Expander graphs and their applications

Shlomo Hoory, Nathan Linial, and Avi Wigderson

George Zirdelis

Structural Complexity 2013-14, CoReLab

July 27, 2014

Edge expansion and a combinatorial definition of expanders

- Let $G(V, E)$ be an **undirected** and d-**regular** graph
- Multiple edges and self loops are allowed
- We denote $n = |V|$
- We can think of each edge as a pair of directed edges. For $S, T \subset V$ we denote $E(S, T)$ the set of directed edges from S to T.
- $E(S, S) \equiv E(S)$.

We give a couple of basic definitions:

Definition

- The **Edge Boundary** of a set S is $\partial S = E(S, V\setminus S)$
- The (edge) **Expansion Ratio** of G, denoted by $h(G)$ is defined as:

$$h(G) = \min_{\{S \mid |S| \leq \frac{n}{2}\}} \frac{|\partial S|}{|S|}$$

Definition (Family of Expander Graphs)

A sequence of d-regular graphs $\{G_i\}_{i\in\mathbb{N}}$ of size increasing with i is a **Family of Expander Graphs** if there exists $\epsilon > 0$ s.t.

$$h(G_i) \geq \epsilon, \quad \text{for all i}$$

The exact determination of $h(G)$, given G, is difficult (co-NP Hard).

In computer science, we are concerned for the **explicit construction** of the objects we study.

Definition

Let $\{G_i\}_i$ be a family of expander graphs where G_i is a d-regular graph on n_i vertices and the integers $\{n_i\}$ are increasing, but not too fast (e.g. $n_i + 1 \leq n_i^2$ will do).

- The family is called **Mildly Explicit** if there is an algorithm that generates the j-th graph in the family G_j in time polynomial in j.
- The family is called **Very Explicit** if there is an algorithm that on input of:
  - an integer i
  - a vertex $v \in V(G_i)$
  - $k \in 1, \ldots, d$

  computes the k-th neighbor of the vertex v in the graph G_i.
  This algorithm's run time should be polynomial in its input length (the number of bits needed to express the triple (i, v, k)).

Examples of expander graphs

- **Very explicit**: A family of 8-regular graphs G_m for every integer m. The vertex set is $V_m = \mathbb{Z}_m \times \mathbb{Z}_m$. The 8 neighbors of the vertex (x, y) are (x+y, y), (x−y, y), (x, y+x), (x, y−x), (x+y+1, y), (x−y+1, y), (x, y+x+1), (x, y−x+1), (all operations are $\mod m$).
- **Mildy explicit**: A family of 3-regular graphs where $V_p = \mathbb{Z}_p$, p is prime and a vertex v is connected to $v + 1$, $v - 1$ and v^{-1}, (all operations are $\mod m$ and we define the inverse of 0 to be 0).
  This family is only mildy explicit, since we are at present unable to generate large primes deterministically.

Graph spectrum and an algebraic definition of expansion

- Let $A = A(G)$ be the **Adjacency Matrix** of a n-vertex graph G.
- Each (u, v) matrix entry is the number of edges between verices u and v
- A is real and symmetric. Real and symmetric matrices have real eigenvalues.
- We denote the n real eigenvalues as:

$$\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$$

- We ofter refer to the eigenvalues of $A(G)$ as the **Spectrum** of the graph G.

Eigevalues encode a lot of information of a d-regular graph:

- $\lambda_1 = d$
- The graph is connected iff $\lambda_1 > \lambda_2$
- The graph is bipartite iff $\lambda_1 = -\lambda_n$

We are more interested in λ_2 because it's closely related to the expansion parameter.

Theorem

Let G be d-regular graph with spectrum $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$

$$\frac{d - \lambda_2}{2} \leq h(G) \leq \sqrt{2d(d - \lambda_2)}$$

We see that $d - \lambda_2$, also known as **Spectral Gap**, provides an estimate on the expansion of a graph.

The Expander Mixing Lemma

The following lemma shows that a small second eigenvalue in a graph implies that its edges are "spread out", a hallmark of random graphs.

Lemma (Expander Mixing Lemma)

Let G be a d-regular graph with n vertices and set $\lambda = \lambda(G)$. Then for all $S, T \subseteq V$:

$$\left| |E(S,T)| - \frac{d}{n}|S|\,|T| \right| \leq \lambda\sqrt{|S|\,|T|}$$

where λ is the largest absolute value of an eigenvalue other than λ_1.

The left-hand side measures the deviation between two quantitiess

- One is $|E(S,T)|$, the number of edges between the two sets
- The other is the expected number of edges between S and T in a random graph of edge density $\frac{d}{n}$.

A small λ implies that this deviation is small, so the graph is nearly random is this sense.

Lemma (Converse of the Expander Mixing Lemma)

Let G be a d-regular graph with n vertices and suppose that the following holds for every two disjoint set $S, T \subseteq V$ and some positive ρ:

$$\left| |E(S,T)| - \frac{d}{n} |S| \, |T| \right| \leq \rho \sqrt{|S| \, |T|}$$

Then, $\lambda \leq O\left(\rho \cdot (1 + \log\left(\frac{d}{\rho}\right)\right)$. *The bound is tight.*

- For when d grows with n, i.e. the K_n graph, we have $d = n - 1$ and $\lambda = 1$.
- For the range we are interested in, i.e. $n \gg d$, is there a lower bound on λ? Yes.

Theorem

For every (n, d) graph (i.e. n vertices and d-regular),

$$\lambda \geq 2\sqrt{d-1} - o_n(1)$$

(n, d) graphs with small ration $\alpha = \frac{\lambda(G)}{d}$ have some significant properties, some of which we mention below:

- Independent Set has cardinality at most αn
- For the k-coloring problem, the chromatic number $\chi(G)$ is at least $\frac{1}{\alpha}$
- The diameter of G is $O(\log n)$

- A key property of the random walk on an expander graph is that it converges rapidly to its limit distribution.
- In many theoretical and practical computational problems in science and engineering it is necessary to draw samples from some distribution $\mathcal{F}$ on a (usually finite but huge) set V.
- Therefore consider a graph G on vertex set V so that the limit distribution of the random walk on G is $\mathcal{F}$.
- A clever choice of G can guarantee that
  - it is feasible to efficiently simulate this random walk
  - the distribution induced on V by the walk **converges rapidly** to $\mathcal{F}$

Rapid mixing of walks

- A **walk** on a graph $G(V, E)$ is a sequence of vertices $v_1, v_2, \ldots, \in V$ s.t. v_{i+1} is a neighbor of v_i for every vertex i
- So when v_{i+1} is selected uniformly at random for every i, this is called a **random walk** on G

- Now let's see more about the speed of convergence of probability distributions defined on V.
- We know that for a finite, connected, nonbipartite graph G, distributions defined on V converge to a limit or **stationary** distribution

- We denote a d-regular graph G, with n nodes where $|\lambda_2|, |\lambda_n| \leq \alpha d$ holds, as an (n, d, α)-graph.
- A vector $\mathbf{p} \in \mathbb{R}^n$ is called a **probability distribution vector** if its coordinates are nonnegative and $\sum_{i=1}^{n} p_i = 1$
- For the uniform distribution on $\{1, \ldots, n\}$ it's $\mathbf{u} = (1, \ldots, 1)/n$

Let's properly define what a random walk on a graph is.

Definition

A random walk on a finite graph $G = (V, E)$ is a discrete-time stochastic process $(X_0, X_1, \ldots)$ taking values in V. The vertex X_0 is sampled from some initial distribution on V , and X_{i+1} is chosen uniformly at random from the neighbors of X_i.

If G is d-regular with adjacency matrix A the its **normalized adjacency matrix** is defined as $\hat{A} = \frac{1}{d}A$. We summarize some facts for a random walk on G.

1. It's a Markov Chain with state set V and transition matrix $\hat{A}$
2. $\hat{A}$ is real, symmetric and doubly stohastic; i.e. every column and every row sums up to 1
3. if $\hat{\lambda_1} \geq \ldots \geq \hat{\lambda_n}$ are the eigenvalues of $\hat{A}$, then $\hat{\lambda_1} = 1$ and $max\left\{\left|\hat{\lambda_2}\right|, \left|\hat{\lambda_n}\right|\right\} \leq \alpha$
4. The corresponding eigenvectors are the same eigenvectors of A
5. Consider an experiment where we sample a vertex x from some probability distribution $\mathbf{p}$ on V and then move to a random neighbor of x. This is equivalent to sampling a vertex from the distribution $\hat{A}\mathbf{p}$.
6. The matrix $\hat{A}^t$ is the transition matrix of the Markov Chain defined by random walks of length t. In other words $(\hat{A}^t)_{ij}$ is the probability a random walk starting at i is at j after t steps.
7. The stationary distribution of the random walk on G is the uniform distribution, namely, $\mathbf{u}\hat{A} = \hat{A}\mathbf{u} = \mathbf{u}$. (This uses the symmetry of A.)

Convergence in the ℓ_1 and ℓ_2 norms

Theorem

Let G be an (n, d, α)-graph with normalized adjacency matrix $\hat{A}$. Then for every distribution vector $\mathbf{p}$ and any positive integer t

$$\left\| \hat{A}^t \mathbf{p} - \mathbf{u} \right\|_1 \leq \sqrt{n} \alpha^t$$

Why we use ℓ_1 instead of ℓ_∞? For probabilities it holds that

$$max_B \left| Pr_p[B] - Pr_q[B] \right| = \frac{1}{2} \|p - q\|_1$$

Theorem

Let G be an (n, d, α)-graph with normalized adjacency matrix $\hat{A}$. Then for every distribution vector $\mathbf{p}$ and any positive integer t

$$\left\| \hat{A}^t \mathbf{p} - \mathbf{u} \right\|_2 \leq \|\mathbf{p} - \mathbf{u}\|_2 \, \alpha^t \leq \alpha^t$$

Random walks resemble independent sampling

- Imagine an abstract sampling problem in which an unknown set B in a universe of size n is "bad in some sense
- We try to sample the universe so as to avoid the bad set as much as possible
- Our task will be to do so, minimizing the number of random bits used
- Say the set B includes all the bad random choices for a probabilistic algorithm, namely, those choices for which it gives the wrong answer

- Let $G(V, E)$ an (n, d, α) and $B \subset V$ with $|B| = \beta n$
- Experiment: We pick $X_0 \in V$ uniformly at random and start from it a random walk $X_0, \ldots, X_t$ on G.
- Denote by (B, t) the event that this random walk is confined to B, i.e. $\forall i, X_i \in B$

Theorem (Ajtai-Komlós-Szemerédi '87, Alon-Feige-Wigderson-Zuckerman '95)

Let G be an (n, d, α)-graph and $B \subset V$ with $|B| = \beta n$. Then the probability of the event (B, t) is bounded by

$$Pr[(B, t)] \leq (\beta + \alpha)^t$$

This can be generalized as follows:

Theorem

Let $B_0, \ldots, B_n$ be vertex sets of densities $\beta_0, \ldots, \beta_n$ in an (n, d, α)-graph G. Let $X_0, \ldots, X_t$ be a random walk on G. Then

$$Pr\left[X_i \in B_i \quad \textit{for all } i\right] \leq \prod_{i=0}^{t-1} \left(\sqrt{\beta_i \beta_{i+1}} + \alpha\right)$$

Let's say we now want a subset of bad choices.

Theorem

For every subset $K \subset \{0, \ldots, t\}$ and vertex subset B of density β,

$$Pr[X_i \in B \quad \textit{for all } i \in K] \leq (\beta + \alpha)^{|K|-1}$$

Efficient error reduction in probabilistic algorithms

- Let A be a probabilistic algorithm to decide membership in language $\mathcal{L}$
- For input x, algorithm samples a string $r \in \{0,1\}^k$ and computes in polynomial time $A(x,r)$
- Remember that in the complexity class **RP** the algorithm makes errors on inputs outside $\mathcal{L}$
- if $x \in \mathcal{L}$, then $A(x,r) = 1$
- if $x \notin \mathcal{L}$, the probability that $A(x,r) = 1$ is at most β
- again our goal is to reduce the probability of error below a threshold without substantial increase in the number of random bits that are required

Let's use expander graphs to see what we can gain!

- Choose explicit an (n, d, α)-graph with $V = \{0,1\}^k$
- choose α sufficiently smaller that β
- choice of α will put a lower bound on d but d can be take to be $O\left(\alpha^{-2}\right)$
- for a given input x let $B_x = B \subseteq \{0,1\}^k$ be the set of all strings r for which the algorithm A errs on x

We introduce a new algorithm A' that uses m random bits and works as follows:

1. pick a vertex $v_0 \in V$ uniformly at random
2. start from it a random walk of length t, say $(v_0, \ldots, v_t)$
3. return $\bigwedge_{i=0}^{t} A(x, v_i)$

But as we have seen with random walks on expander graphs it holds that:

$$Pr[A' \ fails] = Pr[\forall i, \ v_i \in B] \leq (\beta + \alpha)^t$$

A' achieves an exponential reduction in error probability, while the number of random bits used is only $m + t \log d = m + O(t)$

Well, about **BPP**? Things work in a similar fashion. We introduce a new algorithm A' that uses m random bits and works as follows:

1. pick a vertex $v_0 \in V$ uniformly at random
2. start from it a random walk of length t, say $(v_0, \dots, v_t)$
3. return $majority\{A(x, v_i)\}$

A' fails if the majority of the v_i's belong to $B_x = B \subset V$.

- Fix a set of indices $K \subset \{0, \dots, t\}$ with $|K| \geq (t+1)/2$ (majority)
- We have seen that

$$Pr[v_i \in B \text{ for all } i \in K] \leq (\beta + \alpha)^{|K|-1} \leq (\beta + \alpha)^{(t-1)/2}$$

- Assuming $\alpha + \beta \leq 1/8$ and applying the union bound on the possible choices of K we deduce that:

$$Pr[A' \ fails] \leq 2^t(\beta + \alpha)^{(t-1)/2} = O\left(2^{-t/2}\right)$$

Again, A' achieves an exponential reduction in error probability, while the number of random bits used is only $m + t \log d = m + O(t)$

Main parameters of various techniques

Method	**Error probability**	**No. of random bits**
Randomized algorithm A	$1/10$	m
t independent reps of A	2^{-t}	$t \cdot m$
sampling a point and its neighbors in an $(n, t, 1/\sqrt{t})$-graph	$1/t$	m
A random walk of length t on an $(n, d, 1/40)$-graph	$2^{-t/2}$	$m + O(t)$

Hardness of approximating maximum clique size

- Random walks on expanders can be used to enhance hardness of approximation factors, for example in the **clique** problem
- Let G be a graph, then the **clique number** $\omega(G)$ is defined as the largest cardinality of clique in G

Theorem (Feige-Goldwasser-Lovász-Szegedy '91)

*There are two constants $0 < \delta_2 < \delta_1 < 1$ s.t. it's **NP**-Hard to decide for a given n-vertex graph G whether $\omega(G) \leq \delta_2 n$ or $\omega(G) \geq \delta_1 n$.*

Even obtaining a rough approximation of $\omega(G)$ is hard.

Theorem

*If there exists a polynomial-time algorithm A whose output on every n-vertex graph G satisfies $n^{-\epsilon} \leq A(G)/\omega(G) \leq n^{\epsilon}$ for an $\epsilon > 0$, then **NP = P**.*

Theorem (weaker version)

If there exists a polynomial-time algorithm A whose output on every n-vertex graph $G(V,E)$ satisfies $n^{-\epsilon} \leq A(G)/\omega(G) \leq n^{\epsilon}$ for an $\epsilon > 0$, then $\textbf{NP} \subseteq \textbf{RP}$.

- Consider a graph H with vertex set V^t, $t = \log n$
- The vertices $(v_1, \ldots, v_t)$ and $(u_1, \ldots, u_t)$ are adjacent in H, if the subgraph of G induced by the set $(v_1, \ldots, v_t) \cup (u_1, \ldots, u_t)$ is a clique
- whether $\omega(G)$ is below $\delta_2 n$ or above $\delta_1 n$, this is significantly amplified in H

Consider an algorithm B that on input $G(V, E)$ does the following:

1. Pick $m = poly(n)$ random vertices from V^t and compute the subgraph H' of H induced by this set
2. Apply algorithm A to H'
3. Return 1 if $A(H') > \frac{1}{2}\delta_1^t m$, and otherwise return 0

The following two hold, so we can conclude.

- if $\omega(G) \geq \delta_1 n$, then almost surely $\omega(H') \geq \frac{1}{2}\delta_1^t m$
- if $\omega(G) \leq \delta_2 n$, then almost surely $\omega(H') \leq \frac{1}{2}\delta_2^t m$

(Actually this shows that **NP** $\subseteq$ **BPP**)
An estimate, in a polynomial sized sample on a expander graph, is enough to create a conclusion for the problem in the expander graph, and so in the initial graph.

A deterministic reduction also exists.

- Choose some expander (n, d, α)-graph $\mathcal{G}$ with $V_{\mathcal{G}} = V$
- Consider all t-tuples that represent a random walk of $t-1$ length on $\mathcal{G}$
- random walks on $\mathcal{G}$ should behave like random t-tuples in $|V|^t$
- The resulting H' graph has $m = nd^{t-1}$ vectices. d is fixed and $t = \log n$, so it's polynomial on the input

The following two hold, so we can conclude.

- if $\omega(G) \geq \delta_1 n$, then $\omega(H') \geq (\delta_1 - 2\alpha)^t m$
- if $\omega(G) \leq \delta_2 n$, then $\omega(H') \leq (\delta_2 - 2\alpha)^t m$

The zig-zag product

- The k-th power of $G(V, E)$, is denoted by $G^k(V, E_k)$, has the same vertex set as G and an edge $(u, v) \in E_k$ for every path of length k in G from u to v
- For the adjacency matrices we have that: $A_{G^k} = A_G^k$
- If G is an (n, d, α)-graph then G^k is a (n, d^k, α^k)-graph
- The zig-zag product is an asymmetric binary operation
- The product of an (n, m)-graph and an (m, d)-graph is and (nm, d^2)-graph

Theorem (The Zig-Zag Theorem, Reingold-Vadhan-Wigderson '02)

Let G be an (n, m, α)-graph and H be an (m, d, β)-graph. Then $G \textcircled{z} H$ is an $(nm, d^2, \phi(\alpha, \beta))$-graph where the function ϕ satisfies the following:

1. *if $\alpha < 1$ and $\beta < 1$ then $\phi(\alpha, \beta) < 1$*
2. $\phi(\alpha, \beta) \leq \alpha + \beta$
3. $\phi(\alpha, \beta) \leq 1 - (1 - \beta^2)(1 - \alpha)/2$

- The first bound says that the zig-zag product takes two expanders into another expander
- The other two are crucial for applications. The former is useful when α, β are small, and the latter when they are large
- Reingold used bound (3) on ϕ for his proof that $SL = L$

- Let G be an (n, m, α)-graph and H be an (m, d, β)-graph
- For every vertex $v \in V_G$ we fix some numbering of the edges incident with v, i.e. $e_v^1, \ldots, e_v^m$
- Regard the vertex set of H as $[m] = \{1, \ldots, m\}$

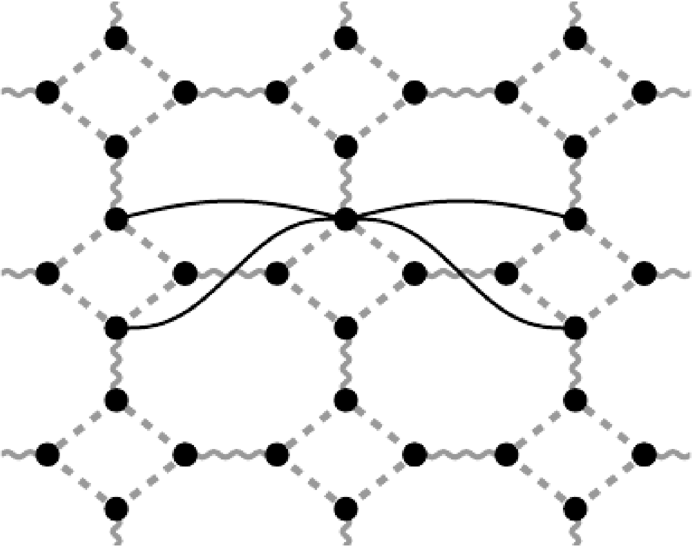
Definition

$G \textcircled{z} H = (V_G \times [m], E')$, where $((v, i), (u, j)) \in E'$ iff there are some $k, l \in [m]$ s.t. $(i, k), (l, j) \in E_H$ and $e_v^k = e_u^l$

Roughly speaking, the zig-zag product $G \textcircled{z} H$ replaces each vertex of G with a copy (cloud) of H, and connects the vertices by moving a small step (zig) inside a cloud, followed by a big step (zag) between two clouds, and finally performs another small step inside the destination cloud.

- The vertex set of $G \textcircled{z} H$ is the cartesian product $V_G \times V_H$
- Define the replacement product (on the same vertex set)
- Edges of $G \textcircled{r} H$ are the union of
  1. the original edges of G (shaded lines)
  2. n copies of the edges of H, one copy per cloud (wiggly edges between clouds)
- edges of $G \textcircled{z} H$ arise form walks of length three in $G \textcircled{r} H$: "dashed-wiggly-dashed"

The zig-zag product of the grid $\mathbb{Z}^2$ with the 4-cycle



Construction of an expander family using zig-zag

- Let H be a $(d^4, d, 1/4)$-graph for some constant d
- There exists such a graph (probabilistic proof & exhaustive search to find it, but more efficient construction exists)
- Using the building block H, we inductively define the infinite sequence G_n by:

$$G_1 = H^2, \quad G_{n+1} = G_n^2 ⓩ H, \quad \text{for } n \geq 1$$

- Graph G_n is a $(d^{4n}, d^2, 1/2)$-graph for all n
- This construction is only **mildly** explicit
- To make it **strongly** explicit, in every iteration we must take the tensor product of G_n with itself

SL=L (Reingold '05)

SL is the complexity class of problems log-space reducible to **USTCON** (undirected s-t connectivity), i.e. USTCON is SL-complete

- Assume you arrive in an unfamiliar city with no map
- You want to get to your hotel whose street name you know
- You can create your own map to avoid loops, but suppose you don't have that much memory available
- That is, suppose you have only **logarithmic** memory instead of linear to the size of the city
- This is the same as exploring a graph (or determine if an s-t path exists)

A probabilistic logspace algorithm

- Aleliunas, Karp, Lipton, Lovász and Rackoff '79
- Perform a polynomial length random walk starting from s
- Algorithm uses logarithmic space, needs to remember the goal t and its current position
- Assume that the input graph G is an expander graph
- ...then the diameter of G is of logarithmic size
- Then one can enumerate all the logarithmically long paths and check if there is and s-t path

- But what if the input graph is not an expander? We will make one out of it
- First make it D-regular, e.g. by adding self loops
- Input graph is now an (n, D, α)-graph
- Assume $D = d^{16}$ and that we have an $(d^{16}, d, 1/2)$-graph H
- We construct graphs G_i as follows:

$$G_1 = G, \quad G_{i+1} = (G_i ⓩ H)^8 \quad \text{for } i \geq 1$$

- For $k = O(\log n)$ the graph G_k is and $(nd^{16k}, d^{16}, 3/4)$-graph
- Neighborhood queries for G_k can be answered in logspace
- Large expander graph constructed by zig-zag product are **very explicit**
- We need only **constant** amount of additional space to create G_k
- We achieve this by using a data structure (with rotation maps)

[HLW06] Shlomo Hoory, Nathan Linial, and Avi Wigderson.
Expander graphs and their applications.
Bull. Amer. Math. Soc., 43(04):439–562, August 2006.

Thank you!