CS 0368-4246: Combinatorial Methods in Algorithms (Spring 2025) May 19, 2025

Lecture 8: Expanders and Tanner Codes

Instructor: *Or Zamir* Scribes: *Elad Almogi*

1 Introduction

The following lecture is about expander graphs. We will go over 4 subjects:

1. Definitions and examples.
2. Properties of expander graphs.
3. Uses of expander graphs in Computer Science.
4. Using expander graph for error correction codes.

2 Expander Graphs

Expander Graphs are, intuitively:

- "Very Connected"
- Sparse
- (For CS uses) Has an explicit construction

We will go over a few different definitions for expander graphs:

Definition 1. *For every $A, B \in V$ it holds that $|E(A,B)| \approx |A| \cdot |B| \cdot \frac{m}{\binom{n}{2}}$.*

Definition 2. ***Edge Expansion***
For every $S \subseteq V$, $|S| \leq \frac{n}{2}$:

$$|\partial S| := |E(S, S^c)| \geq \underbrace{\varphi}_{expansion\ constant} \cdot |S|$$

Definition 3. ***Vertex Expansion***
For every $S \subseteq V$, $|S| \leq \frac{n}{2}$:

$$\underbrace{|N(S)|}_{Neighbors\ of\ S\ and\ not\ in\ S} \geq \varphi \cdot |S|$$

Definition 4. ***Conductance***
For this definition we will define first the Volume of $S \subseteq V$ as:

$$Vol_G(S) = \sum_{v \in S} deg_G(v)$$

Notice that for a d-regular graph G $Vol_G(S) = d \cdot |S|$ for every $S \subseteq V$.

We will define the conductance of the a cut S, S^c as:

$$\phi_G(S) = \frac{|E(S, S^c)|}{min\{Vol_G(S), Vol_G(S^c)\}}$$

G is defined expander if for every $S \subseteq V$:

$$\phi_G(S) \geq \varphi$$

There is a spectral definition of expander graphs (using terms on the eigenvalues of the neighbors matrix). We will elaborate about this definition in future lectures.

3 Examples of Expander Graphs (or not?)

1. **The Path $\mathbf{P_n}$** - not an expander. Consider choosing a subset $S \subset V$ that contains $\frac{n}{2}$ adjacent vertices. The cut (S, S^c) consists of 1 or 2 edges. Thus, the expansion rate is $O(\frac{1}{n})$.

2. **The cycle $\mathbf{C_n}$** - Not an expander, similarly to P_n.

3. **A Star Shaped Graph $\mathbf{K_{1,n-1}}$** - Depends on the definition:
   The graph is an expander in the edge expansion or conductance definitions with $\varphi = 1$.
   However, it is not a vertex expander: the set of all peripheral vertices expands only to the center vertex.

4. **$\mathbf{K_n}$** - An expander graph with conductance $\geq \frac{1}{2}$.

5. **Random Graph G** - Should be expander.

For conclusion, graphs that have "lightweight" cut are presumably not expanders.

4 Connectivity - Dealing With Edge Deletion

Problem 5. *Let G a graph that represents a network. We would like to know which vertex is connected to which vertex. We could use BFS or DFS to map the connected components.*
Now, let us consider the event that the edges $F \subseteq E$ erased. We would like to know if it is still possible to reach from one vertex to another.

Our goal - to show that expanders can deal with the problem of erasing edges. Note that if $\partial\{u\} \subseteq F$, then u is completely disconnected from the graph. This can occur in a sparse or d-regular graphs.

Lemma 6. *Let G be an expander with conductance φ and $F \subseteq E$. We will mark $C_1, C_2, \ldots, C_r$ the connected components of $G - F$ for which:*

$$Vol_G(C_i) \leq \frac{1}{2} \cdot Vol_G(V)$$

(Notice there is only 1 component that doesn't hold this term). Thus, we get that:

$$\sum_{i=1}^{r} Vol_G(C_i) = O(\frac{|F|}{\varphi})$$

Proof.

$$\sum_{i=1}^{r} \varphi \cdot Vol_G(C_i) \underset{(*)}{\leq} \sum_{i=1}^{r} E(C_i, C_i^c) \underset{(**)}{\leq} 2|F|$$

$(*)$ - $\phi_G(S) \geq \varphi$ and because $Vol_G(C_i) \leq \frac{1}{2} \cdot Vol_G(V)$ thus $min\{Vol_G(C_i), Vol_G(C_i^c)\} = min\{Vol_G(C_i)$
$(**)$ - We created the components $C_1, \ldots, C_r$ through deletion of at most $|F|$ edges (some of them might be in any one of the components). We count every edge we delete twice.

We get that:

$$\sum_{i=1}^{n} Vol_G(C_i) \leq \frac{2|F|}{r \cdot \varphi} = O(\frac{|F|}{\varphi})$$

□

Claim 7. *Given a φ-expander G, $F \subseteq V$ and $u, v \in V$, it is possible to check if v and u are connected in $G - F$ in $O(\frac{|F|}{\varphi})$ time.*

Proof. We can run BFS from u and from v separately until we see $\frac{2|F|}{\varphi}$ edges. If the BFS stops before exploring $\frac{2|F|}{\varphi}$ edges, the vertex is in one of the "small" components $C_i, \ldots, C_r$. Each of them has at most $\frac{2|F|}{\varphi}$ edges - thus we've seen every vertex in the component. Otherwise, if both BFS stopped because of the limit we've set, both are in the one "big" component. □

Theorem 8. *G is a φ-expander, thus:*

$$Diameter(G) = O(\frac{log\ n}{\varphi})$$

Notice in a sparse graph the best result we can achieve is $O(log\ n)$ because the number of P_k-s from a vertex $v \leq \Delta^k$ ($\Delta = \underset{v \in V}{max}\ deg_G(v)$). Thus, the diameter can't be less than $O(log\ n)$ because paths of this length don't even reach every vertex in the graph (It could reach only $\Delta^k < \Delta^{O(log\ n)} \approx n$ vertices) - thus the diameter is larger.

Proof. Let us mark $B(v, r)$ as the set of vertices that are at-most r edges from v. Notice that:

$$Vol_G(B(v,r)) \leq \frac{1}{2} Vol_G(V) \rightarrow |E(B(v,r), B(v,r)^c)| \geq \varphi \cdot Vol_G(B(v,r))$$

Thus:

$$Vol_G(B(v, r+1)) \geq (\underset{*}{1} + \underset{**}{\varphi}) \cdot Vol_G(B(v,r))$$

$*$ - The volume of the vertices before adding those $r+1$ edges from v. $**$ - Every edge in the cut $(B(v,r), B(v,r)^c$ were added in $B(v, r+1)$ and is being counted in the volume of the newly added vertices.

Therefore, we can use this term recursively and get that for each r that holds $Vol_G(B(v,r)) \leq \frac{1}{2}Vol_g(V)$ it holds that:

$$Vol_G(B(v,r)) \geq (1+\varphi)^r$$

The largest r for which this term still hold would be $\lceil log_{1+\varphi}(n+1) \rceil$ which is $O(\frac{log\ n}{\varphi})$:

$$log_{1+\varphi}\ n = \frac{log\ n}{log(1+\varphi)} \underset{1+x \leq e^x}{\leq} \frac{log\ n}{\varphi}$$

Therefore, $B(v, \frac{100\ log\ n}{\varphi})$ and $B(u, \frac{100\ log\ n}{\varphi})$ should have common vertices (Both reached more than half the graph) and thus there is a path at length $O(\frac{log\ n}{\varphi})$ □

5 Expander Construction

Definition 9. *Bipartite graph G is a $(n, m, d, \gamma, \alpha$-expander if the following holds:*
1. It has 2 disjoint sets in sizes n and m (Let us mark these sets as L, R and w.l.o.g $|L| = n \geq m = |R|$).
2. For every $v \in L$: $deg(v) = d$.
3. Every set $S \subseteq L$, if $|S| \leq \gamma \cdot n$ then $|N(S)| \geq \alpha \cdot |S|$.

Theorem 10. *For every $n \geq m$ and $\epsilon > 0$ there are $\gamma \geq 0$ and $d \in \mathbb{N}$ s.t there exists $(n, m, d, \gamma, (1-\epsilon)d)$-expander. And explicitly, there is one with:*

$$\gamma = \Theta(\frac{\epsilon \cdot m}{d \cdot n}),\ d = \Theta(\frac{log\frac{n}{m}}{\epsilon})$$

Proof. We will show a sketch for a proof that random graph will hold these term "in a good probability" (and that it is indeed exist). We won't see any explicit constructions for now (maybe in future lectures). For the ease of the proof, let us assume that $m = \Theta(n)$. We'd like to take a particular $S \subseteq L$ and show that the probability that it won't expand is very low, then we could use union-bound on every selection of $S \subseteq L$.

How could we bound the probability that S doesn't expand?
We will enumerate S vertices as $v_1, v_2, \ldots, v_{|S|}$. In each step, we randomly select d neighbors for the i-th vertex of S in R. Notice that for every i:

$$N(\{v_1, v_2, \ldots, v_{i-1}\}) \leq d \cdot |S| \leq \gamma \cdot n \cdot d \underset{\gamma\ definition}{\leq} \frac{\epsilon \cdot m}{1000}$$

Therefore, the set of vertices in R that are connected to S until the i-th vertex $\leq \frac{\epsilon \cdot m}{1000}$. Thus we could hope that v_i would add at least $(1-\epsilon) \cdot d$ new neighbors.

What is the probability that v_i will randomly select at least $\frac{\epsilon}{2} \cdot d$ neighbors that we've already seen? from union bound:

$$\leq \binom{d}{\frac{\epsilon}{2} \cdot d} \cdot (\frac{\epsilon}{1000})^{\frac{\epsilon}{2} \cdot d} \underset{(*)}{\leq} (\frac{e \cdot d}{\frac{\epsilon}{2} \cdot d})^{\frac{\epsilon}{2} \cdot d} \cdot (\frac{\epsilon}{1000})^{\frac{\epsilon}{2} \cdot d} \leq (\frac{2e}{\epsilon})^{\frac{\epsilon}{2} \cdot d} \cdot (\frac{\epsilon}{1000})^{\frac{\epsilon}{2} \cdot d} \leq (\frac{1}{10})^{\epsilon \cdot d}$$

$(*)$ - Using Stirling approximation we can prove that:

$$\binom{n}{k} \leq \frac{n^k}{k!} \underset{Stirling}{\leq} \frac{n^k}{(\frac{k}{e})^k} = (\frac{en}{k})^k$$

For this sketch we can assume $d = \Theta(\frac{log\frac{1}{\epsilon}}{\epsilon})$. Thus, the probability above equals:

$$(\frac{1}{10})^{\epsilon \cdot d} \leq (\frac{1}{10})^{\epsilon \cdot 100 \frac{log\frac{1}{\epsilon}}{\epsilon}} = (\frac{1}{10})^{100 \cdot log\frac{1}{\epsilon}} < (\frac{1}{e})^{100 \cdot log\frac{1}{\epsilon}} = \frac{1}{e^{100 \cdot log\frac{1}{\epsilon}}} = \epsilon^{100}$$

Notice that this probability is i.i.d for each vertex in S. Thus, the number of vertices that chose more than $\frac{\epsilon}{2}$ joint neighbors are $Bin(|S|, \epsilon^{100})$. Using Chernoff's bound we can see that:

$$Pr[Bin(|S|, \epsilon^{100}) \geq \underbrace{\frac{\epsilon}{2} \cdot |s|}_{(1+(\frac{\epsilon}{2\epsilon^{100}}-1)) \cdot \epsilon^{100} \cdot |S|}] \leq e^{-10 \cdot \epsilon \cdot |S|}$$

If this tiny probability didn't apply, we got that S expanded, because there at most $\frac{\epsilon}{2} \cdot |S|$ vertices that might have all joint neighbors, and $(1 - \frac{\epsilon}{2}) \cdot |S|$ that might have at most $\frac{\epsilon}{2} \cdot d \cdot |S|$ joint neighbors each (Thus each has at least $(1 - (1 - \frac{\epsilon}{2})) \cdot d$ new neighbors. Indeed we get that:

$$N(S) \geq ((1 - \frac{\epsilon}{2}) \cdot d) \cdot (1 - \frac{\epsilon}{2}) \cdot |S| + d \cdot (\frac{\epsilon}{2}) \cdot |S| \geq (1 - \epsilon) \cdot d \cdot |S|$$

We will use Union Bound on the choosing of S. Let us assume $|S| = k$:

$$\binom{n}{k} \cdot e^{-10\epsilon k} \leq (\frac{en}{k})^k \cdot e^{-10\epsilon k} \underset{(*)}{\approx} (\frac{en}{\gamma \cdot n})^{\gamma \cdot n} \cdot e^{-10\epsilon \cdot \gamma \cdot n}$$

$(*)$ - The expression grows as k does. Thus, choosing the largest k possible will give as the order of the size of the expression.

We can choose γ as small as needed for the probability's bound above will be as small as needed. Thus, we found γ and d for them we could construct $(n, m, d, \gamma, \alpha$-expander. □

6 Error Correction Codes (Using Expanders)

Error correction code is a function that codes binary string of length k to a binary string of length n that is resistant to certain precentage of errors ($\frac{\delta}{2}$).

Parameters:

- **Rate:** $\frac{k}{n}$
- **Distance:** δ.

Remember we can fix up to half the distance between two different words.

We are familiar with construction (not explicit ones) for "good" codes (for general n but constant Rate and Distance). For example random code or random linear code (Choosing $k \times n$ matrix that will act as the function mentioned above).

Problem 11. *While coding or decoding can be done relatively fast, even in random linear code, fixing error would take exponential time.*

Our goal: Construct a "good" code that can fix errors "fast".

7 Tanner's Code (1981) [1]

We will construct a linear code that will be able to fix errors relatively fast. Let us use bipartite expander as above with $(n, \frac{n}{2})$ sizes of each side and $\epsilon \leq \frac{1}{4}$. The code will be the linear subspace of $\mathbb{F}_2^n$ that derives from assigning binary values in L (marked as x_u) s.t for every $v \in R$:

$$\bigoplus_{u \in N(v)} x_u = 0$$

(The right side acts as parity check)

The rate of the code: The block length is n, and the length of the message would be the dimension of the valid codewords which is $\underset{(*)}{\geq} n - m = \frac{n}{2}$

$(*)$ - Each linear constraint might reduce the dimension by one. Thus, the rate is $\frac{1}{2} = \Omega(1)$.

Lemma 12. *For every $S \subseteq L$, $|S| \leq \gamma \cdot n$:*

$$|U(S)| \geq (1 - 2\epsilon) \cdot d|S|$$

When $U(S) \subseteq N(S)$ is the group of neighbors of S in R that are neighbors of only single vertex in S.

Proof. If there are more than $\epsilon \cdot d|S|$ neighbors of S that appear at least twice, $|N(S)| < (1 - \epsilon) \cdot d|S|$ and that's in contradiction the assumption we have on the expander. □

Claim 13. *Distance* $= \Omega(1)$

Proof. Let us assume in contradiction that there are 2 different codewords with distance $\leq \gamma \cdot n$. Notice that the subtraction of the two words is a valid codeword S (because they lie in the subspace representing the code), which is not the 0 codeword and has weight $\leq \gamma \cdot n$. From the lemma, we know that in those $\gamma \cdot n$ vertices in L (corresponding with S) there is at least one neighbor $v \in R$ that is connected only to one vertex from L that is 1 (from the $\gamma \cdot n$ vertices that do). Thus, this codeword is not valid, because

$$\bigoplus_{u \in N(v)} x_u \neq 0$$

Thus, distance $\geq \gamma \cdot n$ (Often considered γ).
It is possible to prove that distance $\geq 2(1 - \epsilon) \cdot \gamma$. □

Theorem 14. *It is possible to fix up to $(1 - w\epsilon) \cdot \gamma$ errors in $O(n)$.*

The following algorithm should be able to achieve the theorem:

1. We will put the noised codeword in L.

2. While there is $u \in L$ s.t most constraints in R that it's connected to are violated - we will flip its' bit.

Let's prove why is that correct.

Claim 15. *If the distance of the noised codeword from a valid codeword is $\leq (1-2\epsilon)\cdot\gamma$ thus there is a vertex in L that over half of his neighbors in R represent violated constraints.*

Proof. Let us mark the set of the vertices in L for which we don't agree with the nearest valid codeword as S. We know that $|S| \leq (1-2\epsilon)\cdot\gamma$. As we've seen in "Lemma 12", S has at least $(1-2\epsilon)\cdot d\cdot|S|$ unique neighbors. Every one of them is violated constraint (Because it agrees on the rest of the vertices but only disagree with the single vertex in S). Let's take the vertex in S that is the most unique across these $(1-2\epsilon)\cdot d\cdot|S|$ constraints. It should take part in at least $(1-2\epsilon)\cdot d$ constraints. Notice we've chosen $\epsilon < \frac{1}{4}$ thus $(1-2\epsilon)\cdot d\cdot|S| > \frac{d}{2}$. Therefore, for this perticular vertex most of the constraints are indeed violated. □

Claim 16. *In every flip we reduce the number of violated constraints.*

Observation 17. *Notice that because we flip only vertices that more than half of their constraints are violated - when we flip it we reduce the number of violated constraints.*

Conclusion 18. *After at most $m = \frac{n}{2}$ we will reach a valid codeword.*

Claim 19. *During the process, we will never reach distance $> (1-\epsilon)\cdot\gamma\cdot n$ from the original codeword.*

Proof. If the distance of a noised codeword from a codeword is *distance* $= D \leq \gamma\cdot n$, the number of violated constraints is at least the number of unique neighbors of the subtraction from the noised codeword, which is $\geq (1-2\epsilon)\cdot D\cdot d$ (according to Lemma 12).

Therefore, during the process we remind near the original codeword, otherwise the number of violated constraints would rise (and we showed it's only reducing). □

Claim 20. *The process takes $O(n)$ or $\tilde{O}(n)$*

Proof. We've done at most $\frac{n}{2}$ iteration of going through n vertices and check $O(1)$ neighbors (to check the constraints). Thus - $O(n^2)$.
However, it's easy to improve this to $O(n)$ by saving every vertex in L in a list according to the number of violated constraints it has. Every time we will flip a bit we'll update the number of constraints (for those in L that are connected to the constraints that have changed). This would take us $O(1)$ or at most $\tilde{O}(1)$.

Therefore, we can conclude that this proccess can be done in $O(n)$ or $\tilde{O}(n)$. □

References

[1] R. Tanner. A recursive approach to low complexity codes. *IEEE Transactions on Information Theory*, 27(5):533–547, 1981.