Spectral Graph Theory and its Applications Lecture 10

Expander Codes

Lecturer: Daniel A. Spielman November 7, 2004

10.1 Expander Codes

In this lecture, I will present two ways to construct error-correcting codes from expander graphs. I will begin with a brief explanation of one of the goals of coding theory.

10.2 Coding Theory in 5 Minutes

Error-correcting codes enable one to store or transmit a message in such a way that one can later recover the message even if it is partially corrupted. We will examine a commonly used abstraction of this problem. Our messages will be vectors in $\{0,1\}^d$, and we will encode a message through an encoding function $C : \{0,1\}^d \to \{0,1\}^n$, where $n > d$ is called the *block length* of the code. For $x \in \{0,1\}^d$, we call $C(x)$ a codeword. The first important parameter of the code that we will identify is its rate $r = d/n$, which measures how many bits we transmit for each bit of information communicated. Codes with higher rates are preferred.

The decoder is presented with a corrupted version of $C(x)$, $C(x)+e$, where e is an error vector and addition is modulo 2. We will describe decoding algorithms as functions $D : \{0,1\}^n \to \{0,1\}^d \cup ?$. We say that a decoding algorithm D δ-decodes a code C if for all e with at most δn ones,

$$D(C(x)+e) = x.$$

That is, the decoding algorithm can recover the original message x even if it has been subjected to δn errors. In this class, we will see families of codes in which r and δ remain constant as n goes to infinity. Moreover, D can be realized by a linear-time algorithm.

Note that this property of D implies that for all $x^1 \neq x^2 \in \{0,1\}^d$, $C(x^1)$ and $C(x^2)$ must differ in at least $2\delta n$ coordinates. This is a difficult condition to achieve on its own.

Before I proceed with the rest of the lecture, I should make one disclaimer. This abstraction is mathematically appealing and served the coding community well for over 30 years. However, it eventually slowed progress by distracting researchers from the fundamental problem of communication under random noise. I was part of a revolution in which we rejected this model. But, in respect for the material I want to present today, I'll save my criticisms of this model for after class.

10.3 Codes from high expansion

Let's assume that we have a bipartite graph (V, W, E) with a set V of n vertices on one side and a set W of $n/2$ vertices on the other in which

a. all the vertices in V have degree d,

b. all the vertices in W have degree $2d$,

c. there exists an $\alpha > 0$ such that for every set $S \subset V$ such that $|S| < \alpha n$, $|N(S)| > (d/2)\,|S|$.

I will now explain how to construct an error-correcting code from this graph. Instead of describing the encoding function, I will describe the set of codewords. We'll figure out how to encode later.

The codewords will be vectors $y \in \{0,1\}^n$. For each vertex $w \in W$, we will impose the following constraint on the codewords:

$$\sum_{(v,w)\in E} y_v = 0.$$

To make this make sense, I should identify the vertices V with the numbers $1, \ldots, n$. So, a vector $y \in \{0,1\}^n$ corresponds to having a 0 or 1 at each vertex of V. The constraint says that for each $w \in W$, the sum modulo 2 of the bits at its neighbors must be 0. As we have imposed $n/2$ constraints on an n-dimensional vector space, we know that the dimension of the space that satisfies these constraints is at least $n/2$. Thus, there are at least $2^{n/2}$ codewords satisfying these constraints. As the codewords form a linear space, it is possible to encode them by a matrix multiplication, and we will let C be a map realizing this encoding. We have $k \geq n/2$, and so the rate of this code is at least 1/2.

I will now prove that all codewords differ in at least αn coordinates.

Lemma 10.3.1. *For all distinct x^1 and x^2 in $\{0,1\}^k$, $C(x^1)$ and $C(x^2)$ differ in at least αn coordinates.*

Proof. Let $y^1 = C(x^1)$ and $y^2 = C(x^2)$. As y^1 and y^2 are codewords, $y^1 + y^2$ is as well. As $y^1 + y^2$ is 1 where y^1 and y^2 differ, Thus, it suffices to prove that no codeword has fewer than αn ones.

Assume by way of contradiction that there exists a codeword y with fewer than αn ones, and let S be the set of indices of these ones. By our expansion assumption, the set S has more than $(d/2)\,|S|$ neighbors. As each vertex in S only has d edges, this implies that some vertex $w \in W$ has just one edge to a vertex in S. But then, the sum of the bits on the neighbors of w is 1, and so its constraint is not satisfied, a contradiction. $\square$

10.4 Decoding

I will now explain how to decode these codes, by assuming just a little more expansion. In particular, we will assume that for all sets $S \subset V$ of size at most αn, $|N(S)| > (3d/4)\,|S|$. In this case, I will describe a simple $\alpha/2$-decoding algorithm. For sufficiently large d, a random graph satisfies this condition.

To describe the algorithm, I will refer to each vertex in W as a check. I will say that a check is *satisfied* if the sum of its neighbors is 0 modulo 2, and I will say that it is *unsatisfied* otherwise. Consider what happens to the checks if I flip the value of y_v: each check w that is a neighbor of v that was satisfied becomes unsatisfied, and *vice versa*. Also notice that all checks are satisfied if and only if y is a codeword. This suggests the following decoding algorithm.

> If there is a vertex v such that most of its neighbors are unsatsified, flip $y(v)$. Repeat.

The one thing we know about this algorithm is that it decreases the number of unsatisfied constraints. However, we do not know if it is correcting an error when it flips a bit or introducing a new one.

We will now show that this algorithm can remove up to $\alpha n/2$ errors. We first observe that it terminates.

Lemma 10.4.1. *The algorithm terminates after at most $n/2$ iterations.*

Proof. At each iteration, the number of unsatisfied checks decreases. □

The key to our analysis is the following lemma:

Lemma 10.4.2. *If the algorithm is applied to a vector of the form $C(x) + e$, where $e_v = 1$ for $v \in S$ and $|S| \leq \alpha n$, then there are more than $(d/2)\,|S|$ unsatisfied checks.*

Proof. Let $U \subseteq W$ denote the unsatisfied checks, and let $R \subseteq W$ denote the satisfied checks that are neighbors of S. Let $u = |U|$, $r = |R|$ and $s = |S|$. From our expansion assumption, we know

$$u + r > (3d/4)s.$$

As each vertex in u connects to at least one edge from S, each vertex in r connects to at least two edges from S, and there are ds edges attached to S,

$$u + 2r \leq ds.$$

Subtracting the first inequality from the second we find

$$r < (d/4)s,$$

and so

$$u > (d/2)s,$$

as desired. □

This lemma tells us that if $|S| \leq \alpha n$, then there exists a vertex $v \int V$ most of whose neighbors are unsatisfied.

Lemma 10.4.3. *If $y = C(x) + e$, where $e_v = 1$ for $v \in S$ and $1 \leq |S| \leq \alpha n$, then there is an unsatisfied check.*

Proof. If each vertex $v \in V$ had at most $(d/2)$ unsatisfied neighbors, then there would be at most $(d/2)\,|S|$ unsatisfied checks, contradicting Lemma 10.4.2. $\square$

Finally, we observe that if the algorithm is fed an input with at most $\alpha n/2$ errors, then at no point will the algorithm produce a vector with more than αn errors.

Lemma 10.4.4. *Let y^0 be the input to the algorithm, and let y^t be the vector present at time t. If there exists an x such that $y^0 = C(x) + e^0$, where e^0 is the characteristic vector of a set S^0 satisfying $\left|S^0\right| \leq \alpha n/2$, $y^t = C(x) + e^t$ where e^t is the characteristic vector of a set of size at most αn.*

Proof. Let u^t be the number of unsatisfied checks at time t. We have $u^0 > u^1 > u^2 > \cdots > u^t$. As $\left|S^0\right| \leq \alpha n/2$, $u^0 \leq (d/2)\alpha n$. If we had $\left|S^t\right| = \alpha n$, then Lemma 10.4.2 would imply $u^t > (d/2)\alpha n$, a contradiction. $\square$

Together, these lemmas imply

Theorem 10.4.5. *If $y = C(x) + e$, where $e_v = 1$ for $v \in S$ and $|S| \leq \alpha n/2$, then on input y the decoding algorithm outputs $C(x)$.*

Proof. By Lemma 10.4.4, we know that at each time t, $\left|S^t\right| \leq (d/2)\alpha n$. By Lemma 10.4.3, this implies that there exists a vertex most of whose neighbors are unsatisfied, unless $\left|S^t\right| = 0$. However, when the algorithm terminates there is no such vertex. We may conclude that the algorithm terminates with $\left|S^t\right| = 0$. $\square$