CSE 6512 Lecture 12 Notes

Taken by Levon Nazaryan
October 06, 2011

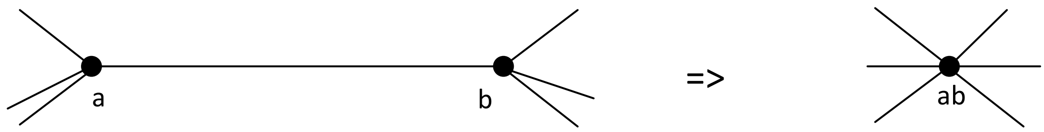
1 FINDING A MIN-CUT

Input: $G(V,E)$, AN UNDIRECTED MULTIGRAPH.
Output: A MIN-CUT

Definition: A cut is a set of edges whose removal results in 2 or more components. A MIN-CUT is a cut of minimum size.

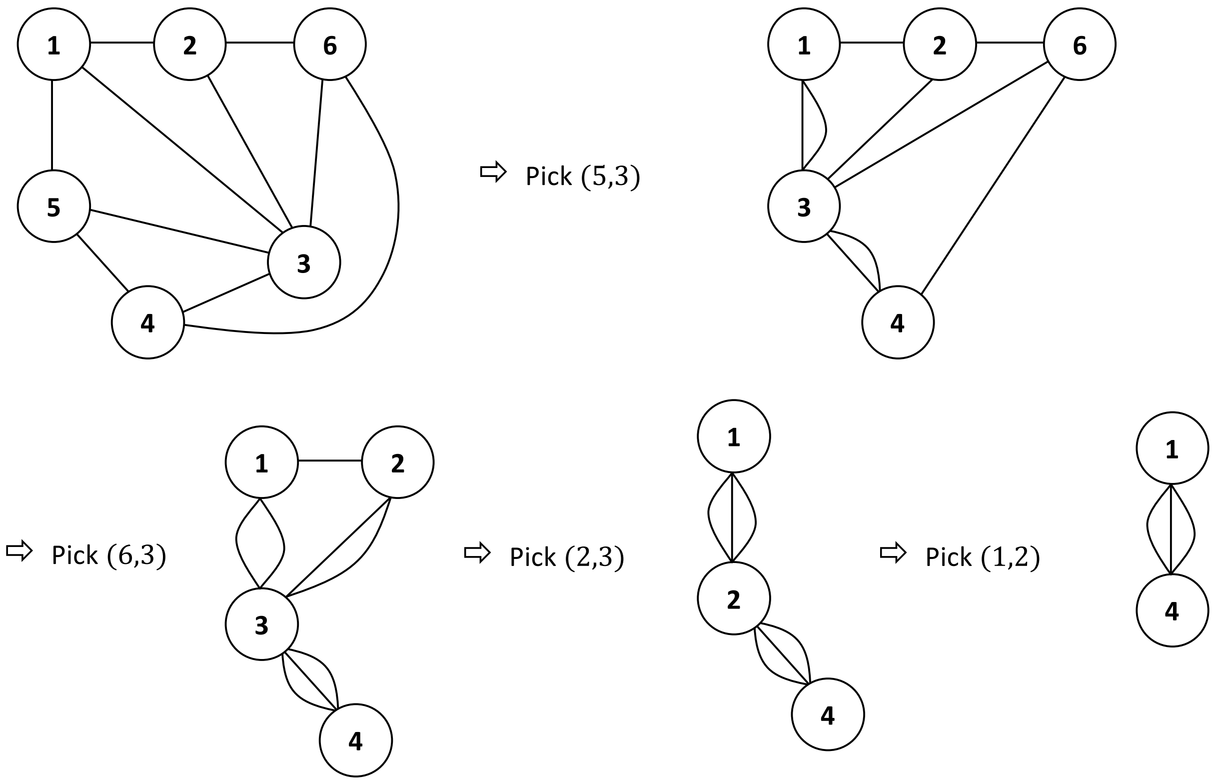
A RANDOMIZED ALGORITHM.

CONTRACTION: Pick a random edge (a,b) and merge a and b. All the edges incident on a and b will be preserved.



Each contraction results in one less node. Also the size of the min-cut does not change with contractions. Do $(n-2)$ such contractions, until only 2 nodes remain. Output the edges between them.

Example:



Analysis: Let k be the size of the min-cut, and let $C = \{e_1, e_2, \cdots, e_k\}$ be a min-cut. We'll calculate the probability that the cut output by the above randomized algorithm is C. Note that the number of edges in G is $\geq \frac{kn}{2}$. Otherwise, it will mean that the degree of at least one node is less than k and if we remove the edges incident on this node we'll get at least two components.

Let E_i be the event that the edge picked in contraction i **is not from** C.

What is $Prob[\bigcap_{i=1}^{n-2} E_i]$?

$Prob[\bar{E}_1] \leq \frac{k}{\frac{nk}{2}} = \frac{2}{n}$. This implies that $Prob[E_1] \geq 1 - \frac{2}{n}$.

After the **first contraction**, the number of nodes is $(n-1)$.

Number of edges in the reduced graph is $\geq \frac{k(n-1)}{2}$.

$\Rightarrow Prob(\overline{E_2}/E_1) \leq \frac{2}{n-1}$. As a result, $Prob(E_2/E_1) \geq 1 - \frac{2}{n-1}$

$\Rightarrow Prob\left[E_i / \bigcap_{j=1}^{i-1} E_j\right] \geq 1 - \frac{2}{n-i+1}$.

Fact: $Prob(E_1 \cap E_2) = Prob(E_1) \cdot Prob(E_2/E_1)$.

$$Prob\left[\bigcap_{i=1}^{n-2} E_i\right] \geq \left(1 - \frac{2}{n}\right)\left(1 - \frac{2}{n-1}\right) \cdots \left(1 - \frac{2}{3}\right)$$

RHS $= \frac{(n-2)}{n} \frac{(n-3)}{(n-1)} \frac{(n-4)}{(n-2)} \frac{(n-5)}{(n-3)} \cdots \frac{1}{3} = \frac{2}{n(n-1)} \geq \frac{2}{n^2}$

Algorithm:

for $i := 1$ to q do
    Repeat the contraction process $(n-2)$ times to find a cut.
    Keep the min seen so far.
Output the min-cut seen.

Probability of finding C in 1 iteration is $\geq \frac{2}{n^2}$

=> Probability of not finding C in one iteration is $\leq \left(1 - \frac{2}{n^2}\right)$

=> Probability of not seeing C in q iterations is $\leq \left(1 - \frac{2}{n^2}\right)^q$

We want this to be $\leq n^{-\alpha}$ => $\left(1 - \frac{2}{n^2}\right)^q = n^{-\alpha}$

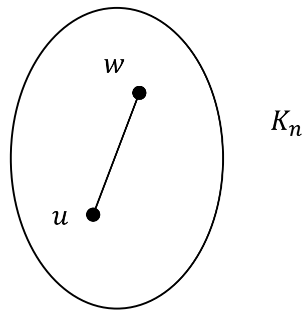
i.e., $e^{-q\frac{2}{n^2}} = n^{-\alpha}$ => $-q\frac{2}{n^2} = -\alpha \log n$ => $q = \frac{1}{2} n^2 \alpha \log n$ ■

2 RANDOM WALKS ON GRAPHS
Input: $G(V, E) \rightarrow$ undirected.
We start from a node u. Then go to a random neighbour of u. From there go to a random neighbour, and so on.

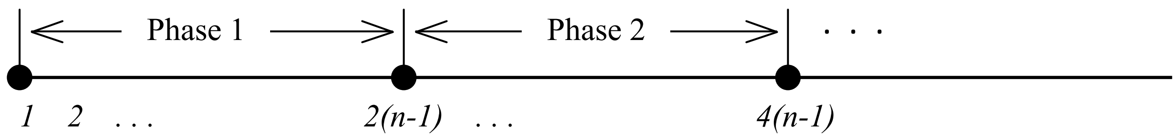
Questions:

1. How much time does it take before each node is visited at least once?
2. If we start from a node u how long will it take to visit another specific node w?

Example: Let K_n be a complete graph on n nodes.
If we start from u, the expected number of steps before visiting $w = (n-1)$.



How long does it take before each node is visited at least once?



TIME →

Let X_w be the time needed to visit w.
$E[X_w] = (n-1) = \mu$
$Prob[X_w \geq 2\mu] \leq \frac{1}{2}$ using Markov's inequality.

Probability of not visiting w in any one of the phases is $\leq \frac{1}{2}$.
=> $Prob[not\ visiting\ w\ in\ (\alpha+1)\log n\ phases] \leq \left(\frac{1}{2}\right)^{(\alpha+1)\log n} = n^{-(\alpha+1)}$

=> $Prob[\exists\ w\ that\ has\ not\ been\ visited\ in\ the\ FIRST\ \log n\ (\alpha+1)\ phases] \leq n^{-\alpha}$

=> The time needed to visit all the nodes is $\tilde{O}(n \log n)$.

Example: 2 SAT
Input: $F = C_1 \wedge C_2 \wedge \cdots \wedge C_m$
Let $S = \{s_1, s_2, \cdots, s_n\}$ be a specific satisfying assignment. Call these values as correct values.

A Randomized Algorithm works as follows:

0) Start from a random assignment.
1) Pick a random clause which is not satisfied.
2) Pick a literal in it randomly and change its value.
3) Repeat steps 1 and 2 until a satisfying assignment is found.

NUMBER OF CORRECT VALUES FOUND →

Note that this corresponds to a Random Walk in the above graph. The algorithm terminates when the node n is visited for the first time (or possibly before since there could be other satisfying assignments).

Fact: The expected RUN TIME of this algorithm is $O(n^2)$. This will be proven soon.