Lecture 6: Expander Codes: Tanner Codes

Anup Rao

October 14, 2019

We have been working with expander graphs in order to design efficient codes. Last time we saw the spectral definition of expanders and proved the expander mixing lemma:

Lemma 1 (Bipartite Expander mixing)**.** *Suppose G is a D-regular bipartite graph with N vertices on the left and right, $N \times N$ symmetric adjacency matrix A, and with second largest magnitude eigenvalue λ. Suppose S, T are subset of vertices. If e denotes the number of edges between S and T, then*

$$\left| e - \frac{D \cdot |S| \cdot |T|}{N} \right| \leq \lambda \sqrt{|S| \cdot |T|}.$$

To see that every set in such a bipartite graph expands, suppose S is a subset of the vertices on the left, of size at most αN, and T is its neighborhood. Then we have

$$\begin{aligned} \lambda \sqrt{|S| \cdot |T|} &\geq D \cdot |S| - D \cdot |S| \cdot \frac{|T|}{N} \\ &\geq D \cdot |S| - D \cdot |S| \cdot \alpha D \\ \frac{|T|}{|S|} &\geq \left(\frac{(1 - \alpha D) D}{\lambda} \right)^2. \end{aligned}$$

If α is very small (say less than $(100D)^{-1}$), this gives expansion close to $(D/\lambda)^2$. The best expanders have $\lambda \approx 2\sqrt{D}$, which gives expansion close to $D/4$. There is a better argument that shows that the expansion is closer to $D/2$ if λ is close to $2\sqrt{D}$. Nevertheless, this is not strong enough to apply the ideas we saw from the last lecture to get codes — we cannot guarantee that most neighbors of the set are unique neighbors.

See `https://www.cs.princeton.edu/~zdvir/expanders/Kahale.pdf`.

Explicit spectral expanders

There are very simple constructions of explicit expanders. For p a large prime, define a bipartite graph whose vertices correspond to $\mathbb{F}_p$. Connect x to $x+1$ and $x-1$ for all $x \in \mathbb{F}_p$. Connect 0 on the left to 0 on the right. Connect every non-zero element x to x^{-1}. This is a 3-regular expander with $\lambda < 3$, where λ does not depend on p.

Here is another construction. If $p > 2$ is a large prime number, we can define a 4-regular bipartite graph whose vertices correspond to all invertible 2×2 matrices over $\mathbb{F}_p$. There are $\Theta(p^4)$ such matrices.

To count the number of such matrices, choose a non-zero vector for the first row, and then a linearly independently non-zero vector for the second row.

Connect two vertices A, B if and only if either AB^{-1} or BA^{-1} lies in the set

$$\left\{ \begin{bmatrix} 1 & \pm(p-1)/2 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ \pm(p-1)/2 & 1 \end{bmatrix} \right\}.$$

Then this graph is a 4-regular expander with $\lambda < 4$, and λ does not depend on p.

See https://www.cs.princeton.edu/~zdvir/expanders/amir-3steps.pdf for some expander construcitons.

You can always square the matrix defining any graph to obtain the graph A^2 which is a D^2-regular graph with magnitude of second largest eigenvalue λ^2. This has the effect of squaring the ratio λ/D, so this ratio can be made arbitrarily close to 0. In this way, we can construct expanders with stronger and stronger expansion, by increasing the degree.

Codes from spectral expanders

NOW, LET US RETURN to the project of constructing efficient codes over small alphabet using these very explicit expanders that we discussed in the last section. Key to the proof here will be the strong bounds given by the expander mixing lemma — not only do these expanders expand, but the number of edges between sets is what you expect.

Start with a symmetric D-regular bipartite graph with N vertices on each side of the partition that is an expander with eigenvalue of second largest magnitude λ. For example, you could use one of the graphs given by powering the adjacency matrices of the explicit constructions given in the last sections. This might create a bipartite graph with multiedges, but that is not going to be a problem for any of the ideas we discuss here.

Let $C' \subseteq \mathbb{F}^D$ be an arbitrary linear code of dimension k', and relative distance δ'. We obtain a new code $C \subseteq \mathbb{F}^{ND}$ by viewing every vector in F^{ND} as an assignment of a vector of length 2 to the edges of the bipartite graph. The subspace C corresponds exactly to the set of vectors where every vertex on the left and the right of the graph sees D symbols that correspond to a codeword of C'.

This kind of code is called a *Tanner* code. The analysis we give here is due to Sipser, Spielman and Zemor.

Let us start by computing the rate of the code. We have an ND dimensional space, and we have added $2N(D-k')$ linear constraints. So, the rate of the code is at least

$$\frac{ND - 2N(D-k')}{ND} = 2(k'/D) - 1,$$

which is positive as long as the rate of the original code is bigger than $1/2$.

To get a code with rate bigger than $1/2$, we will have to use a finite field with more than 2 elements.

What about the distance? Since this is a linear code, it is enough to give a lower bound on the weight of a non-zero codeword. Consider

any non-zero codeword. This corresponds to some set of edges E. Let S be the set of vertices on the left that are incident to E in the bipartite graph and T be the set of vertices that are incident to E on the right. Since this is a codeword, every element of S and T must touch δD edges in E. Thus, the number of edges going from S to T in the bipartite graph is at least $|E| \geq \delta D|S| \geq \delta D\sqrt{|S||T|}$.

On the other hand, the expander mixing lemma says:

$$\delta D\sqrt{|S||T|} \leq |E| \leq \frac{D|S||T|}{N} + \lambda\sqrt{|S||T|},$$

which implies

$$\sqrt{|S||T|} \geq (\delta - \lambda/D)N, \tag{1}$$

and so $|E| \geq \delta(\delta - \lambda/D)ND$. So, the relative distance of the code is at least $\delta(\delta - \lambda/D)$.

Finally, let us discuss a decoding algorithm. In each step, look at all the vertices on the left of the bipartite graph and decode the edges there to codewords. Repeat the same operation from the perspective of the vertices on the right.

To analyze this process, suppose the code C' can recover from ϵ fraction errors. We shall prove that the decoding can recover from $0.9\epsilon(\epsilon - \lambda/D)$ fraction of errors. Let S denote the set of vertices on the left that are incident to an error after the first left-decoding step, and let T be the set of edges on the right that are incident to an error after the next right-decoding step. Let E be the set of edges that correspond to an error in between the two decoding processes.

Since the decoding in C' fails only if the number of errors exceeds ϵ, we get

$$|S| \cdot \epsilon D \leq 0.9 \cdot \epsilon(\epsilon - \lambda/D)ND,$$

so

$$|S| \leq 0.9 \cdot (\epsilon - \lambda/D)N.$$

Similarly, we have

$$|T| \cdot \epsilon D \leq |E|.$$

Applying the expander mixing lemma and the AM-GM inequality, we conclude

$$|T| \cdot \epsilon D \leq \frac{D|S||T|}{N} + \lambda\sqrt{|S||T|} \leq \frac{D|S||T|}{N} + \lambda\frac{|S| + |T|}{2}.$$

Plugging in the bound on $|S|$ from above, we get

$$|T| \cdot \epsilon D \leq |T| \cdot D \cdot 0.9 \cdot (\epsilon - \lambda/D) + \lambda\frac{|S| + |T|}{2},$$

which after rearranging gives

$$|T| \leq |S| \cdot \frac{\lambda}{0.2\epsilon D + 0.8\lambda} = \alpha|S|,$$

for some constant $\alpha < 1$, as long as $\lambda < \epsilon D$. This means that in each iteration, the set of vertices that see errors decreases geometrically. So in $O(\log N)$ steps, the algorithm terminates. When it terminates, there must be 0 errors. This gives an algorithm that runs in time $O(N \log N)$. In fact, there is a more clever linear time algorithm.