a non-maximum flow; we can always increase the value further and further until we eventually arrive at a maximum flow.

In order to prove this, we introduce some convenient notations. We prefer not to talk about "zig-zag paths", but rather reinterpret these "zig-zag paths" as (literal) paths of an appropriately chosen digraph. This has the advantage of allowing us to use known properties of paths without having to first generalize them to "zig-zag paths".

The appropriately chosen digraph is the so-called **residual digraph** of a flow; it is defined as follows:

Definition 1.3.7. Let N be a network consisting of a multidigraph $D = (V, A, \psi)$, a source $s \in V$, a sink $t \in V$ and a capacity function $c : A \to \mathbb{N}$.

(a) For each arc $a \in A$, we introduce a new arc a^{-1}, which should act like a reversal of the arc a (that is, its source should be the target of a, and its target should be the source of a). We don't add these new arcs a^{-1} to our digraph D, but we keep them ready for use in a different digraph (which we will define below).

Here is what this means in rigorous terms: For each arc $a \in A$, we introduce a new object, which we call a^{-1}. We let A^{-1} be the set of these new objects a^{-1} for $a \in A$. We extend the map $\psi : A \to V \times V$ to a map $\widehat{\psi} : A \cup A^{-1} \to V \times V$ as follows: For each $a \in A$, we let

$$\widehat{\psi}(a) = (u, v) \qquad \text{and} \qquad \widehat{\psi}\left(a^{-1}\right) = (v, u),$$

where u and v are defined by $(u, v) = \psi(a)$.

For each arc $a \in A$, we shall refer to the new arc a^{-1} as the **reversal** of a, and conversely, we shall refer to the original arc a as the **reversal** of a^{-1}. We set $\left(a^{-1}\right)^{-1} := a$ for each $a \in A$.

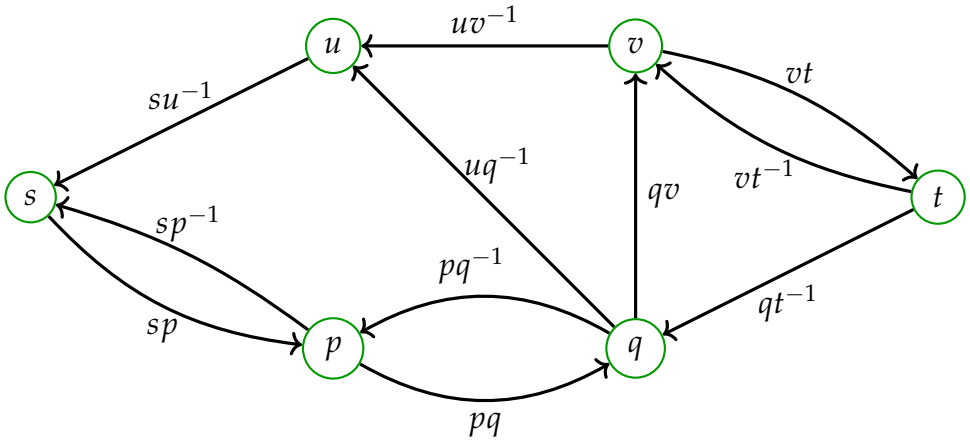
We shall refer to the arcs $a \in A$ as **forward arcs**, and to their reversals a^{-1} as **backward arcs**.

(b) Let $f : A \to \mathbb{N}$ be any flow on N. We define the **residual digraph** D_f of this flow f to be the multidigraph $\left(V, A_f, \psi_f\right)$, where

$$A_f = \{a \in A \mid f(a) < c(a)\} \cup \left\{a^{-1} \mid a \in A;\ f(a) > 0\right\}$$

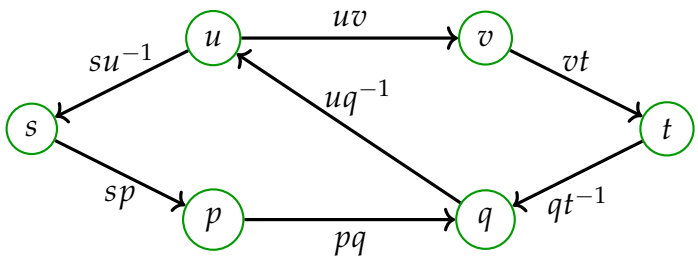
and $\psi_f := \widehat{\psi}\,|_{A_f}$. (This is usually not a subdigraph of D.) Thus, the residual digraph D_f has the same vertices as V, but its arcs are those arcs of D that are not used to their full capacity by f as well as the reversals of all arcs of D that are used by f.

Example 1.3.8. Let f be the flow f from Example 1.1.8. Then, the residual digraph D_f is



Notice that the digraph D_f has cycles even though D has none!

Example 1.3.9. Let f be the non-maximum flow from Example 1.3.5. Then, the residual digraph D_f is



This digraph D_f has a path from s to t, which corresponds precisely to the "zig-zag path" (s, p, q, u, v, t) we found in Example 1.3.6.

You can think of the residual digraph D_f as follows: Each arc of D_f corresponds to an opportunity to change an arc flow $f(a)$; namely, a forward arc a of D_f means that $f(a)$ can be increased, whereas a backward arc a^{-1} of D_f means that $f(a)$ can be decreased. Hence, the paths of the residual digraph D_f are the "zig-zag paths" of D that allow the flow f to be increased (on forward arcs) or decreased (on backward arcs) as in Example 1.3.6. Thus, using D_f, we can avoid talking about "zig-zag paths".

The following crucial lemma tells us that such "zig-zag path increases" are valid (i.e., turn flows into flows), and are sufficient to find a maximum flow (i.e., if no more "zig-zag path increases" are possible, then our flow is already maximal):

Lemma 1.3.10 (augmenting path lemma)**.** Let N be a network consisting of a multidigraph $D = (V, A, \psi)$, a source $s \in V$, a sink $t \in V$ and a capacity function $c : A \to \mathbb{N}$. Let $f : A \to \mathbb{N}$ be a flow.

(a) If the digraph D_f has a path from s to t, then the network N has a flow f' with a larger value than f.

(b) If the digraph D_f has no path from s to t, then the flow f has maximum value (among all flows on N), and there exists a subset S of V satisfying $s \in S$ and $t \notin S$ and $|f| = c\left(S, \overline{S}\right)$.

Proof. **(a)** Assume that the digraph D_f has a path from s to t. Pick such a path, and call it $\mathbf{p}$. Each arc of $\mathbf{p}$ is an arc of D_f.

For each forward arc $a \in A$ that appears in $\mathbf{p}$, we have $f(a) < c(a)$ (since a is an arc of D_f), and thus we can increase the arc flow $f(a)$ by some positive $\varepsilon \in \mathbb{N}$ (namely, by any $\varepsilon < c(a) - f(a)$) without violating the capacity constraints.[1]

For each backward arc $a^{-1} \in A^{-1}$ that appears in $\mathbf{p}$, we have $f(a) > 0$ (since a^{-1} is an arc of D_f), and thus we can decrease the arc flow $f(a)$ by some positive $\varepsilon \in \mathbb{N}$ (namely, by any $\varepsilon \leq f(a)$) without violating the capacity constraints.

Let now

$$\varepsilon := \min\Big(\{c(a) - f(a) \mid a \in A \text{ is a forward arc that appears in } \mathbf{p}\} \cup \left\{ f(a) \mid a^{-1} \in A^{-1} \text{ is a backward arc that appears in } \mathbf{p} \right\} \Big).$$

This ε is a positive integer (since it is a minimum of a set of positive integers[2]). Let $f' : A \to \mathbb{N}$ be the map obtained from f as follows:

- For each forward arc $a \in A$ that appears in $\mathbf{p}$, we increase the arc flow $f(a)$ by ε (that is, we set $f'(a) := f(a) + \varepsilon$).
- For each backward arc $a^{-1} \in A^{-1}$ that appears in $\mathbf{p}$, we decrease the arc flow $f(a)$ by ε (that is, we set $f'(a) := f(a) - \varepsilon$).
- For all other arcs a of D, we keep the arc flow $f(a)$ unchanged (i.e., we set $f'(a) := f(a)$).

[1] Of course, such a unilateral increase will likely violate the conservation constraints.

[2] because

- for each forward arc $a \in A$ that appears in $\mathbf{p}$, we have $f(a) < c(a)$ and thus $c(a) - f(a) > 0$;
- for each backward arc $a^{-1} \in A^{-1}$ that appears in $\mathbf{p}$, we have $f(a) > 0$.

This new map f' still satisfies the capacity constraints[3]. We claim that it also satisfies the conservation constraints. To check this, we have to verify that $(f')^{-}(v) = (f')^{+}(v)$ for each vertex $v \in V \setminus \{s,t\}$. So let us do this.

Let $v \in V \setminus \{s,t\}$ be a vertex. We know that $f^{-}(v) = f^{+}(v)$ (since f is a flow). We must prove that $(f')^{-}(v) = (f')^{+}(v)$.

The path $\mathbf{p}$ is a path from s to t. Thus, it neither starts nor ends at v (since $v \in V \setminus \{s,t\}$). Hence, if v is a vertex of $\mathbf{p}$, then the path $\mathbf{p}$ enters v by some arc and exits v by another. Hence, we are in one of the following five cases:

Case 1: The vertex v is not a vertex of the path $\mathbf{p}$.

Case 2: The path $\mathbf{p}$ enters v by a forward arc and exits v by a forward arc.

Case 3: The path $\mathbf{p}$ enters v by a forward arc and exits v by a backward arc.

Case 4: The path $\mathbf{p}$ enters v by a backward arc and exits v by a forward arc.

Case 5: The path $\mathbf{p}$ enters v by a backward arc and exits v by a backward arc.

Now, we can prove $(f')^{-}(v) = (f')^{+}(v)$ in each of these five cases by hand. Here is how this can be done in the first three cases:

First, we consider Case 1. In this case, v is not a vertex of the path $\mathbf{p}$. Hence, each arc $a \in A$ with target v satisfies $f'(a) = f(a)$ (because neither a nor a^{-1} appears in $\mathbf{p}$). Therefore, $(f')^{-}(v) = f^{-}(v)$. Similarly, $(f')^{+}(v) = f^{+}(v)$. Hence, $(f')^{-}(v) = f^{-}(v) = f^{+}(v) = (f')^{+}(v)$. Thus, we have proved $(f')^{-}(v) = (f')^{+}(v)$ in Case 1.

Let us now consider Case 2. In this case, the path $\mathbf{p}$ enters v by a forward arc and exits v by a forward arc. Let b be the former arc, and c the latter. Then, both b and c are arcs of D, and the vertex v is the target of b and the source of c. The definition of f' yields that $f'(b) = f(b) + \varepsilon$, whereas each other arc $a \in A$ with target v satisfies $f'(a) = f(a)$. Hence, $(f')^{-}(v) = f^{-}(v) + \varepsilon$. Similarly, using the arc c, we see that $(f')^{+}(v) = f^{+}(v) + \varepsilon$. Hence, $(f')^{-}(v) = \underbrace{f^{-}(v)}_{=f^{+}(v)} + \varepsilon = f^{+}(v) + \varepsilon = (f')^{+}(v)$. Thus, we have proved $(f')^{-}(v) = (f')^{+}(v)$ in Case 2.

Let us next consider Case 3. In this case, the path $\mathbf{p}$ enters v by a forward arc and exits v by a backward arc. Let b be the former arc, and c^{-1} the latter. Then, both b and c are arcs of D, and the vertex v is the target of both b and c. The definition of f' yields that $f'(b) = f(b) + \varepsilon$ (since $\mathbf{p}$ uses the forward arc b) and $f'(c) = f(c) - \varepsilon$ (since $\mathbf{p}$ uses the backward arc c^{-1}), whereas each other arc $a \in A$ with target v satisfies $f'(a) = f(a)$. Hence, $(f')^{-}(v) = f^{-}(v) + \varepsilon - \varepsilon = f^{-}(v)$. Moreover, $(f')^{+}(v) = f^{+}(v)$ (since none of the arcs of D with source

[3] since the definition of ε shows that

- for each forward arc a that appears in $\mathbf{p}$, we have $\varepsilon \leq c(a) - f(a)$ and thus $f(a) + \varepsilon \leq c(a)$;
- for each backward arc $a^{-1} \in A^{-1}$ that appears in $\mathbf{p}$, we have $\varepsilon \leq f(a)$ and thus $f(a) - \varepsilon \geq 0$.

v appears in $\mathbf{p}$, nor does its reversal). Hence, $(f')^{-}(v) = f^{-}(v) = f^{+}(v) = (f')^{+}(v)$. Thus, we have proved $(f')^{-}(v) = (f')^{+}(v)$ in Case 3.

The other two cases are similar (Case 4 is analogous to Case 3, while Case 5 is analogous to Case 2). Thus, altogether, we have proved $(f')^{-}(v) = (f')^{+}(v)$ in all five cases.

Forget that we fixed v. We thus have shown that each vertex $v \in V \setminus \{s,t\}$ satisfies $(f')^{-}(v) = (f')^{+}(v)$. In other words, the map f' satisfies the conservation constraints. Since f' also satisfies the capacity constraints, we thus conclude that f' is a flow.

What is the value $|f'|$ of this flow? The path $\mathbf{p}$ starts at s, so it exits s by some arc γ (it must have at least one arc, since $s \neq t$) and never comes back to s again. If this arc γ is a forward arc b, then $f'(b) = f(b) + \varepsilon$ and therefore $(f')^{+}(s) = f^{+}(s) + \varepsilon$ and $(f')^{-}(s) = f^{-}(s)$. If this arc γ is a backward arc c^{-1}, then $f'(c) = f(c) - \varepsilon$ and therefore $(f')^{-}(s) = f^{-}(s) - \varepsilon$ and $(f')^{+}(s) = f^{+}(s)$. Thus,

$$\begin{aligned}
\left(f'\right)^{+}(s) - \left(f'\right)^{-}(s) &= \begin{cases} \left(f^{+}(s) + \varepsilon\right) - f^{-}(s), & \text{if } \gamma \text{ is a forward arc;} \\ f^{+}(s) - \left(f^{-}(s) - \varepsilon\right), & \text{if } \gamma \text{ is a backward arc} \end{cases} \\
&= \begin{cases} f^{+}(s) - f^{-}(s) + \varepsilon, & \text{if } \gamma \text{ is a forward arc;} \\ f^{+}(s) - f^{-}(s) + \varepsilon, & \text{if } \gamma \text{ is a backward arc} \end{cases} \\
&= \underbrace{f^{+}(s) - f^{-}(s)}_{\substack{=|f| \\ \text{(by the definition of } |f|\text{)}}} + \varepsilon = |f| + \varepsilon.
\end{aligned}$$

However, the definition of the value $|f'|$ yields

$$\left|f'\right| = \left(f'\right)^{+}(s) - \left(f'\right)^{-}(s) = |f| + \varepsilon > |f| \qquad (\text{since } \varepsilon > 0).$$

In other words, the flow f' has a larger value than f. Thus, we have found a flow f' with a larger value than f. This proves Lemma 1.3.10 **(a)**.

(b) Assume that the digraph D_f has no path from s to t. Define a subset S of V by

$$S = \left\{v \in V \mid \text{the digraph } D_f \text{ has a path from } s \text{ to } v\right\}.$$

Then, $s \in S$ (because the trivial path (s) is a path from s to s) and $t \notin S$ (since we assumed that D_f has no path from s to t). We shall next show that $|f| = c\left(S, \overline{S}\right)$.

Indeed, we shall obtain this from Proposition 1.2.2 **(d)**. To do so, we will first show that

$$\left(f(a) = 0 \text{ for all } a \in \left[\overline{S}, S\right]\right) \tag{2}$$

and

$$\left(f(a) = c(a) \text{ for all } a \in \left[S, \overline{S}\right]\right). \tag{3}$$

[*Proof of (2):* Let $a \in \left[\overline{S}, S\right]$. Assume that $f(a) \neq 0$. Thus, $f(a) > 0$ (since the capacity constraints yield $f(a) \geq 0$). Hence, the backward arc a^{-1} is an arc of

the residual digraph D_f. Let u be the source of a, and let v be the target of a. Since $a \in \left[\overline{S}, S\right]$, we thus have $u \in \overline{S}$ and $v \in S$. From $v \in S$, we see that the digraph D_f has a path from s to v. Let $\mathbf{q}$ be this path. Appending the backward arc a^{-1} (which is an arc from v to u) and the vertex u to this path $\mathbf{q}$ (at the end), we obtain a walk from s to u in D_f. Hence, D_f has a walk from s to u, thus also a path from s to u (by Corollary 1.2.8 in Lecture 10). This entails $u \in S$ (by the definition of S). However, this contradicts $u \in \overline{S} = V \setminus S$. This contradiction shows that our assumption (that $f(a) \neq 0$) was wrong. Therefore, $f(a) = 0$. This proves (2).]

[*Proof of (3):* Let $a \in \left[S, \overline{S}\right]$. Assume that $f(a) \neq c(a)$. Thus, $f(a) < c(a)$ (since the capacity constraints yield $f(a) \leq c(a)$). Hence, the forward arc a is an arc of the residual digraph D_f. Let u be the source of a, and let v be the target of a. Since $a \in \left[S, \overline{S}\right]$, we thus have $u \in S$ and $v \in \overline{S}$. From $u \in S$, we see that the digraph D_f has a path from s to u. Let $\mathbf{q}$ be this path. Appending the forward arc a (which is an arc from u to v) and the vertex v to this path $\mathbf{q}$ (at the end), we obtain a walk from s to v in D_f. Hence, D_f has a walk from s to v, thus also a path from s to v (by Corollary 1.2.8 in Lecture 10). This entails $v \in S$ (by the definition of S). However, this contradicts $v \in \overline{S} = V \setminus S$. This contradiction shows that our assumption (that $f(a) \neq c(a)$) was wrong. Therefore, $f(a) = c(a)$. This proves (3).]

Now, Proposition 1.2.2 **(d)** yields that $|f| = c\left(S, \overline{S}\right)$ holds (since (2) and (3) hold).

We have now found a subset S of V satisfying $s \in S$ and $t \notin S$ and $|f| = c\left(S, \overline{S}\right)$. In order to prove Lemma 1.3.10 **(b)**, it suffices to show that the flow f has maximum value (among all flows on N). However, this is now easy: Any flow g on N has value $|g| \leq c\left(S, \overline{S}\right)$ (by Proposition 1.2.2 **(c)**, applied to g instead of f). In other words, any flow g on N has value $|g| \leq |f|$ (since $|f| = c\left(S, \overline{S}\right)$). Thus, the flow f has maximum value. This completes the proof of Lemma 1.3.10 **(b)**. □

We are now ready to prove the max-flow-min-cut theorem (Theorem 1.3.3):

Proof of Theorem 1.3.3. We let $f : A \to \mathbb{N}$ be the zero flow on N (see Example 1.1.12 for its definition). Now, we shall incrementally increase the value $|f|$ of this flow by the following algorithm (known as the **Ford-Fulkerson algorithm**):

1. Construct the residual digraph D_f.
2. If the digraph D_f has a path from s to t, then Lemma 1.3.10 **(a)** shows that the network N has a flow f' with a larger value than f (and furthermore, the proof of Lemma 1.3.10 **(a)** shows how to find such an f' efficiently[4]). Fix such an f', and replace f by f'. Then, go back to step 1.

[4]Of course, this requires an algorithm for finding a path from s to t in D_f. But there are many efficient algorithms for this (see, e.g., homework set #4 exercise 5).

3. If the digraph D_f has no path from s to t, then we end the algorithm.

The replacement of f by f' in Step 2 of this algorithm will be called an **augmentation**. Thus, the algorithm proceeds by repeatedly performing augmentations until this is no longer possible.

I claim that the algorithm will eventually end – i.e., it cannot keep performing augmentations forever. Indeed, each augmentation increases the value $|f|$ of the flow f, and therefore it increases this value $|f|$ by at least 1 (because increasing an integer always means increasing it by at least 1). However, the value $|f|$ is bounded from above by the capacity $c\left(S, \overline{S}\right)$ of an arbitrary cut $\left[S, \overline{S}\right]$ (by Proposition 1.2.2 **(c)**), and thus cannot get increased by 1 more than $c\left(S, \overline{S}\right)$ many times (since its initial value is 0). Therefore, we cannot perform more than $c\left(S, \overline{S}\right)$ many augmentations in sequence.

Thus, the algorithm eventually ends. Let us consider the flow f that is obtained once the algorithm has ended. This flow f has the property that the digraph D_f has no path from s to t. Thus, Lemma 1.3.10 **(b)** shows that the flow f has maximum value (among all flows on N), and there exists a subset S of V satisfying $s \in S$ and $t \notin S$ and $|f| = c\left(S, \overline{S}\right)$. Consider this S.

Since the flow f has maximum value, we have

$$|f| = \max\left\{|g| \ \mid \ g \text{ is a flow}\right\}.$$

On the other hand, for each subset T of V satisfying $s \in T$ and $t \notin T$, we have

$$c\left(S, \overline{S}\right) = |f| \leq c\left(T, \overline{T}\right)$$

(by Proposition 1.2.2 **(c)**, applied to T instead of S). Hence,

$$c\left(S, \overline{S}\right) = \min\left\{c\left(T, \overline{T}\right) \ \mid \ T \subseteq V;\ s \in T;\ t \notin T\right\}.$$

Comparing this with

$$c\left(S, \overline{S}\right) = |f| = \max\left\{|g| \ \mid \ g \text{ is a flow}\right\},$$

we obtain

$$\max\left\{|g| \ \mid \ g \text{ is a flow}\right\} = \min\left\{c\left(T, \overline{T}\right) \ \mid \ T \subseteq V;\ s \in T;\ t \notin T\right\}.$$

In other words, the maximum value of a flow equals the minimum capacity of a cut. This proves Theorem 1.3.3. (Of course, we cannot use the letters f and S for the bound variables in $\max\left\{|g| \ \mid \ g \text{ is a flow}\right\}$ and $\min\left\{c\left(T, \overline{T}\right) \ \mid \ T \subseteq V;\ s \in T;\ t \notin T\right\}$, since f and S already stand for a specific flow and a specific set.) $\square$

Remark 1.3.11. All the theorems, propositions and lemmas we proved in this lecture still work if we replace the set $\mathbb{N}$ by the set $\mathbb{Q}_+ := \{\text{nonnegative rational numbers}\}$ or the set $\mathbb{R}_+ :=$

{nonnegative real numbers}. However, their proofs get more complicated. The problem is that if the arc flows of f belong to $\mathbb{Q}_+$ or $\mathbb{R}_+$ rather than $\mathbb{N}$, it is possible for $|f|$ to increase endlessly (cf. Zeno's paradox of Achilles and the tortoise), as we make smaller and smaller improvements to our flow but never achieve (or even approach!) the maximum value.

With rational values, this fortunately cannot happen, since the lowest common denominator of all arc flows $f(a)$ does not change when we perform an augmentation. (To put it differently: The case of rational values can be reduced to the case of integer values by multiplying through with the lowest common denominator.) With real values, however, this misbehavior can occur (see [ForFul74, §I.8] for an example). Fortunately, there is a way to avoid it by choosing a **shortest** path from s to t in D_f at each step. This is known as the **Edmonds-Karp version of the Ford-Fulkerson algorithm** (or, for short, the **Edmonds-Karp algorithm**). Proving that it works takes a bit more work, which we won't do here (see, e.g., [Schrij17, Theorem 4.4]). Incidentally, this technique also helps keep the algorithm fast for integer-valued flows (running time $O\left(|V| \cdot |A|^2\right)$).

1.4. Application: Deriving Hall-König

Now, let us apply the max-flow-min-cut theorem. Recall the following:

Theorem 1.4.1 (Hall-König matching theorem)**.** Let (G, X, Y) be a bipartite graph. Then, there exist a matching M of G and a subset U of X such that

$$|M| \geq |N(U)| + |X| - |U|.$$

We stated this in Lecture 24, and used this to derive the HMT and König's theorem; but we didn't prove Theorem 1.4.1. Let us do this now.

Proof of Theorem 1.4.1 (sketched). (This is an outline; see [17s-lec16, proof of Lemma 1.42] for details.[5]) As explained in Example 1.1.13, we can turn the bipartite graph (G, X, Y) into a network so that the matchings of G become the flows f of this network. The max-flow-min-cut theorem (Theorem 1.3.3) yields that

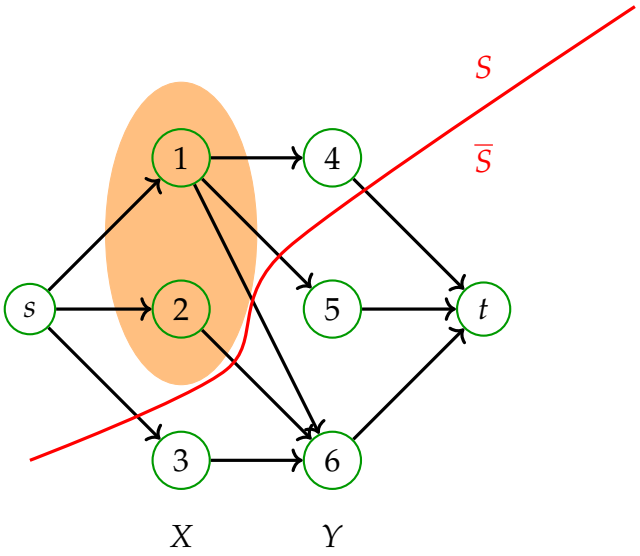
$$\max\{|f| \mid f \text{ is a flow}\} = \min\left\{c\left(S, \overline{S}\right) \mid S \subseteq V;\ s \in S;\ t \notin S\right\},$$

[5]Note that [17s-lec16, Lemma 1.42] is stated only for a simple graph G, not for a multigraph G. However, this really makes no difference here: If (G, X, Y) is a bipartite graph with G being a multigraph, then $\left(G^{\text{simp}}, X, Y\right)$ is a bipartite graph as well, and clearly any matching of G^{simp} yields a matching of G having the same size (and the set $N(U)$ does not change from G to G^{simp} either). Thus, in proving Theorem 1.4.1, we can WLOG assume that G is a simple graph.

where V is the vertex set of the digraph that underlies our network. Thus, there exist a flow f and a cut $\left[S, \overline{S}\right]$ of this network such that $|f| = c\left(S, \overline{S}\right)$. Consider these f and S. Thus, S is a subset of V such that $s \in S$ and $t \notin S$.

Let M be the matching of G corresponding to the flow f (that is, we let M be the set of all edges e of G such that $f\left(\overrightarrow{e}\right) = 1$). Thus, $|M| = |f|$.

Let $U := X \cap S$. Then, U is a subset of X. Here is an illustration of the cut $\left[S, \overline{S}\right]$ on a simple example (the flow f is not shown):



(the orange oval is the set U).

Now, we have

$$|M| = |f| = c\left(S, \overline{S}\right) = \underbrace{c\left(\{s\}, \overline{S}\right)}_{\substack{=|X\setminus U| \\ \text{(why?)}}} + c\left(\underbrace{X \cap S}_{=U}, \overline{S}\right) + \underbrace{c\left(Y \cap S, \overline{S}\right)}_{\substack{=|Y\cap S| \\ \text{(why?)}}}$$

(since S is the union of the disjoint sets $\{s\}$, $X \cap S$ and $Y \cap S$)

$$= \underbrace{|X \setminus U|}_{=|X|-|U|} + \underbrace{c\left(U, \overline{S}\right) + |Y \cap S|}_{\substack{\geq |N(U)| \\ \text{(since each vertex } y \in N(U) \text{ either belongs to } Y\cap S \\ \text{and thus contributes to } |Y\cap S|, \text{ or belongs to } \overline{S} \\ \text{and thus contributes to } c\left(U, \overline{S}\right))}}$$

$$\geq |X| - |U| + |N(U)| = |N(U)| + |X| - |U|.$$

This proves Theorem 1.4.1. □

Further applications of the max-flow-min-cut theorem include:

- A curious fact about rounding matrix entries (stated in terms of a digraph in [Schrij17, Exercise 4.13]): Let A be an $m \times n$-matrix with real entries.

Assume that all row sums[6] of A and all column sums[7] of A are integers. Then, we can round each non-integer entry of A (that is, replace it either by the next-smaller integer or the next-larger integer) in such a way that the resulting matrix has the same row sums as A and the same column sums as A.

- An Euler-Hierholzer-like criterion for the existence of an Eulerian circuit in a "mixed graph" (a general notion of a graph that can contain both undirected edges and directed arcs) [ForFul74, §II.7].

- A proof [Berge91, §6.3] of the Erdös–Gallai theorem, which states that for a given weakly decreasing n-tuple $(d_1 \geq d_2 \geq \cdots \geq d_n)$ of nonnegative integers, there exists a simple graph with n vertices whose n vertices have degrees $d_1, d_2, \ldots, d_n$ if and only if the sum $d_1 + d_2 + \cdots + d_n$ is even and each $i \in \{1, 2, \ldots, n\}$ satisfies

$$\sum_{i=1}^{k} d_i \leq k\left(k-1\right) + \sum_{i=k+1}^{n} \min\left\{d_i, k\right\}.$$

(The "only if" part of this theorem was Exercise 6 on homework set #2.)

References

[17s-lec16] Darij Grinberg, *UMN, Spring 2017, Math 5707: Lecture 16 (flows and cuts in networks)*, 14 May 2022.
`https://www.cip.ifi.lmu.de/~grinberg/t/17s/5707lec16.pdf`

[Berge91] Claude Berge, *Graphs*, North-Holland Mathematical Library **6.1**, 3rd edition, North-Holland 1991.

[ForFul74] L. R. Ford, Jr., D. R. Fulkerson, *Flows in Networks*, 7th printing, Princeton University Press, 1974.

[Schrij17] Alexander Schrijver, *A Course in Combinatorial Optimization*, March 23, 2017.
`https://homepages.cwi.nl/~lex/files/dict.pdf`

[Schrij03] Alexander Schrijver, *Combinatorial Optimization: Polyhedra and Efficiency*, Springer 2003.
See `https://homepages.cwi.nl/~lex/co/` for errata.

[6] A **row sum** of a matrix means the sum of all entries in some row of this matrix. Thus, an $m \times n$-matrix has m row sums.

[7] A **column sum** of a matrix means the sum of all entries in some column of this matrix. Thus, an $m \times n$-matrix has n column sums.