What's the big deal with Expanders?

Ian Chen

June 22, 2026

1 Introduction

Expanders seem to show up everywhere in TCS, or even outside. They have a simple definition — a sequence of graphs G_n is an expander family if they have spectral gap bounded away from 0. These notes will describe the many properties of such families and isolate how they are natural constructions in algorithmic design.

Expanders are common – cliques are always expanders, and Erdös-Renyi graphs tend to be too. Miraculously, *constant degree* expanders exist for essentially every size n and d. We will take their existence for granted here, but their constructions show many connections between algebra, probability, and combinatorics. Ramanujan graphs conjecture, ... Products of graphs, ...,

2 Graph Invariants

We have seen in the previous notes that the spectral gap directly bounds the conductance, via Cheeger's inequality, and so in an expander family, there are no small conductance subsets. In particular, this means that random walks converge quickly, in logarithmic time! Whenever we have a Monte-Carlo algorithm, we would love if it was running on an expander, because then efficient algorithms can run on even exponentially sized chains.

For example, we will demonstrate how this bounds the diameter. Consider the lazy walk matrix $W = \frac{1}{2}(I + AD^{-1})$, i.e. taking a step on the undirected graph with probability $1/2$, and with probability $1/2$ staying in the same place. The diameter is the first t such that $(W^t)_{uv} > 0$ for all $u, v \in V$.

To bound this, fix a vertex u, with $\pi^{(0)} = \mathcal{X}_u$. We want to find t such that $\pi_v^{(t)} = W^t \pi_v^{(0)} > 0$. Of course, as $t \to \infty$, then $\pi^{(t)} \to \frac{1}{n}$, the stationary distribution (ergodic Markov chain). In the intermediate, we can write the spectral decomposition of W, i.e.

$$W = \sum_{i=1}^{n} \mu_i v_i v_i^T$$

for $1 = \mu_1 > \mu_2 \geq \ldots \geq \mu_n = 0$ the eigenvalues ($\mu_2 < 1$ by connectivity, and $\mu_n \geq 0$ by Gershgorin), and v_i the respective orthonormal eigenvectors (with $v_1 = \pi^*$ stationary). Finally,

$$\|\pi^{(t)} - \pi^*\|_1 \leq \|\sum_{i=2}^{n} c_i \mu_i^t v_i\|_1 \leq \sqrt{\|\mathbf{c}\|_2 \times \|\sum_{i=2}^{n} \mu_i^t v_i\|_2} \leq \sqrt{n}(1-\beta)^t$$

for $\beta = \max_j(1-\mu_j)$ the spectral gap. Moreover, this shows that each coordinate is bounded by β^t individually. Thus, as we need to hit a limit of $\left|\pi^{(t)}(v) - \pi^*(v)\right| < 1/n$, then $(1-\beta)^t < e^{-\beta t} \leq 1/n$ for $t = \frac{\log n}{\beta}$, our bound.

To prove a stronger bound, we note that if we find *any* polynomial of degree d such that $p(W)_{uv} > 0$, then that bounds the diameter — we don't have to restrict ourselves to monomials! In fact, a sufficient condition for such a polynomial

is for $p(\mu_1) = p(1) = 1$, and $|p(\mu_i)| < \frac{1}{n}$ for $i \geq 2$. Writing the spectral decomposition once again,

$$e_a^T p(M) e_b = \sum_{i=1}^{n} p(\mu_i) v_i(a) v_i(b) = (1)\frac{1}{\sqrt{n}}\frac{1}{\sqrt{n}} + \sum_{i=2}^{n} p(M) v_i(a) v_i(b)$$
$$\geq \frac{1}{n} - \sum_{i=2}^{n} |p(\mu_i)||v_i(a)||v_i(b)| \geq \frac{1}{n} - \max_{i \geq 2} |p(\mu_i)| > 0$$

again using Cauchy-Schwartz in the penultimate step. We won't go into the details here, but Chebyschev's equi-oscillatory polynomials are useful for constructing a polynomial of degree $(1 + \frac{1}{\sqrt{2\beta}}) \ln(2n)$, a tighter bound. Of course, iterating powers of this polynomial rapidly increases the mixing of random walks.

We extend these results to arbitrary (non-regular) graphs by studying the symmetrized adjacency

$$\tilde{A} = D^{-1/2} A D^{-1/2} = D^{-1/2} W D^{1/2}$$

which is diagonally similar to the random walk matrix (i.e. has the same spectral gap). Properly relating eigenvectors, then $\|\pi^{(t)} - \pi^*\|_2 \leq (1-\beta)^t \sqrt{\frac{\max_x d(x)}{\min_y d(y)}}$, and $\mathrm{diam}(G) \leq \left(1 + \frac{1}{\sqrt{2\beta}}\right) \log \frac{4m}{\min_x d(x)}$ (same bound on d-regular).

3 Network Design

3.1 Nonblocking Flows

One measure of expanders being highly connected is that there exist many vertex disjoint paths. Here, we exploit this to construct a network with n left terminals, n right terminals, such that there exist a vertex disjoint paths from i to $f(i)$, for any $f : [\![n]\!] \to [\![n]\!]$ injective. We will find such a network with bounded degree and $\tilde{O}(n)$ size, and moreover show how non-blocking flows can be efficiently found.

The particular combinatorial expander we need is for $N(S)$ to be larger than S. In particular, a left d-regular bipartite graph $G = (L \cup R, E)$ is an (α, β) expander if for any $S \subset L$ of size $\alpha|L| = \alpha|R| = \alpha n$, then $|N(S)| > \beta|S|$. A d-multibutterfly network is constructed out of these expanders by having n vertices on each of $\log n$ layers, such that layer ℓ is divided into $n2^{-\ell}$ evenly-sized blocks, with each block on layer ℓ connected to 2 blocks on layer $\ell + 1$ via an (α, β) expander. Indeed, nodes are of bounded degree, with a unique path from each $\ell \mapsto r$.

Moreover, this expansion property means that we can route any permutation of αn inputs to αn outputs. In particular, as there are $\leq \alpha n$ that needs to be routed between blocks, a matching exists via Hall's theorem, i.e. a vertex disjoint path. Stacking $\frac{1}{\alpha}$ of these networks allows us to connect any n inputs to n outputs.

Finally, there exists a simple distributed routing algorithm to find these matchings from S to T, where each $t \in T$ is matched with some $s \in S$ if s is the only unmatched in $N(t)$. Indeed, write $A \subset T$ the matched vertices, so

$$|A| \geq \beta|S| - |N(S) \setminus A| \geq \beta|S| - \frac{d|S| - |A|}{2} \implies |A| \geq (2\beta - d)|S|$$

by bounding $|N(S)|$ by $d|S|$, and $|N(S) \setminus A|$ with degree at least 2. Therefore, the number of unmatched S decreases geometrically, with rate $1 - \frac{2\beta - d}{d} < 1$, as long as $\beta > \frac{d}{2}$.

3.2 Sorting Networks

Sorting networks are a different algorithmic model, often developed for *oblivious* algorithms, i.e. their memory accesses are independent of the input (or more precisely, of previous computations). Many developments in modern oblivious RAM or oblivious data structures use these as building blocks. Given an array, a *comparator* takes in $(a, b) \mapsto (\min(a, b), \max(a, b))$. There exist deterministic sorting networks with $O(n \log n)$ such comparators ar-

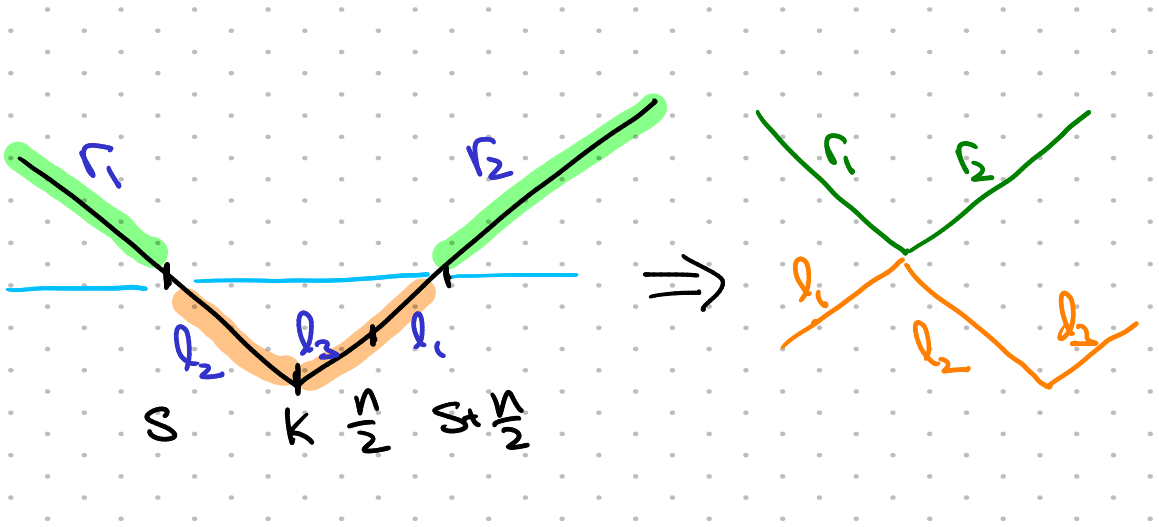

Figure 1: **Diagrammatic proof of lemma 1:** s divides the sequence into two halves of size $n/2$, which get reassembled in $L = \ell_1 \circ \ell_2 \circ \ell_3$ and $R = r_1 \circ r_2$ respectively. The case $s \geq n/2$ is similar.

ranged in $O(\log n)$ depth, that sorts arrays of length n. These are all variants of an approach by AKS (CITE), but won't describe their full construction here.

First, a preliminary is bitonic sort, with $O(n \log^2 n)$ work and $O(\log^2 n)$ depth. It makes the observation that a simple circuit can sort bitonic sequences, i.e. which are rotationally similar to a V-shape ($\exists k$ s.t. decreasing until k, and increasing after index k):

1. Compute $L = (\min(a_0, a_m), \min(a_1, a_{m+1}), \ldots, \min(a_{m-1}, a_{2m-1})$
2. Compute $R = (\max(a_0, a_m), \max(a_1, a_{m+1}), \ldots, \max(a_{m-1}, a_{2m-1})$
3. Bitonic sort L and R, then concatenate

Lemma 1. *L and R are both bitonic, and* $\max(L) \leq \min(R)$.

Proof. WLOG (by rotating and reversing), say the sequence is a V and $k \leq n/2$. Then, find s that splits the array into an upper and lower parts of size $n/2$. The sequences L and R are bitonic, as shown in figure 1. □

Finally, Batcher sort is similar to mergesort, using this $O(n \log n)$ work and $O(\log n)$ depth bitonic sort as a stand-in for merge. In particular, in the j-th round (starting from $j = 0$, we form blocks of size 2^j, which inductively are sorted. Then, in chunks of 2 blocks, we form bitonic sequence by $B_1 \circ \text{rev}(B_2)$, and sort it.

The barrier to getting our desired $O(n \log n)$and $O(\log n)$ work/depth bounds is the halver: it simply cannot be made into constant depth. The breakthrough is thus to build an ϵ-approximate halver, i.e. given $|L| = |R| = n$, rearranges the elements such that

- For $k \leq n$, at most ϵk of the largest k elements in $L \cup R$ are in L
- For $k \leq n$, at most ϵk of the smallest k elements in $L \cup R$ are in R

Suppose we are given a d-regular bipartite graph $G = (L \cup R, E)$, such that $|N(S)| \geq \min(|S| \frac{1-\epsilon}{\epsilon}, (1-\epsilon)n)$. Then, by Hall's theorem, it is a union of d matchings $M_1 \cup \ldots M_d$. In round i, we apply the comparator to each endpoint

$e \in M_i$ (in parallel), moving the smallest to L and the larger to R, using nd comparisons over depth d. Finally, for S the misclassified elements among the largest k in $A \cup B$, if $S > \epsilon k$, then $|S| + |N(S)| > \epsilon k + (1-\epsilon)k = k$. However, as each $s \in S$ can only have been compared against elements larger than it, in order to be misclassified, contradiction. The case S being the misclassified smallest elements is similar.

The AKS and Zig-Zag sorting networks design sufficient fail-safes so that the misclassified elements eventually get shuffled to the right places. We won't describe this here, as they are involved and besides the point.

4 Randomness Reduction (PRNGs)

4.1 RP and BPP

RP is the class of randomized algorithms that have false negative rate (say) $1/2$, and 0 false positive rate. In particular, we will think of these as being given a seed of m random bits. To amplify the success probability to $1-\delta$, then we can run them using a fresh seeds, using $m \times \log_2 n$ random bits. We show how to use only $m + O(\log 1/\delta)$ bits.

We start with G, with 2^m vertices and of spectral gap γ. Each vertex is an m-length bitstring, and we generate a sequence $v_1, \ldots, v_t$ of pseudorandom bit-strings to feed into our algorithm. For RP, we can say it is a negative after all t instances of our algorithm returns negative, but if even one says positive, then we return positive.

We choose v_1 uniformly at random, and v_{i+1} via the undirected random walk. Let B be the set of vertices that fail, i.e. $|B| \leq \frac{1}{2} \times 2^m$, so that we need to analyze the probability that all $v_i \in B$. Namely, we can show that it is $\leq \gamma^t$, so setting $t = \log \frac{1}{\gamma} \log \frac{1}{\delta} = O(\log \frac{1}{\delta})$ wins, for expander G.

To prove this, let P project probability masses onto B, and W be the walk matrix, so that by Courant-Fisher,

$$\mathbf{Pr}(\text{success}) = \frac{\mathbf{1}^T (PWP)^t \mathbf{1}}{\mathbf{1}^T \mathbf{1}} \leq (\lambda_{\max}(PWP))^t$$

Bounding $\lambda_{\max}(W)$ by γ, with conjugation by P not increasing eigenvalues, then indeed we get the γ^t bound.

We can prove similar results for BPP, which now allow two-sided error, each with probability at most (say) $1/3$ (has to be less than $1/2$, to be amplified by the majority vote). We need the Hoeffding-like result:

Theorem 1. *For G with spectral gap $\gamma \geq 1-\epsilon$, $f : V \to [0,1]$, $\mu = \mathbf{E}_{v \sim \mathrm{Unif}(V)}(f(v))$, then our walk satisfies*

$$\mathbf{Pr}\left(\frac{1}{t} \sum_{i=1}^{t} f(v_i) \geq (1+\epsilon)\mu + \epsilon^2 \right) \leq \mathbf{exp}\left(\epsilon(1+\epsilon)(1-\epsilon t)\mu\right)$$

When $t \geq (1+\epsilon)/\epsilon^2$, this bound simplifies to $\mathbf{Pr}\big(\bar{f}(v_i) \geq (1+\epsilon)\mu\big) \leq O(e^{-\epsilon^2 t \mu})$, and so choosing t sufficiently large we get the amplification for BPP.

4.2 Connectivity

in which we discuss the deterministic $O(\log n)$-space algorithm to check if s and t are connected

5 Error Amplification

5.1 Constant Rate Codes

A left d-regular bipartite graph $G = (L \cup R, E)$ is an (α, β) expander if for any $S \subset L$ of size $\alpha|L| = \alpha|R| = \alpha n$, then $|N(S)| > \beta|S|$. These graphs do exist, namely choosing its d neighbors uniformly at random forms an $(\alpha, d-2)$

expander for some $\alpha(d) > 0$. Indeed, for any set $S \subset L$,

$$\mathbf{Pr}(|N(S)| \leq (d-2)|S|) \leq \underbrace{\binom{d|S|}{2|S|}}_{2|S| \text{ overlapping neighbors}} \underbrace{\left(\frac{d|S|}{n}\right)^{2|S|}}_{|N(S)| \leq d|S|}$$

so that using $\binom{a}{b} \leq \left(\frac{ae}{b}\right)^b$, then choosing $\alpha(d) = d^{-4}e^{-3}$, we obtain via union bound,

$$\mathbf{Pr}(\text{any } |S| = k \text{ fails}) \leq \binom{n}{k}\binom{kd}{2k}\left(\frac{kd}{n}\right)^{2k} \leq \left(\frac{n}{k}\right)^k e^k \left(\frac{d}{2}\right)^{2k} e^{2k}\left(\frac{kd}{n}\right)^{2k} = \left(\frac{d^4e^3}{4}\frac{k}{n}\right)^k \leq 4^{-k}$$

Finally, summing over all $k > 0$, we have at most $\frac{1}{1-(1/4)} - 1 = \frac{1}{3} < 1$ chance of failing.

Given these expanders exist, we can form codes out of them. Suppose we have 2^n symbols that we want to transmit over a noisy channel; we will encode these symbols into $(2n)$-bit strings such that they are robust to errors. This robustness will be measured by *gap*, i.e. if $\leq k$ bits flip, then we will decode to the correct codeword. Let G be a left d-regular $(\alpha, \frac{3d}{4})$ expander, with $|L| = 2n$ and $|R| = n$. Assigning each $v \in L$ to a bit b_v, we form 2^n codewords by enforcing n constraints, namely that $\sum_{v \in N(r)} b_v \equiv_2 0$, for each $r \in R$.

Theorem 2. *There exists a linear time decoder for the above with αn gap.*

Proof. To decode, find $v \in L$ with $> d/2$ unsatisfied neighbors (in R). Flip b_v, and repeat until no such more v.

First, note that this algorithm must terminate within n iterations. Indeed, let $\varphi(j)$ be the number of unsatisfied $r \in R$ at iteration j. Then, as $\varphi(0) \leq n$, then this algorithm converges within n steps (and also $O(n)$ time):

$$\varphi(j+1) = \varphi(j) - (\text{number of unsatisfied neighbors}) + (\text{number of satisfied neighbors}) < \varphi(j)$$

Now, we must argue that it has αn gap. Each codeword must differ by at least $2\alpha n$ bits, since if we let S be the 1-bits of $c_1 \oplus c_2$, and if $|S| \leq 2\alpha n$, then by our expansion property,

$$E(S, N(S)) \leq d|S|, \qquad E(N(S), S) > 2 \times \frac{3}{4}d|S| > d|S|$$

where the former is by left d-regular, and the latter by $N(S)$ must have an even number of neighbors to S (to fix parity). In general, there cannot be an $|N(S)| \geq \frac{d}{2}|S|$. Moreover, each codeword must differ by at least $4\alpha n$ bits, since if we split S into S_1 and S_2 each of size at most $2\alpha N$, then

(i) $|N(S_1) \cap N(S_2)| \leq \frac{d}{8}|S|$, then $|N(S_1) \setminus N(S_2)| > \left(\frac{3d}{4} - \frac{d}{4}\right)|S_1|$

(ii) $|N(S_1) \cap N(S_2)| \geq \frac{d}{8}|S|$, then $E(T, S) \geq 2|T| \geq 2\left(\frac{3d}{4} - \frac{d}{8}\right)|S| > d|S|$

Finally, we show that the decoding procedure converges to a codeword, and moreover if we start from $\leq \alpha n$ distance to c, then we stay $< 2\alpha n$ distance to c, hence converging to c by the triangle inequality. Indeed, if we have S the set of wrong bits, then ℓ, the number of satisfied bits in $N(S)$, is bounded by

$$d|S| = E(S, N(S)) > \underbrace{2\ell}_{\text{satisfied}} + \underbrace{(3d/4|S| - \ell)}_{\text{unsatisfied}} = \frac{3d}{4}|S| + \ell$$

and so $\ell < \frac{d}{4}|S|$, i.e. $\varphi(\cdot) > \frac{d}{2}|S|$ (assuming $|S| \leq 2\alpha n$). Thus, as long as there is an incorrect bit, then some neighbor will have $> d/2$ unsatisfied neighbors, and our decoding algorithm only stops at a valid codeword. Moreover, we cannot reach $2\alpha n$ distance from c, since we would have $> d\alpha n$ unsatisfied neighbors, more than $\varphi(0)$. □

5.2 PCP theorem

in which we discuss Dinur's proof of PCP theorem. CSPs and hardness of approximation, a second application of powering (skipping alphabet reduction)

6 Closing Remarks

Here, we have discussed many constructions based on various expansion properties. Edge expanders naturally have many matchings, leading to its uses in routing and low-depth circuits. The uniformly mixed distribution of edges lead to its error amplifying properties, and usage in codes. Moreover, random walks converge quickly in expanders, and Monte Carlo algorithms, even on exponentially sized graphs, run efficiently.

My impression is that expanders are so broadly applicable not for any individual property, but because miraculously, sparse (bounded degree) graphs exist with all such properties simultaneously. Their constructions are not simple — the AKS expander-based sorting network is called a galactic algorithm for a reason — but given they exist, they are a good first object to consider for any counter-examples or constructions.

7 Bibliographic Notes

Randomized algorithms courses. TCS toolkit. Avi's expander survey. The PCP theorem resources I've found.