

Advanced Algorithms

Spectral methods and algorithms

尹一通 栗师 刘景铖

Probability amplification

Say you have a randomized algorithm that fails with probability β

To boost success probability, we can run it multiple times until it succeed

Run independently for t rounds, the failure probability becomes β^t

Q: Can we save randomness while still achieving the same probability amplification?

Imagine a random walk on the $N = 2^n$ random bits

There is a set B of size βN that we try to escape from (or avoid)


We want that the escape probability close to β^t

Q: Can we use a sparse expander instead of a complete graph for the random walk?

Hitting property of expander walks

Let G be a d -regular graph with n vertices, $\alpha = \epsilon d$ be its spectral radius and B be a set of size at most βn .

Then, starting from a uniformly random vertex, the probability that a t -step random walk has never escaped from B , denoted by $P(B, t)$, is at most $(\beta + \epsilon)^t$.


$$\Pr[X_0 \in B, X_1 \in B, X_2 \in B, \dots, X_t \in B]$$

Remarks before a proof:

- Compare this to a sequence of independent samples.
- Expander mixing lemma is like $t = 2$: Note that $\varphi(S) = \Pr(X_2 \notin S \mid X_1 \sim \pi_S)$
- Bound can be strengthened \rightarrow see Chapter 4 of ***Pseudorandomness, by Vadhan***
- Applications to error reduction for randomized algorithms
 - Instead of using kt bits of randomness, only need $k + O(t \log d)$
 - for one-sided error, escaping the bad set of “random bits”
 - for two-sided error, a Chernoff type bound can also be shown \rightarrow then take the majority of the answers

$$\Pi_B = \begin{matrix} & B & V \setminus B \\ \begin{matrix} B \\ V \setminus B \end{matrix} & \begin{bmatrix} I_B & 0 \\ 0 & 0 \end{bmatrix} \end{matrix}$$

$$\Pi_B \Pi_B = \Pi_B$$

Hitting property of expander walks

Proof. Observe that $P(B, t) = \|(\Pi_B W)^t \Pi_B u\|_1$

$$W = \frac{1}{d} A$$

$$u = \frac{1}{n} \vec{1}$$

To see this, notice that $\Pr[X_0 \in B] = \|\Pi_B u\|_1$

$$\Pr[X_0 \in B, X_1 \in B] = \|\Pi_B W \Pi_B u\|_1$$

And so on and so forth.

Suppose that we can show $\forall f: f$ is a probability distribution, we have

$$\|\Pi_B W \Pi_B f\|_2 \leq (\beta + \epsilon) \|f\|_2$$

Then,

$$\begin{aligned} \|(\Pi_B W)^t \Pi_B u\|_1 &\leq \sqrt{n} \|(\Pi_B W)^t \Pi_B u\|_2 \\ &= \sqrt{n} \|(\Pi_B W \Pi_B)^t u\|_2 \\ &\leq \sqrt{n} (\beta + \epsilon)^t \|u\|_2 \\ &= (\beta + \epsilon)^t \end{aligned}$$

Cauchy-Schwarz inequality:

$$\langle u, v \rangle \leq \sqrt{\langle u, u \rangle} \cdot \sqrt{\langle v, v \rangle}$$

Hitting property of expander walks

$$\Pi_B = \begin{array}{cc} & \begin{matrix} B & V \setminus B \end{matrix} \\ \begin{matrix} B \\ V \setminus B \end{matrix} & \begin{bmatrix} I_B & 0 \\ 0 & 0 \end{bmatrix} \end{array}$$


$$W = \frac{1}{d}A \text{ has } \lambda_2(W^\top W) = \epsilon^2$$

Proof (cont'd): It remains to show $\forall f: f$ is a probability distribution,

$$\|\Pi_B W \Pi_B f\|_2 \leq (\beta + \epsilon) \|f\|_2$$

Without loss of generality, we can assume f is supported only on B .

$$\|\Pi_B W \Pi_B f\|_2 = \|\Pi_B W f\|_2 = \|\Pi_B W(u + v)\|_2 \leq \|\Pi_B u\|_2 + \|\Pi_B W v\|_2$$


$$u = \frac{1}{n} \vec{1}, \text{ so } \frac{\langle f, u \rangle}{\langle u, u \rangle} u = u, \text{ then } v \perp \vec{1}$$

Next, $\|\Pi_B W v\|_2 \leq \|W v\|_2 \leq \epsilon \|v\|_2 \leq \epsilon \|f\|_2$.

On the other hand, $\|\Pi_B u\|_2 = \sqrt{\frac{\beta}{n}} \leq \beta \|f\|_2$,

where last inequality follows from Cauchy-Schwarz:

$$1 = \|f\|_1 = \langle 1_B, f \rangle \leq \sqrt{\beta n} \|f\|_2$$

Combined together, we have $\|\Pi_B W \Pi_B f\|_2 \leq (\beta + \epsilon) \|f\|_2$ as desired.

Cauchy-Schwarz inequality:

$$\langle u, v \rangle \leq \sqrt{\langle u, u \rangle} \cdot \sqrt{\langle v, v \rangle}$$

Hitting property of expander

To get a tail bound, consider
 $P(S, t) = \|\Pi_{Z_t} W \Pi_{Z_{t-1}} W \dots \Pi_{Z_1} u\|_1$
where $S = (Z_t, Z_{t-1}, \dots, Z_1)$
indicates whether $Z_i \in \{B, \bar{B}\}$

Proof. Observe that $P(B, t) = \|(\Pi_B W)^t \Pi_B u\|_1$

Suppose that we can show $\forall f: f$ is a probability distribution, we have $\|\Pi_B W \Pi_B f\|_2 \leq (\beta + \epsilon) \|f\|_2$. Then,
 $\|(\Pi_B W)^t \Pi_B u\|_1 \leq \sqrt{n} \|(\Pi_B W)^t \Pi_B u\|_2 = \sqrt{n} \|(\Pi_B W \Pi_B)^t u\|_2 \leq \sqrt{n} (\beta + \epsilon)^t \|u\|_2 = (\beta + \epsilon)^t$

It remains to show $\forall f: f$ is a probability distribution,

$$\|\Pi_B W \Pi_B f\|_2 \leq (\beta + \epsilon) \|f\|_2$$

Without loss of generality, we can assume f is supported only on B .

$$\|\Pi_B W \Pi_B f\|_2 = \|\Pi_B W f\|_2 = \|\Pi_B W(u + v)\|_2 \leq \|\Pi_B u\|_2 + \|\Pi_B W v\|_2$$

Next, $\|\Pi_B W v\|_2 \leq \|W v\|_2 \leq \epsilon \|v\|_2 \leq \epsilon \|f\|_2$.

On the other hand, $\|\Pi_B u\|_2 = \sqrt{\frac{\beta}{n}} \leq \beta \|f\|_2$,

The last inequality follows from Cauchy-Schwarz:

$$1 = \|f\|_1 = \langle 1_B, f \rangle \leq \sqrt{\beta n} \|f\|_2$$

Combined together, we have $\|\Pi_B W \Pi_B f\|_2 \leq (\beta + \epsilon) \|f\|_2$ as desired.

Graph Sparsification

Given an undirected graph $G=(V,E)$ with a weight $w(e)$ on each edge e in E , we are interested in finding a “sparse” graph H that approximates G well.

There are different notions of sparsifiers, depending on what properties to preserve.

NOT in this class:

- spanners, which approximately preserves pairwise distance
- Gomory-Hu tree for all pairs mincut

We saw that, in many ways, a d -regular expander behaves like a complete graph.

Our journey starts with a notion called cut sparsifier.

Cut Approximator

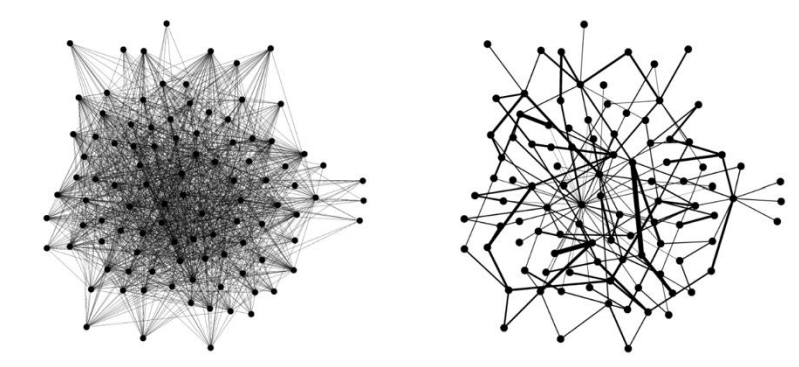
Let $\delta_G(S)$ be the set of edges with one endpoint in S and one endpoint in $V-S$.

Let $w(\delta_G(S)) = \sum_{e \in \delta_G(S)} w(e)$ be the total weight of edges in $\delta_G(S)$.

Definition (ϵ -cut approximator)

A weighted graph $H=(V,F)$ is an ϵ -cut approximator of $G=(V,E)$ if for **every** subset $S \subseteq V$,

$$(1 - \epsilon)w(\delta_G(S)) \leq w(\delta_H(S)) \leq (1 + \epsilon)w(\delta_G(S))$$



Note that edges in G and H could have different weights.

First Result in Cut Sparsification

Karger came up with the first sparsification result and found surprising applications.

Assumptions: The graph G is unweighted and the min-cut value of G is $\Omega(\log n)$.

With the assumption on the min-cut value, there is a very simple algorithm for graph sparsification.

Algorithm: Set a sampling probability p .

For each edge e , with probability p , put e in H with weight $1/p$.

Expectation

For each edge e in G , the expected weight of e in H is 1.

By linearity of expectation, for every $S \subseteq V$,

$$E[|\delta_H(S)|] = p \cdot |\delta_G(S)| \quad \text{and} \quad E[w(\delta_H(S))] = |\delta_G(S)|.$$

So, the expectation is correct for every cut S .

But does it mean that all cuts will have value close to the expectation simultaneously?

Karger's Theorem

Theorem (Karger) For any $0 < \epsilon \leq 1$, set $p = \frac{15 \ln n}{\epsilon^2 c}$, where c is the min-cut value of G .

Then H is an ϵ -cut approximator of G with $O(p \cdot |E(G)|)$ edges with probability $\geq 1 - \frac{4}{n}$.

Note that when c is less than $15 \ln(n)$, then p is at least one and there is no sparsification.

We need to prove that the weight in **every** cut is close to its expected value.

ϵ -cut approximator

For every $S \subseteq V$,

$$(1 - \epsilon)w(\delta_G(S)) \leq w(\delta_H(S)) \leq (1 + \epsilon)w(\delta_G(S))$$

Chernoff-Hoeffding: For any $0 < \epsilon < 1$, let $\mu = E[X]$, where $X = \sum_{i=1}^n X_i$ and each $X_i \in [0,1]$ is independent,

$$\Pr(|X - \mu| \geq \epsilon\mu) \leq 2e^{-\frac{\mu\epsilon^2}{3}}$$

Analysis of One Cut

Consider a subset $S \subseteq V$. Let $|\delta_G(S)| = k$.

Recall that $E[|\delta_H(S)|] = p \cdot |\delta_G(S)| = pk$

Since each edge is included iid with prob. $p = \frac{15 \ln n}{\epsilon^2 c}$, by Chernoff's inequality:

$$\begin{aligned} & \Pr\left(|w(\delta_H(S)) - w(\delta_G(S))| \geq \epsilon \cdot w(\delta_G(S))\right) \\ &= \Pr(||\delta_H(S)| - pk| \geq \epsilon \cdot pk) \leq 2e^{-\frac{pk\epsilon^2}{3}} = 2n^{-\frac{5k}{c}} \end{aligned}$$

Min cut value is c , so the prob. is at most $2n^{-5}$

Simply applying union bound does not work: there are 2^n cuts!

An important observation

The probability that a cut violates the requirement is smaller when the cut size is bigger.

The failure probability is biggest for minimum cuts.

But we know that there can be at most $O(n^2)$ of them!

The following is a generalization of this result.

Lemma. The number of cuts with at most αc edges is at most $n^{2\alpha}$ for any $\alpha \geq 1$.

Proof (Sketch). Similar to the case when $\alpha = 1$, which follows from Karger's randomized min-cut algorithm.

A more careful union bound would work

$$\begin{aligned} \Pr(\text{Some cut is violated}) &\leq \sum_{S \subseteq V} \Pr(\text{cut } S \text{ is violated}) \\ &\leq \sum_{\alpha \geq 1} \sum_{S \subseteq V: \delta_G(S) = \alpha c} \Pr(\text{cut } S \text{ is violated}) \\ &\leq \sum_{\alpha \geq 1} n^{2\alpha} \cdot 2n^{-\frac{5\alpha c}{c}} \\ &\leq 2 \sum_{\alpha \geq 1} n^{-3\alpha} \\ &\leq \frac{4}{n^3} \end{aligned}$$

Example

An almost complete graph has an ϵ -cut approximation with $O\left(\frac{n \log n}{\epsilon^2}\right)$ edges.

Improved Result via Non-Uniform Sampling

Theorem [Benczur-Karger] Any graph G has an ϵ -cut approximator with $O\left(\frac{n \log n}{\epsilon^2}\right)$ edges.

Time permitted, we can see an algebraic generalization of this result later, known as spectral sparsification. One way to get it is by a matrix Chernoff bound.

Standard Applications

Design a fast approximation algorithm for the minimum s-t cut problem.

Minimum Cuts in Near Linear Time (Sketch)

Karger came up with a highly original approach to the min-cut problem, using a classical result about spanning tree packing.

Theorem (Tutte) If G has min-cut value c , then G has $c/2$ edge-disjoint spanning trees.

If we have these edge-disjoint trees $T_1, \dots, T_{\frac{c}{2}}$, then we know that one of these trees, say T , crossed a minimum cut S at most twice.

Karger realized that if we are given T , then we can find S in near linear time using dynamic programming.

Minimum Cuts in Near Linear Time (Sketch)

There is an $\tilde{O}(mc)$ time algorithm to find these disjoint trees, but too slow for our purpose.

Karger's idea, of course, is to use his graph sparsification result.

He used the uniform sampling algorithm described above to sparsify the graph so that its min-cut value is $O(\log n)$.

Then we can use the above algorithm to find the disjoint trees in $\tilde{O}(m)$ time.

Then, one of these trees would cross a min-cut at most twice.

So, running the dynamic programming on these $O(\log n)$ trees, we find a min cut in $\tilde{O}(m)$ time.

UPDATE: Deterministic Mincut in Almost-Linear Time, by Jason Li