Minimum Cut in a Graph

497 - Randomized Algorithms

Sariel Har-Peled

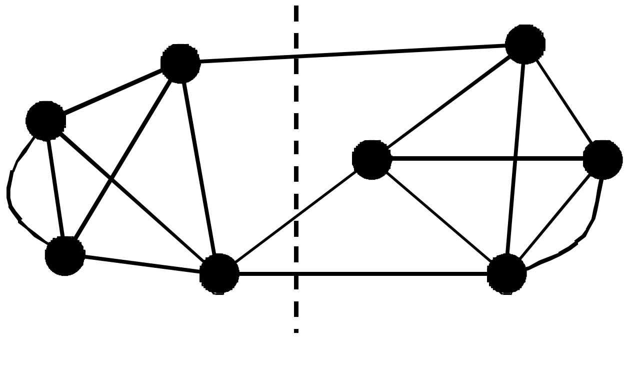
September 3, 2002

Paul Erdős (1913-1996) - a famous mathematician, believed that God has a book, called "The Book" in which god maintains the perfect and most elegant mathematical proofs. Every once in a while, god let mortals peek in the book and "steal" a beautiful proofs. Erdős also said that you need not believe in God, but, as a mathematician, you should believe in The Book.

See: "Proofs from THE BOOK", Aigner and Ziegler, 2001.

1 Min Cut

Compute the cut with minimum number of edges in the graph. Namely, find $S \subseteq V$ such that $(S \times (V - S)) \cap E$ is as small as possible, and S is neither empty nor all the vertices of the graph $G = (V, E)$.



1.1 Some Definitions

Definition 1.1 The conditional probability of X given Y is

$$\Pr\left[X = x \,\middle|\, Y = y\right] = \frac{\Pr\left[(X = x) \cap (Y = y)\right]}{\Pr\left[Y = y\right]}.$$

An equivalent, useful statement of this is that:

$$\Pr\left[(X = x) \cap (Y = y)\right] = \Pr\left[X = x | Y = y\right] * \Pr\left[Y = y\right]$$

Two events X and Y are independent, if $P\left[X = x \cap Y = y\right] = P\left[X = x\right] * P\left[Y = y\right]$. In particular, if X and Y are independent, then

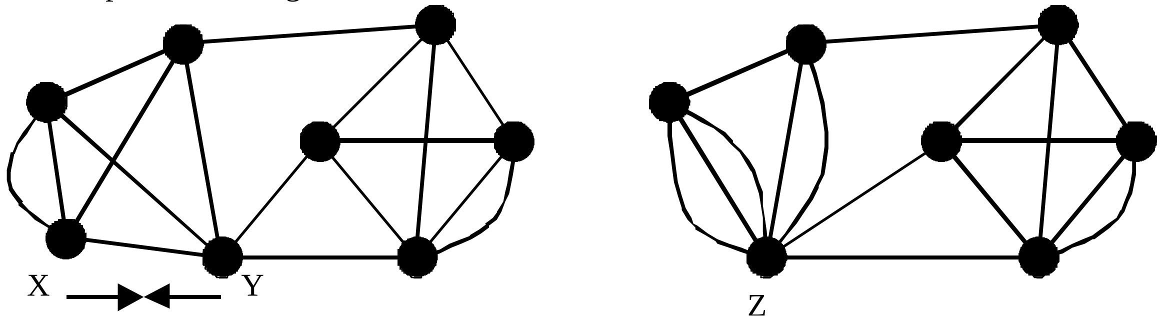
$$\Pr\left[X = x \,\middle|\, Y = y\right] = \Pr\left[X = x\right].$$

Let $\eta_1, \ldots, \eta_n$ be n events which are not necessarily independent. Then,

$$\Pr\left[\cap_{i=1}^{n} \eta_i\right] = \Pr\left[\eta_1\right] * \Pr\left[\eta_2 \,\middle|\, \eta_1\right] * \Pr\left[\eta_3 \,\middle|\, \eta_1 \cap \eta_2\right] * \ldots * \Pr\left[\eta_n \,\middle|\, \eta_1 \cap \ldots \cap \eta_{n-1}\right]$$

2 Algorithm

The basic operation, is edge contraction:



The new graph is denoted by G/xy.
Operation can be implemented in $O(n)$ time for a graph with n vertices (how?).

Observation 2.1 *The size of the in G/xy is at least as large as the cut in G (as long as G/xy as at least one edge). Since any cut in G/xy has a corresponding cut of the same cardinality in G.*

Observation 2.2 *Let $e_1, \ldots, e_{n-2}$ be a sequence of edges in G, such that none of them is in the min-cut, and such that $G' = G/\{e_1, \ldots, e_{n-2}\}$ is a single multi-edge. Then, this multi-edge correspond to the min-cut in G.*

Idea: Let us find a sequence of edges $e_1, \ldots, e_{n-2}$, such that $G/\{e_1, \ldots, e_{n-2}\}$ corresponds to the minimum cut.

Problem: Argumentation is circular, how can we find a sequence of edges that are not in the cut without knowing what the cut is???

Lemma 2.3 *If a graph G has a min-cut of size k, and it has n vertices, then $|E(G)| \geq kn/2$.*

Proof: Vertex degree is at least k. Now count the number of edges... ■

Lemma 2.4 *If we pick in random an edge e from a graph G, then with probability at most $2/n$ it belong to the min-cut.*

Proof: There are at least $nk/2$ edges in the graph and exactly k edges in the cut. Thus, the probability of picking an edge from the min-cut is small then $k/(nk/2) = 2/n$. ■

```
Algorithm MinCutInner(G)
        $G_0 \leftarrow G$
        $i = 0$
        while $G_i$ has more than two vertices do
                Pick randomly an edge $e_i$ from the edges of $G_i$
                $G_{i+1} \leftarrow G_i / e_i$
                $i \leftarrow i + 1$

        Let $(S, V - S)$ be the cut in the original graph corresponding to $G_i$

        return $(S, V - S)$
```

Observation 2.5 *`MinCutInner` runs in $O(n^2)$ time.*

Observation 2.6 *The algorithm always outputs a cut, and the cut is not smaller than the minimum cut.*

Lemma 2.7 *`MinCutInner` outputs the min cut in probability $\geq 2/n(n-1)$.*

Proof: Let η_i be the event that e_i is not in the min-cut of G_i. Clearly, MinCut outputs the minimum cut if $\eta_0, \ldots, \eta_{n-3}$ all happen (namely, all edges picked are outside the min cut).

By the above lemma,

$$\Pr\left[\eta_i \,\middle|\, \eta_1 \cap \ldots \cap \eta_{i-1}\right] \geq 1 - \frac{2}{|V(G_i)|} = 1 - \frac{2}{n-i}$$

Thus,

$$\begin{aligned}\Pr\left[\eta_0 \cap \ldots \cap \eta_{n-2}\right] &= \Pr\left[\eta_0\right] * \Pr\left[\eta_1 \,\middle|\, \eta_0\right] * \Pr\left[\eta_2 \,\middle|\, \eta_0 \cap \eta_1\right] \\ &\quad * \ldots * \Pr\left[\eta_{n-3} \,\middle|\, \eta_0 \cap \ldots \cap \eta_{n-4}\right]\end{aligned}$$

Thus,

$$\Pr\left[\eta_0 \cap \ldots \cap \eta_{n-2}\right] \geq \prod_{i=0}^{n-3}\left(1 - \frac{2}{n-i}\right) = \prod_{i=0}^{n-3} \frac{n-i-2}{n-i} = \frac{n-2}{n} * \frac{n-3}{n-1} * \frac{n-4}{n-2} \cdots$$
$$= \frac{2}{n \cdot (n-1)}.$$ ■

Definition 2.8 (informal) Amplification is the process of running an experiment again and again till the things we want to happed with good probability.

Let MinCut be the algorithm that runs `MinCutInner` $n(n-1)$ times and return the minimum cut computed.

Lemma 2.9 *The probability that MinCut fails to return the min-cut is* < 0.14.

Proof: The probability of failure is at most

$$\left(1 - \frac{2}{n(n-1)}\right)^{n(n-1)} \leq \exp\left(-\frac{2}{n(n-1)} \cdot n(n-1)\right) = \exp(-2) < 0.14,$$

since $1 - x \leq e^{-x}$ for $0 \leq x \leq 1$, as you can (*and should*) verify. ■

Theorem 2.10 *One can compute the min-cut in $O(n^4)$ time with constant probability to get a correct result. In $O\left(n^4 \log n\right)$ time the min-cut is returned with high probability.*

Note: that the algorithm is extremely simple, can we push the basic idea further and get faster algorithm? (or alternatively, can we complicate things, and get a faster algorithm?)

So, why is the algorithm needs so many executions? Because the probability deteriorates very quickly once the graph becomes small. The probability for success in contracting the graph till it has t vertices is:

$$\Pr\left[\eta_0 \cap \ldots \cap \eta_{n-t-1}\right] \geq \prod_{i=0}^{n-t-1}\left(1 - \frac{2}{n-i}\right) = \prod_{i=0}^{n-t-1} \frac{n-i-2}{n-i}$$
$$= \frac{n-2}{n} * \frac{n-3}{n-1} * \frac{n-4}{n-2} \cdots = \frac{t(t-1)}{n \cdot (n-1)}.$$

Thus, as long as t is large (that is $t \geq n/c$, where c is a constant), the probability of hitting the cut is pretty small.

Observation 2.11 *As the graph get smaller, the probability to make a bad choice increases.*

Observation 2.12 *Intuitive idea: Run the algorithm more times when the graph get small.*

```
Contract( $G,t$ )
    while $|V(G)| > t$ do
        Pick a random edge $e$ in $G$.
        $G \leftarrow G/e$

    return $G$
```

Namely, `Contract`(G, t) shrinks G till it has only t vertices.

```
FastCut($G = (V, E)$)
  INPUT: $G$ multigraph
begin
    $n \leftarrow |V(G)|$
    if $n \leq 6$ then
            compute min-cut of $G$ using brute force and return cut.
    $t \leftarrow n/\sqrt{2}$
    $H_1 \leftarrow$ Contract($G$,$t$)
    $H_2 \leftarrow$ Contract($G$,$t$) /* Contract is randomized!!! */
    $X_1 \leftarrow$ FastCut ($H_1$)
    $X_2 \leftarrow$ FastCut ($H_2$)

    return the smaller cut out of $X_1$ and $X_2$.
end
```

Lemma 2.13 *The running time of* `FastCut(G)` *is* $O\left(n^2 \log n\right)$*, where* $n = |V(G)|$.

Proof: Well, we perform two calls to `Contract`(G,t) which takes $O(n^2)$ time. And then we perform two recursive calls, on the resulting graphs. We have:

$$T(n) = O\left(n^2\right) + 2T\left(\frac{n}{\sqrt{2}}\right)$$

The solution to this recurrence is $O\left(n^2 \log n\right)$ as one can easily (and should) verify. ∎

Exercise 2.14 *Show that one can modify* `FastCut` *so that it uses only* $O(n^2)$ *space.*

Lemma 2.15 *The probability that* `Contract`(G,t) *had* not *contracted the min-cut is at least* $1/2$.

Theorem 2.16 `FastCut` *finds the min-cut with probability larger than* $\Omega(1/\log n)$.

Proof: Let $P(t)$ be the probability that the algorithm succeeds on a graph with t vertices.

The probability to succeed in the first call on H_1 is the probability that `Contract` did not hit the min cut (this probability is larger than $1/2$ by the above lemma), times the probability that the algorithm succeeded on H_1 (those two events are independent. Thus, the probability to succeed on the call on H_1 is at least $(1/2) * P(t/\sqrt{2})$, Thus, the probability to fail on H_1 is $\leq 1 - \frac{1}{2}P\left(\frac{t}{\sqrt{2}}\right)$.

The probability to fail on both H_1 and H_2 is smaller than

$$\left(1 - \frac{1}{2}P\left(\frac{t}{\sqrt{2}}\right)\right)^2.$$

And thus, the probability for the algorithm to succeed is

$$P(t) \geq 1 - \left(1 - \frac{1}{2}P\left(\frac{t}{\sqrt{2}}\right)\right)^2 = P\left(\frac{t}{\sqrt{2}}\right) - \frac{1}{4}\left(P\left(\frac{t}{\sqrt{2}}\right)\right)^2.$$

We need to solve this recurrence. Divide both sides of the equation by $P\left(t/\sqrt{2}\right)$ we have:

$$\frac{P(t)}{P(t/\sqrt{2})} \geq 1 - \frac{1}{4}P(t/\sqrt{2})$$

It is now easy,[1] to verify that this inequality holds for $P(t) = 1/\log t$. Indeed,

$$\frac{1/\log t}{1/\log(t/\sqrt{2})} \geq 1 - \frac{1}{4(\log(t/\sqrt{2}))} \iff \frac{\log t - \log\sqrt{2}}{\log t} \geq \frac{4(\log t - \log\sqrt{2}) - 1}{4(\log t - \log\sqrt{2})}$$

Let $\Delta = \log t$. Then,

$$\begin{aligned}
& \frac{\Delta - \log\sqrt{2}}{\Delta} \geq \frac{4(\Delta - \log\sqrt{2}) - 1}{4(\Delta - \log\sqrt{2})} \\
\iff\ & 4(\Delta - \log\sqrt{2})^2 \geq 4\Delta(\Delta - \log\sqrt{2}) - \Delta \\
\iff\ & -8\Delta\log\sqrt{2} + 4\log^2\sqrt{2} \geq -4\Delta\log\sqrt{2} - \Delta \\
\iff\ & \Delta - 4\Delta\log\sqrt{2} + 4\log^2\sqrt{2} \geq 0 \\
\iff\ & \left(\Delta - 2\log\sqrt{2}\right)^2 \geq 0,
\end{aligned}$$

which is definitely true. Thus, we just verified that $P(t) \geq 1/\log t$, for t large enough. We conclude, that the algorithm succeeds in finding the min-cut in probability $\geq 1/\log t = 1/\log n$. ■

Exercise 2.17 *Prove, that running of `FastCut` $c \cdot \log^2 n$ times, guarantee that the algorithm outputs the min-cut with probability $\geq 1 - 1/n^2$ for c a constant large enough.*

3 Notes

The `MinCutInner` algorithm was developed by David Karger during his PhD thesis in Stanford. The fast algorithm is a joint work with Stein. David Karger is currently a processor to CS in MIT. Clifford Stein is a coauthor of CLRS [CLRS01], and he is currently in Columbia university.

The basic algorithm of the min-cut is described in [MR95, pages 7–9], the faster algorithm is described in [MR95, pages 289–295].

References

[CLRS01] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press / McGraw-Hill, Cambridge, Mass., 2001.

[MR95] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, New York, NY, 1995.

[1]easy = I did it in less than five days.