Tuesday, November 1

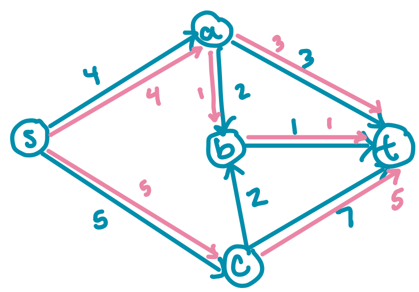
1. Welcome!
2. Homework 6 due Thursday
3. Thank you for small work
   - ↳ if not done: send by end of day
4. Progress Report due in 2 weeks
5. Questions?
6. Flows day 1
7. Outro
   - ↳ small work: none, get hw in.

Math 494: Discrete Optimization 11/1

Today we'll start our discussion of flows! The math behind network flows got its origins during the Cold War when tanks needed to be sent via rail around Russia. However, they still have applications in operations research, considering shipments, data, water, and transportation in general. Let's describe a network below:

Definition. *(network) A* network (G, c, s, t) *is a directed graph G and distinguished vertices s and t, often called the source and target. Every edge is assigned a capacity, $c : E \to R^+$.*

Example: Draw a network with five vertices. Be sure to give the edges capacity!



flow

Capacity

value = 9

Our goal will be to push as much "stuff", or flow, through the network as we can. There are two general rules: we cannot send more flow along an edge than it has capacity for, and for every vertex that's not s or t, we cannot generate or eliminate flow. Think, essentially, flow in equals flow out.

Example: Using the discussion above, complete the definition below

Definition. *(flow, value) A* flow $f : E \to R^+$ *is a function such that*

(1) $f(e) \leq c(e)$ (flow cannot exceed capacity)

(2) *For* $v \neq s, t$

δ^+: originate @ v

summing over u $\sum_{v \to u} f(vu) = \sum_{u \to v} f(uv) \to \sum_{e \in \delta^+(v)} f(e) - \sum_{e \in \delta^-(v)} f(e) = 0$

δ^-: terminate @ v.

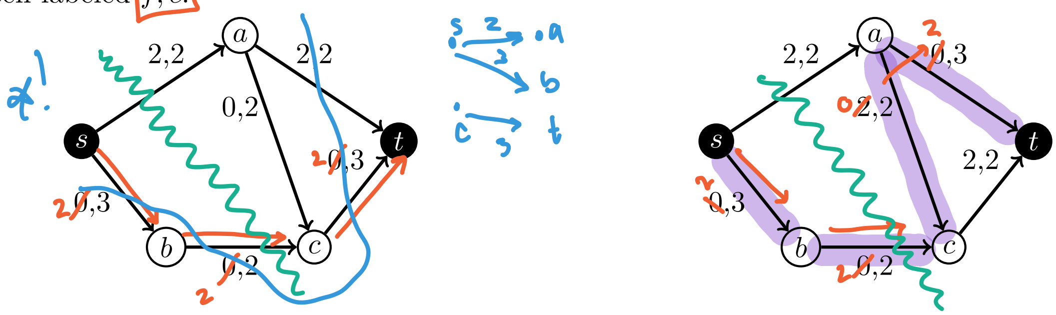
(3) *For s and t,*

$$\sum_{e \in \delta^+(s)} f(e) = \sum_{e \in \delta^-(t)} f(e)$$

The value *of a flow, denoted $|f|$, is* how much we send.

Example: Find the maximum flow on your example above. Is your solution unique? How do you know you're maxed out?

Example: How can we improve the flows on the networks shown below? Note, edges have been labeled $\boxed{f, c.}$



Follow up: Can you continue to improve these flows? How do you know you've maximized?

↳ nope!

all flow must cross these green lines, and the capacity only allows 4.

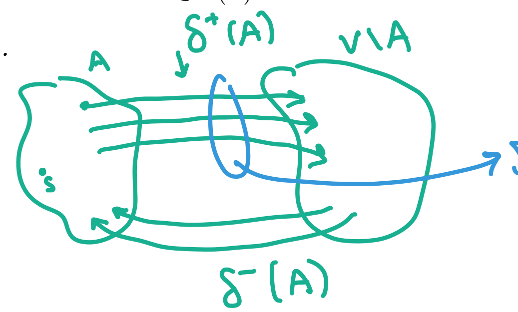
Let's formalize this certificate of maximality business by considering the value of a cut.

Definition. *(cut, capacity) For a network* (G, c, s, t)*,* $A \subset V$ is *a cut if* $s \in A$*,* $t \notin A$*. Let* $\delta^+(A)$ *denote edges going from* A *to* $V \setminus A$*, or equivalently,*

$$\delta^+(A) = \{uv \in E : u \in A, v \notin A\}.$$

The capacity of the cut is $\sum_{e \in \delta^+(A)} c(e)$*.*

In a picture...



sum these capacities for capacity of the cut! $c(A)$

Example: Consider the network on the left in the example above. Calculate the capacity for every possible cut.

$A: \{s, b\}$, $c(A) = 4$ $\quad$ $A: \{s\}$ $\quad c(A) = 5$ $\quad$ $A = \{s, b, c\}$ $\quad c(A) = 5$

$A = \{s, a\}$, $c(A) = 7$ $\quad$ $A = \{s, a, b\}$ $\quad c(A) = 6$ $\quad$ $A = \{s, a, b, c\}$ $\quad c(A) = 5$

$A = \{s, c\}$, $c(A) = 8$ $\quad$ $A = \{s, a, c\}$ $\quad c(A) = 8$

Follow up: for a network on n vertices, how many cuts exist?

2^{n-2}

Example: Prove the following lemma.

Lemma. *For any cut A and any flow f,*

a. $|f| = \sum_{e\in\delta^+(A)} f(e) - \sum_{e\in\delta^-(A)} f(e)$

b. $|f| \le \sum_{e\in\delta^+(A)} c(e)$

Pf: a) by conservation of flow

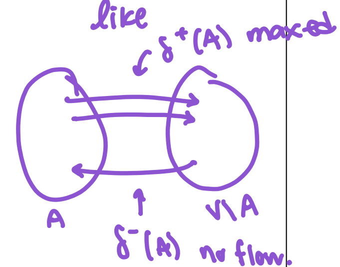
$$|f| = \sum_{e\in\delta^+(s)} f(e) = \sum_{v\in A}\Big(\sum_{e\in\delta^+(v)} f(e) - \sum_{e\in\delta^-(v)} f(e)\Big)$$

$$= \sum_{e\in\delta^+(A)} f(e) - \sum_{e\in\delta^-(A)} f(e)$$

b) note, since $\sum_{e\in\delta^-(A)} f(e) \ge 0$,

$$|f| \le \sum_{e\in\delta^+(A)} f(e) \le \sum_{e\in\delta^+(A)} c(e)$$

across min cut, max flow should look like



So we've got an idea about the maximization problem, an upper bounding structure, and maybe even the beginnings of some sort of incremental improvement idea that could lead to an algorithm. Let's take our tool kit to it: LP, dual, algorithm.

Example: To start, construct a linear program for the maximum flow problem. Consider what sorts of decisions you make when constructing flow and how conservation affects the constraints.

$$\begin{cases} \max \sum_{e\in\delta^+(s)} f(e) \quad \to |f|! \\ \text{st. } \sum_{e\in\delta^+(v)} f(e) - \sum_{e\in\delta^-(v)} f(e) = 0 \quad \forall v \ne s,t \\ f(e) \le c(e), \quad f(e)\ge 0 \end{cases}$$

↓ sketch

$$\begin{cases} \max \begin{array}{c} \text{edge} \\ \begin{bmatrix} \quad d \quad \end{bmatrix} \end{array} \begin{bmatrix} f \end{bmatrix}, \quad d = \begin{cases} 1 & s\to \\ 0 & \text{else} \end{cases} \\ \text{st } \begin{array}{cc} & \text{edges} \\ \text{vert} \ne s,t & \begin{bmatrix} \quad M \quad \end{bmatrix} \end{array} \begin{bmatrix} f \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \\ \begin{bmatrix} I \end{bmatrix}\begin{bmatrix} f \end{bmatrix} \le \begin{bmatrix} c \end{bmatrix} \end{cases} \quad \begin{bmatrix} M \\ I \end{bmatrix}\begin{bmatrix} f \end{bmatrix} \begin{matrix} = \\ \le \end{matrix} \begin{bmatrix} 0 \\ c \end{bmatrix}$$

$$M_{ve} = \begin{cases} 0 & v \notin e \\ 1 & v\to \\ -1 & \to v \end{cases} \qquad f(e)\ge 0$$

almost! → no var. for s,t

$$\begin{cases} \min \begin{bmatrix} \quad c \quad \end{bmatrix}\begin{bmatrix} \ell \end{bmatrix} \\ \text{st } \begin{bmatrix} M^T \mid I \end{bmatrix} \begin{bmatrix} y \\ \ell \end{bmatrix} \ge \begin{bmatrix} d \end{bmatrix} \\ \ell \ge 0, \ y \text{ free} \end{cases}$$

y: vertex indexed

ℓ: edge indexed

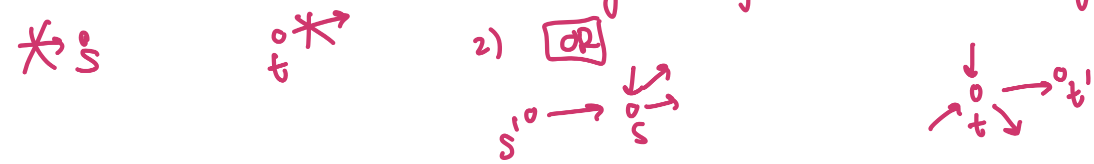
$$M^T y + \ell \ge d$$

Follow up: Dualize your program above.

Example: We're really hoping this dualization yields some sort of LP that corresponds to <u>Cuts!</u>. It will! We're just going to have to do some work to get it there.

First, assume s and t have no in and out edges, respectively. Why is this a reasonable assumption to make?

1) argue why flow on those edges $=0$



So we've sorta got two types of variable kicking around in our dual: one for every vertex and one for every edge. We're going to divide the edge variables into four different groups:

E_1 : $u \to v$: $u, v \notin \{s, t\}$ $\quad$ E_3 $u \to t$ $\quad u \neq s$

E_2 : $s \to u$ $\quad u \neq t$ $\quad$ E_4: $\{st\}$ or $\emptyset$

We can then rewrite the program as follows.

$$\begin{cases} \min\ c^T \ell \\ \text{st}\quad y_u - y_v + \ell_{uv} \geq 0 & E_1 \\ \quad -y_v + \ell_{sv} \geq 1 & E_2 \\ \quad y_u + \ell_{ut} \geq 0 & E_3 \\ \quad \ell_{st} \geq 1 & E_4 \\ \ell \geq 0,\ y \text{ free} \end{cases} \rightarrow \begin{cases} \min\ c^T \ell \\ \text{st}\quad y_s = -1,\ y_t = 0 \\ y_u - y_v + \ell_{uv} \geq 0 \text{ for all } uv \in E \\ y \text{ free},\ \ell \geq 0 \end{cases}$$

breaking matrix eq. apart.

we can introduce this!

We can now prove the following theorem by strong duality. Or, your book can (pg 179-181). We'll sketch the big ideas.

Theorem. *For a network $N = (G, c, s, t)$,*

$$\max\{|f| : f \text{ flow on } N\} = \min\{c(A) : A \text{ is an } s-t \text{ cut}\}$$

<u>Pf</u>: (highlights)

- both LP and dual are feasible: $f(uv) = 0$ for primal, $y_v = 0$ $(v \neq s)$ $\ell_{uv} = 1$ for dual.
- by duality, both have optimal solutions
- the signed incidence matrix M is totally unimodular, so dual has an optimal integral solution. Call it $\bar{\ell}, \bar{y}$
- Let $W = \{u \in V : \bar{y}_u \leq -1\}$. Note, $s \in W$, $t \notin W$, so W is a cut! Then the objective is

$$\sum c_e \bar{\ell}_e = \sum_{e \in \delta^+(A)} c_e \bar{\ell}_e + \underbrace{\sum_{e \notin \delta^+(A)} c_e \bar{\ell}_e}_{\geq 0} \geq c(\delta^+(W)) = c(W)$$

$$\bar{\ell}_{uv} \geq \underset{\geq 0}{\bar{y}_v} - \underset{\leq -1}{\bar{y}_u} \geq 1$$

- Use W to define y', ℓ': $y'_v = \begin{cases} -1 & v \in W \\ 0 & \text{else} \end{cases}$, $\ell'_e = \begin{cases} 1 & e \in \delta^+(W) \\ 0 & \text{else} \end{cases}$. We can show that this solution is optimal w/ objective val $c(W)$