Complexity Theory II Notes

Fall 2022

1 Lecture 1

Scribe: Naman, BMC202026

1.1 Introduction

Grading Scheme:

1. Scribe Notes
2. Assignments
3. Take-Home Exam/Project

Books:

- *Computational Complexity: A Modern Approach*, Boaz Barak and Sanjeev Arora
- *Pseudorandomness*, Salil Vadhan

1.2 Randomized Algorithms

Over the past several years, computer scientists have determined a variety of randomized algorithms to efficiently solve difficult problems with low time/space complexities. This leads to a number of questions:

- Can every Randomized Algorithm be derandomized?
- Can every algorithm that uses randomness be substituted with a similarly efficient algorithm that does not?

A major conjecture in computer science is that, *yes*, all random algorithms can be derandomized.

1.3 Polynomial Identity Testing

Consider the following problem: given two multivariate polynomials f and g in $\mathbb{F}[x_1, x_2, \ldots, x_n]$, is it the case that $f \equiv g$?

This problem, when posed with an *efficient* representation of polynomials, is known as **Polynomial Identity Testing**.

Note 1. It is clear that representing the polynomial as a set of monomials is inefficient, as the number of possible monomials for n-variate polynomials with degree d is $\binom{n+d}{d}$ and grows with n^d.

1.3.1 Arithmetic Circuit Representation

We can represent polynomials as arithmetic circuits, which is a circuit that has $+$ and $\times$ gates. We can at this point refine our original question and ask the following problem: Consider a polynomial $h(x_1, x_2, \ldots, x_n)$ of degree $\leq d$ by an arithmetic circuit. Can we decide if $h \equiv 0$?

This problem is considered to be the most important problem in the study of psuedorandomness, and is known to be solvable in randomized polynomial time.

1.3.2 Schwartz-Zippel Lemma

Theorem 1. Let $f(x_1, x_2, \ldots, x_n)$ be an n-variate polynomial of degree $\leq d$ and $S \subseteq \mathbb{F}$. Then $\Pr[f(a_1, \ldots, a_n) = 0] \leq d/|S|$ if the a_i have been uniformly sampled from S.

Proof. We proceed by induction. Consider the case for $n = 1$. Since the number of roots that a $1-$variate polynomial of degree d can have is at most d, it follows immediately that the probabilty is $\leq d/|S|$.

Now consider the n variate case. Any polynomial in this regard can be written as a 1-variate polynomial in the x_n variable:

$$\sum_k f_k(x_1, \ldots, x_{n-1}) x_n^k$$

Let the highest nonzero degree of x_n be j. Then the degree of f_j is $\leq d - j$. We know from the induction hypothesis that the probability of f_j being zero when supplied with $a_1, \ldots, a_{n-1}$, is $d - j/|S|$. The probability of x_j itself being zero is $j/|S|$ by the base case. So the probability that the entire term is 0 is

$$\Pr[f_j(a_1, \ldots, a_{n-1})a_n = 0] \leq \frac{d-j}{|S|} + \frac{j}{|S|} = \frac{d}{|S|}$$

and we are done.

So for solving this problem in a randomized manner, simply choose a big-enough subset from which selection is being made and then repeatedly evaluate the polynomial until we can be fairly confident that it is nonzero.

2 Lecture 2

(Scribe: Naman, BMC202026)

2.1 UREACH $\in$RL

The UREACH problem asks to find whether given an undirected graph G and two vertices s, t, the two vertices are in the same connected component.

2.1.1 Randomized Algorithm for UREACH

Idea Make a sufficiently long random walk from s and if we don't reach t, reject (G, s, t).

We can assume WLOG that G is $d-$regular, since we can always simply add self-loops in order to maintain regularity.

Adjacency Matrix Consider the normalized adjacency matrix of the graph, which is defined as $A_G(i, j) = 1/d$ if $(i, j) \in E$, and 0 otherwise.

- A_G is a real, symmetric matrix.
- A_G is *doubly-stochastic*, ie. rows and columns add up to 1 (the rows and columns look like probability distributions).
- Eigenvalues of A_G are reals.
- If the eigenvectors of A_G are normalized, $||v_i||_2 = \langle v_i, v_i \rangle = 1$, they form an orthonormal basis for $\mathbb{R}^n$.

Procedure Consider the probability distribution of the current node as a vector of n values, which represents the probability of being on each node as the nth coordinate.

The vector at the beginning will be $w = (1, 0, \ldots, 0)$ assuming that s is taken to be the first vector. After a single step, the vector will become $A_G w$, and in particular for every neighbor of s the probability will become $1/d$.

Observations:

1. U is an eigenvector for A_G of eigenvalue 1.
2. We arrange the eigenvectors in order of decreasing magnitude of eigenvalues. The uniform vector, $u = v_1$, is the largest since the eigenvalue is 1. We label the eigenvalues as $v_1, v_2, \ldots, v_n$.

Definition 1. Spectral Gap. The spectral gap, $\gamma(G) = 1 - \lambda_2(A_G)$, is the difference between the biggest and second biggest eigenvalue.

We now write $w = u + \sum_{j=2}^{n} \mu_i v_i$, where w is the distribution vector. It follows that

$$A_G w = u + \sum_{j=2}^{n} \mu_j A_G v_j = u + \sum_{j=2}^{n} \mu_j \lambda_j v_j$$

Taking the norm, it follows that

$$||A_G w - u|\,|_2^2 = ||\sum_{j=2}^{n} \mu_j \lambda_j v_j||_2^2 \leq |\lambda_2|^2 ||w||_2^2$$

This shows that after applying A_G, we get closer and closer to the normal vector u.

If we make t rounds of random walks, then the probability vector $A_G^t w$ will be such that

$$||A_G^t w - u||_2 \leq |\lambda_2|^t$$

Since $\lambda_2 < 1$, we are getting closer and closer to the uniform distribution $(1/n, 1/n, \ldots, 1/n)$.

Definition 2. Alon-Bopanna Bound. If we have a d-regular, non-bipartite, connected graph, then

$$\gamma(G) \geq \Omega\left(\frac{1}{d^2 n^2}\right)$$

Using this, we get that

$$|\lambda_2|^t \leq e^{-\Omega\left(\frac{1}{d^2 n^2}\right)t} \leq \frac{1}{n^{O(1)}}$$

It follows that we can get the result if we let t be such that

$$n^{O(1)} \leq e^{\frac{t}{d^2 n^2}} \implies t \geq O(d^2 n^2 \log n). \square$$

2.2 How is UREACH derandomized?

Reference. Expander Graphs and their Applications, Hoory, Linial & Wignerson.

2.2.1 Expander Graphs

Definition 3. Vertex Expander. A graph family (over n vertices) is a (K, A)-vertex expander if for any $S \subset V$ such that $|S| \leq K$, $|N(S)| \geq A|S|$. Here,

$$N(S) = \{u \mid \exists v \in S \text{ such that } (u, v) \in E\}$$

Definition 4. Spectral Expander. A family of graphs $\{G_n\}$ is called a spectral expander if $|\lambda_2(G_n)| \leq \epsilon$.

These two notions are equal, proved by Jeff Cheeger.

3 Lecture 3

(Scribe: Diptaksho, BMC202015)

Lemma 1. $\lambda_2 = \max_{x \perp u} \frac{\|Ax\|}{\|x\|}$. Here λ_2 is the second largest (by magnitude) eigenvalue, u is the uniform distribution vector $\begin{bmatrix} \frac{1}{n} & \dots & \frac{1}{n} \end{bmatrix}$, A is the normalised adjacency matrix, and unless mentioned otherwise, the norm is $L2$.

Consider the eigenbasis for $A = \{u, u_2, \cdots, u_N\}$. Given any $x \perp u$, write it as a linear combination $\sum_{j=2}^{N} \beta_j u_j$. Note that we start from $j = 2$ as $x \perp u$. Now,

$$\begin{aligned} Ax &= \sum_{j=2}^{N} \lambda_j \beta_j u_j \\ \Rightarrow \|Ax\|^2 &= \sum_{j=2}^{N} \lambda_j^2 \cdot \beta_j^2 \\ &\leq \lambda_2^2 \sum_{j=2}^{N} \beta_j^2 = \lambda_2^2 \|x\|^2 \\ \Rightarrow \lambda_2 &\geq \frac{\|Ax\|}{\|x\|} \end{aligned}$$

Taking $x = u_2$, we get the equality. □

Theorem 2. Spectral Expansion ≡ Vertex Expansion [Cheeger]

3.1 Few Definitions and Observations

Let Π be a probability distribution over $\{1, \cdots, N\}$. Define $Coll(\Pi)$ (collision probability) as $Coll(\Pi) = \|\Pi\|_2{}^2$. Notation: Π is the vector whose i^{th} component is the probability distribution of Π at i, i.e. $\Pi(i)$.

Observation Write $\Pi = \alpha \cdot u + \Pi^\perp$. Now, $\langle \Pi, u \rangle = \alpha \langle u, u \rangle \Rightarrow \frac{1}{N} = \alpha \cdot \frac{1}{N} \Rightarrow \alpha = 1$. The first simplification of the LHS comes from the fact that Π is a probability distribution, hence the terms add up to 1. Then $\Pi = u + \Pi^\perp \Rightarrow \|\Pi\|^2 = \|u\|^2 + \|\Pi^\perp\|^2 = \frac{1}{N} + \|\Pi - u\|^2 \Rightarrow Coll(\Pi) = \frac{1}{N} + \|\Pi - u\|^2$.

Cauchy-Schwarz Inequality Let a, b be two vectors. Then $\langle a, b \rangle \leq \|a\| \cdot \|b\|$.

Define the support of a probability distribution Π as $supp(\Pi) = \{i \in \{1, \cdots N\} \mid \Pi(i) > 0\}$.

Observation $Coll(\Pi) \geq \dfrac{1}{|supp(\Pi)|}$. Define the vector $1_{supp(\Pi)}$ as:

$$1_{supp(\Pi)}[i] = \begin{cases} 1, & i \in supp(\Pi) \\ 0, & \text{otherwise} \end{cases}$$

Note that $\langle \Pi, 1_{supp(\Pi)} \rangle = 1$, and $\|1_{supp(\Pi)}\|^2 = |supp(\Pi)|$. By C-S Inequality, $1 \leq \|\Pi\| \cdot \|1_{supp(\Pi)}\| \Rightarrow \|\Pi\| \geq \dfrac{1}{\|1_{supp(\Pi)}\|} \Rightarrow Coll(\Pi) \geq \dfrac{1}{|supp(\Pi)|}$. □

3.2 Proof of Spectral Expansion ⇒ Vertex Expansion

Given $S \subset V$, a subset of vertices, let π be the uniform distribution over S. The size of S is αN. Note that $A\pi$ is also a probability distribution. Now, $Coll(\pi) = \frac{1}{N} + \|\pi - u\|^2$.

$$\begin{aligned} Coll(A\pi) &= \frac{1}{N} + \|A\pi - u\|^2 \\ &= \frac{1}{N} + \|A(\pi - u)\|^2 \\ &\leq \frac{1}{N} + \lambda_2^2 \|\pi - u\|^2 \end{aligned}$$

$$\begin{aligned} &\Rightarrow \quad \frac{1}{N} + \lambda_2^2 \|\pi - u\|^2 \geq \frac{1}{|supp(\pi)|} = \frac{1}{|N(S)|} \\ &\Rightarrow \quad \frac{1}{N} + \lambda_2^2 \left(Coll(\pi) - \frac{1}{N} \right) \geq \frac{1}{|N(S)|} \\ &\Rightarrow \quad \lambda_2^2 \left(\frac{1}{|S|} - \frac{1}{N} \right) \geq \frac{1}{|N(S)|} - \frac{1}{N} \\ &\Rightarrow \quad \lambda_2^2 \frac{N - |S|}{N \cdot |S|} \geq \frac{N - |N(S)|}{N \cdot |N(S)|} \\ &\Rightarrow \quad |N(S)|(|S| + \lambda_2^2 N - \lambda_2^2 |S|) \geq N \cdot |S| \\ &\Rightarrow \quad \frac{|N(S)|}{|S|} \geq \frac{N}{N(\alpha + \lambda_2^2 - \alpha \cdot \lambda_2^2)} = \frac{1}{\alpha + \lambda_2^2 - \alpha \cdot \lambda_2^2} \end{aligned}$$

Thus we get a $\frac{1}{\alpha + \lambda_2^2 - \alpha \cdot \lambda_2^2}$-expander graph from the spectral expansion graph. □

3.3 Expander Mixing Lemma

Lemma 2. Expander Mixing Lemma Suppose we have two subsets $S, T \subset V$. Denote the number of edges flowing from S to T as $\#E(S,T)$. Then we have the following inequality:

$$\left| \#E(S,T) - \frac{d|S||T|}{N} \right| \leq d\lambda_2 \sqrt{|S||T|}$$

Note 2. Good expanders have high spectral expansion ($\gamma = 1 - \lambda_2$), implying λ_2 must be small. The best known bound for λ_2 in D-regular graphs is $\lambda_2 \approx \frac{1}{\sqrt{D}}$. Such graphs are called Ramanujan graphs.

Exercise Prove that $\dfrac{d|S||T|}{N}$ is a good approximation for $\#E(S,T)$ in random graphs.

3.3.1 Proof of Expander Mixing Lemma

For any subset $U \subset V$, define the characteristic vector 1_U as:

$$1_U[i] = \begin{cases} 1, & i \in U \\ 0, & \text{otherwise} \end{cases}$$

Define 1_S and 1_T in this way.

Observation $\#E(S,T) = \langle 1_S, A \times 1_T \rangle \times d$ [Multiplication by d because A is normalised].

Write $1_S = \alpha \cdot u + 1_S^\perp$. Now, $\langle 1_S, u \rangle = \alpha \langle u, u \rangle \Rightarrow \frac{|S|}{N} = \alpha \cdot \frac{1}{N} \Rightarrow \alpha = |S|$. Similarly, writing $1_T = \beta \cdot u + 1_T^\perp$, we get $\beta = |T|$.

$$\begin{aligned}
\#E(S,T) &= d \times \langle 1_S, A \times 1_T \rangle \\
&= d \times \langle \alpha \cdot u + 1_S^\perp, A \times (\beta \cdot u + 1_T^\perp) \rangle \\
&= d \times \langle \alpha \cdot u + 1_S^\perp, \beta \cdot u + A \times 1_T^\perp \rangle \\
&= d \times \left(\frac{\alpha \cdot \beta}{N} + \langle 1_S^\perp, A \times 1_T^\perp \rangle \right) \\
&= \frac{d|S||T|}{N} + d \times \langle 1_S^\perp, A \times 1_T^\perp \rangle
\end{aligned}$$

$$\begin{aligned}
\therefore \quad \left| \#E(S,T) - \frac{d|S||T|}{N} \right| &= d |\langle 1_S^\perp, A \times 1_T^\perp \rangle| \\
&\leq d \cdot \|1_S^\perp\| \cdot \|A \times 1_T^\perp\| \quad \text{[C-S Inequality]} \\
&\leq d \cdot \sqrt{|S|} \cdot \lambda \cdot \sqrt{|T|} \\
&= d\lambda_2 \sqrt{|S||T|}
\end{aligned}$$

This is the desired result. □

Random Walk on Expanders We have seen that $\|A^t w - u\| \leq \lambda_2{}^t$. Write $\lambda_2 = 2^{-\beta}$. Taking $t = O(\log N)$, the RHS becomes $\frac{1}{N^{O(1)}}$. A difference of less that $\frac{1}{N}$ implies the component of each vertex in $A^t w$ is non-zero.

Corollary The diameter of a Spectral Graph is $O(\log N)$.

3.4 Next Class

Suppose $\mathcal{A}$ is an RP algorithm with error probability $\leq \frac{1}{3}$, and $\mathcal{A}$ uses $m(n)$ random bits for inputs of size n. If we want to reduce the error probability to say $\frac{1}{3^k}$, the algorithm would require $m \times k$ random bits. But using expanders, we can bring the number of random bits required to $O(m + k)$.

4 Lecture 4: Error Reduction in RP using Expander

(Scribe: Somnath, MCS202221)

Error reduction in **RP** and **BPP** is one of the applications of Expander graphs in the area of probabilistic algorithm. In the standard way of error reduction, we perform the the random event k times independently and output the majority value of the algorithm. Now it surely reduces the error $\frac{1}{3}$ to $2^{-\Omega(k)}$ (to be specific the error will be $2^{-|x|^d}$ where $k = 8|x|^{2d+1}$, for the detailed proof we will refer to the section 7.4.1 of the book by Arora and Barak). But it uses too many random bits, if each event takes m many random bits and if we are repeating it k many times then the amount of random bits will be mk.

The main problem is we are executing the events independently, but in expander graph uses of random coins are correlated, hence it reduces the number of random bits, we will see it reduces to $m + O(k)$

Central Idea:

We will deal with **RP** algorithm in this class.. The key idea is simple, if each event takes m random bits, then we will take $N = 2^m$ vertex, λ-expander, D-regular graph G. We will further more assume that G is ***strongly explicit***, i.e., for any vertex v in G we can find the neighbours of v efficiently. Note each vertex of G represents one random string. Here also we will perform the event k many times, but the random strings say $r_1, \ldots, r_k$ will not be independent. We will pick r_1 uniformly at random and then start a k-length random walk on G, at each point of the walk we will take that corresponding random string for the algorithm. If any of this r_i outputs 1 then we will output 1, else output 0.

So apart from the constructing such expander graph G, we are left with two questions. First what is the number of random bits used here? It is easy to see that only r_1 is picked uniformly at random, and the remaining r_is are random neighbour of r_{i+1}, since the graph is D-regular, the number of random bits used is $m + (k-1)\log D$. So clearly number of random bits is reduced.

The next question is the efficiency compare to the standard method ,ie, what is the amount of the error? Actually **theorem** 1 will imply that *expander walks are ergodic*, that means our current covered vertex set will expand with high probability during random expander walk, so we eventually end up on the vertices which are responsible for correct answers. We will proof this in details in the next section.

Correctness:

Before starting the error calculation we need to define ***Operator norm.*** For a linear operator $A : \mathbb{R}^n \to \mathbb{R}^n$, the operator norm of A, denoted by $||A||$, is defined by $\max\limits_{x \in \mathbb{R}^n} \dfrac{||Ax||_2}{||x||_2}$ where $||.||_k$ is the l_k norm.

From the definition it is clear that $||Av||_2 \leq ||A||.||v||_2$. Also in the previous classes we have proved that for any doubly stochastic non negative matrix A, $||A|| = \lambda_2$, the largest eigen value after 1.

Define $J := \begin{bmatrix} \frac{1}{N} & \cdots & \frac{1}{N} \\ \vdots & \ddots & \vdots \\ \frac{1}{N} & \cdots & \frac{1}{N} \end{bmatrix}$ which is the normalized adjacency matrix of a complete graph. Then next lemma will indicate Expander graph G approximates complete graph.

Lemma 1. Let G be the N-vertex λ-expander D-regular graph G with normalized adjacency matrix A then $\exists\ E \in \mathbb{R}^{N\times N}$ with $||E|| \leq 1$ s.t. $A = (1-\lambda)J + \lambda E$

Proof : We will basically do the reverse calculation. Define $E = \frac{1}{\lambda}(A - (1-\lambda)J)$ Now It is easy to observe sum of the rows of E= $\frac{1}{\lambda}$(sum of the rows of $A - (1-\lambda)$sum of the rows of J)=1

Similarly sum of the columns of E is also 1. Hence it is doubly stochastic. So $||E|| =$ second largest eigen value ≤ 1 (Since 1 is the largest eigen value of any doubly stochastic non-negative matrix). □

Let $B \subseteq [N]$ be the set of all random strings which leads us to an error. Clearly for error μ

$$|B| \leq \mu N$$

The next theorem will imply that if we start the walk from some r_1 and ends at r_k, then with high probability depending on k, r_k is outside of B, that is our walk will expand from the set B in expander graph.

Theorem 1. Let $r_1, \ldots, r_k$ denoting a k-step random walk on G from r_1 where r_1 is chosen uniformly in $[N]$ then

$$\Pr[\forall i \quad r_i \in B] \leq \mu((1-\lambda)\mu + \lambda)^{k-1}$$

Let P be the projection of B in to $[N]$,i.e., P be the $N \times N$ matrix s.t.

$$P[i,j] = \begin{cases} 1 & \text{if } i = j \in B \\ 0 & \text{otherwise} \end{cases}$$

Note P is a diagonal matrix with diagonal entries $0/1$, hence $P^2 = P$ (Alternative proof is any projection matrix is Idempotent because for P^2u, Pu is already in the projective set, so $P^2u = P(Pu) = Pu$, hence P^2 and P is the same operator).

One more observation is $\Pr[r_1 \in B] = ||Pu||_1$ where $u = (\frac{1}{N}, \ldots, \frac{1}{N})$. More generally we can say probability that after t step we are still in B, ie,

$$\begin{aligned} \Pr[\forall_{i\leq t} r_i \in B] &= ||(PA)^{t-1}Pu||_1 \\ &= ||(PAP)^{t-1}Pu||_1 \\ &= \sqrt{\mu N}||(PAP)^{t-1}Pu||_2 && \text{(as support of } Pv \leq |B| = \mu N) \\ &\leq \sqrt{\mu N}||PAP||^{t-1}||Pu||_2 \\ &= \mu||PAP||^{t-1} && \text{(as } ||Pu||_2 = \sqrt{\tfrac{\mu}{N}}) \end{aligned}$$

..............(i)

From **lemma** 1

$$\begin{aligned}||PAP|| &= ||P[(1-\lambda)J+\lambda E]P|| \\ &\leq (1-\lambda)||PJP|| + \lambda||PEP||\end{aligned}$$

...............(ii)

It can be easily observed that $||P|| \leq 1$ (because support of Pv is atmost support of v, it vanishes all the components of v which are not in B), hence $||PEPv||_2 \leq ||P||.||E||.||P||.||v||_2$, or, $||PEP|| \leq 1$ ....(iii)

Now say $Px = y$, then $Jy = \left(\frac{1}{N}\sum y_i, \ldots, \frac{1}{N}\sum y_i\right) = (\sum y_i)u$
So

$$\begin{aligned}||PJPx||_2 &= ||PJy||_2 \\ &= ||(\sum y_i)Pu||_2 \\ &\leq |\sum y_i|.||Pu||_2 \\ &\leq ||y||_1.||Pu||_2 \\ &\leq \sqrt{\mu N}||y||_2.\sqrt{\frac{\mu}{N}} \\ &\leq \mu||Px||_2 \\ &\leq \mu||x||_2 \\ \implies ||PJP|| &\leq \mu\end{aligned}$$

Combining this with (i),(ii),(iii) we get

$$\Pr[\forall i \quad r_i \in B] \leq \mu((1-\lambda)\mu+\lambda)^{k-1}$$

□

Now clearly if input $x \notin L$ then $|B| = 0$, so our algorithm will always output 0, If $x \in L$, the error will occur if after k steps we are still in B, which has probability $\mu((1-\lambda)\mu+\lambda)^{k-1} = O(2^{\Omega(k)}$ for small λ. Hence the error reduction is same as the standard method, but randomness is reduced to $m + O(\log k)$ from mk.

5 Lecture 5

(Scribe: Diptaksho, BMC202015)

5.1 Error Reduction in BPP Algorithms

Chernoff Bound Suppose $X_1, X_2, \cdots, X_n$ are independent random variables distributed on $[0,1]$. Let $X = \frac{\sum_{i=1}^n X_i}{n}$ and $\mathbb{E}(X) = \mu$. Then $\Pr[|X-\mu| \geq \epsilon] \leq 2e^{-\Omega(\epsilon^2 n)}$.

Theorem 2. Let G be an (N, D, λ) graph. Let $f : [N] \to \{0,1\}$ be a random variable with expected value $\mu(f)$. Let $X_i = f(v_i)$, and $X = \sum_{i=1}^n X_i$. Then

$$\Pr_{X_1,\cdots,X_t}\left[\left|\frac{X}{t} - \mu(f)\right| \geq \lambda + \epsilon\right] \leq 2e^{-\Omega(\epsilon^2 t)}.$$

We will prove one direction of this theorem.

Lemma 2. $\Pr[X \geq t(\mu + \lambda + \epsilon)] \leq e^{-\Omega(\epsilon^2 t)}$.

Proof: Let r be a parameter we fix later. We want to study $\mathbb{E}[e^{rX}] = \mathbb{E}[\prod_{i=1}^t e^{rX_i}]$. Define the matrix

$$P = \begin{bmatrix} e^{rf(v_1)} & 0 & \cdots & 0 \\ 0 & e^{rf(v_2)} & \cdots & 0 \\ \vdots & 0 & \ddots & \vdots \\ 0 & \cdots & 0 & e^{rf(v_N)} \end{bmatrix}.$$

Then $|Pu|_1$ is the expectation for a length 1 walk. Similarly, $|(PM)^{t-1}Pu|_1$ is the expectation for a length t walk. As $Mu = u$, we have

$$\begin{aligned} |(PM)^{t-1}Pu|_1 &= |(PM)^t u|_1 \\ &\leq \sqrt{N} \times \|(PM)^t u\|_2 \quad \text{[by C-S Inequality]} \\ &\leq \sqrt{N} \times |(PM)^t\| \times \|u\|_2 \\ &= \|(PM)^t\| \\ &= \|PM\|^t \\ &= \|P((1-\lambda)J + \lambda E)\|^t \quad \text{[From Lecture 4]} \\ &\leq ((1-\lambda)\|PJ\| + \lambda\|PE\|)^t. \end{aligned}$$

Now, $\|P\| \leq \max e^{rf(v_i)} \leq e^r$. Then

$$\begin{aligned} \lambda\|PE\| &\leq \lambda\|P\| \\ &\leq \lambda e^r \\ &= \lambda(1 + r + O(r^2)). \end{aligned}$$

Next,

$$\begin{aligned}
\|PJy\|_2^2 &= \left\| \left(\sum y_i\right) Pu \right\|_2^2 \\
&= \left|\sum y_i\right| \|Pu\|_2^2 \\
&\leq N \times \|y\|_2^2 \times \|Pu\|_2^2 \quad \text{[C-S Inequality]} \\
&= N \times \|y\|_2^2 \times \frac{1}{N^2} \times \left(\sum e^{2rf(v_i)}\right) \\
&= \frac{1}{N}\left(\sum_{i=1}^{N} 1 + 2rf(v_i) + O(r^2)\right) \times \|y\|_2^2 \\
&= \left(1 + 2r\mu(f) + O(r^2)\right) \times \|y\|_2^2.
\end{aligned}$$

It follows that $\|PJ\|^2 \leq \left(1 + 2r\mu(f) + O(r^2)\right) \Rightarrow \|PJ\| \leq 1 + r\mu(f) + O(r^2)$. Combining all the inequalities, we get

$$\begin{aligned}
\|(PM)^t u\|_1 &\leq \left((1-\lambda)(1 + r\mu + O(r^2)) + \lambda(1 + r + O(r^2))\right)^t \\
&= \left(1 + O(r^2) + r(\lambda + \mu)\right)^t \\
&\leq 1 + rt(\lambda + \mu) + O(r^2).
\end{aligned}$$

This gives us

$$\begin{aligned}
\Pr\left[e^{rX} \geq e^{rt(\mu+\lambda+\epsilon)}\right] &\leq \frac{\mathbb{E}[e^{rX}]}{e^{rt(\mu+\lambda+\epsilon)}} \\
&\leq \frac{e^{1+rt(\lambda+\mu)+O(r^2)}}{e^{rt(\lambda+\mu+\epsilon)}} \\
&\leq e^{-\Omega(rt\epsilon)}.
\end{aligned}$$

Taking $r = \frac{\epsilon}{c}$, we get the desired result. □

5.2 Construction of Expander Graphs

Definition 5. Graph Product Let $G = (N, D, \lambda)$ be a graph. Let G^2 be the distance 2-graph for this. Then the degree of G^2 is D^2. The adjacency matrix for G^2 will be M^2. Then the second largest eigenvalue for M^2 is λ^2. Thus we get $G^2 = (N, D^2, \lambda^2)$.

Definition 6. Tensor Product of Graphs For two matrices A and B, their tensor product is defined as

$$A \otimes B = \begin{bmatrix} a_{1,1}B & \cdots & a_{1,n}B \\ \vdots & \ddots & \vdots \\ a_{n,1}B & \cdots & a_{n,n}B \end{bmatrix}.$$

Suppose we have two graphs $G_1 = (N_1, D_1, \lambda_1)$ and $G_2 = (N_2, D_2, \lambda_2)$. Then $G' = G_1 \otimes G_2$ is constructed by taking $V' = V_1 \times V_2$ and $M' = M_1 \otimes M_2$. Thus $((u_1, u_2), (v_1, v_2)) \in E' \iff (u_1, v_1) \in E_1$ and $(u_2, v_2) \in E_2$. The degree of G' is $D_1 \times D_2$. The eigenvectors for G' are tensor products of eigenvectors for G_1 and G_2. Thus $\lambda' = \max(\lambda_1, \lambda_2)$.

6 Lecture 6

Scribe: Rohan Goyal, BMC202151

6.1 Zig-zag product

Let $G = (N_1, D_1, \gamma_1)$ be the *outer graph* on $[N_1]$ and $H = (D_1, D_2, \gamma_2)$ be the *inner graph* on $[D_1]$.[1].

Now, to take the zig zag product, replace each vertex in G with "clouds" of D_1 vertices and think of every cloud as a copy of H. Thus, we replace G with a graph on the vertex set $[N_1] \times [D_1]$. We now only use this model of "copying H" for identification of vertices and haven't described the edge set yet. We call this vertex set V.

Now, consider any bijection $f : V \mapsto V$ between vertices of different clouds such that $f((u, i)) = (u', i') \implies (u, u') \in E(G)$ and we have maps from the cloud u to cloud u' with multiplicity of edge (u, u') in G only.

Now the edges of (u, i) are to the set $\{(u', j')|(i, i') \in E(H), f((u, i')) = (u', j), (j, j') \in E(H)\}$ and multiplicities are according to multiplicities of $E(H)$. So, we take a jump using H followed by a jump using our bijection and another jump using H.

Let $\hat{A}$ be the permutation matrix corresponding to the bijection and let A, B be the normalized adjacency matrices of G, H respectively.

Now, the above described graph is a $(N_1 D_1, D_2^2, \gamma)$ expander but we need to find γ. Also note that the degree of $G Ⓩ H$ is independent of G.

So, let M be the normalized adjacency matrix of $G Ⓩ H$. Now,

$$M = (I_{N_1} \otimes B)\hat{A}(I_{N_1} \otimes B)$$

Since H is a γ_2 expander, we have $B = \gamma_2 J + (1 - \gamma_2)E$ where J is the complete graph on D_1 vertices and E is the error graph and $||E|| \leq 1$.

Now, let x which is orthogonal to the uniform vector be any vector in $M_{N_1 D_1, 1}$. We now consider $\frac{||Mx||}{||x||}$ and observe that anything in the expansion of M involving E can be estimated to be having operator norm at most 1. Now,

$$\frac{||Mx||}{||x||} \leq \gamma_2^2 \frac{||(I_{N_1} \otimes J)\hat{A}(I_{N_1} \otimes J)}{||x||} + 1 - \gamma_2^2$$

Let $\hat{J} = (I_{N_1} \otimes J)$. Now, observe that $\hat{J}\hat{A}\hat{J} = A \otimes J$.

[1]γ_1 and γ_2 are respectively the spectral expansions of G and H i.e. $1 - \lambda_2^{(G)}, 1 - \lambda_2^{(H)}$.

Thus, $\lambda_2^{\hat{J}\hat{A}\hat{J}} = \lambda_2^{A \otimes J} = \lambda_2^G$. Thus, $\frac{||Mx||}{||x||} \leq (1-\gamma_2^2) + \gamma_2^2(1-\gamma_1) = 1 - \gamma_2^2\gamma_1$. Thus, we get a lower bound on the spectral expansion as well. $\gamma \geq \gamma_2^2\gamma_1$. Thus, $G \textcircled{z} H$ is atleast a $(N_1D_1, D_2^2, \gamma_2^2\gamma_1)$ expander.

We can now use this to create expanders by alternating between normal graph products and zig zag products.

7 Lecture 7

(Scribe: Tejas, GCS202102)

7.1 Recap

Let G (the outer graph) be (N_1, D_1, γ_1) on $[N_1]$ and H (the inner graph) be (D_1, D_2, γ_2) on $[D_1]$. Here $\gamma_1 = 1 - \lambda_2^{(1)}$ and $\gamma_2 = 1 - \lambda_2^{(2)}$ are the spectral expansions of the 2 graphs. $G \textcircled{z} H$, the zig-zag product of G and H is a multigraph on the vertex set

$$V = \{(u, i) : u \in [N_1], i \in [D_1]\}.$$

To define $E(G \textcircled{z} H)$, consider the auxiliary graph, A on V as follows: For each $u \in G$, $C_u = \{(u, i) : i \in [D_1]\}$ is thought of as a 'cloud' around $u \in G$. The second co-ordinates of vertices in a cloud have the same edge relations as H. So, the 'intra-cloud' edges are determined by $E(H)$. Define a permutation, Φ on V as follows. For each $u \in G$, fix some ordering on $N_G(u)$. If v is the i-element of $N_G(u)$ (in the ordering), then $\Phi((u, i))$ is set to be some element of C_v. As G is D_1 regular and as each cloud has D_1 many elements, Φ can be made into a permutation. For all $x \in V$, put $x - \Phi(x)$ in $E(A)$. So, $E(G)$ and Φ determine the 'inter-cloud' edges in A.

Add an edge $((u, i), (v, j))$ to $E(G \textcircled{z} H) \iff (v, j)$ is reachable in A from (u, i) using the following transitions in order

1. Go from (u, i) to some (u, s) if $(i, s) \in E(H)$ (this uses the intra-cloud edges of C_u).
2. Go from (u, s) to (v, l) for some l (this uses the inter-cloud edges between C_u and C_v).
3. Go from (v, l) to (v, j) if $(l, j) \in E(H)$ (this uses the intra-cloud edges of C_v).

Fix some (u, i). As H is D_2 regular, it can go to D_2 many points in C_u in step (1). Now, it can transition to a unique point via Φ in (2). In (3), it can again go to D_2 many points (similarly to (1)). So, $G \textcircled{z} H$ is D_2^2-regular multigraph. We showed that for any G and any H, a $(D^4, D, 7/8)$ graph, the family $(G_k)_k$, defined inductively by: $G_k = G_{k-1}^2 \textcircled{z} H$ has spectral gap atleast $3/4$.

7.2 Reingold's Theorem (2004)

We show that USTCONN is in Log-space. Two key observations are:

1. USTCONN for expanders is in L for expanders: We showed before that the diameter of an expander is $O(log(N))$. So, to check s-t connectivity, we start from s and take $O(log(N))$ steps and output 'yes' iff we reach t. This takes $O(log(N))$ space as the indices of the vertices in the path need to be stored (to ensure that the $s - t$ path considered is acyclic)
2. Every graph can be 'expanderized' while maintaining reachability (this will be made precise in what follows)

Given G, s, t, the algorithm is as follows: Add self loops to G to make it regular and to break bipartiteness. Fix H, a $(D^4, D, 3/4)$ graph. Let $G_0 := G$ and recursively define

$$G_k = G_{k-1}^2 \textcircled{z} H.$$

Check $USTCONN$ in $G_{O(log(N))}$.

7.2.1 Analysis

Let $\gamma()$ denote the expansion and $\lambda()$, the second largest eigenvalue. As $\lambda(S^2) = \lambda^2(S)$ for any S,

$$\gamma(G_{k-1}^2) = 1 - \lambda(G_{k-1}^2) = 1 - \lambda^2(G_{k-1}) = 1 - (1 - \gamma(G_{k-1}))^2$$
$$= 2\gamma(G_{k-1})(1 - \gamma(G_{k-1})/2)$$

This and properties of ⓩ give,

$$\gamma(G_k) = \gamma(G_{k-1}^2 ⓩ H) \geq \gamma(G_{k-1}^2)\gamma^2(H) = \gamma^2(H)2\gamma(G_{k-1})(1 - \gamma(G_{k-1})/2)$$
$$= (9/8)\gamma(G_{k-1})(1 - \gamma(G_{k-1})/2)$$

Case 1: $\gamma(G_{k-1}) < 1/18$. Then,

$$(9/8)\gamma(G_{k-1})(1 - \gamma(G_{k-1})/2) \geq (9/8)\gamma(G_{k-1})(35/36) = \gamma(G_{k-1})(35/32).$$

Case 2: $\gamma(G_{k-1}) \geq 1/18$. Note that $\gamma(G_k) \geq \gamma(G_{k-1})$ as the expansion cannot decrease by powering a graph and by taking its ⓩ with another graph. So, $\gamma(G_k) \geq 1/18$.

So, by both cases, $\gamma(G_k) \geq \min\{\gamma(G_{k-1})(35/32), 1/18\}$

So, in one step, the expansion either has gone over $1/18$ or increases by a factor of $35/32 > 1$. So, for $k = O(log(n))$, $\gamma(G_k) > 1/18$.

So, given any G with N vertices, we get a $G_{O(log(N))}$ with (1) $D^{O(log(N))}$ nodes and (2) with atleast $1/18$ expansion. We showed before that a spectral graph has diameter $O(log(N))$, where the constant in the O(). depends on the spectral expansion. As, $G_{O(log(N))}$ has an expansion of atleast $1/18$, it has a diameter of $log(D^{O(log(N))}) = Clog(n))$, where C depends on $1/18$ and is hence independent of N. To summarize, we took *any* G and expanderized it into a graph ($G_{O(log(n))}$) with the same reachability properties as G, with diameter $O(log(|G|))$ and hence on which, USTCONN can be solved in $O(log(|G|))$ space. I.e., USTCONN on G can be solved in $O(log(|G|))$ space. This shows the correctness.

8 Lecture 8

(Scribe: Tejas, GCS202102)

8.1 Coding theory

We will not cover classical coding theory (for example block codes) but will rather focus on those aspects of coding theory relevant to complexity theory. An excellent reference is: Essential Coding Theory by Venkatesan Guruswami, Atri Rudra and Madhu Sudan.

8.2 The Basic set-up

Roughly, we want to send a message m across a potentially noisy channel. To do this, we encode m as $enc(m)$ and send $enc(m)$ through the channel. Due to noise, a corrupted version of $enc(m)$ is received, which is then decoded to recover m.

Formally: The messages will be words of length n over some alphabet A. So, $M \subseteq A^n$ is the set of messages. We will encode them into length $\hat{n}$ words over the alphabet Σ. A code, C is a subset of $\Sigma^{\hat{n}}$ containing the encoded messages. I.e., for a message, $m \in M$, its encoding, $enc(m)$ will be in C. $\hat{n}$ is called the block length of C. The $enc()$ function is a bijection and so M is in bijection with $enc(M) \subseteq C \subseteq \Sigma^{\hat{n}}$. The word received on sending $enc(m) \in C$ through the channel will be an element of $\Sigma^{\hat{n}}$. The rate of a code is given by

$$Rate(C) = n/\hat{n}$$

A higher block length will allow for better decoding (by increasing redundancy) but will decrease the rate. We begin with a few basic notions:

Definition 7. Hamming distance. Given $x \neq y \in \Sigma^{\hat{n}}$,

$$d_H(x, y) := Pr_{i\in[\hat{n}]}((x(i) \neq y(i)).$$

This is called the relative Hamming distance between x and y. The Hamming distance between x and y is the number of co-ordinates where x and y differ. We use the relative Hamming distance as it is a probability and amongst other things, lends itself nicely to define the agreement:

Definition 8. Agreement. Given $x \neq y \in \Sigma^{\hat{n}}$,

$$agr(x, y) = 1 - d_H(x, y).$$

Definition 9. Distance of a code, C. Given $C \subseteq \Sigma^{\hat{n}}$,

$$dist(C) = \min_{x\neq y, x,y\in C}(d_H(x, y)).$$

8.3 List Decoding

A (δ, L)-list decoder does the following: Given a (received) a word, $r \in \Sigma^{\hat{n}}$, it outputs all messages m such that $d_H((enc(m), r) \leq \delta$ and it also has the property that the total number of messages outputed for a given r is $\leq L$. Clearly, it is conceivable that there are δ, L for which no (δ, L)-list decoders exist.

8.4 Examples

8.4.1 Hadamard Code

Small bold letters denote vectors. All inner products are taken mod 2. $A = \Sigma = \{0, 1\}$. $\hat{n} = 2^n$. The encoding function is:

$$enc(\mathbf{a}) = (\langle \mathbf{a}, \mathbf{b} \rangle : \mathbf{b} \in \mathbf{A^n}) \in \mathbf{A^{\hat{n}}}.$$

I.e., a message $\mathbf{a}$ is encoded as the word given by taking its inner product with all the $\hat{n}$ many vectors in A^n. So, $enc(\mathbf{a}) \in \mathbf{A^{\hat{n}}}$. The Hadamard code is $C = enc(A^n)$. Note that $enc()$ defines a bijection between A^n and C. We show that this code has distance = 1/2: Fix any two distinct vectors in C. So, there are $\mathbf{a} \neq \mathbf{a'}$ in A^n such that these two distinct vectors in C are $enc(\mathbf{a})$ and $enc(\mathbf{a'})$. So, $d_H(enc(\mathbf{a}), enc(\mathbf{a'})) = \mathbf{Pr_{b \in [\hat{n}]}(enc(a)(b) \neq enc(a')(b))} = \mathbf{Pr_{b \in [\hat{n}]}}(\langle \mathbf{a}, \mathbf{b} \rangle \neq \langle \mathbf{a'}, \mathbf{b} \rangle)$

$$= \Pr_{\mathbf{b} \in [\hat{\mathbf{n}}]}(\langle \mathbf{a} - \mathbf{a'}, \mathbf{b} \rangle \neq \mathbf{0}) = \mathbf{1/2}.$$

The last equality follows as the inner product with $\mathbf{a} - \mathbf{a'} \neq \mathbf{0}$ of a vector picked uniformly randomly from A^n is 0 w.p. 1/2 and 1 w.p. 1/2. In fact, we have shown that an element of C is at a distance of 1/2 from *every* other element of C.

8.4.2 Reed-Solomon Code

Let q be a prime power. The message space will be $\mathbb{F}_q^d$ (i.e., $A = \mathbb{F}_q, n = d$). The encoding function is: For a message , $a = (a_0, .., a_{d-1})$ let

$$P_a(x) = \sum_i a_i x^i.$$

The encoding is:

$$enc(a) = (P_a(b) : b \in \mathbb{F}_q) \in \mathbb{F}_q^q.$$

The code, C is $enc(\mathbb{F}_q^d)$ and the rate is $n/\hat{n} = d/q$.

$$enc(a) \neq enc(a') \implies \exists b, P_a(b) \neq P_{a'}(b) \implies (P_a(x) - P_{a'}(x)) \neq 0 \implies$$

$$Pr_{b \in \mathbb{F}_q}((P_a(b) - P_{a'}(b)) \neq 0) \geq 1 - (d-1)/q \geq 1 - d/q$$

as the non-zero polynomial, $P_a(x) - P_{a'}(x)$ has degree atmost $d - 1$ and so has atmost $d - 1$ roots in $\mathbb{F}_q$. So, the distance of this code is atleast $1 - d/q$.

8.4.3 Reed-Muller Code

This is the multi-variate version of the Reed-Solomon code. Let q be a prime power. The message space will be $\mathbb{F}_q^N$ (i.e., $A = \mathbb{F}_q, n = N$) where N is taken to be C_d^{n+d} for some n, d. There are atmost N many monomials of degree d over $x_1, .., x_n$ and let they be listed as $m_1, .., m_N$. The encoding function is: For a message , $a = (a_0, .., a_N)$ let

$$P_a(x_1, .., x_n) = \sum_i a_i m_i.$$

The encoding is:

$$enc(a) = (P_a(b) : b \in \mathbb{F}_q^n).$$

The code, C is $enc(\mathbb{F}_q^N)$, has block length, q^n and rate N/q^n.

$$enc(a) \neq enc(a') \implies \exists b, P_a(b) \neq P_{a'}(b) \implies (P_a(x_1, .., x_n) - P_{a'}(x_1, .., x_n)) \neq 0 \implies$$

$$Pr_{b \in \mathbb{F}_q^n}((P_a(b) - P_{a'}(b)) \neq 0) \geq 1 - d/q$$

by the same argument as in Schwartz-Zippel applied to the non-zero polynomial, $P_a(x_1, .., x_n) - P_{a'}(x_1, .., x_n)$ with degree atmost $d - 1$. So, the distance is atleast $1 - d/q$.

8.5 Johnson Bound

We show the following theorem:

Theorem 3. Let $C \subseteq \Sigma^{\hat{n}}$ be a code of distance $1-\epsilon$ for some $0 < \epsilon < 1$.

Then C is $((1-O(\sqrt{\epsilon})), O\left(\frac{1}{\sqrt{\epsilon}}\right))$ list decodable.

Proof. Fix the C as in the theorem statement. To show the theorem we need to show that for any received word, r, there are atmost $\leq O\left(\frac{1}{\sqrt{\epsilon}}\right)$ messages which are $(1-O(\sqrt{\epsilon}))$-close to r. Note that there is a bijection between the set of messages, M and the subset $enc(M) \subset C$. So, it suffices to show that for any received word, r, if $\{c_i\}_{i \in s}$ is the set of codewords in C which are at distance $\leq 1-\epsilon' := 1-\sqrt{\epsilon}$ from r, then $s \leq O\left(\frac{1}{\sqrt{\epsilon}}\right)$: To this end, fix r and $\{c_i\}_{i \in s}$ as above.

Let N be the places in r which agree with some code word in $\{c_i\}_{i \in s}$. So, $N/\hat{n} \leq 1$. For a fixed c_i in $\{c_i\}_{i \in s}$, let

$$A_i := \{j \in [\hat{n}] : r(j) = c_i(j)\}$$

So, $\bigcup_{i \leq s} A_i$ is the set of places where r agrees with some c_i in $\{c_i\}_{i \in s}$. So, by inclusion-exclusion,

$$N = |\bigcup_{i \leq s} A_i| = \sum_i |A_i| - \sum_{i \neq j} |A_i \cap A_j|$$

Dividing by $\hat{n}$ and using the above,

$$1 \geq \sum_i |A_i|/\hat{n} - \sum_{i \neq j} |A_i \cap A_j|/\hat{n}$$

Now, $|A_i|/\hat{n} = agr(r, c_i) = 1 - d_H(r, c_i) \geq \epsilon'$ by choice of the c_i's.

Also, c_i agrees with c_j on $A_i \cap A_j$ as $c_i(t) = r(t) = c_j(t)$ for any $t \in A_i \cap A_j$ and so $agr(c_i, c_j) \geq |A_i \cap A_j|/\hat{n}$. So,

$$dist(C) = 1 - \epsilon \leq d_H(c_i, c_j) = 1 - agr(c_i, c_j) \leq 1 - |A_i \cap A_j|/\hat{n},$$

giving that $|A_i \cap A_j|/\hat{n} \leq \epsilon$. Putting these in the above,

$$1 \geq \sum_i \epsilon' - \sum_{i \neq j} \epsilon \geq s\epsilon' - s^2\epsilon = s\epsilon'(1 - s\epsilon'),$$

from which we have that, $s\epsilon' \leq 1/2$ which gives that $s = O(1/\epsilon')$ as needed. □

9 Lecture 9 (1 Sep 2022 - Thursday)

(Scribe: Harish Chandramouleeswaran, GCS202101)

9.1 List Decoding of Reed-Solomon Codes

Let q be a prime power. Let $\mathbb{F}_q = \{y_1, \cdots, y_q\}$. The message space is given by $\mathbb{F}_q^{d+1}$. Let $\bar{a} := (a_0, \cdots, a_d)$, and let $f_{\bar{a}}(x) := \sum\limits_{i=0}^{d} a_i x^i$. Note that $\deg f = d$. The encoding is then given by :

$$Enc(\bar{a}) := (f_{\bar{a}}(y_1), \cdots, f_{\bar{a}}(y_q)) \in \mathbb{F}_q^q.$$

The vector $(f_{\bar{a}}(y_1), \cdots, f_{\bar{a}}(y_q))$ is then sent on the communication channel, and the vector retrieved at the receiver's end is given by the vector $r := (r_1, \cdots, r_q)$ as a result of noise in the channel (say).

This received word r is interpreted as a function $r \colon \mathbb{F}_q \to \mathbb{F}_q$, which is defined as follows :

$$\forall i \in \{1, \cdots, q\},\ r(y_i) := r_i.$$

Fix an $\epsilon > 0$. We will now consider the set of code words, whose encoding agrees with the final received code word on $\geq \epsilon$ fraction of the q coordinates.

$$LIST(r, \epsilon) := \{\bar{a} \in \mathbb{F}_q^{d+1} \mid Agree(Enc(\bar{a}, r)) \geq \epsilon)\}.$$

Let $\bar{a} \in LIST(r, \epsilon)$. The receiver will now interpolate r using a bivariate polynomial $Q(Y, Z) := \sum\limits_{\substack{0 \leq i \leq d_Y \\ 0 \leq j \leq d_Z}} c_{i,j} Y^i Z^j$, where $d_Y := \deg_Y(Q(Y, Z)$, and $d_Z := \deg_Z(Q(Y, Z)$.

What we need :

$$\forall y \in \mathbb{F}_q, \quad Q(y, r(y)) = 0. \quad (q \text{ constraints}) \tag{9.1}$$

$$\text{Unknowns to solve for : } c_{(i,j)}. \quad ((1 + d_Y)(1 + d_Z) \text{ unknowns})$$

$$\text{For a nontrivial solution (that is, for } Q(Y, Z) \not\equiv 0\text{), we need } (1 + d_Y)(1 + d_Z) > q. \tag{9.2}$$

Let's say we have found a $Q(Y, Z)$ which satisfies the q constraints (9.1). Let $\mathfrak{F} := \{(y, f_{\bar{a}}(y)) \mid y \in \mathbb{F}_q\}$. For at least ϵq-many i in $\{1, \cdots, q\}$, $f_{\bar{a}}(y_i) = r(y_i)$ $(= r_i)$, since $\bar{a} \in LIST(r, \epsilon)$. So, for at least ϵq-many elements $(y, f_{\bar{a}}(y))$ of $\mathfrak{F}$, we have $Q(y, f_{\bar{a}}(y)) = 0$.

Now, consider the (univariate) polynomial $Q(Y, f_{\bar{a}}(Y))$ in the variable Y, obtained by substituting $Z := f_{\bar{a}}(Y)$ in the expression for $Q(Y, Z)$. We note that $\deg Q(Y, f_{\bar{a}}(Y)) \leq d_Y + dd_Z$ (as a polynomial in Y).

$$\text{Suppose we can ensure that } \epsilon q > d_Y + dd_Z. \tag{9.3}$$

If (9.3) is satisfied, then we have $Q(Y, f_{\bar{a}}(Y)) \equiv 0$, since a nonzero polynomial of degree t has at most t roots.

In other words, if (9.3) is satisfied, then $f_{\bar{a}}(Y)$ is a root of $Q(Y, Z)$ considered as a polynomial in the variable Z with coefficients in $\mathbb{F}_q(Y)$ (the field of fractions of $\mathbb{F}_q[Y]$).

$$\text{Formally let } \tilde{Q}(Z) := Q(Y, Z) = \sum_{i=0}^{d_Z} \tilde{c_j} Z^j, \quad \text{where, for each } j, \tilde{c_j} \in \mathbb{F}_q(Y).$$

Our discussion so far is then summarized as : For each $\bar{a} \in LIST(r, \epsilon)$, its contents $a_0, \cdots, a_d$ are the coefficients of the polynomial $f_{\bar{a}}(Y) := \sum\limits_{i=0}^{d} a_i Y^i$, and this $f_{\bar{a}}(Y)$ will be one of the roots of $\tilde{Q}(Z)$, provided (9.1), (9.2), and (9.3) are met.

So, factoring $\tilde{Q}$ using an appropriate factoring algorithm will give us all the original code words, possibly along with some other extra factors.

We note that d is chosen by the sender, and we have no control over it. We set $d_Y \approx \frac{\epsilon q}{2}$ and $d_Z \approx \frac{\epsilon q}{2d}$ to satisfy (9.3), and this requires $\epsilon \approx 2\sqrt{\frac{q}{d}} \approx \sqrt{\rho}$, where ρ is the rate of the code given by $\rho := \frac{d+1}{q}$.

Summarising, given a list-decoding radius $\epsilon > 0$, we can efficiently correct the errors of a Reed-Solomon code with rate $\rho \approx \Theta(\epsilon^2)$ using the procedure outlined in this lecture.

Note : It is in fact possible to efficiently correct the errors of a Reed-Solomon code with rate $\rho \approx \Theta(\epsilon)$ using the Guruswami-Sudan list-decoding Algorithm!

10 Lecture 10

Scribe: Saswata, MCS202220

10.1 Introduction

In last class we have seen (r, ϵ) list decoding algorithm for Reed-Solomon Code, where rate of the code $\rho \sim \epsilon^2$. Now we shall see another code, that is **Parvaresh–Vardy Codes** and (r, ϵ) list decoder for it, where $\epsilon \sim \rho \log \rho$.

10.2 Parvaresh–Vardy Codes

q be a prime power and $h, m \in \mathbb{N}$ which we shall set later suitably.
$E(Y) \in \mathbb{F}_q[Y]$ be an irreducible polynomial of degree $> d$ (say, $d + 1$).

1. Alphabet set $\Sigma = \mathbb{F}_q^m$.

2. For a message $a = (a_0, a_1, \ldots, a_d) \in \mathbb{F}_q^{d+1}$, say, $f(Y) = \sum_{i=0}^{d} a_i Y^i$.
   Enumerate $\mathbb{F}_q = \{y_1, \ldots, y_q\}$.
   $Enc(a_0, \ldots, a_d) = ((f_0(y_1), \ldots, f_{m-1}(y_1)), \ldots, (f_0(y_q), \ldots, f_{m-1}(y_q)))$.

   For each $j \in [q]$, $(f_0(y_j), \ldots, f_{m-1}(y_j)) \in \Sigma$ and $f_i(Y) = (f(Y))^{h^i} (\text{mod } E(Y))$.

We visualise recieved message r as a function $r : \mathbb{F}_q \to \mathbb{F}_q^m = \Sigma$, where $r(y_j) = (r_0, \ldots, r_{m-1})$.
For each input $y \in \mathbb{F}_q$ epresent r as $(y, r_0, \ldots, r_{m-1})$, this $m + 1$-tuple.

Rate: Rate of this code $= \dfrac{d+1}{mq} \approx \dfrac{d}{mq}$.

10.3 Construction of List Decoder

On input $r : \mathbb{F}_q \to \mathbb{F}_q^m$, first construct nonzero polynomial $Q(Y, Z_0, \ldots, Z_{m-1})$, by interpolation such that,

$$\forall y \in \mathbb{F}_q, \;\; Q(y, r_0(y), \ldots, r_{m-1}(y)) = 0$$

Say, $d_Y = \deg_Y(Q)$ and $d_{Z_i} = \deg_Q(Z_i)$. We want, for all i, $d_{Z_i} \leq h - 1$.

Number of monimials in Q is $(1 + d_Y)(1 + d_{Z_0}) \ldots (1 + d_{Z_{m-1}}) \leq (1 + d_Y)h^m$. Therefore to have such Q, we need,

$$(1 + d_Y)h^m > q \ldots\ldots \text{(I)} \;\; [\text{ As there are } q \text{ number of } m+1 \text{ tuples } (y, r_0(y), \ldots, r_{m-1}(y))]$$

Now, consider the polynomial $\tilde{Q}(Y) = Q(Y, f_0(Y), \ldots, f_{m-1}(Y))$.
$f_i(Y) = (f(Y))^{h^i} (\text{mod } (E(Y))) \implies \deg(f_i) \leq d$ (As, $\deg_Y E = d + 1$).
Therefore, $\deg(\tilde{Q}) \leq d_Y + d(h-1)m$ (As, $\deg_{Z_i} Q \leq h - 1$).

If we can say, $\tilde{Q} \equiv 0$, that is $Q(Y, f(Y), \ldots, f^{h^{m-1}}(Y))(\text{mod } (E(Y))) \equiv 0$, then, f will be a root of $Q^*(Y, Z) = Q(Y, Z, \ldots, Z^{h^{m-1}})$ over $\mathbb{F}_q/(E(Y))$. In that case, we can factor the polynomial $Q^*(Y, Z) = Q(Y, Z, \ldots, Z^{h^{m-1}})$ over $\mathbb{F}_q[Y]/(E(Y))$ and get $(Z - f(Y))$ as a factor, where $agr(Enc(f), r) > \epsilon$.

Number of f such that $agr(Enc(f), r) > \epsilon$, is at least ϵq. To have our required conditions to be satisfied, we need,

$$\epsilon q > d_Y + d(h-1)m \ldots\ldots \text{(II)}.$$

Because, if number of roots of a univariate polynomial is greater than it's degree then the polynomial is identically zero.

Lemma 3. $Q^*(Y, Z) = Q(Y, Z, Z^h, \ldots, Z^{h^{m-1}}) \neq 0$.

Proof. $Q = \sum c(Y) Z_0^{e_0} \ldots Z_{m-1}^{e_{m-1}}$ where $c(Y) \in \mathbb{F}_q[Y]$ and $e_i \leq h-1$.
$\implies Q^* = \sum c(Y) Z^{e_0 + he_1 + \cdots + h^{m-1} e_{m-1}}$, $e_i \leq h-1$.
As we know the map, $(e_0, \ldots, e_{m-1}) \mapsto \sum_i e_i h^i$ is one-one, if $e_i \leq h-1$, the substitution of Z_i by Z^{h^i} does not cause non-trivial cancellation of monomials. Hence, Q^* is non-zero. □

Thus, $Q^* \neq 0$ and it will also be nonzero on $(\text{mod } E(Y))$.

So, if condition (I) and (II) holds, then list decoder can factor nonzero polynomial $Q^*(Z)$ over $\mathbb{F}_q[Y]/(E(Y))$ and gives output the roots of it.
The factoring cam be done in $poly(q, d, h^m)$ time.

Now, we can take $h = 2$, i.e., $\deg_{Z_i} Q \leq 1$ and $m = \log(q/d)$.
By condition (II), take $d_Y \approx \epsilon q - dhm$.
Then, By condition (I), $(q\epsilon - dhm)h^m > q \implies \epsilon \approx \dfrac{1}{h^m} + \dfrac{dhm}{q} \approx \dfrac{d}{q} + hm\dfrac{d}{q} \approx \rho \log \rho$.

11 Lecture 11

Scribe: Sagnik Dutta, MCS202112

In this class, we will see connections between list-decodable codes and expander graphs. Using these connections, we will construct graphs with good vertex expansion from Parvaresh-Vardy codes.

11.1 Two views

1. **Graph view:** Consider a bipartite graph $G = (N, M, D)$ where the bipartition has the vertex sets $[N]$ and $[M]$ and every vertex in $[N]$ has degree D. The function $\Gamma : [N]\times[D] \to [M]$ is defined by $\Gamma(x, y) = y$-th neighbour of x in $[M]$.

2. **Code view:** Consider a code Enc: $[N] \to [M]^d$. Let $M' = DM$. Think of $[M']$ as $[D] \times [M]$. The function $\Gamma : [N] \times [D] \to [M']$ is defined by $\Gamma(x, y) = (y, (\text{Enc}(x))_y)$.

Definition 10. For $T \subseteq [M]$, define

$$\text{LIST}_\Gamma(T, \epsilon) = \{x \in [N] \ : \ \Pr_y[\Gamma(x, y) \in T] > \epsilon\}$$
$$\text{LIST}_\Gamma(T, 1) = \{x \in [N] \ : \ \forall y\, \Gamma(x, y) \in T\}.$$

Exercise 1. Enc is a $(1 - 1/q - \epsilon, k)$ LIST-decodable code iff $\forall r \in [M]^D$, $|\text{LIST}_\Gamma(T_r, 1/q + \epsilon)| \le k$ where $T_r = \{(y, r_y) \in [M']\}$.

Definition 11. Fix k. A bipartite graph G is a $(= k, A)$ vertex-expander if $\forall S$, $|S| = k$, we have $|N(S)| \ge Ak$.

Observation: G is a $(= k, A)$ expander $\iff$ $\forall k' \le k$, G is a $(= k', A)$ expander.

Lemma 4. Fix k. $\Gamma : [N] \times [D] \to [M]$ is an $(= k, A)$ expander iff $\forall T \subseteq [M]$ such that $|T| < kA$, we have $|\text{LIST}_\Gamma(T, 1)| < k$.

Proof. $\exists T \subseteq [M]$ such that $|T| < kA$ and $|\text{LIST}_\Gamma(T, 1)| \ge k$
$\implies$ $\exists S \subseteq \text{LIST}_\Gamma(T, 1)$, such that $|S| = k$ and $N(S) \subseteq T$ implying $|N(S)| < kA$.
$\implies$ Γ is not an $(= k, A)$-expander.

Γ is not an $(= k, A)$-expander.
$\implies$ $\exists S \subseteq [N]$ such that $|S| = k$ and $|N(S)| < kA$.
$\implies$ For $T = N(S)$, we have $|T| < kA$ and $S \subseteq \text{LIST}_\Gamma(T, 1)$ implying $|\text{LIST}_\Gamma(T, 1)| \ge k$. $\square$

11.2 Parvaresh-Vardy codes

Every message $\boldsymbol{a} = (a_0, \dots, a_{n-1})$ in the message space $\mathbb{F}_q^n$ is viewed as a function $f^{\boldsymbol{a}}(x) = \sum_{i=0}^{n-1} a_i x^i$. Enumerate the elements of $\mathbb{F}_q$ as $y_1, \dots, y_q$. For an irreducible polynomial $E(Y)$ of degree n, define $f_i(Y) = f^{h^i}(Y) \bmod E(Y)$. Define $\Gamma : \mathbb{F}_q^n \times \mathbb{F}_q \to \mathbb{F}_q^m$ as $\Gamma(\boldsymbol{a}, y) = (f_0^{\boldsymbol{a}}(y), \dots, f_{m-1}^{\boldsymbol{a}}(y))$.

Theorem 4. Γ is a $(= k_{max}, A)$-expander for $k_{max} = h^m$ and $A = q - nhm$.

Proof. By Lemma 4, it suffices to show that $\forall T \subseteq \mathbb{F}_q^m$ such that $|T| \leq Ak_{max} - 1$, we have $|\text{LIST}_\Gamma(T, 1)| \leq k_{max} - 1$.

Interpolate T via a low-degree multivariate curve:

We want to find a polynomial $Q(Y, Z_0, Z_1, \ldots, Z_{m-1})$ with $\deg_Y(Q) = d_Y = A-1$ and $\deg_{Z_i}(Q) \leq h-1$ for $0 \leq i \leq m-1$ such that $Q(y, r_0(y), \ldots, r_{m-1}(y)) = 0$ for all $y \in \mathbb{F}_q$. Such Q exists since the possible number of monomials in Q is $(1 + d_Y)h^m$ and

$$(1 + d_Y)h^m = Ah^m > Ak_{max} - 1 \geq |T|.$$

Want to argue that $Q(y, f_0(y), f_1(y), \ldots, f_{m-1}(y)) = 0$ for all $f \in$ LIST$_\Gamma(T, 1)$:

This is ensured when $q > d_Y + hmn = A - 1 + hmn$ [$\because$ if the number of roots of a univariate polynomial is greater than the degree of the polynomial, then the polynomial is identically zero].

After ensuring this, we will have $Q(y, f, f^h, \ldots, f^{h^{m-1}}) = 0 \bmod E(Y)$ for all $f \in \text{LIST}_\Gamma(T, 1)$. Thus, all such f are roots of the polyomial $\widetilde{Q}(Z) = Q(y, Z, Z^h, \ldots, Z^{h^{m-1}})$ over $\mathbb{F}_q[Y]/E(Y)$.

We have $|\text{LIST}_\Gamma(T, 1)| \leq \deg_Z(\widetilde{Q}) \leq (h-1) + h(h-1) + h^2(h-1) + \ldots + h^{m-1}(h-1) = (h-1).\dfrac{h^m - 1}{h-1} = h^m - 1 = k_{max} - 1.$ □

Setting the parameters suitably, we get

For every $\alpha > 0$, every $N, k, \epsilon > 0$, there is an explicit $(k, (1-\epsilon)D)$ vertex expander with left vertex set $[N]$, and right vertex set $[M]$ with $D = O((\log N)(\log k)/\epsilon)^{1+1/\alpha}$ and $M \leq D^2 k^{1+\alpha}$.

12 Lecture 12

Scribe: Aryan Agarwala, BMC202010

In this lecture we will describe the Impagliazzo, Nisan, Wigderson (INW) pseudorandom generator.

We begin by formally defining two classes:

1. BPSPACE(f) - this is the set of languages L for which there exists a randomised Turing machine M that uses $f(n)$ work tape space on inputs of length n. On any given input x, M correctly determines whether $x \in L$ with probability $\geq \frac{2}{3}$.

2. RSPACE(f) - this is the one-sided error version of BPSPACE(f). If $x \notin L$, then M must output 0 with probability 1. If $x \in L$, then M must output 1 with probability $\geq \frac{1}{2}$.

Note that RSPACE(f) $\subseteq$ NSPACE(f) $\subseteq$ DSPACE(f^2)

The first containment above is trivial, and the second is by Savitch's theorem. Note that the first containment does not hold true for BPSPACE. This begs the question: does a similar statement hold true for BPSPACE? The INW pseudorandom generator answers this question in the affirmative.

12.1 Pseudorandom Generator

We define an ϵ-PRG to be a function $G : \{0,1\}^d \to \{0,1\}^m$ with $m >> d$ such that

$$\forall x \, |\Pr[M(x,y) = 1] - \Pr[M(x, G(y)) = 1]| < \epsilon$$

.

We will now see Impagliazzo, Nisan, and Wigderson's construction of a pseudorandom generator for space bounded computation.

Let's say that we have a randomised Turing machine M that runs in space $s(n)$ on inputs of length n.

We have seen before that if $s(n) = \Omega(\log n)$ then the time taken by machine M is $2^{O(s(n))}$.

We look at the computation tableau of M on an input y of length n. We divide this tableau into blocks of $r = \Theta(s(n))$ configurations. There are at most $2^{O(s(n))}$ such blocks. Each of these blocks can be thought of as an individual program that uses at most r random bits. Let's label these blocks from 1 to k. The work-tape configuration after the i^{th} block runs can be referred to as A_i, which is a function of the work tape configuration after $(i-1)^{th}$ block, along with the random bits fed to the i^{th} block. We will somewhat abuse notation and write this as $A_i(A_{i-1}, r_i)$ where r_i is the $s(n)$ length random string fed to the i^{th} block.

The idea now is that we will pair up adjacent blocks. Instead of using completely random strings r_i and r_{i+1}, we will use randomness to obtain the string r_i, and then obtain the string r_{i+1} by traversing the edge of an expander graph with 2^r vertices. Let's say that the expander has degree 2^m. This reduces the total randomness from kr to $\frac{k}{2} \cdot (r+m)$. Our hope is that m is much smaller than r. However, regardless of how small m is, this reduces the total randomness by at most a factor of 2. How do we reduce it further? Let's consider 4 adjacent blocks i, $i+1$, $i+2$, $i+3$. $(i, i+1)$ and $(i+2, i+3)$ are paired blocks that take $(r+m)$ random bits each. We now pair up these pairs. We obtain the random string for $(i, i+1)$ from pure randomness. In order to obtain the random string for $(i+2, i+3)$, we again traverse the edge of an expander graph with 2^{r+m} vertices. Again, let the degree of the expander be 2^m. This reduces the total randomness to $\frac{k}{4} \cdot (r + 2m)$. We repeat this process again and again, until we are left with total randomness of $r + \log k \cdot m = r + sm$. We will show a family of expanders such that $m = O(s) \implies$ the total randomess is $O(r + s^2) = O(s^2)$.

The family of strongly explicit expander graphs we are going to use was defined by Lubotzky, Phillips, Sarnak. This family has the property that, for an expander with degree k, the spectral gap λ is $\leq \sqrt{k^{-1}}$. We select k to be 2^{6s}, which implies that $\lambda \leq 2^{-3s}$. Let the i^{th} neighbour of vertex v be denoted by $n_i(v)$. We now use the following version of the expander mixing lemma:

Lemma 3 (Expander Mixing Lemma)**.** For any two sets $S, T \subseteq V$,

$$\frac{|E(S,T)|}{|E|} - \frac{|S||T|}{|V|^2} \leq \lambda$$

We will now bound the error made by the generator at the very first step of computation. It is simple to extend this argument inductively.

Consider the first two blocks of computation. We want to show that $\forall b \in \{0,1\}^s$,

$$\Pr_{x \in \{0,1\}^r, y \in \{0,1\}^r} [A_2(A_1(x), y) = b]$$

is similar to

$$\Pr_{x \in \{0,1\}^r, y \in \{0,1\}^{6s}} [A_2(A_1(x), n_y(x)) = b]$$

Let $b' \in \{0,1\}^s$. We define two sets:

$$S_{b'} = \{x \in \{0,1\}^r | A_1(x) = b'\}$$

$$T_{b'} = \{x \in \{0,1\}^r | A_2(b', x) = b$$

Now we have,

$$\Pr_{x \in \{0,1\}^r, y \in \{0,1\}^r} [A_2(A_1(x), y) = b] = \sum_{b' \in \{0,1\}^s} \frac{|S_{b'}||T_{b'}|}{2^{2r}}$$

Similarly,

$$\Pr_{x \in \{0,1\}^r, y \in \{0,1\}^{6s}} [A_2(A_1(x), n_y(x)) = b] = \sum_{b' \in \{0,1\}^s} \frac{|E(S_{b'}, T_{b'})|}{|E|}$$

Therefore, by triangle inequality, the difference between the two is bounded by

$$\sum_{b' \in \{0,1\}^s} |\frac{|E(S_{b'}, T_{b'})|}{|E|} - \frac{|S_{b'}||T_{b'}|}{2^{2r}}|$$

By lemma 3, this is

$$\leq 2^s \cdot \lambda = 2^s \cdot 2^{-3s} = 2^{-2s}$$

In total there are 2^s generators in this system. An extension of this argument will show that the probability of error in any of the generators is $\leq 2^{-2s}$. Therefore, the overall probability of an error is $\leq 2^{-s}$.

Let G denote our generator from $O(s^2)$ bits to $O(2^s)$ bits. Therefore we have that for any Turing machine M with s work tape space on input x, $\forall b \in \{0,1\}^s$

$$|\Pr_{r \in \{0,1\}^{2^s}}[M(x,r) = b] - \Pr_{r \in \{0,1\}^{s^2}}[M(x, G(r)) = b]| \leq 2^{-s}$$

We can reduce the error by constant powers of 2 by simply increasing the degree of the expander.

13 Lecture 13

Scribe: Naman Kumar, BMC202026

We will see some basic proofs related to pseudorandom generators in motivation of our goal to determine whether BPP = P.

Motivating Question We wish to determine the question of whether BPP = P, where BPP is the class of problems which can be solved in polynomial time using polynomial number of random bits.

Pseudorandom Generator We recall the concept of a pseudorandom generator, which is a function $G : \{0,1\}^d \to \{0,1\}^m$ with $m >> d$ such that the generated bits are computationally indistinguishable from random bits. If such a generator can be built with $d = O(\log m)$ for all BPP problems, then BPP = P, as a brute force search through all strings d works.

We now define the following notion.

Computational Indistinguishability. Let X and Y be two distributions over $\{0,1\}^m$. X, Y are (t, ϵ)-computationally indistinguishable if for all non-uniform algorithms (circuits) of runtime t, X, Y can take different values with at most ϵ. In other words, for all algorithms T such that $|T| \le t$,

$$\left| \Pr_{x \sim X}[T(x) = 1] - \Pr_{y \sim Y}[T(y) = 1] \right| \le \epsilon$$

Exercise. Using the probabilistic method, show that $\forall m, \epsilon > 0$, there exists an (m, ϵ)-PRG $G : \{0,1\}^d \to \{0,1\}^m$ such that $d = O(\log m/\epsilon)$.

13.1 Hybrid Algorithms

One of the most useful tools to analyze pseudorandom generators (introduced by Yao) are *hybrid arguments.*

Lemma 5. If X, Y are (t, ϵ) indistinguishable, $X^{\otimes k}$ and $Y^{\otimes k}$ are $(t, k\epsilon)$-indistinguishable.

Proof. We proceed through contradiction. Suppose $X^{\otimes k}$ and $Y^{\otimes k}$ are not (k, ϵ)-indistinguishable. Then there exists a circuit (or a nonuniform algorithm) family T of size at most t such that

$$\left| \Pr_{x \sim X^{\otimes k}}[T(x) = 1] - \Pr_{y \sim Y^{\otimes k}}[T(y) = 1] \right| > k\epsilon$$

We now define the following set of distributions.

$$H^{\otimes i} = X^{\otimes i} \times Y^{\otimes k-i}$$

Then we can rewrite our original equation as $|\Pr[T(H^{\otimes 0}) = 1] - \Pr[T(H^{\otimes k}) = 1]| > k\epsilon$, and we get

$$\left|\sum^{k} \Pr[T(H^{\otimes i-1}) = 1] - \Pr[T(H^{\otimes i}) = 1]\right|$$
$$\leq \sum^{k} \left|\Pr[T(H^{\otimes i-1}) = 1] - \Pr[T(H^{\otimes i}) = 1]\right| > k\epsilon$$

And thus, from the pigeonhole principle, there exists an i for which

$$\left|\Pr[T(H^{\otimes i-1}) = 1] - \Pr[T(H^{\otimes i}) = 1]\right| > \epsilon$$

Both the H differ on the ith value, and it follows that there are $x \in X^{\otimes i}$ and $y \in Y^{\otimes(i-1)}$ for which the circuit evaluation difference is greater than ϵ. However, such a circuit does not exist since X and Y are (t, ϵ)-indistinguishable, and the proof follows. □

13.2 Next-bit Predictors

We now define the following concept, which first came up in the context of cryptography.

Next Bit Predictivity. Suppose we have a distribution $X \sim \{0,1\}^m$. X is then (t, ϵ)-next bit unpredictable if for all circuits of size $\leq t$,

$$\Pr[T(x_1 x_2 \ldots x_{i-1}) = x_i] \leq \frac{1}{2} + \epsilon$$

These definitions are equivalent, as Yao proved the following theorem:

Theorem 5. Next-bit unpredictability is equivalent to pseudorandomness.

Proof. We proceed through proving the contrapositive. Assume that some distribution X is not next-bit unpredictable. Then suppose that $\exists T$ of size t such that

$$\Pr[T(x_1 \ldots x_{i-1}) = x_i] > \frac{1}{2} + \epsilon$$

Then we can define the following test $\hat{T}$, which works as

$$\hat{T}(x_1 x_2 \ldots x_{i-1}) = \begin{cases} 1 & \text{if } T(x_1 \ldots x_{i-1}) = x_i \\ 0 & \text{otherwise} \end{cases}$$

Then a circuit for this test can be easily shown to determine whether X is computationally indistinguishable from the uniform distribution, which is $t - O(1)$. □

We now show the other direction. To do this, suppose that X is not $(m\epsilon)$-pseudorandom. This imploes that there exists some test $|T| \leq t$ for which

$$|\Pr[T(X_1X_2\ldots X_m) - 1] - \Pr[T(U_1U_2\ldots U_m) = 1]| > m\epsilon$$

We define the distributions $H^{\otimes i}$ as we did in the previous hybrid argument. Repeating the proof, we get that $\exists i$ such that

$$\left|\Pr[T(H^{\otimes i-1}) = 1] - \Pr[T(H^{\otimes i}) = 1]\right| > \epsilon$$

Then considering X_i, U_i, we can define U_i as $\frac{1}{2}\left(X_i + \overline{X_i}\right)$. Then substituting into the equation, we have,

$$\frac{1}{2}\left|\Pr[T(X_1X_2\ldots X_iU_{i+1}\ldots U_m) = 1] - \Pr[T(X_1X_2\ldots \overline{X_i}U_{i+1}\ldots U_m) = 1]\right| > \epsilon$$

We will use this information to construct a next-bit predictor for the bit i. □

14 Lecture 14

Scribe: Aryan Agarwala, BMC202010

14.1 Next Bit Unpredictability and Pseudorandomness

In this section we will continue our proof from last time, and show that if a distribution is next bit unpredictable, then it must be pseudorandom.

$X \sim \{0,1\}^m$

Suppose X is not (t, m, ϵ) pseudorandom.

Then there exists a non uniform algorithm T which takes time at most t such that

$$|\Pr_{x \sim X}[T(x) = 1] - \Pr_{u \sim \{0,1\}^m}[T(u) = 1]| > m\epsilon$$

$$\implies |\Pr_{x \sim X}[T(x_1, x_2, \cdots, x_m) = 1] - \Pr_{u \sim U}[T(u_1, u_2, \cdots, u_m) = 1] > m\epsilon|$$

By an averaging argument, there must exist i such that

$$\Pr[T(x_1, \cdots, x_i, u_{i+1}, \cdots, u_m) = 1] - \Pr[T(x_1, \cdots, x_{i-1}, u_i, u_{i+1}, \cdots, u_m) = 1] > \epsilon$$

Note that $u_i = 0.5x_i + 0.5\bar{x_i}$

$$\begin{aligned}\implies |\Pr[T(x_1, \cdots, x_i, u_{i+1}, \cdots, u_m) = 1] - \frac{1}{2}\Pr[T(x_1, \cdots, x_{i-1}, \bar{x_i}, \cdots, u_m) = 1]\\ -\frac{1}{2}\Pr[T(x_1, \cdots, x_i, u_{i+1}, \cdots, u_m) = 1]| > \epsilon\\ \implies |\Pr[T(x_1, \cdots, x_i, u_{i+1}, \cdots, u_m) = 1] - \Pr[T(x_1, \cdots, x_{i-1}, \bar{x_i}, \cdots, u_m) = 1]| > 2\epsilon\end{aligned}$$

Let this value be δ. Also, without loss of generalisation, assume that $\Pr[T(x_1, \cdots, x_i, u_{i+1}, \cdots, u_m) = 1] > \Pr[T(x_1, \cdots, x_{i-1}, \bar{x_i}, u_{i+1}, \cdots, u_m) = 1]$.

We define our next bit predictor $P(x_1, \cdots, x_{i-1})$ as follows:
We evaluate T on uniformly random bits $u_i, \cdots, u_m$ as $T(x_1, \cdots, x_{i-1}, u_i, \cdots, u_m)$. If the result is 1, then we output $x_i = u_i$, else we output $x_i = \bar{u_i}$.

Now we show that our predictor is correct with probability $> 0.5 + \epsilon$,

$$\begin{aligned}\Pr[P(x_1, \cdots, x_{i-1}) = x_i] = \Pr[T(x_1, \cdots, x_{i-1}, u_i, \cdots, u_m) = 1 | u_i = x_i] \Pr[u_i = x_i]\\ + \Pr[T(x_1, \cdots, x_{i-1}, u_i, \cdots, u_m) = 0 | u_i \neq x_i] \Pr[u_i \neq x_i]\\ = \frac{1}{2}(\Pr[T(x_1, \cdots, x_i, u_{i+1}, u_m) = 1] + (1 - \Pr[T(x_1, \cdots, \bar{x_i}, u_{i+1}, \cdots, u_m) = 1]))\\ \geq \frac{1}{2} + \delta > \frac{1}{2} + \epsilon\end{aligned}$$

Therefore, X is not (t, ϵ) next bit unpredictable.

This proves that (t, ϵ) next bit unpredictability implies (t, m, ϵ) pseudorandomness.

Combined with our proof in the previous lecture, this shows that next bit unpredictability and pseudorandomness are equivalent.

14.2 Average Case Hardness

Given parameters (s, δ), a function $\{0,1\}^l \mapsto \{0,1\}$ is defined to be (s,δ)-hard if for any (non-uniform) algorithm A of size $\leq s$,

$$\Pr_{x \sim \{0,1\}^l}[A(x) = f(x)] \leq 1 - \delta$$

By setting $\delta = \frac{1}{2} - \epsilon$, we can alternatively define a function to be (s, ϵ)-hard if

$$\Pr_{x \sim \{0,1\}^l}[A(x) = f(x)] \leq \frac{1}{2} + \epsilon$$

Our goal to show that that the existence of average case hard functions implies the existence of pseudorandom functions. In order to do this, we first need to introduce the concept of a design.

14.3 Design

An (m, d, l, a) design is $S_1, S_2, \cdots, S_m$ such that each $S_i \subset [d]$, $|S_i| = l$ and for all pairs i, j, we have $|S_i \cap S_j| \leq a$.

Consider the finite field F_p where p is a prime. Now, let $p_1, \cdots$ be the set of polynomials in $F_p[x]$ with degree $\leq k$. There are p^k such polynomials.

Consider the sets $S_1, \cdots, S_{p^k}$ defined by

$$S_i = \{(j, p_i(j) | j \in F_p\}$$

$|S_i \cap S_j| = \{(x \in F_p | p_i(x) = p_j(x)\}$

By the Schwartz-Zippel lemma, this is bounded by k.

Therefore, $S_1, \cdots, S_{p^k}$ defines a $(p^k, [p] \times [p], p, k)$ design.

14.4 Average Case Hardness and Pseudorandomness

Let's say we have a (m, d, l, a) design $S_1, \cdots, S_m$, along with an average case hard function $f : \{0,1\}^l \mapsto \{0,1\}$. Let $S_i(j)$ refer to the j^{th} element of S_i.

For $x \in \{0,1\}^d$, we define $x_{S_i} \in \{0,1\}^l$ for $1 \leq i \leq m$ by $(x_{S_i})_j = 1$ if and only if the $x_{S_i(j)} = 1$.

Now we define $g : \{0,1\}^d \mapsto \{0,1\}^m$ by

$$g(x) = f(x_{S_1}) \cdot f(x_{S_2}) \cdots f(x_{s_m})$$

We claim that g is (t, ϵ) pseudorandom for $t = s - ma2^a$.

We will do this by showing that it is $(t, \epsilon/m)$ next bit unpredictable.

Assume that that there exists a predictor P such that $\Pr[P(f(x_{S_1}), f(x_{S_2}), \cdots, f(x_{S_{i-1}})) = f(x_{S_i})] > \frac{1}{2} + \frac{\epsilon}{m}$

We can fix the bits in $[d]\ S_i$ such that, by an averaging argument, the following still holds:

$\Pr_{x_{S_i(1)} x_{S_i(2)} \cdots x_{S_i(l)}}[P(f(x_{S_1})f(x_{S_2})\cdots f(x_{S_{i-1}})) = f(x_{S_i})] > \frac{1}{2} + \frac{\epsilon}{m}$

Since a is small, you can embed this function deterministically in a circuit with small size. This will contradict the average case hardness of the function.

15 Lecture 18: Razborov-Smolensky: Lower bound for Parity (Oct 13)

Scribe: Tejas Bhojraj, GCS202102

Today we start with boolean circuit lower bounds. Recall that $AC^0[d]$ is the class of families circuits with constant depth, d and unbounded fanin. Define the boolean function on n bits: $PARITY_n(x_1, .., x_n) := \sum_i x_i (\text{mod } 2)$. Using the polynomial method of Razborov and Smolensky, we show that:

Theorem 6. If $(C_n)_n \in AC^0[d]$ computes the family $(PARITY_n)_n$ (i.e., $PARITY_n = C_n$ for all n), then $|C_n| \geq 2^{\Omega(n^{1/2d})}/\sqrt{n}$.

The main idea is "$AC^0[d]$ circuits can be well-approximated by low-degree polynomials but $PARITY_n$ cannot". A polynomial $p(x_1, .., x_n) \in \mathbb{F}_3[x_1, .., x_n]$ is said to be proper if it takes value 0 or 1 on inputs from $\{0, 1\}^n$. The following 2 lemmas imply the above theorem

Lemma 6. Fix d and a depth d AC^0 circuit C on n variables. For any t, there is a degree $(2t)^d$ proper polynomial, $p \in \mathbb{F}_3[x_1, .., x_n]$ that disagrees with C on atmost $|C|2^{n-t}$ points.

Lemma 7. Any proper $p \in \mathbb{F}_3[x_1, .., x_n]$ with degree atmost $\sqrt{n}$ disagrees with $PARITY_n$ on atleast $2^n/\sqrt{n}$ points.

The proof of the main theorem follows:

Proof. Take a C_n assumed by the Theorem. Let $t = n^{1/2d}$ in the first lemma to get a proper $p \in \mathbb{F}_3[x_1, .., x_n]$ of degree $(2t)^d \leq \sqrt{n}$ which disagrees with C_n on the set D. The 2 lemmas give that $2^n/\sqrt{n} \leq |D| \leq |C_n|2^{n-n^{1/2d}}$ implying that $|C_n| \geq 2^{n^{1/2d}}/\sqrt{n}$ □

The first lemma says that an AC^0 circuit can be well approximated by a low-degree polynomial while the second says that $PARITY_n$ cannot.

Proof of the first Lemma:

Proof. Assume that C has only NOT and OR gates (this is wlog as replacing all AND gates with an OR and a NOT gates will increase the depth and size of C by a factor of atmost 2). For each gate, g of C we define a proper $p_g \in \mathbb{F}_3[x_1, .., x_n]$ inductively. As p_g will be defined using randomness when g is an OR gate, p_o, the polynomial at the output gate o depends on the random choices made while picking p_g's for OR gates g. The required p will be p_o for a suitable fixing of random choices.

The inductive construction is: For a leaf, $l = x_i$, let $p_l = x_i$ (p_l is proper as x_i is boolean). If g is a NOT gate with input h, then let $p_g = 1 - p_h$. As p_h is proper (inductively), so is p_g. Suppose now that g is an OR gate with inputs, $g_1, .., g_k$. Inductively, we have proper polynomials, $p_i := p_{g_i} \in \mathbb{F}_3[x_1, .., x_n]$ for all i. Pick sets, $S_1, .., S_t \subseteq [k]$ independently, u.a.r. and for all $j \in [t]$, let

$$q_j = \big(\sum_{i \in S_j} p_i\big)^2$$

and let

$$p_g = 1 - \prod_j (1 - q_j).$$

q_j is proper for all j (as $2^2 = 1$) and hence so is $\prod_j(1-q_j)$ and hence p_g is proper. If $b =$ the maximum degree of the p_j's, then $deg(q_j) \leq 2b$ and so $deg(p_g) \leq 2bt$. So, the degree of the polynomial increases by a factor of atmost $2t$ at every level. As there are d levels, $deg(p_o) \leq (2t)^d$ as required, where o is the output gate.

Note that naively defining $p_g := 1 - \prod_{i \leq k}(1-p_i)$ would have ensured that $p_g = 1 \iff \exists i, p_i = 1$ and so there is no error in this case. Defining $p_g := 1 - \prod_{i \leq k}(1-p_i)$, however only gives that $deg(p_g) \leq bk$. So the degree may increase by a factor of k at each level giving merely that $deg(p) \leq k^d$, which is bad as k (the fanin) is unbounded. While the naive choice would have ensured that the output polynomial, albeit of high degree, equals C_n at all input points, our choice of p is low-degree but disagrees with C_n at some points. We now show that the disagreement is small:

Consider the error at the OR gate g as above. Suppose that for all $i, p_i =$ the boolean function computed by g_i (i.e., all p_i are correct). If $p_i = 0$ for all i, then $q_j = 0$ for all j and hence $p_g = 0$, which is correct. Now, suppose that $p_l = 1$ for some l. An error occurs at $g \iff p_g = 0 \iff q_j = 0$ for all j.
Let's first compute

$$Pr_{S_j}(q_j = 0) = Pr_{S_j}[\big(\sum_{i \in S_j} p_i\big)^2 = 0]$$

given that $p_l = 1$ for some l. Let $T-$ be the set of subsets of $[k]$ not containing l and let $T+$ be the set of subsets of $[k]$ containing l. So $|T+| = |T-| = 2^{k-1}$. $S \mapsto \pi(S) := S \cup \{l\}$ is a bijection from $T-$ to $T+$. For $j \in \{0,1,2\}$, let $T-_j := \{S \in T- : \sum_{i \in S} p_i = j\}$ and $T+_j := \{S \in T+ : \sum_{i \in S} p_i = j\}$. Note the following

1. If $S \in T-_0$, then $\pi(S) \in T+_1$ and if $S \cup \{l\} \in T+_1$, then $\pi^{-1}(S \cup \{l\}) = S \in T-_0$. So, $\pi|_{T-_0}$ is a bijection between $T-_0$ and $T+_1$, showing that $|T-_0| = |T+_1|$.
   Similarly, we see that $|T-_1| = |T+_2|$ and $|T-_2| = |T+_0|$.

2. So, $|T-_0| \leq |T-| - |T-_2| = 2^{k-1} - |T+_0|$, giving that $|T+_0| + |T-_0| \leq 2^{k-1}$

3. $q_j = 0 \iff S_j \in T+_0 \cup T-_0$.

4. As S_j is picked u.a.r from $T+ \cup T-$, $Pr_{S_j}(q_j = 0) = |T+_0 \cup T-_0|/2^k \leq 1/2$.

As the S_j's are picked independently, $Pr(p_g$ makes an error$) = Pr(\forall j \in [t], q_j = 0) = \prod_i Pr(q_j = 0) \leq 2^{-t}$ where the Pr is over the random choice of the $S_1, .., S_t$. p_o is wrong $\iff$ one of the p_g's is wrong. So, by the union bound, $Pr(C \neq p_o) = Pr(\exists g \in C, p_g$ makes an error$) \leq |C|2^{-t}$.

In summary, we have shown that for a fixed input $a_1, .., a_n$, $Pr(p(a_1, .., a_n) \neq C(a_1, .., a_n)) \leq |C|2^{-t}$, where the probability is over the choice of the S_i's at all OR gates.

Fix some big enough R such that a random choice of the S_i's at all OR gates can be represented by a string in $\{0,1\}^R$. Denote by $p_{o,r}$, the p_o polynomial obtained when $r \in \{0,1\}^R$ is the random string used to pick the S_i's. Take a $2^n \times 2^R$ matrix, M the rows indexed by $\{0,1\}^n$, the inputs to C and the columns indexed by random strings, $\{0,1\}^R$. Let $M(x,r) = 1$ if $C(x) = p_{o,r}(x)$ and let $M(x,r) = 0$ otherwise. The fraction of zeros in the row indexed by x, equals $Pr_r(p_{o,r}(x) \neq C(x))$. So, by above, each row of M has atmost $|C|2^{R-t}$ zeros and so M has $\leq |C|2^{n+R-t}$ zeros. So, there must be a column, indexed by some r' which has $\leq |C|2^{n-t}$ zeros (If all columns have $> |C|2^{n-t}$ zeros, then M has $> |C|2^{R+n-t}$ zeros). So, $p_{o,r'}(x) \neq C(x)$ for $\leq |C|2^{n-t}$ many x's. This $p_{o,r'}$ is the required p. $\square$

Proof of the second lemma

Proof. Suppose $p \in \mathbb{F}_3[x_1,..,x_n]$ with $deg(p) \leq \sqrt{n}$ and let $S' \subseteq \{0,1\}^n$ be the set on which $p = PARITY_n$. Define a transformation $f : \{0,1\} \longrightarrow \{1,-1\}$ (mod 3) by $f(x_i) = 1 + x_i$ (mod 3). So, $f(0) = 1, f(1) = -1$. By abuse of notation, let f also be the transformation $f : \{0,1\}^n \longrightarrow \{1,-1\}^n$ (mod 3) obtained by applying f to each co-ordinate. So, the transformation changes $PARITY_n : \{0,1\}^n \longrightarrow \{0,1\}$ to another function $\{1,-1\}^n \longrightarrow \{1,-1\}$ (mod 3) which should be 1 when its input has even many -1's and should be -1 if its input has odd many -1's. We now describe this transformed function: Define the function $PARITY'_n : \{-1,1\}^n \longrightarrow \{-1,1\}$ (mod 3) by

$$PARITY'_n(y_1,..,y_n) = \prod_i y_i.$$

Then,

$$PARITY_n(x_1,..,x_n) = 1 \iff PARITY'_n(f(x_1,..,x_n)) = -1,$$

and so,

$$f(PARITY_n(x_1,..,x_n)) = PARITY'_n(f(x_1,..x_n)).$$

So, $PARITY'_n$ is the required transformed function. Let p' be the transformed version of p. I.e., p is a polynomial such that $f(p(\overline{x})) = p'(f(\overline{x}))$. So, for any $\overline{x} \in S'$,

$$PARITY'_n(f(\overline{x})) = f(PARITY_n(\overline{x})) = f(p(\overline{x})) = p'(f(\overline{x})),$$

where the second equality follows by the the choice of S'. I.e., $f(S') := S$ is the set on which p' agrees with $PARITY'_n$. $|S'| = |S|$ as f is a bijection.
Let $\mathcal{F}$ be the class of functions from S to $\mathbb{F}_3$ (So, $|\mathcal{F}| = 3^{|S|}$). To any $f \in \mathcal{F}$, we associate a polynomial $p_f \in \mathbb{F}_3[x_1,..,x_n]$ defined on S as follows :

$$p_f = g_f|_S,$$

where

$$g_f(x_1,..,x_n) = \sum_{\overline{y} \in S} f(\overline{y}) \prod_i (y_i x_i + 1).$$

Note that (letting n be even) $f(\overline{x}) = p_f(\overline{x})$ for all $\overline{x} \in S$. As $-1^2 = 1 = 1^2$, we may assume that p_f is multilinear by replacing any x_i^2 by 1. Let T be any monomial in p_f with degree $> n/2 + \sqrt{n}$. As $PARITY'_n = p'$ on S, for all $(x_1,..,x_n) \in S$, we have that

$$p'(x_1,..,x_n) = PARITY'_n(x_1,..,x_n) = \prod_{i \in [n]} x_i = \prod_{i:x_i \in T} x_i \prod_{i:x_i \notin T} x_i.$$

Multiplying both sides by $\prod_{i:x_i \notin T} x_i$,

$$p'(x_1,..,x_n) \prod_{i:x_i \notin T} x_i = \prod_{i:x_i \in T} x_i.$$

Now, $deg(p') = deg(p) \leq \sqrt{n}$. As T has $> n/2 + \sqrt{n}$ many variables (by multilinearity), there are $\leq n - (n/2 + \sqrt{n}) = n/2 - \sqrt{n}$ many x_i's in the product on the LHS. So, the LHS has degree $\leq deg(p') + n/2 - \sqrt{n} \leq n/2$. The monomial T is a scalar multiple of the RHS ($\prod_{i:x_i \in T} x_i$). By replacing this by the LHS, the degree of T can be reduced to $\leq n/2$. Doing this for all monomials T with $deg(T) > n/2 + \sqrt{n}$ transforms p_f into a polynomial with degree $\leq n/2 + \sqrt{n}$.
In summary, for any $f \in \mathcal{F}$, there is a polynomial, $p_f \in \mathbb{F}_3[x_1, .., x_n]$ defined on S with degree $\leq n/2 + \sqrt{n}$. There are atmost

$$m := \sum_{i=0}^{n/2+\sqrt{n}} \binom{n}{i} \leq (49/50)2^n/\sqrt{n}$$

many monomials on n variables of degree $\leq n/2 + \sqrt{n}$. So, there are 3^m polynomials defined on S in $\mathbb{F}_3[x_1, .., x_n]$ with degree atmost $n/2 + \sqrt{n}$. So, $3^{|S|} = |\mathcal{F}| \leq 3^m$ and so $|S| \leq m \leq (49/50)2^n/\sqrt{n}$. So, p disagrees with PARITY$_n$ on $2^n - |S| \geq 2^n - 2^n/\sqrt{n} \geq 2^n/\sqrt{n}$. □

16 Lecture 21

Scribe: Rohan Goyal, BMC202151

We now analyze the gap amplification under σ' in G'.

A verifier does the following test via an ASRW.

16.1 Idea:

1. Do an ASRW from an arbitrary vertex. Say the path is from A to B.
2. If for any (u, v) on the path, if $dist_G(a, u) \leq t$, $dist_G(v, b) \leq t$ and $\sigma' : V \mapsto \Sigma'$, we have $(\sigma'(a)_u, \sigma'(b)_v)$ falsifies (u, v) in the original constraint, we reject.
3. Else, accept.

σ' best assignment, extract an assignment σ for the older graph. $\sigma(v) =$ From v, do a BSRW, conditioning on the fact that BSRW stops within t steps.

This gives a distribution on the vertices reachable within t steps. Let this distribution be $\{P_{v,w}\}$.

$$\forall a \in \Sigma : P_a = \sum_{(\sigma'(w))_v = a} P_{v,w}$$

Assign $\sigma(v) = a$ s.t P_a is maximized.

16.2 Faulty Edges:

Let F be the set of falsified edges under σ. $\implies gap \leq \frac{|F|}{|E|}$.

Faulty Edges: In verifier's ASRW walk an edge $(u, v) \in E$ is faulty:

1. $(u, v) \in F$.
2. $d(a, u) \leq t \wedge (\sigma'(a))_u = \sigma_u$
3. $d(b, v) \leq t \wedge (\sigma'(b))_v = \sigma_v$

Let N be the random variable that counts the number of faulty steps in the ASRW, then:

$$gap' \geq Pr[N > 0] \geq \frac{\mathbb{E}[N]^2}{\mathbb{E}[N^2]}$$

This is true since the verifier rejects if $N > 0$ and for the second part we use Second Moment Method. Now, we try to get bounds on $\mathbb{E}[N]$ and $\mathbb{E}[N^2]$.

16.3 Analysis of $\mathbb{E}[N]$

Lemma 8. $\mathbb{E}[N] \geq \frac{t|F|}{8|\Sigma|^2|E|}$

Proof. Let $N_{u,v}$ be the expected value of (u, v) being faulty in the given walk.

$$N = \sum_{(u,v)\in F} N_{u,v} \implies \mathbb{E}[N] = \sum_{(u,v)\in F} \mathbb{E}[N_{u,v}]$$

Thus, it is enough to show that

$$\mathbb{E}[N_{u,v}] \geq \frac{t}{8|\Sigma|^2|E|}$$

Now,

$$\begin{aligned} Pr[d(a,u) \leq t \wedge \sigma'(a)_u = \sigma(u) | k\ u \to v \text{ steps.}] &= Pr[d(a,u) \leq t \wedge \sigma'(a)_u = \sigma(u)] \\ &\geq \left(1 - \frac{1}{t}\right)^t Pr[\sigma'(a)_u = \sigma(u) | d(a,u) \leq t] \\ &\geq \frac{1}{2|\Sigma|} \end{aligned}$$

$$\begin{aligned} \mathbb{E}[N_{u,v}] &= \sum kPr[k\ u \to v \text{ steps}] \cdot Pr[u \mapsto v \text{ step is faulty} | k\ u \to v \text{ steps}] \\ &= \sum kPr[k\ u \to v \text{ steps}] \cdot Pr[d(a,u) \leq t \wedge (\sigma'(a)_u = \sigma(u)) \\ &\qquad \wedge (\sigma'(b)_v = \sigma(v)) \wedge d(v,b) \leq t | k\ u \to v \text{ steps}] \\ &\geq \left(\frac{1}{2|\Sigma|}\right)^2 \sum kPr[k\ u \to v \text{ steps}] \\ &\geq \frac{t}{8|\Sigma|^2|E|} \end{aligned}$$

□

c

17 Lecture 22

Scribe: Rohan Goyal, BMC202151

We now analyze $\mathbb{E}[N^2]$. For that we begin with the following lemma:

Lemma 9. Let $G = (n, d, \gamma)$ expander where λ is $d\lambda'$ where λ' is the spectral expansion of the expander. Now, given a set $F \subset E$, and the condition that the zeroth step of a random walk starts from a random edge $\in F$, the probability that the random walk visits an edge in F in the t^{th} step is $\leq \frac{|F|}{|E|} + \left(\frac{\lambda}{d}\right)^{t-1}$

Proof. Let x be the probability distribution on the vertices of G of the walk starting. Let x_v be the probability that the walk starts at v. Now, if A is the normalized adjacency matrix of G and y is the probability distribution of taking an edge in F in the tth step. Now, $y_w = \frac{2|F|x_w}{d}$ as x_w is the number of edges in F adjacent to w divided by $2|F|$ but y_w is the same value divided by d.

We have that the requisite probability is

$$P = \left\langle A^{t-1}x, y \right\rangle = \frac{2|F|}{d} < A^{t-1}x, x >$$

We can now let $x = x^{\parallel} + x^{\perp}$ where $x^{\parallel}$ is the component of x along the uniform vector and $x^{\perp}$ is the component perpendicular to the uniform vector.

Thus,

$$P = \frac{2|F|}{d}\left(\left\langle x^{\parallel}, x^{\parallel}\right\rangle + \left\langle A^{t-1}x^{\perp}, x^{\perp}\right\rangle\right) \leq \frac{2|F|}{dn} + \frac{2|F|}{d}\left(\frac{\lambda}{d}\right)^{t-1}\left\langle x^{\perp}, x^{\perp}\right\rangle \leq \frac{|F|}{|E|} + \frac{2|F|}{d}\left(\frac{\lambda}{d}\right)^{t-1}\langle x, x\rangle$$

Thus, we just want $\langle x, x\rangle \leq \frac{d}{2|F|}$. Now, $\langle x, x\rangle \leq \max_v x_v(\sum x_v) \leq \max_v x_v \leq \frac{d}{2|F|}$ if all its neighbours are in F.

□

Now, let χ_i be the $0-1$ random variable which is 1 iff the ith step of the random walk is faulty.

$$\begin{aligned}
\mathbb{E}[N^2] &= \mathbb{E}[(\sum_{i=1}^{t}\chi_i)^2] \\
&\leq 2\sum_{i\geq 1} Pr[\chi_i = 1]\left(\sum_{j\geq i}\chi_j = 1|\chi_i = 1\right) \\
&\leq 2\sum_{i\geq 1} Pr[\chi_i = 1]\left(1 + \sum_{j\geq 1}^{\infty}\left(\frac{|F|}{|E|} + \left(\frac{\lambda}{d}\right)^{j-1}\right)(1 - \frac{1}{t})^j\right) \\
&\leq O(1)\mathbb{E}[N](1 + \frac{|F|t}{|E|}) \\
&\leq O(1)\mathbb{E}[N]
\end{aligned}$$

since we can assume $t|F| \leq |E|$ as we will work with t large constant and we would have amplified gap to a constant already.

Thus, we get $gap' \geq Pr[N > 0] \geq \frac{\mathbb{E}[N]^2}{\mathbb{E}[N^2]} \geq O(1)\mathbb{E}[N] \geq O(\frac{t|F|}{|\Sigma|^2|E|})$