CS 810: Complexity Theory	2/16/2007

Lecture 12: Expanders

Instructor: Dieter van Melkebeek — Scribe: Nathan Collins

In the last lecture we introduced randomized computation in terms of machines that have access to a source of random bits and that return correct answers more than $\frac{1}{2}$ of the time. We showed that BPP has polynomial-size circuits and the conjecture in the community is that BPP = P.

Today we'll introduce expanders, a type of graph that is useful in improving (amplifying) randomized algorithms with little or no additional random bit overhead. But first we'll prove a theorem relating BPP to the polynomial time hierarchy.

1 Hierarchy Results for BPP

Generally, whenever we introduce a new computational model, we look for hierarchy results: Does the class of problems we can solve using the model in question depend on the resources we are allowed? Most hierarchy results use a computable enumeration of all machines of the type in question, but the following can be shown.

Exercise 1. *There does not exist a computable enumeration of the randomized machines with error probability bounded away from* $\frac{1}{2}$.

Therefore, typical hierarchy arguments fail for randomized machines. The typical hierarchy arguments can be tailored to prove a hierarchy theorem for a modified model of randomized computation called promise-BPP. See HW 1.2. However, no hierarchy result is known for BPP.

2 BPP and the Poly Time Hierarchy

Although we don't know if BPP = P, or even if BPP $\subseteq$ NP, we do know that BPP $\subseteq$ PH:

Theorem 1. BPP $\subseteq \Sigma_2^p \cap \Pi_2^p$

Proof. Fix M a randomized polytime machine that accepts $L \in$ BPP, and let r be the number of random bits M uses when running on an input x of size $n = |x|$. We will now show that BPP $\subseteq \Sigma_2^p$. Since BPP = *co*BPP it follows that BPP $\subseteq co\Sigma_2^p = \Pi_2^p$, completing the proof.

We need to remove randomness from the computation. For a given input x, the space $\{0,1\}^r$ of r-bit strings gets partitioned into two pieces: $Acc(x)$, the set of random strings on which M accepts x, and $Rej(x)$, the set of strings on which M rejects x. If the error rate ε of M is small, then $Acc(x)$ will be much larger than $Rej(x)$ when $x \in L$, and $Acc(x)$ will be much smaller than $Rej(x)$ when $x \notin L$. See Figure 1. If our random computation has a small error rate then it will be correct on most random bit strings. We'll turn "most" into "all." The idea is that when $x \in L$ a few "shifts" of $Acc(x)$ will cover the whole space $\{0,1\}^r$ of r-bit random sequences, while the same few shifts of $Acc(x)$ will fail to cover the whole space when $x \notin L$.

If $S \subseteq \{0,1\}^r$ and $\sigma \in \{0,1\}^r$, then $S \oplus \sigma = \{s \oplus \sigma | s \in S\}$, the *shift* of S by σ.[1] Since shifting is invertible we see that $|S \oplus \sigma| = |S|$.

[1]The symbol "$\oplus$" denotes XOR, or, equivalently, addition in $\mathbb{Z}_2^r$

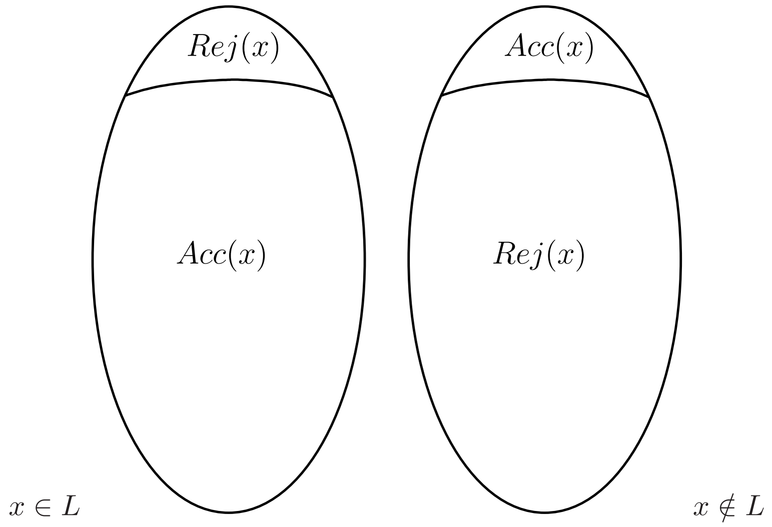

Figure 1: If the error rate ε of M is small, then $Acc(x)$ will be much larger than $Rej(x)$ when $x \in L$, and $Acc(x)$ will be much smaller than $Rej(x)$ when $x \notin L$.

We will use shifts to give a Σ_2-predicate using the intuition discussed above. Namely, consider

$$x \in L \iff \exists \sigma_1, \ldots, \sigma_t \forall \rho \in \{0,1\}^r [\rho \in \bigcup_{i=1}^{t} Acc(x) \oplus \sigma_i]. \tag{1}$$

Now, r is poly in n, and so if we can pick t poly in n as well, then the above will be a Σ_2^p-predicate, provided that we can verify the membership of ρ in $\bigcup_{i=1}^{t} Acc(x) \oplus \sigma_i$ in time poly in n. The membership check is no problem since we can equivalently check that $\rho \oplus \sigma_i \in Acc(x)$ for some i, which is polytime since it corresponds to running M on x with the random bit string $\rho \oplus \sigma_i$, for poly-many σ_i.

We'll show we can pick t a suitable poly to make (1) true by showing that we can choose the σ_is randomly with a high rate of success. There are two cases to consider:

1. $x \in L$: For ρ fixed,

$$\Pr_{\sigma_1,\ldots,\sigma_t}[\rho \notin \bigcup_{i=1}^{t} Acc(x) \oplus \sigma_i] \leq (\mu(Rej(x))^t \leq \varepsilon^t,$$

where $\mu(Rej(x))$ is the probability that M rejects when it should accept, and is hence not larger than ε. So, by union bound,

$$\Pr[\{0,1\}^r \not\subseteq \bigcup_{i=1}^{t} Acc(x) \oplus \sigma_i] \leq |\{0,1\}^r|\varepsilon^t = 2^r \varepsilon^t,$$

and hence if $2^r \varepsilon^t < 1$ then some choice of the σ_is must work.

2. $x \notin L$: We need to be sure that for any choice of the σ_is that some bit string ρ makes M reject x. But

$$\mu(\bigcup_{i=1}^{t} Acc(x) \oplus \sigma_i) \leq \sum_{i=1}^{t} \mu(Acc(x) \oplus \sigma_i) = t\varepsilon,$$

since $\mu(Acc(x)) \leq \varepsilon$ when $x \notin L$, and hence we need $t\varepsilon < 1$.

So, we need to choose t so that both $t\varepsilon$ and $2^r\varepsilon^t$ are less than 1. So $t = r$ works, provided that $\varepsilon < \frac{1}{r}$. From the definition of BPP we only know that $\varepsilon < \frac{1}{3}$, but, using the "majority vote" trick introduced during the last lecture, we can in k runs reduce the error to $e^{-2k\delta^2}$, where $\delta = \frac{1}{2} - \varepsilon > 0$. Since k runs will need kr random bits, we see that it suffices to choose k poly and large enough that $e^{-2k\delta^2} < \frac{1}{rk}$. In fact, $k = O(\log r)$ works, and so an examination of the above shows that the time complexity is $O(T^2 \log T)$ if the original run-time was T. □

3 Expanders

The proof of Theorem 1 used the "majority vote" trick to get an exponential in k increase in accuracy using a linear in k increase in random bit usage. Expander graphs, which are introduced here, lead to an "amplification" technique that gives an exponential in k accuracy improvement using only a constant increase in random bit usage (rk versus $r + k$). Expanders have many other uses, some of which we mention in this lecture, and others we may see in later lectures.

Definition 1 ((k, c)-expanding)**.** *A graph $G = (V, E)$ is* (k, c)-expanding *if $S \subset V$ with $|S| \leq k$ implies that $|\Gamma(S)| \geq c|S|$, where $\Gamma(S) = \{v \in V | \exists s \in S, (s, v) \in E\}$ is the* neighborhood *of S in G.*

Notice that any graph is trivially (k, c)-expanding for all k for all $c \leq 1$.

Definition 2 (Expander family)**.** *An* expander family *is an infinite sequence of graphs $G_1, G_2, \ldots$ and fixed numbers $c > 1$ and d such that each G_n has degree $\leq d$ and is $(\frac{N}{2}, c)$-expanding, where $N = |V_n|$ is the number of vertices in G_n.*

Intuitively, expander graphs are "very connected" in that the number of vertices reachable from a given subset of vertices is proportional to the size of that subset, at least when the subsets aren't so large as to make this impossible.

3.1 Graph Theoretic Properties

From now on we'll assume all our graphs are d-regular. We describe a d-regular graph using a normalized adjacency matrix. This view of a expander graphs proves useful in the analysis of their properties and randomized algorithms that make use of them.

Definition 3 (Normalized adjacency matrix)**.** *Given a d-regular $G = (V, E)$ its* normalized adjacency matrix *A is defined by*

$$A_{ij} = \begin{cases} \frac{1}{d}, & (i, j) \in E \\ 0, & \textit{otherwise} \end{cases}$$

The normalized adjacency matrix describes the Markov chain of a random walk on G:

$$A_{ij} = \Pr[\text{go to state } i \text{ from state } j],$$

and if p is a column vector with p_i the probability of being at vertex i, then $(Ap)_i$ gives the probability of being at vertex i after one random step in G.

The normalized adjacency matrix has a number of properties that can be proved using basic linear algebra. We will use the following properties, but we do not prove them here.

Proposition 1. *A is real-symmetric and so: all eigenvalues of A are real, and A has a basis of orthogonal eigenvectors.*

It turns out that the eigenvalues of the normalized adjacency matrix of an expander are closely connected to the expanders graph properties. The following are some basic properties, whose proofs we omit.

Proposition 2. *Let A be a normalized adjacency matrix of a graph on N vertices.*

1. *Each eigenvalue λ of A satisfies $|\lambda| \leq 1$.*
2. *1 is an eigenvalue of A, with corresponding eigenvector given by the uniform distribution on $(1/N, 1/N, ..., 1/N)$.*
3. *The multiplicity of the eigenvalue 1 is greater than 1 iff G is disconnected.*
4. *A has -1 for an eigenvalue iff G is bipartite.*

As the uniform distribution is always an eigenvector, we look only at the remaining eigenvector/values. It turns out that the second largest eigenvalue in absolute value is often useful to work with.

Definition 4 ($\lambda(G)$)**.** *Let $\Lambda = \{\lambda | \exists e_\lambda [Ae_\lambda = \lambda e_\lambda \wedge \langle e_\lambda, u\rangle = 0]\}$. Then*

$$\lambda(G) = \max_{\lambda \in \Lambda} |\lambda|$$

is the largest eigenvalue corresponding to an eigenvector orthogonal to u.

So, if G is connected and not bipartite then $\lambda(G) < 1$. The next two theorems will not be proven here. See the notes for CS 880 Spring 2006 for proofs.

Theorem 2. *If G is $(\frac{N}{2}, c)$-expanding then $\lambda(G) < f(c,d)$ where $f(c,d) < 1$ if $c > 1$ and is a function whose value depends only on c and d.*

Corollary 1. *Expanders have positive spectral gaps and expander families have spectral gaps bounded away from 0, where the spectral gap of a graph G is $1 - \lambda(G)$.*

Theorem 3. *If $\lambda(G) \leq \lambda < 1$ then G is $(\frac{N}{2}, g(\lambda))$-expanding where $g(\lambda) > 1$ when $\lambda < 1$ and is a function whose value depends only on λ.*

In light of Theorems 2 and 3 we see that we could just as well have defined expander families in Definition 2 as d-regular families with positive spectral gaps.

So, for expanders, the uniform distribution u is the only fixed point. The following Proposition tells us that the larger the spectral gap the faster an arbitrary probability distribution converges to the uniform distribution via a random walk on G.

Lemma 1. *For any probability distribution vector p*

$$\|A^t p - u\|_1 \leq \sqrt{N}\lambda^t,$$

where $\|v\|_q = \left[\sum_i |v_i|^q\right]^{1/q}$ is the q-norm of v and $\lambda = \lambda(G)$.

Proof. Since $Au = u$ we have $A^t p - u = A^t(p-u)$. Now, $(p-u)\perp u$:

$$\langle p-u, u\rangle = \langle p,u\rangle - \langle u,u\rangle = \sum_{i=1}^{N} p_i/N - \sum_{i=1}^{N} 1/N^2 = 1/N - 1/N,$$

since p is a probability distribution.

Write $p - u = \sum_i a_i e_i$, where the $\{e_i\}$ form an orthogonal eigen basis for A and $Ae_i = \lambda_i e_i$. Then, since the u-component of $p-u$ is 0, we have

$$\|A^t(p-u)\|_2 = \|A^t \sum_i a_i e_i\|_2 = \|\sum_i \lambda_i^t a_i e_i\|_2 \le \lambda^t \|\sum_i a_i e_i\|_2 = \lambda^t \|p-u\|_2$$

Now

$$\|p-u\|_2^2 + \|u\|_2^2 = \|p\|_2^2 \le \|p\|_1^2 = 1,$$

since $(p-u)\perp u$, and $\|v\|_2^2 \le \|v\|_1^2$ for all vectors v. By the Cauchy-Schwartz inequality, we have

$$\begin{aligned}\|A^t(p-u)\|_1 &= |\langle(-1^{\sigma_1}, \ldots, -1^{\sigma_N}), A^t(p-u)\rangle| \\ &\le \|(-1^{\sigma_1}, \ldots, -1^{\sigma_N})\|_2 \|A^t(p-u)\|_2 \\ &= \sqrt{N}\|A^t(p-u)\|_2,\end{aligned}$$

where $\sigma_k = \begin{cases} 0, & (A^t(p-u))_k \ge 0 \\ 1, & \text{otherwise} \end{cases}$. Combining all of the above we get

$$\begin{aligned}\|A^t(p-u)\|_1 &\le \sqrt{N}\|A^t(p-u)\|_2 \\ &\le \sqrt{N}\lambda^t\|(p-u)\|_2 \\ &\le \sqrt{N}\lambda^t,\end{aligned}$$

completing the proof. □

This lemma can be used to prove that the random walk algorithm for the undirected path problem needs only polynomially many steps, i.e. that PATH $\in$ BPL. The proof uses Lemma 1 and the following exercise.

Exercise 2. *If G is connected and not bipartite then $\lambda(G) \le 1 - \frac{1}{dN^2}$.*

3.2 Constructions

To prove that expanders exist one can argue that a randomly chosen d-regular graph G has a high probability of being an expander. However, we want to use expanders to reduce the amount of randomness needed in random algorithms, so using randomness to construct expanders won't help us. We want explicit constructions for which given a vertex v and index i we can compute v's ith neighbor in time poly$(|v|, |i|)$.

Example: For any integer $m \ge 2$, we can get an expander G on vertices $V = \mathbb{Z}_m \times \mathbb{Z}_m$ with edges given by the relations

$$\Gamma(\{(x,y)\}) = \{(x, y \pm x), (x, y \pm (x+1)), (x \pm y, y), (x \pm (y+1), y)\}.$$

If $m \geq 4$ then the graph has degree 8. The proof that this construction works is non-trivial. See the notes for CS 880 for a partial proof using harmonic analysis. ⊠

There are other efficient constructions of expanders, but the aforementioned expander suffices for our needs. See the notes for CS880 Spring 2006 for other constructions.

3.3 Applications

We'll use expanders for two purposes:

1. *Deterministic amplification*: Given a randomized algorithm R that uses r random bits we reduce the error to be less than some arbitrary ε, without using any additional random bits. This requires running R poly$(1/\varepsilon)$ times.

2. *Randomness efficient amplification*: With R and r as above we reduce the error to be less than some arbitrary ε using $r + O(\log(1/\varepsilon))$ additional random bits and running R $O(\log(1/\varepsilon))$ times.

To accomplish the above amplifications we use an expander graph G whose vertices are in one-to-one correspondence with the bit strings in $\{0,1\}^r$. For application 1 we run R, look at the vertex v in G corresponding to the random bits used by R's run, and then run R once for each neighbor v' of v, with random bits corresponding to v'. For application 2 we use the same G and v. Then, starting from v, we perform a random walk in G starting from v of length $t = \log(1/\varepsilon)$. This gives t vertices $v_1, \ldots, v_t$. We then run R once for each vertex v_i with random bits corresponding to v_i. In both cases, we finish by taking the "majority vote" of R's runs. We discuss these applications and prove their correctness in the next lecture.

4 Next Time

In the next lecture we prove the correctness of the algorithms given in the applications above. To accomplish this, we also prove

Lemma 2 (Expander mixing lemma)**.** *Given a d-regular expander graph $G = (V, E)$ and $S, T \subseteq V$*

$$\left| \frac{|E(S,T)|}{dN} - \mu(S)\mu(T) \right| \leq \lambda(G)\sqrt{\mu(S)(1-\mu(S))}\sqrt{\mu(T)(1-\mu(T))}$$

where $E(S,T)$ is the set of edges connecting S to T.

Notice that dN is the number of ways to choose an edge in G: N choices of starting vertex and then degree d choices for ending vertex. Also, $\mu(S)\mu(T)$ is the probability that, given two randomly chosen vertices, one lies in S and the other in T.