AN INTRODUCTION TO EXPANDERS

JAEHOON KIM

ABSTRACT. In this note, we learn basics on expander graphs. The materials on this note are based on the survey [1] by Hoory, Linial and Wigderson.

1. Motivating examples

We will learn about expander graphs. Expander graphs has several definitions.

(1) Combinatorial aspects: Any vertex set U has comparatively large boundary ($|E(U,\overline{U})|$ or $N_G(U)$).
(2) Probabilistic aspects: Random walk mixes fast, i.e. k-th vertex on the random walk is almost as if we choose a vertex at random.
(3) Algebraic aspects: All eigenvalues except the first one are small.

We plan to learn about these aspects and how they are related. We will also see some examples where expanders are useful, and we will see how one can construct such expander graphs. First, we see some simple examples showing why such graphs can be useful.

Imagine that Alice and Bob are far way and wants to communicate messages. Suppose Alice has a message of k bits which she want to send to Bob over some communication channel. However, as there can be noises in the channel which can disrupt the message. If we assume that at most a p-fraction of the bits can be altered, how can we make sure that Alice and Bob can still communicate without ambiguity? If such communication is possible, how short can Alice make the bits she send?

How can one solve this problem? What Shannon suggested is to consider 0/1 sequence of length n for each 0/1 sequence of length k where two length n sequences are difficult to confuse each other. To be more precise, consider a collection $\mathcal{C} \subseteq \{0,1\}^n$ of size $|\mathcal{C}| = 2^k$ (which is called a *code*) and a bijective map $\phi : \{0,1\}^k \to \mathcal{C}$.

If Bob receives an altered string $y \in \{0,1\}^n$ then finds a code $z \in \mathcal{C}$ which is closest to y, and decode this to $\phi^{-1}(y)$. If the Hamming-distance between any two code in $\mathcal{C}$ is larger than $2pn$, then Bob always correctly decode the message into the original message. Hence, now the question becomes how large code $\mathcal{C}$ can we find while ensuring this pairwise distance condition.

Definition 1.1. *Let $\mathcal{C} \subseteq \{0,1\}^n$ be a code. Its rate is defined to be $\frac{1}{n}\log|\mathcal{C}|$ and normalized distance is defined to be $\frac{1}{n}\min_{c\neq c'} d_H(c,c')$, where $d_H(c,c')$ measures the Hamming distance between c and c' that is the number of coordinates in which c and c' differ.*

How can one find a good code? In other words, can one always construct a code with rate and normalized distance at least some absolute constant?

The answer is yes. We see one way of constructing such an example, which was discovered by Sipser and Spielman in 1996.

We first consider a graph satisfying the following property.

Definition 1.2 (Magical graph). *Let G be a bipartite graph with vertex partition (L,R). We say that G is an $(n,m;d)$-magical graph if $|L| = n$, $|R| = m$ and every vertex $v \in L$ satisfies $d_G(v) = d$ where the following two properties hold.*

Date: July 26, 2021.

(M) If $S \subseteq L$ with $|S| < \frac{n}{10d}$, then $|N(S)| \geq \frac{5d}{8}|S|$.

We will later show that such a graph actually exists. Assuming the existence of such a graph, we can observe one good property of such a graph.

In an $(n, 3n/4, d)$-magical graph G with $n > 20d$, for each nonempty $S \subseteq L$ with $|S| \leq \frac{n}{10d}$, there exists a vertex $u \in R$ having exactly one neighbor in S. (1.1)

Indeed, the edge set $E(G[S, N(S)])$ contains $d|S|$ edges while $|N(S)| \geq \frac{5d}{8}|S|$ by (M). Hence, the vertices in $N(S)$ has $8/5 < 2$ vertices on average. On the other hand, every vertex in $N(S)$ has at least one neighbor in S. Thus (1.1) holds.

Claim 1. *There exists a code $\mathcal{C} \subseteq \{0,1\}^n$ with rate at least $1/4$ and distance at least $\frac{1}{10d}$.*

Proof. Consider a $(n, 3n/4, d)$-magical graph G. Let A be the 0/1-matrix with row set R and column set L where $a_{r\ell} = 1$ if and only if r and ℓ are adjacent in G.

Let $\mathcal{C} = \{x \in \{0,1\}^n : Ax = 0\}$ be our code. As the rank of A is at most $3n/4$, the right kernel $\mathcal{C}$ of A has dimension at least $n/4$, hence $|\mathcal{C}| \geq 2^{n/4}$.

Now we prove a lower bound on the distance. As $\mathcal{C}$ is a linear space, $x, y \in \mathcal{C}$ implies that $x - y \in \mathcal{C}$. Hence the distance of the code is same as the minimum number of 1s in a nonzero code in $\mathcal{C}$. Let $x \in \mathcal{C}$ be a non-zero vector in $\mathcal{C}$ with the support $S = \{j \in L : x_j = 1\}$. If $|S| < \frac{n}{10d}$, then (1.1) implies that there exists $r \in R$ with $|N(r) \cap S| = 1$. This implies that r-th coordinate of Ax is 1, hence x is not in the right kernel of A, which is a contradiction to the definition of $\mathcal{C}$. Hence, the normalized distance of $\mathcal{C}$ is at least $1/(10d)$. □

This construction is a special case of Low Density Parity Check (LDPC) code. Explicitly constructing expander to make the above argument feasible was a difficult task. Such an expander is called 'lossless expander'.

Consider one more motivating example. Imagine you want to check whether a given number is prime or not. One easy way is when a number m is given, see if each integer from 1 to $\sqrt{m}$ divides m. However, as this takes a long time, one can choose some random number y and see if y divides m, and conclude from this. Of course such an algorithm can give a wrong answer. However, if you repeat this long enough, then the probability of getting right answer is getting sufficiently high.

Let's consider the following more general scenario. Suppose that there is a collection $\mathcal{L} \subseteq \{0,1\}^k$ of strings, which is called a language. Assume that there exists a polynomial algorithm that calculates a function $f(x, r)$ for given $x \in \{0,1\}^k$ and $r \in \{0,1\}^k$ satisfying the following:

(1) $f(x, r) = 1$ if $x \in \mathcal{L}$,
(2) If $x \notin \mathcal{L}$, then $|\{r \in \{0,1\}^k : f(x, r) = 1\}| \leq \frac{1}{16} \cdot 2^k$.

In other words, if we choose r uniformly at random from $\{0,1\}^k$, then for given $x \notin \mathcal{L}$, we have $f(x, r) = 0$ with probability at least $15/16$. In short, in order to check whether a given $x \in \{0,1\}^k$ belongs to $\mathcal{L}$, we can choose a random $r \in \{0,1\}^k$ and compute $f(x, r)$. It will give you correct answer with probability at least $1 - 1/16$.

One can consider x as a binary representation of given number m and r as a binary representation of the random number y. The choice $1/16$ is rather arbitrary here.

Our aim here is to find a way to use this f to obtain an algorithm with a probability better than $1 - 1/16$. Obvious way is to repeat this d times, then our success probability will become at least $1 - (1/16)^d$. However, one drawback of this is that 'randomness' is very expensive concept.

How do we generate a random number? One can see a digital clock at the moment and see the time. If it's 5.12323204839.. minutes after 4 o'clock, then you take the 11th number after the decimal point. Ok, suppose that's a random number. Then how do we generate the second random number independent from the first one? If we use a clock again, then would it be independent from the first choice? If we instead use a thermometer and choose a number from it, then would it be independent of the first random number? There are many complications on this. Hence, taking many numbers independently at random is a pricy task. So, people want to design an algorithm which uses as small amount of randomness as possible.

We measure 'randomness' by the number of bits on the random number we use. In the above simple algorithm, we choose $r \in \{0,1\}^k$ in random, using k-bits of randomness.

Consider the following algorithm. For given integer d, fix an $(n,n;d)$-magical graph G with bipartition (L,R) and $n = 2^k$. Associate strings in $\{0,1\}^k$ with the vertices in L and the vertices in R in a bijective way.

Now, for given x, choose a random k-bits $r \in \{0,1\}$. Let $v_r \in L$ be the vertex associated with r, and consider $u_{a_1}, \dots, u_{a_d} \in N_G(v_r)$ be the vertices in L which are associated with $a_1, \dots, a_d \in \{0,1\}^k$. Now the algorithm outputs 1 if and only if $f(x,a_1) = \cdots = f(x,a_d) = 1$.

Let $B = \{a \in \{0,1\} : f(x,a) = 1\}$. The algorithm gives a wrong answer only when $x \notin \mathcal{L}$ and $r_1, \dots, r_d \in B$. Let $A \subseteq L$ be the set of vertices that makes the algorithm to give a wrong answer, i.e. $A = \{v \in L : N_G(v) \subseteq B\}$. Note that $N(A) \subseteq B$. If $|A| > \frac{n}{10d}$, then we have

$$|B| \geq |N(A)| > \frac{5d}{8}(\frac{n}{10d}) \geq \frac{n}{16},$$

a contradiction to the assumption that $|B| < \frac{n}{16}$. Hence $|A| \leq \frac{n}{10d}$, so the algorithm succeeds with probability at least $1 - 1/(10d)$. By using no additional randomness, we boosted our success probability from $1 - 1/16$ to $1 - 1/(10d)$.

However, for usage of this algorithm, we need to construct such a magical graph in an efficient way. Also, for given a vertex of G, we need to generate its neighbors in efficient way.

Also, by using d dependent samples, we achieve the error probability to $O(1/d)$. However, this is much worse that simple repetition error bound of $2^{-O(d)}$). Later we will see that by using a bit more randomness, one can achieve such an exponential decay on the error probability.

One thing that remains is to show that there exists such a magical graph. This can be shown by taking a random graph.

Lemma 1.3. *There exists n_0 such that for every $d \geq 32$ and $n \geq n_0$, $m \geq 3n/4$, there exists an $(n,m;d)$-magical graph.*

Proof. Let L be n vertices and R be m vertices. Let G be a random bipartite graph where each vertex in L chooses d vertices in R uniformly at random and connects. (We allow multi-edges)

We first show that (M) holds with high probability. Let $S \subseteq L$ be a set of size $s \leq \frac{n}{10d}$, and let $T \subseteq R$ be a set of size $t < \frac{5ds}{8}$. Let $X_{S,T}$ be an indicator random variable for the event that all the edges from S to go T. If $\sum X_{S,T} = 0$ then (M) holds. The probability of the event $X_{S,T}$ is $(t/m)^{sd}$, so

$$\mathbb{P}[\sum_{S,T} X_{S,T} > 0] \leq \sum_{s=1}^{n/(10d)} \binom{n}{s}\binom{m}{5ds/8}(\frac{5ds}{8m})^{sd} \leq \sum_{s=1}^{n/(10d)} (\frac{ne}{s})^s(\frac{8me}{5ds})^{5ds/8}(\frac{5ds}{8m})^{sd} < \frac{1}{10}.$$

Hence there exists $(n,m;d)$-magical graph. □

Note that this shows that most of the graphs are magical. However, explicitly finding one such graph is not easy. For example, in order for the previous application, one should be able to deterministically (and efficiently) find the neighbors of a given vertex. Hence, this probabilistic argument shows existence, but is not useful for the application.