Lecture 13

Expander Graphs and Derandomization

Suppose that we have a collection of points $V = \{1, ..., N\}$, where N is very large. Our goal is to uniformly sample points from V using as little space as possible. To do this, we place a graph on V and select the points with a random walk. What types of graphs would be good for this procedure? We could throw in all edges — this will optimize the mixing time. But then at any step we need to choose amongst N possible neighbors, and this can be a complicated task; especially in situations such as we encountered in the last lecture, where determining if a point is a neighbor requires to do some computation (in that case, we had to make a call to the membership oracle, which could be expensive). To reduce the number of choices that the random walk has to make at each step, we want to minimize the degree. But we don't want to sacrifice the mixing time either. *Expander graphs* are often used because they reach the optimal tradeoff, achieving the best possible mixing time (or more precisely, the largest possible second eigenvalue) for a fixed degree.

13.1 Definitions of expanders

There are several definitions of expanders which are all more-or-less, but not quite, equivalent. The one that we will focus on is that of (two-sided) *spectral expanders*.

Definition 13.1. Given $d \in \mathbb{N}$ and $\gamma \in (0, 1)$, a one-sided (resp. two-sided) (d, γ) spectral expander is a graph G such that:

- G is d-regular with degree d.
- $|\lambda_2(\bar{L}) - 1| \leq \gamma$ (resp. $\forall i \geq 2, |\lambda_i(\bar{L}) - 1| \leq \gamma$), where $\overline{L}$ is the normalized Laplacian of G.

Two-sided expanders have all their eigenvalues close to 1, except $\lambda_1 = 0$.; one-sided expanders can also have larger eigenvalues, up to the maximum possible of 2 (and in particular they can be bipartite).

Remark 13.2. Using the mixing lemma from last lecture we find that the lazy random walk on expanders mixes fast:

$$\tau_\epsilon = O\Big(\frac{\log(\frac{n}{\epsilon})}{1-\gamma}\Big).$$

For the case of expanders (and especially two-sided expanders, which are guaranteed to not be bipartite) we will usually run a normal random walk with walk matrix $W = AD^{-1}$, rather than the lazy random walk with walk matrix $W = \mathbb{I} - \frac{1}{2}D^{1/2}\bar{L}D^{-1/2}$ that we considered previously.

There are two other common definitions of expanders:

Definition 13.3. A (d, γ_e) edge expander is a graph G such that:

- G is d-regular with degree d.
- $\phi(G) = \min\limits_{1 \le |S| \le \frac{n}{2}} \frac{|\partial S|}{d \cdot |S|} \ge \gamma_e$.

Definition 13.4. A (d, γ_v) vertex expander is a graph G such that:

- G is d-regular with degree d.
- $\sigma(G) = \min\limits_{1 \le |S| \le \frac{n}{2}} \frac{|N(S)|}{|S|} \ge \gamma_v$, where $N(S)$ is the set of all neighbors of S lying outside of S. (Here $\sigma(G)$ is called the *vertex expansion* of G.)

The following relation holds between these two notions:

$$\frac{\gamma_v}{d} \le \gamma_e \le \gamma_v.$$

The first inequality follows from the fact that the number of edges entering S is at least the size of its neighborhood. The second inequality follows from the size of the neighborhood being at least the number of edges divided by the degree.

13.2 Limits on expansion

From this point on we will focus on two-sided spectral expanders. Given a target degree d, how small can we make γ?

Theorem 13.5. *For any d-regular graph,*

$$\gamma := \max_{i=2,\dots,n} |\lambda_i(\bar{L}) - 1| \ge 2(1 - o_n(1))\sqrt{\frac{1}{d} - \frac{1}{d^2}}$$
$$\sim 2\sqrt{\frac{1}{d}}.$$

We prove a slightly weaker version of this bound, which is missing the factor 2:

Proof. If A is the adjacency matrix of a graph G then for any $k \geq 1$ the (i,j)-th entry $(A^k)_{ij}$ counts the number of paths from i to j with length exactly k. So $(A^2)_{ii}$ is just the number of edges incident on the i-th vertex, i.e. its degree. Therefore $\mathrm{Tr}(A^2) = nd$. We also know that for a symmetric matrix,

$$\begin{aligned}
\mathrm{Tr}(A^2) &= \sum_{i=1}^{n} \lambda_i^2(A) \\
&= \sum_{i=1}^{n} \lambda_i^2(d(\mathbb{I} - \bar{L})) \\
&= d^2 \sum_{i=1}^{n} (1 - \lambda_i(\bar{L}))^2 \\
&\leq d^2 + d^2(n-1)\gamma^2.
\end{aligned}$$

Therefore $d^2(n-1)\gamma^2 \geq nd - d^2$, i.e.

$$\gamma^2 \geq \frac{n}{d(n-1)} - \frac{1}{n-1}.$$

For large n the last term is very small, and we get the bound $\gamma \geq (1 - o_n(1))\sqrt{\frac{1}{d}}$, which is off by just a factor 2. □

Ramanujan graphs are graphs that achieve the optimal expansion for a given degree:

Definition 13.6. A Ramanujan graph is a d-regular graph such that

$$\gamma = 2\sqrt{\frac{1}{d} - \frac{1}{d^2}}.$$

13.3 Constructions of expanders

Do such good expanders as Ramanujan graphs even exist? The probabilitic method can be used to show that random d-regular graphs are good expanders with high probability:

Theorem 13.7 (Friedman)**.** *$\forall \epsilon > 0$, a random d-regular graph on n vertices satisfies:*

$$\mathbf{Pr}\left(\gamma \leq 2\sqrt{\tfrac{1}{d} - \tfrac{1}{d^2}} + \epsilon\right) = 1 - o_n(1).$$

Unfortunately random graphs are not so useful in practice because they are just that: random. In particular, working with a random graph requires to first compute the whole graph, then store it in memory and perform the random walk. But recall that for our typical applications of expanders we are thinking of working with potentially large graphs for which

we'd like to be able to compute the neighborhood structure locally very efficiently. Such a construction was presented by Margulis; efficiency aside it is the first explicit construction of a good (meaning that both d and γ are constants independent of the size of the graph) family of expanders:

Example 13.8 (Margulis '73)**.** Take $V = \mathbb{Z}_m \times \mathbb{Z}_m$ for some integer m. For each vertex $(x, y) \in V$ connect it to the following eight vertices:

$$N((x,y)) = \begin{cases} (x, y \pm x) \\ (x, y \pm (x+1)) \\ (x \pm y, y) \\ (x \pm (y+1), y) \end{cases}$$

Theorem 13.9 (Margulis)**.** *The graph given above is a $(8, \gamma)$ two-sided spectral expander for some $\gamma < 1$ independent of m.*

The construction of the graph is very simple, but the proof of the theorem is very difficult. This construction also provides a very fast mixing time: it provides a way to mix a 2-dimensional $m \times m$ grid in time $O(\log m)$, whereas as we've seen the regular random walk would require time $O(m^2)$. This only requires us to double the degree and throw in a few "long-distance hops".

The first construction of explicit Ramanujan expanders was given by Lubotzky, Philips and Sarnak:

Theorem 13.10 (LPS '88)**.** *for every $n = p + 1$ where:*

- *p is a prime such that $p \equiv 1[4]$,*
- *$d = q^m + 1$, q prime, m integer,*

there exists a $(d, \gamma = 2\sqrt{\frac{1}{d} - \frac{1}{d^2}})$ spectral expander.

Very recent works by Marcus, Spielman and Srivastava give explicit constructions of *bipartite* expanders for every possible degree d and size n. It is still an open problem whether (non-bipartite) Ramanujan expanders exist for all possible degrees.

13.4 Application to derandomization

In the last lecture we saw a randomized algorithm that could efficiently solve a problem, volume estimation, that we also argued was too hard to solve deterministically. One might ask whether it is still possible to remove the randomness from such algorithms to obtain efficient deterministic algorithms. The process of reducing or eliminating randomness from an algorithm is called **derandomization**. The question of derandomizing all polynomial-time algorithms is the question as to whether $\mathbf{P} = \mathbf{BPP}$, where:

Definition 13.11. **P** is the class of problems that can be solved deterministically in polynomial time. **BPP** is the class of problems for which there is a randomized polynomial-time algorithm that makes the correct decision for every input with probability $\geq \frac{2}{3}$.

In the last lecture with volume estimation we seemed to show that for a specific problem in **BPP** it was impossible to have an equivalent deterministic algorithm — thus $\mathbf{P} \neq \mathbf{BPP}$? However the impossibility result relied on the assumption that the only access to the convex set was through a membership/separation oracle. In "real life" it will in general be the case that other details of the problem are available, and may be used to construct a deterministic polynomial-time algorithm. This is what makes proving separations of complexity classes difficult!

13.4.1 Polynomial Identity Testing

A popular candidate for trying to separate **P** from **BPP** is polynomial identity testing (PIT):

Definition 13.12 (Polynomial Identity Testing)**.** Given a (large) finite field $\mathbb{F}$ and an n-variate polynomial $p \in \mathbb{F}[x_1, ..., x_n]$ provided as an arithmetic circuit (eg. $(x_1+3x_2-x_3)(3x_1+x_4-1)$), determine whether $p \equiv 0$, i.e. all coefficients of p when fully expanded are zero.

To solve the above problem we could simply multiply all the terms out and check whether the final coefficient of every term is 0, but that could take exponential time in the size of the circuit specifying p. No known deterministic algorithm can solve PIT in polynomial time, but there is a simple randomized algorithm that can.

Lemma 13.13 (Schwartz-Zippel)**.** *Given a non-zero $p \in \mathbb{F}[x_1, ..., x_n]$ with degree$(p) \leq d$, let $S \subseteq \mathbb{F}$ and $(s_1, ...s_n)$ be n points selected independently and uniformly at random from S. Then:*

$$\mathbf{Pr}\left(p(s_1, ..., s_n) = 0\right) \leq \tfrac{d}{|S|}.$$

The proof is not hard, and we omit it (hint: proceed by induction on the number of variables). Based on the lemma, if we simply evaluate the polynomial at enough random points, we can easily determine whether $p = 0$ with high probability (assuming $|\mathbb{F}| \gg d$). A sample application is to determine whether a bipartite graph $G = (L, R, E)$ has a perfect matching. Consider the $|L| \times |R|$ matrix M which has a variable x_{ij} in position $(i, j) \in E$ and zero otherwise. Then we see that $\det(M) \neq 0$ as a multivariate polynomial if and only if M has a perfect matching. Using the Schwartz-Zippel lemma we can check this efficiently in randomized polynomial-time. Note there are also efficient deterministic algorithms for finding perfect matchings, so this is not a breakthrough... But it turns out to be an important technique to design fast *parallel* algorithms for constructing perfect matchings.

13.4.2 Pseudorandom Generators

A popular approach for trying to show that $\mathbf{P} = \mathbf{BPP}$ is constructing good pseudorandom generators (PRG).

Definition 13.14. A pseudorandom generator is a function $g : \{0,1\}^s \to \{0,1\}^n$, where $n \geq s$ and s is called the *seed length*. We say that a PRG g ϵ-fools a class of functions $\mathcal{C} \subseteq \{f : \{0,1\}^n \to \{0,1\}\}$ if $\forall f \in \mathcal{C}$:

$$\left| \mathop{\mathbf{Pr}}_{x \in \{0,1\}^n} \left[f(x) = 1 \right] - \mathop{\mathbf{Pr}}_{y \in \{0,1\}^s} \left[f(g(y)) = 1 \right] \right| \leq \epsilon,$$

where both probabilities are taken over a uniformly random choice of x, y respectively. (Intuitively, this means that from the point of view of any function $f \in \mathcal{C}$, the output of g looks random)

Here is how we might use this to prove that $\mathbf{P} = \mathbf{BPP}$. Let $\mathcal{C}$ be the class of all functions that can be computed by a polynomial time-randomized algorithm with success probability at least $2/3$. Such algorithms can be understood as functions f of two variables, the input x to the problem and the randomness r. Fixing all possible inputs x of a certain length gives us a large class of functions $\mathcal{C}'$ which are functions of the randomness only. If we design a PRG that ϵ-fools the class $\mathcal{C}'$ with $\epsilon < \frac{1}{3}$ and $s = O(\log n)$, then we could derandomize **BPP** by trying all $2^s = \text{poly}(n)$ possible seeds to our PRG and computing a good estimate of the probability that f would accept on any input.

13.5 Expanders for derandomization

We will use expanders, not with the intent of producing a PRG, but to derandomize the following problem: Suppose that we have a **BPP** algorithm which given an input generates random bits in $\{0,1\}^r$ and has an error rate of $\frac{1}{100}$. If we run the algorithm t times and take the majority, then we can reduce the error to $(\frac{1}{100})^t$. Doing so requires $t \cdot r$ bits. We will show that by using expanders, we can get a similar result using much fewer bits.

Theorem 13.15. *Given any* **BPP** *algorithm which requires r random bits and has error $\leq \frac{1}{100}$, we can construct an algorithm solving the same problem with error $\leq (\frac{2}{\sqrt{5}})^t$ and using only $r + 9t$ random bits.*

The idea for the algorithm is very simple. We fix a (d, γ) expander G on $V = \{0,1\}^r$ with $d \leq 400$ and $\gamma \leq \frac{1}{10}$. We haven't seen how to construct such a thing but I promise you that it exists, and it's not that hard to get. We then pick a random vertex to start from using r bits, and for each of t iterations we perform a random walk to a neighbor using $\log_2(400) \approx 9$ bits. For each of the vertices traversed we run the algorithm with the corresponding bits and take the majority outcome at the end.

Note that we do not need to actually compute the whole graph (which would have size exponential in r); we only need to be able to access a small number of vertices and their neighbors. The hope is that running the **BPP** algorithm on the highly correlated pseudo-random bits generated by this procedure will generate similar results to uncorrelated random bits. This is not trivial, and in particular it does not follow from the analysis of the

mixing time given in the previous lecture. That analysis only shows that mixing happens in $O(\log 2^r) = O(r)$ steps, but here t could be much smaller than r.

Proof. Given an input x, we know that for any set of random bits y,

$$\mathop{\mathbf{Pr}}_{y\in\{0,1\}^r}\big(\text{Alg fails on input } x \text{ and randomness } y\big) \le \frac{1}{100}.$$

Fix an input x, and let $X = \{y : \text{Alg fails on } y\}$, $Y = \{0,1\}^n \setminus X$, $v_0, v_1, \ldots, v_t$ the vertices selected from the random walk and $S = \{i : v_i \in X\}$. Note that X is at most a fraction $\frac{1}{100}$ of the graph and that since we are taking majority at the end, $\mathbf{Pr}(fail) = \mathbf{Pr}(|S| \ge t/2)$. We will use a regular "non-lazy" random walk, as there is no point in staying at a vertex. This gives us a random walk matrix $W = AD^{-1} = \frac{1}{d}A$ since the graph is d-regular.

Let D_X be the diagonal matrix where entry i is 1 if $i \in X$ and 0 otherwise, and D_Y the diagonal matrix where entry i is 1 if $i \in Y$ and 0 otherwise. So $D_X + D_Y = \mathbb{I}$. Given a certain set of designated walk steps $R \subseteq \{0, \ldots, t\}$ the probability that all such steps are errors and all other steps are correct is:

$$\mathbf{Pr}(R = S) = \mathbf{1}^T D_{z_t} W \cdots D_{z_1} W D_{z_0} \frac{\mathbf{1}}{n}, \qquad \text{where} \qquad D_{z_i} = \begin{cases} D_X \text{ if } i \in R \\ D_Y \text{ if } i \notin R \end{cases}.$$

So for any fixed R,

$$\begin{aligned}
\mathbf{Pr}(R = S) &\le \|\mathbf{1}\|_2 \cdot \|D_{z_t} W\| \cdots \|D_{z_1} W\| \cdot \|D_{z_0} W\| \Big\| \frac{\mathbf{1}}{n} \Big\|_2 \\
&= \sqrt{n} \cdot \|D_X W\|^{|R|} \cdot \|D_Y W\|^{t-|R|} \frac{1}{\sqrt{n}} \\
&\le \|D_X W\|^{|R|},
\end{aligned}$$

where the last inequality follows since $\|W\| = \|\frac{A}{d}\| \le 1 \to \|D_Y W\| \le 1$. Now it remains to bound $\|D_X W\|$. Consider any vector x. We can use the decomposition $x = \alpha\mathbf{1} + y$, $y \perp \mathbf{1}$ (these are not the x and y from the original problem description!). Then

$$\|x\|^2 = \alpha^2 n + \|y\|^2,$$

and

$$D_X W x = \alpha D_X W \mathbf{1} + D_X W y.$$

To bound the first term, use that $W\mathbf{1} = \mathbf{1}$ and so

$$\|\alpha D_X W \mathbf{1}\| = |\alpha|\sqrt{|X|} \le \frac{\|x\|}{\sqrt{n}}\sqrt{|X|} \le \frac{\|x\|}{10}.$$

To bound the second term,

$$\|D_X W y\| \le \|W y\| \le \frac{1}{10}\|y\| \le \frac{\|x\|}{10},$$

where the second inequality follows since $y \perp \mathbf{1}$. Thus we have shown that $\|D_X W\| \leq \frac{1}{5}$. Finally, putting everything together,

$$\begin{aligned}
\mathbf{Pr}(fail) &= \mathbf{Pr}\left(|S| \geq \frac{t}{2}\right) \\
&= \sum_{R:|R|\geq\frac{t}{2}} \mathbf{Pr}[R = S] \\
&\leq \sum_{R:|R|\geq\frac{t}{2}} \|D_X W\|^{|R|} \\
&\leq 2^t (\frac{1}{5})^{\frac{t}{2}}.
\end{aligned}$$

□