Lecture 4 : Explicit constructions using zig-zag product and SL=L

Topics in Pseudo-randomness and Complexity Theory (Spring 2018)
Rutgers University
Swastik Kopparty
Scribe: Vishvajeet N, Mik Zlatin

1 Recap

We started out with a goal to reduce the probability of error in randomized computation. We saw that using independent runs of the randomized algorithm and then taking majority helps us amplify the success probability. However, this method uses a lot more randomness, often a costly resource, and we would like to amplify the success probability using almost the same amount of randomness as the original algorithm. In some sense, we wanted correlated samples which appeared close to random on a larger space. We tried placing an expander on the space of all possible random strings that the algorithm can use, and then choose a random vertex and selected its neighbours as our strings for subsequent runs. This gave us a way to use the same amount of randomness as in one run of the original algorithm, since the only randomness in thi procedure was in choosing one vertex. Once the vertex was chosen, the set of strings used for sequential runs was completely determined. The expander mixing lemma guaranteed us that the probability of a majority of neighbours being bad was low. Despite the randomness being the same, this method did not amplify the success probability as good as independent runs did. We turned our attention to random walks. Random walks involve correlated random steps and we can thus save the total randomness used. If we take a sufficiently long random walk on a graph, the distribution becomes close to uniform (converges). We proved that random walks on expanders converge quickly, and thus if we take one on an expander, it doesn't get stuck in the set of (bad) strings which give the wrong answer and eventually heads out into the (good) set, in probability. Given a good expander graph, we showed that the probability of error could be reduced significantly using this method and thus the problem boiled down to the construction of a family of expanders, deterministically. It's easy to get expanders of high (polynomial) degree by repeated squaring of small graphs, however, that means our random walks require more randomness at each step. We are thus looking for expander graphs of constant degree which can be found deterministically. Zig-zag product is one way to decrease the degree while keeping the expansion close to the original. Thus, we can interleave squaring and zig-zag product to get a constant degree expander, as we shall see.

2 Zig - Zag Product

Let's recall the zig-zag product that we defined in the last class. We took a big D-regular graph G with N vertices and expansion λ_G, and a small d-regular graph H on D vertices with expansion λ_H and claimed that the zig-zag product of these two graphs $G \textcircled{z} H$ is a d^2-regular graph with ND vertices and expansion $1 - (1 - \lambda_H)^2(1 - \lambda_G)$. We will try to show this.

Let's remind ourselves the definition of the zig-zag product.

Definition 1. *Zig-zag product If G is a D-regular graph on the vertex set $[N]$. H is a d-regular graph on the vertex set $[D]$, then $G \textcircled{z} H$ is a d^2-regular graph on the set $[N] \times [D]$. A vertex (i, j) has its (k_1, k_2)-th neighbour as vertex (i^*, j'') i.e. $(i, j)[(k_1, k_2)] = (i^*, j'')$ if it can be obtained using the following set of rules:*

- *Take the edge labelled k_1 out of the vertex j in H, you will reach j' i.e. $j[k_1] = j'$*
- *Next, take the j'-th neighbour of i in G. We land at vertex i^*, of which i is the j^*-the neighbour. i.e. $Rot_G(i, j') = (i^*, j^*)$.*
- *Finally, take the k_2 neighbour of j^* in H and we arrive at j''. i.e. $j^*[k_2] = j''$.*

Before we analyse the expansion of this graph, let's recall this lemma which we have already proved in the last class.

Lemma 2. *G is a λ-expander, if and only if M, the normalized adjacency matrix of G can be written as $M = (1 - \lambda)J + \lambda E$, where J is the all $1/N$ matrix and E has $||E|| \leq 1$*

The above lemma says that if the second eigenvalue of the adjacency matrix is close to 1, then the error term E becomes large and the distribution is further away from the uniform distribution J. Expanders have secdon eigenvalue small, and they can be thus thought of as trying to be like J - the adjacency matrix of a complete graph. For a complete graph, taking just one step means that the distribution becomes completely uniform.

If we can express the normalized adjacency matrix of $G \textcircled{z} H$ as in the above lemma, we're done. Observe that each vertex in G is blown up into a set of vertices of size $[D]$ - a 'cloud' of vertices of H. We look at a step in the random walk on $G \textcircled{z} H$ as a set of 3 steps : (1) A random step within the cloud. (2) A step between clouds, according to the edges of graph G and (3) A random step within the corresponding cloud. The first and the third steps are random, the second step is deterministic. Each vertex in G is replaced by a cloud of vertices in H. Thus, the transition matrix of the first step has copies of H on every vertex of G, and so does the third step. The second step is a permutation matrix, which just gives the cloud to jump to, when we input the cloud which we are in, and the second parameter - edge label. Thus, the adjacency matrix is expressed as $B\ C\ B$, where $B = I_N \otimes M_H$[1] and C is a permutation matrix of order $[N][D] \times [N][D]$ such that :

$$C_{(u,k),(v,l)} = \begin{cases} 1 \;\; if\; u[k] = v \text{ and } v[l] = u \text{ in } G \\ 0\; otherwise \end{cases}$$

Thus, $\mathrm{M}_{G \textcircled{z} H} = BCB = (I_N \otimes M_H)C(I_N \otimes M_H)$

Using lemma 2, can write $\mathrm{M}_H = (1 - \lambda_H)J_D + \lambda_H E_H$

$M_{G \textcircled{z} H}$
$= (I_N \otimes ((1 - \lambda_H)J_D + \lambda_H E_H))C(I_N \otimes ((1 - \lambda_H)J_D + \lambda_H E_H))$

[1]The adjacency matrix of H

$= (1-\lambda_H)^2(I_N \otimes J_D)C(I_N \otimes J_D) + (1-\lambda_H)\lambda_H(I_N \otimes J_D)C(I_N \otimes E_H) + (1-\lambda_H)\lambda_H(I_N \otimes E_H)C(I_N \otimes J_D) + \lambda_H^2(I_N \otimes E_H)C(I_N \otimes E_H)$

Let's go over properties of matrix norms again.

Definition 3. *Matrix norm over the reals*

$||M|| = sup_{x \neq 0} \frac{||Mx||}{||x||}$

where x is a vector over the reals of appropriate dimension.

Some standard properties of matrix norms which we will use are mentioned in the following lemma without proof.

Lemma 4. *Matrix norm properties*

1. $||\alpha A|| = \alpha ||A||$ *where α is a real number*
2. $||A + B|| \leq ||A|| + ||B||$
3. $||A \otimes B|| \leq ||A|| \cdot ||B||$
4. $||A \cdot B|| \leq ||A|| \cdot ||B||$

We analyse the last 3 terms separately using properties of norms.

Claim 5. *Norms*

$||I_N|| = 1$

$||J_D|| = 1$

$||C|| = 1$

$||I_N \otimes J_D|| = 1$

The claim follows from the fact that I_N and C are permutation matrices, so their norm is 1. J_D is the adjacency matrix of the complete graph, so its norm is 1. The last equality follows from property 3 of lemma 4.

Claim 6. $(I_N \otimes J_D)C(I_N \otimes E_H) = E'$ *with* $||E'|| \leq 1$

Proof: Using property 4 of lemma 4,

$(I_N \otimes J_D)C(I_N \otimes E_H)$
$= ||I_N \otimes J_D|| \cdot ||C|| \cdot ||I_N \otimes E_H||$
$= ||I_N|| \cdot ||J_D|| \cdot ||C|| \cdot ||I_N|| \cdot ||E_H||$ *using property 3 of lemma 3* $= 1 * 1 * 1 * 1 * ||E_H||$
$= ||E_H|| \leq 1$

Thus, $(I_N \otimes J_D)C(I_N \otimes E_H)$ is a matrix with norm less than 1. We call it E'.

Similar is the case with
$(I_N \otimes E_H)C(I_N \otimes J_D)$ and $(I_N \otimes E_H)C(I_N \otimes E_H)$. We summarize in the following claim; proof is just as in claim 6.

Claim 7. *$(I_N \otimes E_H)C(I_N \otimes J_D) = E''$ and $(I_N \otimes E_H)C(I_N \otimes E_H) = E'''$ with $||E''|| \le 1$ and $||E'''|| \le 1$*

Thus,

$M_{G \textcircled{z} H}$
$= (I_N \otimes ((1-\lambda_H)J_D + \lambda_H E_H))C(I_N \otimes ((1-\lambda_H)J_D + \lambda_H E_H))$
$= (1-\lambda_H)^2(I_N \otimes J_D)C(I_N \otimes J_D) + (1-\lambda_H)\lambda_H(I_N \otimes J_D)C(I_N \otimes E_H) + (1-\lambda_H)\lambda_H(I_N \otimes E_H)C(I_N \otimes J_D)$
$+ \lambda_H^2(I_N \otimes E_H)C(I_N \otimes E_H)$
$= (1-\lambda_H)^2(I_N \otimes J_D)C(I_N \otimes J_D) + (1-\lambda_H)\lambda_H E' + (1-\lambda_H)\lambda_H E'' + \lambda_H^2 E'''$
$= (1-\lambda_H)^2(I_N \otimes J_D)C(I_N \otimes J_D) + [(1-\lambda_H)\lambda_H + (1-\lambda_H)\lambda_H + \lambda_H^2]E''''$
$= (1-\lambda_H)^2(I_N \otimes J_D)C(I_N \otimes J_D) + [1-(1-\lambda_H)^2]E''''$
for some E'''' with $||E''''|| \le 1$

$(1-\lambda_H)\lambda_H E' + (1-\lambda_H)\lambda_H E'' + \lambda_H^2 E'' = [(1-\lambda_H)\lambda_H + (1-\lambda_H)\lambda_H + \lambda_H^2]E''''$ follows from property 2 of lemma 4.

Let's analyse $(I_N \otimes J_D)C(I_N \otimes J_D)$. It can be seen that it is equal to $M_G \otimes J_D$. Now we apply lemma 2 to M_G and substitute.

Thus,

$M_{G \textcircled{z} H}$
$= (1-\lambda_H)^2((1-\lambda_G)J_N + \lambda_G E_G) \otimes J_D + [1-(1-\lambda_H)^2]E''''$
$= (1-\lambda_H)^2(1-\lambda_G)J_{ND} + (1-\lambda_H)^2\lambda_G E_G + [1-(1-\lambda_H)^2]E''''$
$= (1-\lambda_H)^2(1-\lambda_G)J_{ND} + [1-(1-\lambda_H)^2(1-\lambda_G)]E'''''$

for some E''''' with $||E'''''|| \le 1$

where the last equality follows from property 2 of lemma 4 again.

Using the $\Leftarrow$ direction of lemma 2, we have managed to prove the following theorem.

Theorem 8. *If G is a D-regular graph with N vertices and expansion λ_G, and H small d-regular graph on D vertices with expansion λ_H, the zig-zag product $G \textcircled{z} H$ is a d^2-regular graph with ND vertices and expansion $1-(1-\lambda_H)^2(1-\lambda_G)$.*

3 Deterministic Construction of Expanders

We will use zig-zag product to construct constant degree expanders, the idea being that squaring gives us better expansion at the cost of degree and zig-zaging gives a bigger graph while roughly preserving the expansion and decreasing the degree, so interleave them.

Consider the following algorithm :

Algorithm 9. *Explicit construction*

```
$G_0$ is any non-bipartite connected $D^{1/10}$-regular graph on $N_0$ vertices
For $i = 1$ to $k$ {
    $G'_i = G_{i-1}^{10}$
    $G_i = G'_i ⓩ H$ ;$H$ is a constant degree $d$ expander of size $D$ and $\lambda_H \leq 1 - 0.1682$
}
```

Claim 10. *G_k is a graph of size $N_0 d^k$ and is d^2-regular.*

Analysis of the eigenvalue :

$1 - \lambda_{G_{i+1}} = (1 - \lambda_H)^2 (1 - \lambda_{G_i}^{10}) \geq 1/4(1 - \lambda_{G_i}^{10})$

If $\epsilon_i = 1 - \lambda_{G_i}$,

$1 - \lambda_{G_{i+1}} \geq 1/4(1 - (1 - \epsilon_i)^{10}) \geq 1/4 \cdot (1 - (1 - 5\epsilon_i)) \geq \frac{5}{4}\epsilon_i$

since $(1 - \epsilon_i)^{10} \leq 1 - 5\epsilon$ for all values of $\epsilon_i < 0.1682$[2]

Thus, ϵ_i grows geometrically as long as it is smaller than 0.1682. If we choose k to be $\log N_0$, then the spectral gap becomes $\Omega(1)$, which is what we want. The number of vertices is also polynomial in N_0. In polynomial time, we have constructed a graph which is an expander of constant degree.

Let's analyse the space complexity of the above construction. There are two types of cosntructions :

- Explicit construction

  This type of graph construction means that the adjacency matrix of the graph can be found out in time polynomial in the size of the graph.

- Strongly explicit construction

  In this type of graph construction, given a pair of vertices, we can output whether they are connected or not, in time polynomial in the logarithm of the size of the graph.

Observe that the above construction is explicit but not strongly explicit.

[2]0.1682 is a root of the function $(1 - x)^{10} - (1 - 5x)$

4 Undirected Deterministic s-t Connectivity

The problem: Given an undirected graph $G = (V, E)$, and two vertices $s, t \in V$, we want to determine whether there is a path from s to t in G.

We have seen several algorithms to solve this problem before. The obvious method of using BFS or DFS runs in polynomial time but requires $O(n)$ bits of memory. We then analyzed Savitch's algorithm, which reduces the memory requirement to $O(\log^2 n)$, but takes $O(n^{\log n})$ time, which is not polynomial in n. Finally, we saw a randomized algorithm involving lazy random walks that runs in polynomial time, requires $O(\log n)$ space, and is correct with high probability.

We will now use the theory of expanders to produce a polynomial time deterministic algorithm that only requires $O(\log n)$ space. Note that this essentially means that we are asking for a strongly explicit (deterministic) construction.

Algorithm 1 Reingold(2005)

```
1:  procedure FOO($G, s, t$)
2:      $d \leftarrow 1000$
3:      Fix a $d$-regular expander $H$ on $d^{20}$ vertices.
4:      Let $G_0$ be a $d^2$-regular transformation of $G$
5:      for $i = 0$; $i \leq O(\log n)$; $i$++ do
6:          $G'_{i+1} \leftarrow (G_i)^{10}$
7:          $G_{i+1} \leftarrow G'_{i+1} ⓩ H$
8:      end for
9:      At this point, $G_{O(\log n)}$ has diameter at most $O(\log n)$, since it is a good expander (a random
        walk with $O(\log n)$ steps is close to uniform on good expanders). Thus we can check all possible
        paths of length $O(\log n)$from $s$ to $t$.
10:     return Yes, iff a path was found.
11: end procedure
```

Analysis: Since G is d^2-regular, there are at most $d^{O(\log n)}$ paths of length $O(\log n)$. So the algorithm only checks a polynomial number of paths, and thus takes polynomial time. To iterate through the paths, we just need to store a counter in $[d^2]$ for each depth of the recursion, to keep track of which edge to travel down next. Thus we have to store $O(\log n)$ bits.

Interesting Facts: There are other ways of obtaining deterministic algorithms with this space and time complexity, but they all are connected to the theory of expanders. See Derandomized Squaring by Rozenman-Vadham. Algorithm for *s*-*t* connectivity in directed graphs with $O(\log n)$ space is an open problem.

5 k-Wise Independence of Random Variables

Definition 11. *A probability distribution $(x_1, x_2, \ldots, x_n)$ is k-wise independent if any subset of size at most k is independent.*

Example 12. *Let x_1, x_2 be uniform random variables from $\{0, 1\}$. Let $x_3 = x_1 \oplus x_2$. Then*

(x_1, x_2, x_3) is 2-wise independent. $\diamondsuit$

Example 13. *Let $x_1, x_2, \ldots, x_m$ be uniform random variables from $\{0, 1\}$. Define for each $S \subseteq [m]$ with $|S| > 0$, $x_S = \bigoplus_{i \in S} x_i$. Then each (x_S) are pairwise independent.*

Proof: Take $S_1 \neq S_2$. We want to show that X_{S_1} and X_{S_2} are independent. Let $y_i \in \mathbb{F}_2^m$ be the indicator variable on S_i. Let x be a random vector in $\mathbb{F}_2^m$. We must show that for $b_1, b_2 \in \mathbb{F}_2$:

$$Pr[\langle y_1, x\rangle = b_1 \text{ and } \langle y_2, x\rangle = b_2] = Pr[\langle y_1, x\rangle = b_1] \cdot [\langle y_2, x\rangle = b_2].$$

The R.H.S is clearly equal to $\frac{1}{4}$, since for any vector x that gives $\langle y_1, x\rangle = b_1$, we can associate another vector $\bar{x}$ such that $\langle y_1, \bar{x}\rangle = b_2$, so each term on the R.H.S is a half. To analyze the L.H.S, we look at the values of x that solve:

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

We will establish a bijective correspondence between solutions to this equation for each of the 4 combinations of b_1 and b_2. This will prove that the L.H.S is also $\frac{1}{4}$. Consider x_0 a solution to the above equation for $(b_1, b_2) = (0, 0)$. Since $S_1 \neq S_2$, we know $y_1 \neq y_2$ at some coordinate, say i. Then by flipping the i^{th} bit of x, we obtain a solution for $(b_1, b_2) = (1, 1)$. Flipping any other bit further divides these into solutions to $(b_1, b_2) = (1, 0)$ and $(b_1, b_2) = (0, 1)$. Thus the set of possible x values is partitioned equally into solutions for each possible pair of b values and the L.H.S is $\frac{1}{4}$. $\diamondsuit$

References

- Entropy waves, the zig-zag graph product, and new constant-degree by Omer Reingold, Salil Vadhan, Avi Wigderson : https://arxiv.org/abs/math/0406038
- Undirected ST-connectivity in log-space by Omer Reingold
- Derandomized Squaring of Graph by Eyal Rozenman, Salil Vadhan
- Expander Graphs and Their Applications by Shlomo Hoory, Nathan Linial, Avi Wigderson
- https://lucatrevisan.wordpress.com/2011/03/07/cs359g-lecture-17-the-zig-zag-product/