Randomized algorithms:
First example: Karger's min-cut algorithm.

Input: undirected graph $G=(V,E)$, $n=|V|$, $m=|E|$.

Goal: find the min number of edges to remove so that the resulting graph is disconnected.

Formally:
for subset of vertices $S \subset V$,
let $\delta(S) = \delta_G(S)$ denote the set of edges crossing between S & $\bar{S} = V \setminus S$.
Removing $\delta(S)$ separates G into S & $\bar{S}$ (so ≥ 2 components)
Hence, $\delta(S) = \{(v,w) \in E : v \in S, w \notin S\}$
$\delta(S)$ is a cut

Goal: find the cut $\delta(S)$ of minimum size $|\delta(S)|$.

Closely related is the min st-cut problem

Here the input is $G=(V,E)$ & two vertices are specified $s \in V$ & $t \in V$.

Goal: find cut $\delta(S)$ of min size with the restriction that $s \in S$, $t \notin S$.

If you saw Max-flow before, you'll recall that

size of min st-cut = Value of max st-flow

& max st-flow can be solved in $O(nm\log n)$ time.

Can use this to solve min cut — just fix $s \in V$ & vary $t \in V$ (try all $n-1$ choices), & take the smallest of the $n-1$ solutions.

(can actually do these $n-1$ problems in this same total time.)

We'll see max st-flow algorithms later.

Today: simple <u>randomized</u> algorithm to directly solve min cut.

(simpler & faster)

Randomness in the algorithm.

For every input, the algorithm is likely to succeed.

Take input, flip coins that determines algorithm's trajectory, and with probability close to 1 the algorithm finds the min cut.

Can make as close to 1 as desired, but <u>can't</u> check if it's correct or not.

Idea of Karger's algorithm:

<u>Meta vertex</u> is a vertex representing a set of vertices in the original graph

<u>Multigraph</u> is a graph with possibly multiple edges between pairs of vertices.

<u>Simple graph</u> has at most 1 edge between a pair.

Start with a simple graph as input.

End with a multigraph with 2 metavertices representing set S & $\overline{S}$ and the remaining edges are $\delta(S)$.

Goal: end with metavertices for the min cut $(S, \overline{S})$ then output the remaining edges.

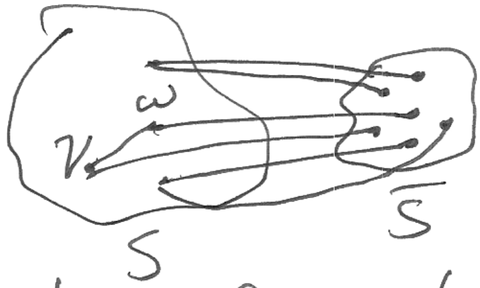
Basic operation: Merge 2 vertices.
We'll always do this along an edge.
Formally: contract an edge $e=(v,w)$, denoted G/e.
($G \setminus e$ means remove edge e)

For $G=(V,E)$, $e \in E$, define G/e as:

1) Replace vertices v & w by a new vertex z.
2) Replace every edge (v,y) by (z,y) & (w,y) by (z,y)
3) Remove self-loops to z, i.e., drop all edges of the form (z,z)
4) Resulting multigraph is G/e.

Key fact:

Look at a cut $(S, \overline{S})$



if we contract an edge $e = (v, w)$ where $v \in S$ & $w \in S$

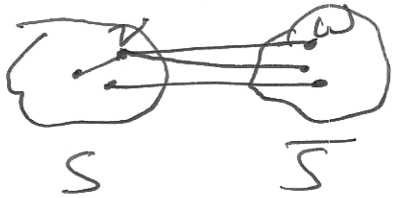
Then the cut $(S, \overline{S})$ is not changed.

In other words,

$$\delta_G(S) = \delta_{G/e}(S)$$

when $v, w \in S$.

But if $v \in S$ & $w \notin S$ then



the cut $(S, \overline{S})$ is destroyed.

How does one even define S in G/e?

include $z = v \cup w$ in S or $\overline{S}$?

So we lose cuts that contain e
& preserve cuts that don't contain e.

$$\text{Cuts in } G/e \xrightarrow{\hspace{3em}} \text{Cuts in } G$$
$$\text{Cuts in } G/e \xleftarrow[\text{not necessarily.}]{\times} \text{Cuts in } G$$

$$\text{Cuts in } G/e \subset \text{Cuts in } G$$

Go from $\approx 2^{n-1}$ cuts $\rightsquigarrow$ 1 cut.

Start with n vertices.
When we contract an edge, # of vertices goes down by 1.
So $n-2$ contractions.
Want that all $n-2$ contractions don't ~~contain~~ use an edge in the min cut $(S, \bar{S})$.

Then we'll be left with this mincut $(S, \bar{S})$.

Let $e_1, e_2, \dots, e_{n-2}$ denote the contracted edges.

Let $(S^*, \overline{S^*})$ be a min cut.

(if more than one min cut, choose one arbitrarily)

How do we choose the first edge e_1?

Let $k = |\delta_G(S^*)|$ = size of the min cut.

Want that $e_1 \notin \delta(S^*)$.

$\delta(S^*)$ is small (since it's a min size cut)

So if we choose a random edge (from E)

then it's unlikely to be in $\delta(S^*)$

Choose e_1 at random.

Then choose e_2 at random, etc.

Here is the algorithm.

Karger's min-cut algorithm:

Input: $G=(V,E)$

Repeat until 2 vertices remain:

1. Choose an edge $e=(v,w)$ uniformly at random from E
2. Set $G = G/e$.

Output the remaining edges between the 2 final metavertices.

Lemma:
Let $(S^*, \overline{S^*})$ be a cut of minimum size of the original graph G. Then,

$$\Pr\left(\begin{array}{c}\text{Karger's algorithm}\\ \text{outputs } \delta_G(S^*)\end{array}\right) \geq \frac{1}{\binom{n}{2}}$$

How do we boost the success probability?

Run the algorithm l times, & take the best of these l ~~set~~ outputs.

What l?

Simplified problem:
biased coin that's heads with probability p & tails with probability $1-p$.

How many coin flips till we see our 1st heads?

In expectation, $1/p$ flips needed.

We have $p \geq 1/n^2$

Heads means we found a min cut.

So want to choose l so that have ≥ 1 heads.

In expectation, choose $l = O(n^2)$.

To boost further, choose $l = O(n^2 \log n)$.

For integer $C > 0$, let $l = Cn^2 \ln n$.

Run Karger's algorithm l times, and take the smallest of the l cuts found.

$$\Pr\left(\text{new algorithm found } \delta_G(S^*)\right) \geq 1 - n^{-C}$$

So setting $C = 10$, get error prob. $\leq n^{-10}$.

Proof of):

Say $p = 1/n^2$. Do $l = Cn^2 \ln n$ flips.

Want at least 1 heads.

$$\Pr\left(\begin{matrix}\text{all } l \text{ flips}\\ \text{are fails}\end{matrix}\right) = (1-p)^l$$

Simplify:

recall, $e^{-x} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} \pm \ldots$

for $0 < x < 1$, $x > x^2 > x^3 > \ldots$

So, $x > \frac{x^2}{2!} > \frac{x^3}{3!} > \ldots$ terms getting smaller.

$$\frac{x^2}{2!} - \frac{x^3}{3!} > 0, \quad \frac{x^4}{4!} - \frac{x^5}{5!} > 0, \ldots$$

Therefore,

$$e^{-x} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} \pm \cdots > 1 - x$$

$$e^{-x} > 1 - x$$

Thus,

$$\Pr\left(\begin{array}{c}\text{all } \ell \text{ flips} \\ \text{are tails}\end{array}\right) = (1-p)^{\ell} < e^{-p\ell}$$

for $p = \frac{1}{n^2}$ & $\ell = C n^2 \ln n$

$$= \frac{1}{n^C}$$

$$\Pr\left(\begin{array}{c}\text{algorithm finds} \\ \delta(S^*)\end{array}\right) = \Pr(\text{at least } 1 \text{ heads})$$

$$= 1 - \Pr(\text{all tails})$$

$$\geq 1 - n^{-C}.$$

$\boxtimes$

Next class: Proof of the lemma that

$$\Pr\left(\begin{array}{c}\text{1 run of Karger's alg.}\\ \text{finds } \delta(S^*)\end{array}\right) \geq \frac{1}{\binom{n}{2}}.$$