Math 482: Linear Programming[1] — Mikhail Lavrov

Lecture 23: Network Flows

March 30, 2020 — *University of Illinois at Urbana-Champaign*

1 What is a network?

We want to go on to model a fancier kind of problem: problems where some materials need to be transported from one location to another. (For example, imagine supplying water or electricity to houses, or shipping raw materials to factories.)

In our abstract setup, a network is a pair (N, A) where:

- N is a set of *nodes*, representing locations in a network;
- A is a set of *arcs*, ordered pairs (i, j) where $i, j \in N$, representing a one-way connection from i to j.

(Nodes and arcs are very similar to vertices and edges, and some people use the same terminology for both, but we'll stick to "nodes" and "arcs" to emphasize that arcs have a direction where edges didn't.)

For the first problem we'll consider, the max-flow problem, we have a bit more information on top of this. For every arc (i, j) there is a capacity c_{ij}: a nonnegative real number. There are two special nodes: a *source* s and a *sink* t. Our goal is to transport as much stuff from s to t. (Later, we will see how to generalize this model to other cases.)

We encode the problem with variables x_{ij} for every arc (i, j), representing the amount of *flow* from i to j: the amount of stuff traveling from i to j along arc (i, j). These flows have to satisfy the following conditions:

- Capacity constraints: $x_{ij} \le c_{ij}$ for all arcs $(i, j) \in A$.
- Nonnegativity constraints: $x_{ij} \ge 0$ for all $(i, j) \in A$.
- Flow conservation: for every node $k \in N$ (except for s and t), the total flow going in is equal to the total flow going out. That is,

$$\sum_{i:(i,k)\in A} x_{ik} = \sum_{j:(k,j)\in A} x_{kj}.$$

We do not require flow conservation at the source s or the sink t. Instead, the difference

$$\sum_{i:(i,t)\in A} x_{it} - \sum_{j:(t,j)\in A} x_{tj}$$

[1] This document comes from the Math 482 course webpage: https://faculty.math.illinois.edu/~mlavrov/courses/482-fall-2019.html

represents the net incoming flow at the sink t. This is called the *value* of the flow $\mathbf{x}$. We would like to maximize the value: to transport as much stuff to t as possible.

Similarly, at the source s, we can take the difference (but switching the two sums):

$$\sum_{j:(s,j)\in A} x_{sj} - \sum_{i:(i,s)\in A} x_{is}$$

and this represents the net outgoing flow at the source s. Because flow is conserved at every vertex other than s and t, the net outgoing flow at s should be equal to the net incoming flow at t, so it's another way to express the value of $\mathbf{x}$.

There's a couple of ways we can simplify the setup for the problem, should we care to:

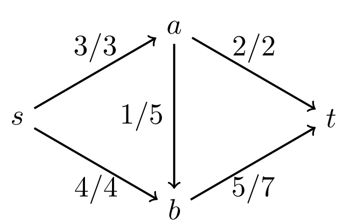
- We can assume that A consists of all the possible arcs, and just set $c_{ij} = 0$ whenever we don't actually want arc (i, j) to do anything. We probably don't want to bother when drawing a network, since it makes the drawing more complicated, but we can do it to make setting up the linear program simpler.
- It never makes sense to have positive flow $x_{is} > 0$ on an arc going into s, or positive flow $x_{tj} > 0$ on an arc leaving t. So we can assume that all of these arcs don't exist, or have capacity 0. This simplifies our expression for the value of $\mathbf{x}$: it can be written as

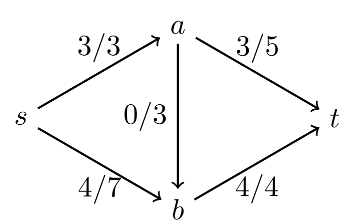
$$\sum_{j:(s,j)\in A} x_{sj} \qquad \text{or} \qquad \sum_{i:(i,t)\in A} x_{it}.$$

2 Upper bounds on flows

We move on to the following question: how can we tell if a feasible flow (that is, an assignment to the variables $x_{ij} : (i, j) \in A$ that satisfies all constraints) is optimal?

Consider the first diagram below. Here, a label of x/y on an arc means that the capacity of that arc is y, and x of it is used. (That is, an arc (i, j) is labeled by x_{ij}/c_{ij}.) Here, the value of the flow 7; we can guarantee that this is best possible, because the total capacity of edges leaving s is 7, so at most 7 flow can leave s.





Now consider the second diagram. Here, things are a bit more complicated. However, we can still see a "bottleneck" in the flow if we think of splitting up the nodes into $\{s, b\}$ and $\{a, t\}$. The total flow going from $\{s, b\}$ to $\{a, t\}$ is 7 (which is still the value of the flow). This cannot be increased, because all edges from s or b to a or t (namely, (s, a) and (b, t)) have the maximum flow possible, and all edges going the other way (namely, (a, b)) have zero flow.

The generalization of this notion is a *cut*. A cut in a network is a partition of the node set N into two sets, S and T, such that $s \in S$ and $t \in T$. (Being a partition requires that $S \cap T = \varnothing$ and $S \cup T = N$: each node is exactly one of the two sets S, T.)

The *capacity* of a cut (S, T) is, informally, the maximum amount of flow that can move from S to T. Formally, it is the sum

$$\sum_{i \in S} \sum_{j \in T} c_{ij}$$

(taking $c_{ij} = 0$ if $(i, j) \notin A$). For example, in the second diagram above, the cut $(\{s, b\}, \{a, t\})$ has capacity $c_{sa} + c_{bt} = 7$, because (s, a) and (b, t) are the only two arcs going from $\{s, b\}$ to $\{a, t\}$.

Low-capacity cuts are bottlenecks in the network: if we have a cut (S, T) with capacity c, then no more than c flow can be sent from s to t. This might make intuitive sense, but just in case, let's prove why this happens.

Theorem 2.1. *If a cut (S, T) has capacity $c(S, T)$, then no more than $c(S, T)$ flow can be sent from s to t in the network.*

Proof. Let $\mathbf{x}$ be a feasible flow in the network, and consider the sum

$$v(\mathbf{x}) := \sum_{k \in S} \left(\sum_{j:(k,j) \in A} x_{kj} - \sum_{i:(i,k) \in A} x_{ik} \right).$$

On one hand, flow conservation tells us that only the $k = s$ term of this sum is allowed to be nonzero. In fact, the $k = s$ term of this sum is equal to the value of $\mathbf{x}$, and therefore the whole sum $v(\mathbf{x})$ simplifies to the value of $\mathbf{x}$.

Now rearrange this sum slightly differently. First, split it up:

$$v(\mathbf{x}) = \sum_{k \in S} \sum_{j:(k,j) \in A} x_{kj} - \sum_{k \in S} \sum_{i:(i,k) \in A} x_{ik}.$$

Now split up each of these sums further: in each sum ranging over all i or all j, consider $i, j \in S$ separately from $i, j \in T$. (To simplify notation, we'll drop the requirement that $(k, j) \in A$ or $(i, k) \in A$: with the convention that the capacity of arcs not in A is 0, this doesn't make a difference.) We get:

$$v(\mathbf{x}) = \left(\sum_{k \in S} \sum_{j \in S} x_{kj} + \sum_{k \in S} \sum_{j \in T} x_{kj} \right) - \left(\sum_{k \in S} \sum_{i \in S} x_{ik} + \sum_{k \in S} \sum_{i \in T} x_{ik} \right).$$

The double sum over $k \in S, j \in S$ cancels with the double sum over $k \in S, i \in S$, because those two sums include the exact same terms, so we have

$$v(\mathbf{x}) = \sum_{k \in S} \sum_{j \in T} x_{kj} - \sum_{k \in S} \sum_{i \in T} x_{ik}.$$

Now we're going to put an upper bound on $v(\mathbf{x})$. For the first sum, we have $x_{kj} \leq c_{kj}$ in each term, and so replacing x_{kj} by c_{kj} can only increase the result. For the second sum, we have $x_{ik} \geq 0$ in each term, and we're subtracting all of these terms, so replacing x_{ik} by 0 can also only increase the result. Therefore

$$v(\mathbf{x}) \leq \sum_{k \in S} \sum_{j \in T} c_{kj} - \sum_{k \in S} \sum_{i \in T} 0.$$

But this is precisely the definition (with slightly different summation variables) of the capacity $c(S, T)$. Therefore $v(\mathbf{x}) \leq c(S, T)$, which is the inequality we wanted. $\square$

We can also see from this proof when $v(\mathbf{x})$ will equal $c(S, T)$ exactly. This happens when $x_{ij} = c_{ij}$ for every arc (i, j) with $i \in S, j \in T$, and $x_{ij} = 0$ for every arc (i, j) with $i \in T, j \in S$.

In both of the examples earlier, we were able to find a cut whose capacity is equal to the value of the flow. We can ask: does this always happen? Will every maximum flow have a cut to match it?