Randomness and Computation

or, "Randomized Algorithms"

Mary Cryan

School of Informatics
University of Edinburgh

Karger's Min-Cut Algorithm

We are given an undirected graph $G = (V, E)$ with $|V| = n$, and are interested in computing a "min cut"; that is, a partition of E into two non-empty sets S, $V \setminus S$, such that the following quantity is minimized*:*

$$\{e = (u, v) : u \in S, v \in V \setminus S\}$$

There are many deterministic algorithms which can solve this problem in polynomial-time. Karger's algorithm (faster) uses random sampling:

Repeatedly, choose an edge uniformly at random (from the not-yet contracted edges) and conjoin its endpoints.

When there are just two "vertices" left, return that cut.

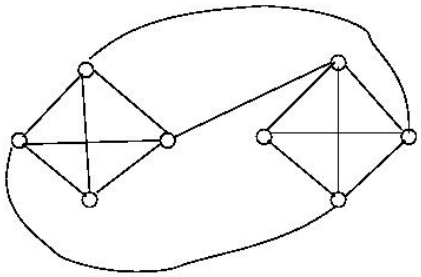
Exact deterministic algorithms for Min-Cut

Deterministic algorithms for this undirected problem already exist:

- "Net Flow": replace each edge with two opposing arcs. Select fixed source vertex s (no matter which). Find the min (s, t)-cut, for each $t \in V \setminus \{s\}$ via a fast Network Flow algorithm (fastest is $\Theta(mn)$, via a merge of Orlin's and King, Rao, Tarjan's algorithms).
  Do $(n-1)$ individual calls (one for each t) $\Rightarrow \Theta(mn^2)$ time in total, and return the minimum cut ever found.
- Best deterministic "no flows" algorithm is Stoer and Wagner's $\Theta(mn + n^2 \log(n))$ algorithm.

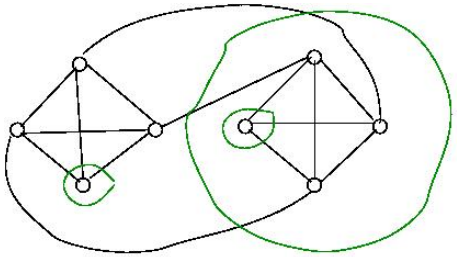
We will see that Karger's algorithm is not guaranteed to return the optimal Min Cut, but is much faster.

Karger's Min-Cut Algorithm - Example



Min degree of any vertex is 3. This means the min cut of the graph is no larger than 3.

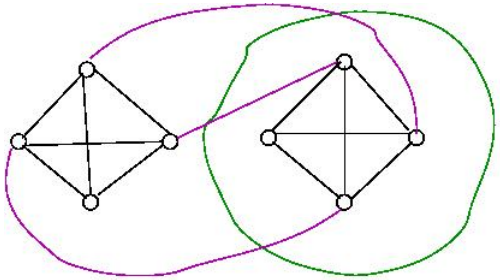
Karger's Min-Cut Algorithm - Example



The green lines are marking groups of vertices which form "minimum–sized" cuts in the graph. The analysis of Karger's algorithm involves showing there is good probability a randomly–chosen edge avoids the cut boundary.

Let's focus on the larger "S" as our target to achieve the min cut value of 3, for the analysis.

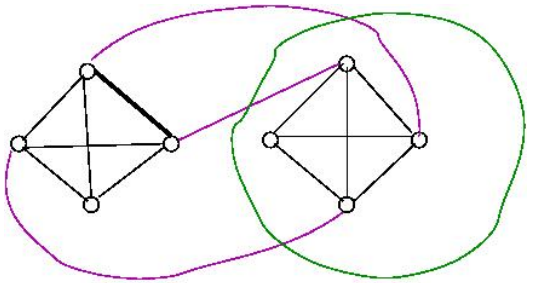
Karger's Min-Cut Algorithm - Example



Let's focus on the larger "S" as our target to achieve the min cut value of 3, for the analysis.

Our hope is that the random process will avoid the magenta edges with some decent probability.

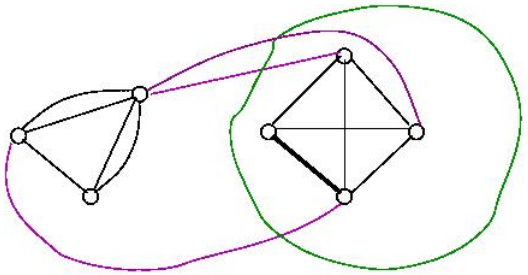
Karger's Min-Cut Algorithm - Example



Randomly choosing our first edge to contract ...

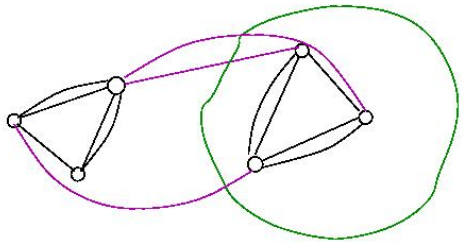
First choice of edge is uniformly at random from all 15 edges, as it happens we avoid the magenta ones (not surprising as there are only 3, probability of hitting one of them is only 1/5)

Karger's Min-Cut Algorithm - Example



Second choice of edge is uniformly at random from remaining 14 edges, again we avoid the magenta ones (not surprising as there are only 3, probability of hitting one of them is only 3/14)

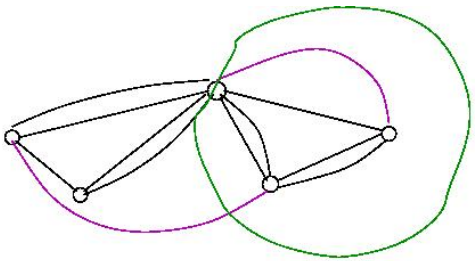
Karger's Min-Cut Algorithm - Example



Again we have lost another vertex (only 6 now) and another edge (only 13 now). Note that the probability we avoided magenta for two steps was (12/15)*(11/14) which is 22/35.

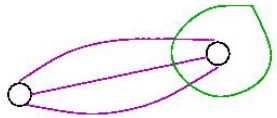
Notice that when we contract an edge, we are putting those vertices into the same "side" of our eventual cut. So we never want to contract a magenta edge. But our algorithm does not know which are the cut edges.

Karger's Min-Cut Algorithm - Example



Want to avoid ever contracting one of the cut edges (as has happened in this step) as we end up with a composite vertex which has hidden one of the cut edges (and apart from hiding this particular cut, we probably have reduced the chance of finding a good one).

Karger's Min-Cut Algorithm - Example



This is the "hoped–for" scenario for the final stage of our algorithm (just two vertices left). We have all the S vertices on one side, all of V\S on the other, and the number of edges between them is hence the (true) min–cut.

Karger's (randomised) Min-Cut Algorithm

Algorithm KARGERONETRIAL($G = (V, E)$)

1. $n' \leftarrow |V|$
2. **while** $(n' > 2)$ **do**
3. Draw $e = (u, v)$ uniformly at random from E
4. Let $\widehat{v}$ be the "vertex" containing v *(may just be $\{v\}$)*
5. Let $\widehat{u}$ be the "vertex" containing u *(may just be $\{u\}$)*
6. $E \leftarrow E \setminus \{(w, z) : w \in \widehat{u}, z \in \widehat{v}\}$ *(merge $\widehat{u}, \widehat{v}$)*
7. $n' \leftarrow n' - 1$
8. **return** $E, |E|$

In line 6. we will definitely delete (u, v), but might delete other parallel edges between the supervertices to be contracted.

Can be implemented in about $\Theta(m \log(m))$ time by using sampling without replacement (from E) and a fast Disjoint Sets Data Structure.

Karger's paper shows how to use a nice trick with connected component calculations to evaluate the result in $\Theta(m)$ time.

Karger's Min-Cut Algorithm - Analysis

Let k be the size of a min cut of G, let $S \subset V$ be a *specific* partition where C_S, the set of edges between S and $V \setminus S$, is of cardinality k.

We must have $deg(v) \geq k$ for every $v \in V$. *(WHY?)*

Let E_j be the event that the jth-random edge is not in C_S. *(this event is what we want)*

Calculating $\Pr[E_1]$, there are k "cut-edges" (from C_s), and at least $k \cdot n/2$ edges overall. Hence

$$\Pr[E_1] \geq 1 - \frac{2k}{kn} \geq 1 - \frac{2}{n}.$$

We next calculate $\Pr[E_2 \mid E_1]$, probability that the *2nd* edge avoids C_s, *conditional that the first edge was outside* C_s. ...

Karger's Min-Cut Algorithm - Analysis

$\Pr[E_2 \mid E_1]$:

- Still have all k C_S edges (since we assumed E_1).
- Graph now has $(n-1)$ "vertices", each must have degree $\geq k$ (discuss); hence the graph now has at least $k \cdot (n-1)/2$ edges overall.

Hence

$$\Pr[E_2 \mid E_1] \;\geq\; 1 - \frac{2k}{(n-1)k} \;=\; 1 - \frac{2}{n-1}.$$

Next we will compute, for any initial sequence of j edge-choices satisfying $\cap_{i=1}^{j} E_i$, a lower bound on $\Pr[E_{j+1} \mid E_1 \cap \ldots \cap E_j]$ ...

Karger's Min-Cut Algorithm - Analysis

For any $j = 1, \ldots, n-3$, we analyse the *conditional* probability $\Pr[E_{j+1} \mid E_1 \cap \ldots \cap E_j]$:

- All k C_S edges still remain (since we assume $E_1 \cap \ldots \cap E_j$).
- We have removed at least j edges before selecting the $j+1$-th (not exactly j, as we might have contracted a "parallel edge" earlier on, which has the effect of removing more than one edge from the graph).
- We have absorbed *exactly* j vertices so far, graph now has $(n-j)$ "vertices", and each must have degree $\geq k$ *(discuss)*; hence the graph now has at least $k \cdot (n-j)/2$ edges overall.

Hence

$$\Pr[E_{j+1} \mid E_1 \cap \ldots \cap E_j] \;\geq\; 1 - \frac{2k}{(n-j)k} \;=\; 1 - \frac{2}{n-j}.$$

Karger's Min-Cut Algorithm - Analysis

We hope that our contraction of random edges will lead us to a scenario where we are left with two "vertices" without contracting any of the C_S edges (min-cut) on the way.

If we achieve this, then one "vertex" will contain all of S, the other "vertex" all of $V \setminus S$, and the parallel edges between both are exactly the edges in the min-cut C_S.

The probability we get to this nice scenario is the probability that E_1 holds, *and* (conditioned on that) that E_2 also holds, *and* (conditioned on $E_1 \cap E_2$) ...) that E_3 also holds, and ... Formally,

$$\begin{aligned}
\Pr[\cap_{j=1}^{n-2} E_j] &= \Pr[E_1] \cdot \Pr[E_2 \mid E_1] \cdot \ldots \cdot \Pr[E_{n-2} \mid \cap_{i=1}^{n-3} E_i] \\
&= \prod_{j=1}^{n-2} \Pr[E_j \mid \cap_{i=1}^{j-1} E_i] \\
&= \prod_{j=1}^{n-2} \left(1 - \frac{2}{n-(j-1)}\right) \;=\; \prod_{j=3}^{n} \left(1 - \frac{2}{j}\right)
\end{aligned}$$

Karger's Min-Cut Algorithm - Analysis

Expanding $\prod_{j=3}^{n}\left(1-\frac{2}{j}\right)$, we have

$$\begin{aligned}&\prod_{j=3}^{n}\frac{j-2}{j}\\ =\ &\left(\frac{1}{3}\right)\left(\frac{2}{4}\right)\left(\frac{3}{5}\right)\left(\frac{4}{6}\right)\cdots\left(\frac{n-4}{n-2}\right)\left(\frac{n-3}{n-1}\right)\left(\frac{n-2}{n}\right)\\ =\ &\frac{2}{n(n-1)}\end{aligned}$$

So the probability that a single "run" of KARGERONETRIAL generates a cut which is Minimal for the original graph is *at least* $\frac{2}{n(n-1)}$.

Could be more in practice. *(WHY?)*

Karger's Min-Cut Algorithm - Repeated iterations

We can improve our result by running KARGERONETRIAL many times, and returning the minimum of all the different cuts.

If we do k trials, the probability that *none* is a min cut is *at most*

$$\left(1-\frac{2}{n(n-1)}\right)^{k}.$$

We can relate this to e using $(1+\frac{1}{n})^{n}<e<(1+\frac{1}{n})^{n+1}$.

$$\Rightarrow\ \left(1-\frac{2}{n(n-1)}\right)^{\frac{n(n-1)}{2}}<e^{-1},$$

and taking $k=\frac{n(n+1)}{2}\cdot\ln(n)$, we get

$$\left(1-\frac{2}{n(n-1)}\right)^{k}=\left(\left(1-\frac{2}{n(n-1)}\right)^{\frac{n(n+1)}{2}}\right)^{\ln(n)}<(e^{-1})^{\ln(n)}=\frac{1}{n}.$$

Wrapping up

- Probability tools used in our analysis were simple: we have used Lemma 1.4 iteratively:

$$\begin{aligned}\Pr[\cap_{j=1}^{n-2}E_j] &= \Pr[\cap_{j=2}^{n-2}E_j\mid E_1]\cdot\Pr[E_1]\\ &= \Pr[\cap_{j=3}^{n-2}E_j\mid E_1\cap E_2]\cdot\Pr[E_1\cap E_2\mid E_1]\cdot\Pr[E_1]\\ &= \ldots\end{aligned}$$

  (also used simple inequalities relating $(1+\frac{1}{n})$ and e)
- No approximation guarantee - analysis does not address likely quality of C_s when it fails to be optimum.

Reading

Original source of the algorithm in today's lecture is "Global Min-cuts in RNC and Other Ramifications of a Simple Mincut Algorithm", by David R Karger, SODA 1993.

Our next topic will be the "Coupon Collector" problem.

- Some of you may have seen the "Coupon Collector" problem in lower level classes.
- We will re-visit it, but as well as deriving the expected value, we will also bound the variance (2nd moment), and look at the implications of that.
- You might want to read sections 2.3, 2.4 and 3.3 of [MI] in advance (if your probability is rusty, also read 2.1, 2.2, 3.1 and 3.2)