6.841: Advanced Complexity Theory Fall 2012

Lecture 12 — October 18, 2012

Prof. Dana Moshkovitz *Scribe: Justin Kopinsky*

1 Overview

Last time we showed

$$\|G^l\vec{p} - \vec{u}\| \leq \lambda^l$$

for any connected graph G and initial distribution $\vec{p}$, where $\vec{u}$ is the uniform distribution and λ is the second eigenvalue of G. This gave us a randomized algorithm for solving undirected $s-t$ connectivity by choosing a random neighbor at each step and accepting if we ever reach t and rejecting if we've taken $l = l(n, \lambda)$ steps without finding t.

We didn't have time to see this last time, but there is the following lemma:

Lemma If G is not bipartite, then $\lambda \leq 1 - \frac{1}{poly(n)}$.

In this lecture we derandomize this algorithm by giving a deterministic log space undirected connectivity algorithm, a result due to Reingold[1].

2 Expanders

The main idea behind our approach is to find a class of graphs for which derandomization is easy. Then, if we can transform any graph into a member of such a class, we can fully derandomize connectivity. So...

Definition A graph on n vertices which is d-regular and has 2^{nd} eigenvalue λ is called an **(n, d, λ)-expander**.

In particular we will require $d = O(1)$, $\lambda < 1$, $\lambda = O(1)$. A family of graphs with growing n which satisfy these constraints, is called an *expander family.*

2.1 Motivation

Suppose we wish to solve undirected connectivity on G, and we are told in advance that G is an expander. Then, because $\lambda = O(1)$, we only need $l = \Theta(\log n)$ steps to get $\lambda^l < 1/2n$ (i.e. $G^l p$

is very close to uniform for any starting distribution p). There are only polynomially many such walks, so we can enumerate and simulate all of them, and because G is d-regular (and $d = O(1)$), this takes only logarithmic space to do, giving us a deterministic undirected connectivity algorithm.

Remark Notice that a path $s \to v_1 \to \cdots \to v_n \to t$ is *not* an expander. After $l = \Theta(\log n)$ steps there will still have been no possible way to get from s to most of the nodes in the path, so $G^l p$ cannot possibly be close to uniform, if p is the vector with 1 at s and 0 everywhere else.

3 Main Algorithm

Here we give a full algorithm for solving undirected connectivity on an *arbitrary* graph, G, deterministically in logarithmic space.

Idea We would like to transform G into an expander by creating "shortcuts". As long as we never connect two nodes that are initially disconnected (or disconnect those which are initially connected), we preserve the integrity of the problem, and can make any transformations we like. Notice that we don't have enough space to represent an entire transformed graph, so we will need to keep track of all the transformations implicitly.

Observation It suffices to maintain the relation $N(\langle v, i\rangle) = \langle u, j\rangle$, where u is the i^{th} neighbor of v ($i \in [d]$) and v is the j^{th} neighbor of u. If we can compute $N(\langle v, i\rangle)$ using logarithmic space, we can do all the operations necessary to simulate random walks on G.

3.1 d-Regularization

We first need to make sure that G is d-regular. For any node with degree $< d$, we can just duplicate some edges incident to that node arbitrarily. Then, for any node with degree $D > d$, we split it into a connected collection of D nodes, each of which is incident to $d-1$ other nodes in the collection (e.g. if $d = 3$, we just get a cycle). Furthermore, each node in the collection should be incident to exactly one of the original neighbors of v, so that in total, each node has degree d.

Observation Let's suppose $d = 3$ and call the nodes in the collection corresponding to the original node v by $\{v_k\}$ for $k \in [D]$. Then it is easy to compute the new neighbor function $N^*(\langle v_k, i\rangle)$ as follows:

$$\begin{aligned} N^*(\langle v_k, 1\rangle) &= \langle N(\langle v, k\rangle), 1\rangle \\ N^*(\langle v_k, 2\rangle) &= \langle v_{k-1}, 3\rangle \\ N^*(\langle v_k, 3\rangle) &= \langle v_{k+1}, 2\rangle \end{aligned}$$

and it is easy to check that there are systematic ways to do this for any d. Unfortunately, this method can ruin our bounds on λ, as previously, we could just walk from v to u, but now we may have to walk all the way around the cycle corresponding to v before being able to get to u, so we will need to do something more clever later.

3.2 Dealing with λ

We've shown how to regularize the degree of G, but what about λ? Recall that we need $\lambda = O(1)$ for our algorithm to work.

Observation Consider G^2, i.e. the graph obtained by putting an edge between any two vertices in G which can be reached by a path of length 2 in G. It is well known that if G is represented as a matrix, then the corresponding matrix G^2 really does represent the correct graph. Furthermore, we know $\lambda(G^2) = (\lambda(G))^2$. Better yet, if G was regular, then G^2 is regular too! If we iterate this process enough times for λ to become small, are we done?

Unfortunately not. Every time we square G, the degree of G is squared as well, so if we square G a logarithmic number of times (enough for λ to shrink appropriately), the degree of G will become polynomial, meaning we can't enumerate its paths.

3.3 Replacement Products

Idea Unfortunately, our cycle method for regularizing graphs can significantly increase λ. We need an alternative method of regularization which doesn't deteriorate λ as much, which we can then alternate with squaring to systematically lower λ without blowing up the degree.

To achieve this, we will use the **replacement product**, $G \circledR H$. We require that $|V(H)| = d = deg(G)$, and that H is an expander graph (i.e. fix some expander family beforehand and choose H from it) and construct $G \circledR H$ by replacing each vertex, v, of G with a copy of H, $H_v = \{H_{v1}, \ldots, H_{vd}\}$. Then, if u is the i^{th} neighbor of v in G (and v is u's j^{th} neighbor), we add $deg(H)$ edges from H_{vi} to H_{vj} (we duplicate edges so that the number of edges between separate H_v is about the same as the number of edges interior to a single H_v). The resulting graph, G', will have degree $\sim 2d$.

We need to prove the following lemma:

Lemma If $\lambda(G) \leq 1 - \epsilon$ and $\lambda(H) \leq 1 - \delta$ then $\lambda(G \circledR H) \leq 1 - \epsilon\delta^2/24$.

We start with the following claim:

Claim If G is an (n, d, λ)-expander, then

$$G = (1-\lambda)\begin{pmatrix} \frac{1}{n} & \cdots & \frac{1}{n} \\ \vdots & \ddots & \vdots \\ \frac{1}{n} & \cdots & \frac{1}{n} \end{pmatrix} + \lambda C$$

where $\|C\| \le 1$. Here $\|\cdot\|$ is the spectral norm, defined by

$$\max_{\|v\|_2^2=1} \|Cv\|_2$$

Proof of claim Write $U = \begin{pmatrix} \frac{1}{n} & \cdots & \frac{1}{n} \\ \vdots & \ddots & \vdots \\ \frac{1}{n} & \cdots & \frac{1}{n} \end{pmatrix}$. We can write $C = 1/\lambda(G-(1-\lambda)U)$. We need to show $\|Cv\|_2^2 \le \|v\|_2^2$ holds for any v. Suppose $\{v_i\}$ is the eigenvector basis relative to G. Write $v = \sum \alpha_i v_i$, with $v_1 = \vec{u}$ (the uniform vector). Write $\vec{w} = \sum_{i>1} \alpha_i v_i$. Then:

$$\begin{aligned} C\vec{u} &= \frac{1}{\lambda}(Gu - (1-\lambda)U\vec{u}) \\ &= \frac{1}{\lambda}(\vec{u} - (1-\lambda)\vec{u}) \\ &= \vec{u} \end{aligned}$$

On the other hand,

$$\begin{aligned} \|Cw\|_2^2 &= \frac{1}{\lambda^2}\|Gw\|_2^2 \quad \text{(because } w \perp u \text{ so } Uw = 0) \\ &= \frac{1}{\lambda^2}\sum_{i>1} \alpha_i^2 \|Gv_i\|_2^2 \\ &\le \frac{1}{\lambda^2}\sum \alpha_i^2 \lambda^2 \|v_i\|_2^2 \\ &= \sum \alpha_i^2 \|v_i\|_2^2 \\ &= \|w\|_2^2 \end{aligned}$$

so combining these, we have

$$\|Cv\|_2^2 = \|Cu\|_2^2 + \|Cw\|_2^2 \le \|u\|_2^2 + \|w\|_2^2 = \|v\|_2^2$$

as desired. With this, we can prove the lemma:

Proof of lemma We'll show $\lambda((G Ⓡ H)^3) \leq 1 - \frac{\epsilon\delta^2}{8}$. The original lemma follows by basic approximation theorems from calculus.

We can calculate the matrix $(G Ⓡ H)^3$ as:

$$(G Ⓡ H)^3 = \left(\frac{1}{2} G \otimes I_d + \frac{1}{2}(I \otimes H)\right)^3$$

(where $\otimes$ is the tensor product, i.e. $A \otimes B = \begin{pmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{n1}B & \dots & a_{nn}B \end{pmatrix}$). Using the claim we just proved, we also get:

$$H = \delta U + (1-\delta)H'$$

for some H' with $\|H\| \leq 1$. Then we can calculate the dominant term of the expansion of $(G Ⓡ H)^3$ to be:

$$\frac{\delta^2}{8}(I \otimes U)(G \otimes I_D)(I \otimes U)$$

(which comes from one particular term in the binomial expansion, the other terms can be checked to be inconsequential). However, we know $\lambda(G) \leq 1 - \epsilon$ by assumption, and $\lambda(U) = 1$, so in total, we get:

$$\lambda(G Ⓡ H)^3 \leq \frac{\delta^2}{8}(1)(1-\epsilon)(1) + (1-\delta)H' = 1 - \frac{\epsilon\delta^2}{8}$$

completing the proof.

3.4 Wrapping up

Now, we play a balancing game between powering G to decrease the eigenvalue and using replacement products to reduce the degree. Suppose our initial graph were a d^{50}-regular graph for some constant d, where d is chosen such that H is a $(d^5 0, d/2, 1-\delta)$ expander graph (which can be stored with constant space and computed in constant time). Because we can brute force search for H, we can make $1 - \delta = 0.01$. We iteratively build $G_k = (G_{k-1} Ⓡ H)^5 0$. Notice that G_k is a $d^{50}n$-vertex graph with degree d, where n is the number of vertices in G. These transformations preserve connectivity. Now, by the lemmas above, we know that each connected component $G_{O(logn)}$ is an expander.

Lemma 1. *For every k, every connected component in G_k is a $(d^{50k}n, d^{50}, 1-\epsilon(k))$-expander, where $\epsilon(k) \geq 2^k/n^4$.*

Proof. This is by induction. For the base case: By the lemma in the previous lecture regarding the second eigenvalue of regular graphs, we saw that $\lambda(G) \leq 1 - 1/n^4$. $G Ⓡ H$ is a d-regular graph with

$d^{50}n$ vertices. Thus, $G_1 = (G \circledR H)^{50}$ is a d^{50}-regular graph, with $d^{50}n$ vertices. So first, $\lambda(G \circledR H) \leq 1 - \delta^2/(24n^4) \leq 1 - 1/(25n^4)$, by the lemma above. Then, $\lambda(G_1) \leq \lambda(G \circledR H)^{50} \leq 1 - 2/n^4$. This establishes the base case. Then, assuming G_{k-1} is a $(d^{50(k-1)}n, d^{50}, 1-\epsilon)$-expander, we know that $\lambda(G_{k-1} \circledR H) \leq 1 - \epsilon/25$, and then $\lambda(G_k) \leq 1 - 2\epsilon$. So G_k is a $(d^{50k}n, d^{50}, 1 - 2\epsilon(k-1))$-expander. $\square$

So after $k = O(\log n)$ many steps, we obtain a graph that is a $(\text{poly}(n), d^{50}, O(1))$-expander. Furthermore, we can compute the neighbors of G_k implicitly in logarithmic space given G. We then enumerate over all $O(\log n)$-length walks in G_k. Thus, computing connectivity on G_k can be done deterministically in logarithmic space, giving the desired result.

References

[1] O. Reingold, *Undirected ST-connectivity in log-space*, STOC, 376-385, 2005.