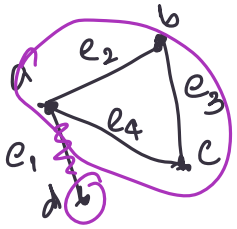
Previous lecture: Intro to Randomized algorithms, RandQS.

1.2 Global Min-Cut algorithm



Cut: e_1, e_4
e_2, e_4
e_3, e_4
e_1

Given a graph $G = (V, E)$.

Cut: A set of edges whose removal results in G being broken into two or more components

min-cut: A cut of minimum cardinality.

Question: Given a graph $G = (V, E)$, find a min-cut.

Recall that s-t min-cut ≡ s-t max-flow. [Max-flow min cut theorem]

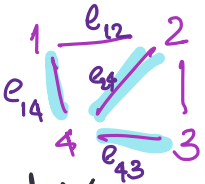
And s-t max-flow can be obtained by a "modified" Ford-Fulkerson algorithm or Edmond-Karp algo which runs in $O(VE^2)$ time. [Thm 26.8 of CLRS]

By running over all possible s,t in V, we get a $O(V^3E^2)$ time algo for mincut.

Question: Can randomization help?

Let us look at Karger's algorithm for min-cut. [Karger 1993]

Some definitions before we proceed:



{2, 3, 4}
e_{12} ( ) e_{14}
{1 3

- "supernode": A set of nodes/vertices.
- "superedge" between two supernodes X and Y:

$\{e \mid e = (u, v)$ such that $u \in X$ and $v \in Y\}$.

{1, 2, 4} {1, 2, 3}
( ) ( )
{3} {4}

Roughly, Karger's algo does the following:

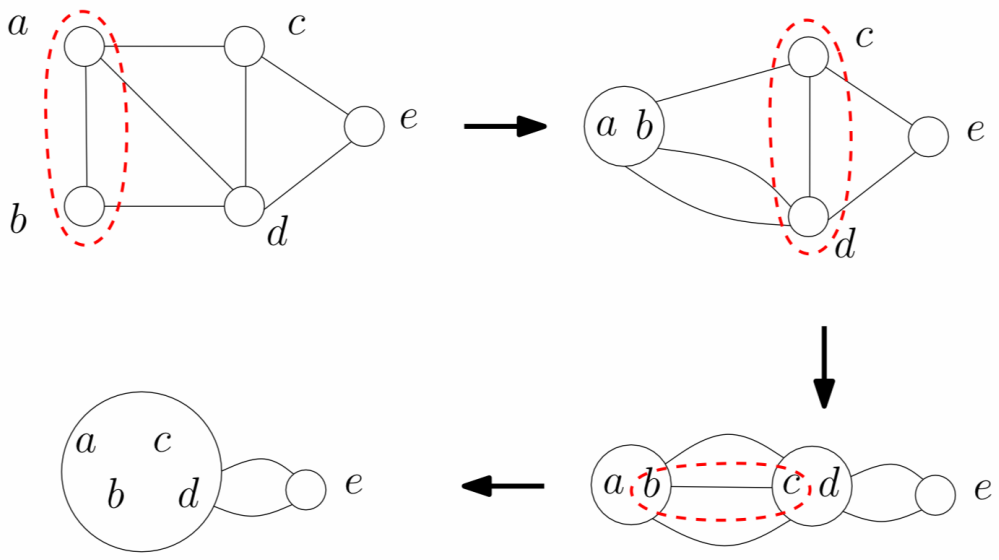
1. While # no. of supernodes > 2, pick an edge from E and merge its endpoints
2. Output the edges between the remaining two supernodes.

Karger shows that the above procedure outputs a min-cut with a non-trivial probability.

Notation:
Γ : set of supernodes
F : set of superedges
$V(u)$: nodes in the supernode u
E_e : edges in the superedge e.

Algorithm(Initialize(G)):
Input: Graph $G=(V,E)$
Output: Γ, F
1. $\Gamma \leftarrow V$
2. $F \leftarrow E$

Ex:



Algorithm (Merge (a, b, Γ)):
Input: Γ is the set of supernodes, and $a, b \in \Gamma$.
Output: Updated Γ with supernodes a, b merged into a new supernode.
1. $x \leftarrow$ new empty supernode.
2. $V(x) \leftarrow V(a) \cup V(b)$ // Merges the sets of nodes
3. For each $d \in \Gamma \setminus \{a, b\}$:
   $E_{dx} \leftarrow E_{da} \cup E_{db}$ // Merges the set of edges.
4. $\Gamma \leftarrow (\Gamma \setminus \{a, b\}) \cup \{x\}$.

choices
Obtains corr answ.

Algorithm (Karger(G)):
Input: Graph $G=(V,E)$
Output: A min-cut in G

1. $\Gamma, F \leftarrow$ Initialize(G)
2. While $|\Gamma|>2$ do:
   - Pick $e=(u,v)$ "uniformly at random" from F
   - $\Gamma \leftarrow$ Merge$(\bar{u},\bar{v},\Gamma)$ // $\bar{u}$ and $\bar{v}$ are supernodes containing u and v resp.
   - $F \leftarrow F \setminus E_{\bar{u}\bar{v}}$.
3. Return E_{xy} where $\Gamma=\{x,y\}$.

$O(n)+O(m)+O(n\cdot|\text{union operations}|)$

Analysis:

Let k be the size of the min cut C of G and $|V|=n$.

Claim 1: G has at least $\frac{kn}{2}$ edges. (min deg $\geq k$)

Proof: The min degree of G is at least k. Otherwise removing a vertex of degree $<k$ gives us a cut that is smaller than k. Further #Edges = (total degree)/2 $\Rightarrow |E| \geq \frac{kn}{2}$.

We shall now show that with a good probability any edge in the cut C was never "contracted".

E_i := Event that an edge from C was not picked at i^{th} step

$1 \leq i \leq n-2$

We want $\Pr\left[\bigcap_{i=1}^{n-2} E_i\right]$ to be non-trivial.

From a generalization of conditional probability we get

$$\Pr\left[\bigcap_{i=1}^{n-2} E_i\right] = \Pr[E_1] \times \Pr[E_2 \mid E_1] \times \Pr[E_3 \mid E_1 \cap E_2] \times \cdots \times \Pr\left[E_{n-2} \mid \bigcap_{j=1}^{n-3} E_j\right].$$

$\Pr[E_1]$ = Probability that edges from C are not chosen.

$$= 1 - \frac{k}{|E|}$$

$$\geq 1 - \frac{k}{\frac{kn}{2}} \quad \text{// Since } |E| \geq \frac{kn}{2}.$$

$$= 1 - \frac{2}{n}$$

Remark: After E_1 happens, we can invoke Claim 1 on the remaining $n-1$ supernodes and get that $\frac{k(n-1)}{2}$ edges remain.

After the first step assuming E_1 happens,

$\Pr[E_2 \mid E_1]$ = Probability of picking an edge outside C over remaining edges.

$$\geq 1 - \frac{k}{\frac{k(n-1)}{2}} = 1 - \frac{2}{(n-1)}.$$

Similarly at i^{th} step assuming $\bigcap_{j=1}^{i-1} E_j$, we get that

there are $(n-i+1)$ supernodes and thus,

$$\Pr\left[E_i \mid \bigcap_{j=1}^{i-1} E_j\right] \geq 1 - \frac{k}{\frac{k(n-i+1)}{2}} = 1 - \frac{2}{n-i+1}.$$

Putting it all together, we get that

$$\Pr\left[\bigcap_{i=1}^{n-2} E_i\right] \geq \prod_{i=1}^{n-2} \left(1 - \frac{2}{n-i+1}\right) = \prod_{i=1}^{n-2} \left(\frac{n-i-1}{n-i+1}\right) = \frac{2}{n(n-1)}.$$

That is, with a probability of at least $\Omega\left(\frac{1}{n^2}\right)$ we get a min-cut.

Claim 2: Runtime of the algorithm is at most $O(n^2)$.
$O(n \cdot$ (complexity of union)$)$.

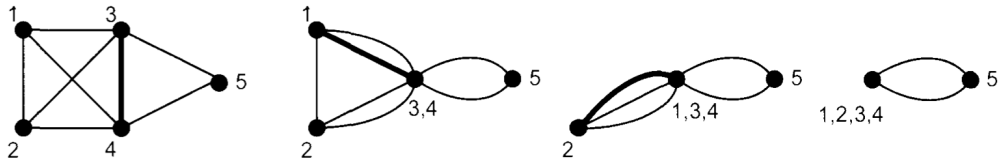
Question: What happens w/ probability of at most $1 - \frac{2}{n(n-1)}$?

$\Pr[\text{Success}] \leftarrow \Pr[\cup \text{ cuts success}]$

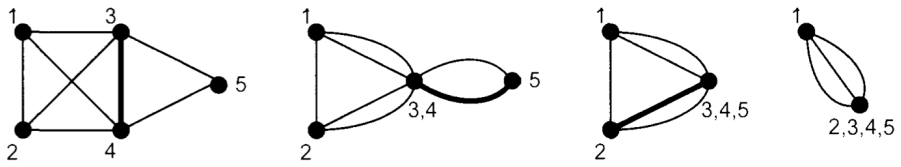
$\geq \Pr[\text{Success wrt a single min cut}]$

$\geq \frac{2}{n(n-1)}$

$\Pr[\cup E_i] \geq \Pr[E_i]$



(a) A successful run of min-cut.



(b) An unsuccessful run of min-cut.

Observation: Karger's algorithm does not always produce a correct min-cut. However, we bound the error prob to at most $1 - \frac{2}{n(n-1)}$.

1.3 Las Vegas and Monte Carlo algorithms

Algorithms where you can bound the error probability are called "Monte Carlo algorithms."

In contrast, RandQS always produced a correct answer but its running time was bound in expectation.

Algorithms that always produce the correct answer and its running time is bound in expectation are called Las Vegas algorithms.

Remark: Las Vegas algorithms are also Monte Carlo algorithms with error prob. 0.

Question: Can we improve the bound on error probability of Karger's algorithm?

1.4 Boosting the success probability

From the above discussion, a run of Karger's algo takes $t = O(n^2)$ steps and outputs a "mincut" w/p of at least $\frac{2}{n(n-1)}$.

- Run s times
- Mincut = min { results of s many runs }

Let us repeat Karger's algorithm s times independently.

Probability that a mincut is not found in any of the s repetitions is at most $\left(1 - \frac{2}{n(n-1)}\right)^s$.

$$\lim_{n\to\infty}\left(1-\frac{1}{n}\right)^n \to e^{-1}$$

$$\left(1 - \frac{2}{n(n-1)}\right)^s = \left(1 - \frac{2}{n(n-1)}\right)^{\frac{n(n-1)}{2} \cdot \frac{2s}{n(n-1)}}$$

$s \sim n^2 \ln n$

$1 - \frac{1}{n^2}$

$$\sim e^{-\frac{2s}{n(n-1)}}.$$

// Choose your s to obtain the bound of your choice.

By setting s to $\frac{n(n-1)}{2}$, we bound the error to atmost $\frac{1}{e}$.

Total running time $= O(s \cdot n^2) = O(n^4)$.

$(1-\frac{1}{e})$ ✓

References:

1. Lecture 16 notes of CS 161 @ Stanford (2016) taught by Moses Charikar.
2. Sections 1.1 and 1.2 of Motwani and Raghavan.
3. Section 1.4 of Mitzenmacher and Upfal.

Claim: Suppose the $|\text{mincut}| = k$ then mindegree of $G \geq k$.