18.405J / 6.841J Survey: The Zig-Zag Product and USTCON $\in L$

Alexandr Wang

Abstract

This survey will cover the zig-zag product, introduced in Reingold et al. (2002), and the application of the zig-zag product for a deterministic log-space algorithm for USTCON in Reingold (2008). In particular, the result implies $SL = L$, where SL is the class of problems solvable by symmetric, non-deterministic, log-space algorithms. The result is a major step in determining if $RL = L$, where RL is the class of problems solved by randomized log-space algorithms with one-sided error. In particular, $RL = L$ would imply that randomization cannot give you significant savings in memory.

1 Introduction

In this survey, we'll cover the landmark result of Reingold (2008) to show that USTCON $\in L$. USTCON is the problem, given as input an undirected graph G and two nodes s, t, determining if s and t are connected in G. We'll first give some background of the related complexity classes. L is the class of problems solvable by deterministic, log-space algorithms. It was previously demonstrated that USTCON is complete for the class SL (Lewis and Papadimitriou (1982)), where SL is the class of problems solvable by symmetric, non-deterministic, log-space algorithms. RL is the class of problems solved by randomized log-space algorithms with one-sided error, and NL is the class of problems solved by non-deterministic log-space algorithms. Prior to this result, we had the following hierarchy for these classes: $L \subseteq SL \subseteq RL \subseteq NL \subseteq L^2$, where the last containment follows from Savitch's theorem.

There has been a lot of work studying the space complexity of USTCON. The first result is that USTCON $\in L^2$ as a result of Savitch's theorem. In 1979, Aleliunas, Karp, Lipton, Lovasz, and Rackoff demonstrated a randomized log-space algorithm for USTCON, which performs a polynomial-length random walk starting at s. Therefore, it follows that USTCON $\in RL$. Therefore, demonstrating a deterministic log-space algorithm for USTCON could be thought of as a de-randomization result, and a step in determining if $RL = L$. Is it possible to de-randomize an algorithm without a significant increase in space?

In 1992, Nisan, Szemeredi, and Wigderson found a deterministic algorithm for solving USTCON that ran in $O(\log^{3/2} n)$ space, and it was later reduced to $O(\log^{4/3} n)$ using the same techniques by Armoni, et. al. in 2000. These results primarily relied on the tools of derandomization. Using different methods, Saks and Zhou showed in 1999 that every RL problem is also in $L^{3/2}$. The next improvement was Reingold (2008), demonstrating that USTCON $\in L$, implying $SL = L$.

The intuition of the algorithm in Reingold (2008) is relatively straightforward. The idea is to improve the connectivity of the input graph until it guaranteed that each connected component has logarithmic diameter. Then, we can simply enumerate all logarithmically-long paths from s and see if any visit t. The transformation to improve the connectivity heavily relies on the results

of Reingold et al. (2002). In particular, they use the zig-zag product (introduced in Reingold et al. (2002)) to improve the connectivity while maintaining that the transformed graph has constant degree. We'll present a thorough discussion of the zig-zag product to understand the intuition behind the transformation, and then we'll present the log-space algorithm for USTCON.

2 Outline

The survey will cover two main topics. The first is the zig-zag product (section 4), originally introduced in Reingold et al. (2002). The zig-zag product is an operation on two expanders, which are informally graphs with good connectivity. It takes in a "large" expander and a "small" expander, and returns an expander with, roughly speaking, the degree of the small expander and the connectivity of the large expander. We will define the zig-zag product, provide motivation behind the construction, and then present the main result: the zig-zag theorem. We'll then provide intuition behind two known proof methods. The first is the original idea presented in Reingold et al. (2002), and the second is a simpler idea presented in Reingold et al. (2005), which used the more elegant methods of Rozenman and Vadhan (2005).

The second topic is the log-space algorithm for USTCON (section 5), thereby implying that $SL = L$. The algorithm presented will use the zig-zag product as an important operation to increase the connectivity of the graph while keeping the degree small. Here's a high-level overview of the algorithm:

1. Let the input graph be G. Turn G into a constant-degree, regular graph G' with each connected component non-bipartite.
2. (subsection 5.1) Turn each connected component of G', in logarithmic number of phases, into an expander. Each phase involves raising the graph to some power and then applying the zig-zag product with a constant-size expander.
3. (subsection 5.2) After our transformation, the diameter of each connected component is logarithmic (Proposition 8) and the degree is constant. So, we enumerate all polynomially many logarithmically long paths starting with s and see if any arrive at t.

Once we properly define the algorithm, it suffices to show each step can be carried out in logarithmic space. Step 1 is relatively simple. We omit the proof here, but the reader may refer to Rozenman and Vadhan (2005) for the construction.

Proposition 1 (Rozenman and Vadhan (2005)) *In logarithmic space and polynomial time, we can transform any undirected graph $G = (V, E)$ to an undirected 4-regular graph with $|V| \cdot |E|$ vertices, contains a loop on every vertex, and is consistently labelled.*

3 Background and Definitions

We'll start by providing some standard definitions and results. For this section, all graphs are undirected, regular, and may have loops and parallel edges.

3.1 Rotations

It is useful to construct a formalized edge labeling as a representation of the graph, which we'll use to represent regular graphs for the rest of the paper. We'll use the notation $[k]$ to denote the set $\{1, \ldots, k\}$.

Definition 2 *For a D-regular undirected graph G on N vertices, the **rotation map** $Rot_G : [N] \times [D] \to [N] \times [D]$ is defined as follows: $Rot_G(v, i) = (w, j)$ if the ith edge incident to v leads to w, and this edge is the jth edge incident to w.*

This allows us to have consistent labelings of edges from the perspective of both endpoints, since $\text{Rot}_G(v, i) = (w, j)$ and $\text{Rot}_G(w, j) = (v, i)$. Notice also that a rotation map fully specifies a graph.

3.2 Eigenvalues and Expanders

The adjacency matrix of an N-vertex graph is the matrix A whose (u, v)'th entry is the number of edges between vertices u and v. If a graph is D-regular, then it follows that each row and column of A have sum D.

Definition 3 *The **normalized adjacency matrix** of a D-regular graph G with adjacency matrix A is $\hat{A} = A/D$.*

Note that the normalized adjacency matrix is also the transition probability matrix of a random walk on G, where every step a token is moved from the current vertex along a uniformly chosen edge to a neighboring vertex. So, $\pi \in \mathbb{R}^N$ is the probability distribution on the vertices at step i, then $\hat{A}\pi$ is the probability distribution at step $i + 1$.

It is clear that the all-1's vector $1^N = (1, 1, \ldots, 1) \in \mathbb{R}^N$ is an eigenvector of $\hat{A}$ of eigenvalue 1. It is a well-known result that all other eigenvalues of $\hat{A}$ have absolute value at most 1, and it turns out the second largest eigenvalue is a good measure of G's expansion properties.

Definition 4 *$\lambda(G)$ denotes the **second largest eigenvalue** (in absolute value) of G's normalized adjacency matrix. Equivalently,*

$$\lambda(G) = \max_{\alpha \perp 1^N} \frac{\langle \alpha, \hat{A}\alpha \rangle}{\langle \alpha, \alpha \rangle} = \max_{\alpha \perp 1^N} \frac{||\hat{A}\alpha||}{||\alpha||}$$

where $\langle \cdot, \cdot \rangle$ is the standard inner product and $||\alpha|| = \sqrt{\langle \alpha, \alpha \rangle}$.

Definition 5 *An (N, D, λ)-graph is any D-regular graph G on N vertices such that $\lambda(G) \leq \lambda$.*

Definition 6 *An family $\mathcal{G}$ of graphs is called a family of **expanders** if there is some constant $\lambda < 1$ such that $\lambda(G) \leq \lambda$ for all $G \in \mathcal{G}$.*

We'll call a graph G an expander if $\lambda(G)$ is bounded by a constant not dependent on the number of vertices in G.

The spectral gap of $\hat{A}$ is then $1 - \lambda(G)$. $\lambda(G)$ is a measure of how quickly a random walk on G will converge to the uniform distribution. To see this, suppose π is a probability distribution on the vertices of G. Due to linear algebra, we can write $\pi = 1^N/N + \pi^\perp$, where $1^N/N$ is the uniform

distribution and $\pi^{\perp}$ is perpendicular to $1^N/N$. By Definition 4, we know that $||\hat{A}\pi^{\perp}|| \leq \lambda(G) \cdot ||\pi^{\perp}||$, implying $\lambda(G)$ measures how quickly it will converge to uniform.

In addition, being an expander implies vertex expansion.

Theorem 7 (Alon (1986)) *If $\mathcal{G}$ is a family of expanders, then there is a constant $\epsilon > 0$ such that for every $G \in \mathcal{G}$ and for any set S of at most half the vertices in G, at least $(1+\epsilon) \cdot |S|$ vertices of G are connected to some vertex in S.*

So, being a good expander is equivalent to having a random walk on the graph converge quickly to uniform. Note that this implies that G has logarithmic diameter. More formally,

Proposition 8 *For any (N, D, λ)-graph G and any vertices s, t in G, there exists a path of length $O(\log N)$ that connects s to t.*

3.3 Graph powering

Our goal is to create an expander out of our initial graph. The simplest way to amplify the expansion of a graph is through powering, although it increases the degree significantly. The t'th power G^t of a graph is the graph with edges corresponding to paths of length t in G. In terms of rotations,

Definition 9 *Let G be a D-regular graph on $[N]$ given by rotation map Rot_G. The t'th power of G is the D^t-regular graph G^t whose rotation map is given by $Rot_{G^t}(v_0, (a_1, a_2, \ldots, a_t)) = (v_t, (b_t, b_{t-1}, \ldots, b_1))$, where these values are computed via the rule $(v_i, b_i) = Rot_G(v_{i-1}, a_i)$ for $i = 1, 2, \ldots, t$.*

Proposition 10 *If G is an (N, D, λ)-graph, then G^t is an (N, D^t, λ^t)-graph.*

This follows from the normalized adjacency matrix of G^t being the normalized adjacency matrix of G raised to the tth power.

4 The Zig-zag Product

As mentioned before, the algorithm for USTCON relies on the zig-zag product to construct an expander out of an arbitrary graph. Here, we'll define the zig-zag product, develop the intuition for it, and prove the Zig-zag Theorem, which is the key result.

4.1 Definition

Definition 11 (zig-zag product Reingold et al. (2002)) *If G is a D_1-regular graph on N vertices with rotation map $Rot_G : [N] \times [D_1] \to [N] \times [D_1]$ and H is a D_2-regular graph on D_1 vertices with rotation map $Rot_H : [D_1] \times [D_2] \to [D_1] \times [D_2]$, then their **zig-zag** product $G \textcircled{z} H$ is defined to be the graph on $[N] \times [D_1]$ vertices whose rotation map $Rot_{G \textcircled{z} H} : ([N] \times [D_1]) \times ([D_2] \times [D_2])$ is as follows:*

$Rot_{G \textcircled{z} H}((v, a), (i, j))$:

1. *Let $(a', i') = Rot_H(a, i)$.*

2. *Let* $(w, b') = Rot_G(v, a')$.

3. *Let* $(b, j') = Rot_H(b', j)$.

4. *Output* $((w, b), (j', i'))$.

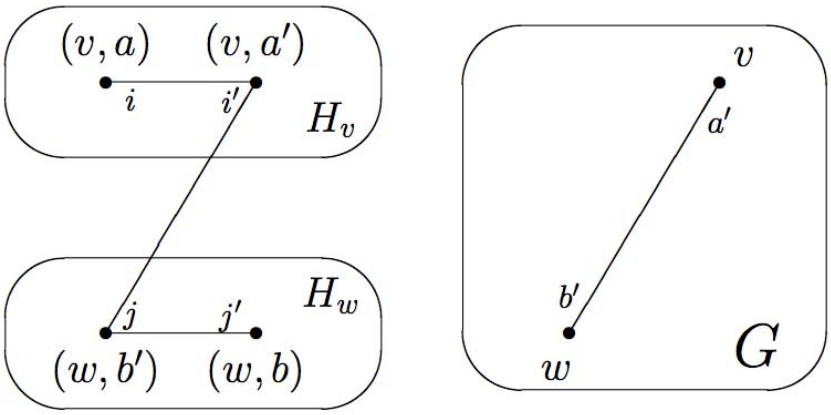

Figure 1: Figure illustrating the zig-zag product, originally in Reingold (2008). The left figure shows the shows each of the small step, the large step, and the small step. The right figure shows the large step when considered as an edge of the larger graph G.

Let's unpack this definition a little. First, we replace every vertex v of G with a cloud of D_1 vertices $(v, 1), \ldots, (v, D_1)$, and denote it H_v. Now, each edge of the zig-zag product is made up of three steps (a zig-zag). First, we make a "small" step within the cloud from (v, a) to (v, a'), where $a \to a'$ corresponds to an edge in H. Second, we make a "big" step between clouds from (v, a') to (w, b'), corresponding to an edge $v \to w$ in G with labels a' and b' in the rotation map. Third, we make another small step from (w, b') to (w, b), where $b' \to b$ corresponds to an edge in H.

Note that the degree of $G \circledz H$ is D_2^2, so if H is a very small graph (which it will be for our algorithm), this results in a graph with significantly smaller degree.

4.2 Motivation for Zig-zag construction

We want $G \circledz H$ to be a good expander as long as both G and H are good expanders. As we saw before, being a good expander is equivalent to having a random walk converge quickly to uniform.

So, first consider the intuition that edges of $G \circledz H$ are defined such that a random step on $G \circledz H$ corresponds to a random step in G, using a random step in H to choose an edge in G. Since H is a good expander, that means a random walk on H should generate close to uniformly random choices on the vertices. So, by using steps in H as the randomness for steps in G, and knowing that truly random walks on G do converge to uniform, we'd expect this initial intuition to correspond to a good expander.

Unfortunately, this initial intuition is not symmetric, i.e. going from (u, a) to (w, b) does not imply you can go from (w, b) to (u, a). So, we correct this by adding another step in H after the step in G. This allows us to construct undirected expander graphs. Moreover, the intuition is still intact. We're using random steps in H, which should be sufficiently random as choices from $[D_1]$, as edge choices for a walk in G.

4.3 The Zig-zag Theorem

The most important property of the zig-zag product is that it takes two expanders into another expander with bounds on the expansion.

Theorem 12 (The Zig-zag Theorem, Reingold et al. (2002)) *Let G be an (N, D_1, α)-graph and H be an (D_1, D_2, β)-graph. Then $G ⓩ H$ is an $(ND_1, D_2^2, \phi(\alpha, \beta))$-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. *$\phi(\alpha, \beta) \leq 1 - (1 - \beta^2) \cdot (1 - \alpha)/2$*

The first bound is stating that the zig-zag product takes two expanders into another expander. The second bound is useful in applications where α and β are small, but isn't used in our algorithm. For our algorithm, we'll mostly be using a fixed expander H, and using it to improve the connectivity of an initially weak expander. In particular, the bound from (3) allows us to decrease the second eigenvalue to a constant in a logarithmic number of iterations, while keeping the degree of the graph constant.

4.4 Intuition for Zig-zag theorem

There are two main intuitions for the Zig-zag theorem, following the two main proof methods:

1. We basically think of H as acting similarly to K_{D_1}, the complete graph on D_1 vertices. If H were the complete graph, then walks on $G ⓩ H$ would actually correspond to exactly random walks in G, and the two small steps in the zig-zag basically don't matter.

   Therefore, we try to write the transition matrix for H as the linear combination of the transition matrix for K_{D_1} and an error matrix. This is exactly the proof method for the proof we will present in subsection 4.5.

2. Here, we consider how the a joint distribution $p_{v,i}(\cdot, \cdot)$ over $[N] \times [D_1]$ on the vertices of $G ⓩ H$ that is not close to uniform would change with the three steps. Roughly speaking, there are two cases to consider:

   (a) Each of the conditional distributions $p_{i|v}(\cdot|v)$ for all $v \in V(G)$ are not close to uniform. These conditionals $p_{i|v}(\cdot|v)$ correspond to the transition probabilities within the cloud for v. So, this means the distribution within each cloud is uneven.

(b) Each of the conditional distributions $p_{i|v}(\cdot|v)$ for all $v \in V(G)$ are uniform, meaning that the distribution $p_v(\cdot)$ obtained from marginalizing over each cloud must be far from uniform.

In the first case, notice that our first small step within a cloud helps that conditional converge to marginal, since H is an expander. Since our large step is just a permutation, and our last small step is a random step on a regular graph, they cannot make our distribution less uniform. So, in this case our random walk would converge towards uniform.

In the second case, our first small step doesn't do anything since it's already close to uniform. But, since the cloud conditional distribution is close to uniform, and that determines which corresponding edge in G we traverse, the large step acts like a real random step in G when marginalizing out the clouds. So, our marginal distribution $p_v(\cdot)$ over the clouds becomes closer to uniform.

But, the large step is also a permutation on all the vertices of $G \textcircled{z} H$, meaning it does no mixing and simply shuffles numbers. So if it causes our marginal $p_v(\cdot)$ to become closer to uniform, then the conditional distributions $p_{i|v}(\cdot|v)$ must have deviated from being uniform. But, now we have the second small step within the cloud, which will bring the conditionals closer to uniform.

This leads to a proof method using Definition 4. For any vector $\alpha \perp 1^{ND_1}$, we write it as the sum of $\alpha^{\|}$ and and $\alpha^{\perp}$, where $\alpha^{\|}$ is uniform on each cloud, and $\alpha^{\perp}$ is orthogonal to uniform on each cloud.

This is the proof method originally presented in Reingold et al. (2002), and does give the best known bounds. An elementary approach will achieve the bound $\phi(\alpha, \beta) \leq \alpha + \beta + \beta^2$, which unfortunately is not strong enough for our USTCON algorithm. Through a geometric bounding argument, this method achieves the results in Theorem 12.

4.5 Proof of Zig-zag Theorem

The proof in Theorem 12 in Reingold et al. (2002) is relatively involved, and basic analysis only gives a weak version of bound (2), namely $\phi(\alpha, \beta) \leq \alpha + \beta + \beta^2$. Instead, we'll present a simplification presented in Reingold et al. (2005). We'll prove a slightly weaker statement of (3) of Theorem 12.

Theorem 13 (Slightly weaker zig-zag theorem) *Let G be an (N, D_1, α)-graph and H be an (D_1, D_2, β)-graph. Then $\lambda(G \textcircled{z} H) \leq 1 - (1 - \alpha) \cdot (1 - \beta)^2$*

First, we'll prove a key lemma.

Lemma 14 (Rozenman and Vadhan (2005)) *Let A be the transition matrix of an (N, D, λ)-graph. Let J_N be the $N \times N$ matrix with all entries equal to $1/N$. Then $A = (1-\lambda)J_N + \lambda C$ where $||C|| \leq 1$.*

Proof: (of Lemma 14)

Write $C = (A - (1-\lambda)J_N)/\lambda$. Let $u_N = 1^N/N$ be the uniform distribution on all N vertices. Since $Au_N = J_N u_N = u_N$, it follows that $Cu_N = u_N$. For $v \perp u_N$ we have that $J_N v \perp u_N$ and $Av \perp u_N$,

which implies $Cv \perp u_N$. Thus, it suffices to show that $||Cv|| \leq ||v||$ for every $v \perp u_N$. Since $Jv = 0$ and $||Av|| \leq \lambda||v||$, then $||Cv|| \leq ||v||$ as desired, proving the lemma. □

The intuition of the lemma is that we can view a random step on a graph as taking us to the uniform distribution with probability $(1 - \lambda)$ and taking us to a distribution that's not any worse from random with probability λ. With it, we can prove the theorem:

Proof: (of Theorem 13, Reingold et al. (2005))

Let M be the transition matrix of a random walk on $G \textcircled{z} H$. Let A be the transition matrix of G and B be the transition matrix of H. We can decompose M into the product of three matrices corresponding to each of the three steps in Definition 11.

Let $\tilde{B}$ be the transition matrix for taking a random G_2-step in the second component of $[N] \times [D_1]$. More explicitly, $\tilde{B} = I_N \otimes B$ where I_N is the $N_1 \times N_1$ identity matrix. Let $\tilde{A}$ be the permutation matrix corresponding to Rot_G. By Definition 11, we have that $M = \tilde{B}\tilde{A}\tilde{B}$.

By Lemma 14, we have that $B = (1 - \beta)J_{D_1} + \beta E$, where J_{D_1} is the $D_1 \times D_1$ matrix with every entry equal to $1/D_1$ and $||E|| \leq 1$. That means $\tilde{B} = (1 - \beta)\tilde{J}_{D_1} + \beta\tilde{E}$, where $\tilde{J} = I_N \otimes J$ and $\tilde{E} = I_N \otimes E$, where we still have $||\tilde{E}|| \leq 1$.

So, we have

$$\begin{aligned} M &= \tilde{B}\tilde{A}\tilde{B} \\ &= \left((1-\beta)\tilde{J}_{D_1} + \beta\tilde{E}\right)\tilde{A}\left((1-\beta)\tilde{J}_{D_1} + \beta\tilde{E}\right) \\ &= (1-\beta)^2\tilde{J}_{D_1}\tilde{A}\tilde{J}_{D_1} + \beta^2\tilde{E}\tilde{A}\tilde{E} + \beta\cdot(1-\beta)\cdot(\tilde{J}_{D_1}\tilde{A}\tilde{E} + \tilde{E}\tilde{A}\tilde{J}_{D_1}) \\ &= (1-\beta)^2\tilde{J}_{D_1}\tilde{A}\tilde{J}_{D_1} + (2\beta - \beta^2)F \end{aligned}$$

where we set F so that $(2\beta - \beta^2)F$ is the sum of the three terms above. We see that

$$||F|| = \frac{1}{2\beta - \beta^2}||\beta^2\tilde{E}\tilde{A}\tilde{E} + \beta\cdot(1-\beta)\cdot(\tilde{J}_{D_1}\tilde{A}\tilde{E} + \tilde{E}\tilde{A}\tilde{J}_{D_1})|| \leq \frac{1}{2\beta - \beta^2}\cdot(\beta^2 + 2\cdot\beta\cdot(1-\beta)) = 1$$

so F has norm at most 1.

Now, I claim that

$$\tilde{J}\tilde{A}\tilde{J} = A \otimes J$$

Let's unpack both sides. We'll show they're equivalent by showing they correspond to the same transition matrix.

The left-hand side corresponds to the following sequence of transitions:

- Starting at state (v, a).
- Choose a' uniformly from $[D_1]$.
- Let $(w, b') = \text{Rot}_G(v, a')$.
- Choose b uniformly from $[D_1]$.
- Go to state (w, b).

The right-hand side corresponds to the following sequence of transitions:

- Starting at state (v, a).
- Let w be a random neighbor of v in G.
- Choose b uniformly from $[D_1]$.
- Go to state (w, b).

These two processes are identical by the definition of a rotation map, so their corresponding transition matrices must be the same, demonstrating the claim.

Finishing up, we have that

$$\begin{aligned} M &= (1-\beta)^2 \tilde{J}_{D_1} \tilde{A} \tilde{J}_{D_1} + (2\beta - \beta^2) F \\ &= (1-\beta)^2 \cdot A \otimes J + (2\beta - \beta^2) F \end{aligned}$$

Therefore,

$$\begin{aligned} \lambda(M) &\leq (1-\beta)^2 \cdot \lambda(A \otimes J) + (2\beta - \beta^2)\lambda(F) \\ &\leq (1-\beta)^2 \cdot \alpha + (2\beta - \beta^2) \\ &= 1 + (1-\beta)^2 \cdot \alpha - (1-\beta)^2 \\ &= 1 - (1-\beta)^2 \cdot (1-\alpha) \end{aligned}$$

as desired □

5 Log-space algorithm for USTCON

We can now use the Zig-zag theorem to formalize our log-space algorithm for USTCON.

5.1 Transforming regular graphs into expanders

First off, given any regular graph G on N vertices, we do have a bound that $\lambda(G) < 1 - \Omega(1/N^2)$. More formally,

Lemma 15 (Alon and Sudakov (2000)) *For every D-regular, connected, non-bipartite graph G on N vertices, we have $\lambda(G) \leq 1 - \frac{1}{DN^2}$.*

Now suppose we're given a (N, D, λ)-graph G where $\lambda < 1 - \frac{1}{DN^2}$. Assume that $D = d^{16}$, and that we have some fixed $(d^{16}, d, 1/2)$-graph H. There are some explicit constructions of constant-degree expanders (some presented in Reingold et al. (2002)), which we'll use to assume the existence and construction of H. Note that although we only demonstrated how to create a 4-regular graph, we can power the graph a constant number of times to obtain a graph with $D = d^{16}$.

Now, we'll iteratively construct graphs G_i, starting with G, such that after a logarithmic number of iterations, our final graph is an expander. The construction is as follows:

$$\begin{aligned} G_1 &= G \\ G_{i+1} &= (G_i \circledS H)^8 \text{ for } i \geq 1 \end{aligned}$$

After $k = O(\log N)$ iterations, we terminate. Note that graph G_i is d^{16}-regular on Nd^{16i} vertices. Moreover, we'll show that G_k is a constant-degree expander by bounding the expansion after each operation. Since d is constant, this is at most a polynomial blow up in the number of vertices of the graph.

Theorem 16 *For $i = 1, \ldots, k-1$, $\lambda(G_{i+1}) \leq \max(\lambda(G_i), 1/2)$.*

Proof: By applying the third bound of Theorem 12 on $G_i \circledz H$, we have

$$\begin{aligned}\lambda(G_i \circledz H) &\leq 1 - \frac{1}{2} \cdot (1 - \lambda(H)^2) \cdot (1 - \lambda(G_i)) \\ &\leq 1 - \frac{1}{2} \cdot \left(1 - \frac{1}{4}\right) \cdot (1 - \lambda(G_i)) \\ &\leq \frac{3}{8} \cdot (1 - \lambda(G_i))\end{aligned}$$

Therefore,

$$\lambda(G_{i+1}) = \lambda((G_i \circledz H)^8) = \lambda(G_i \circledz H)^8 \leq \left(1 - \frac{3}{8} \cdot (1 - \lambda(G_i))\right)^8$$

If $\lambda(G_i) \leq 1/2$, then we have $\lambda(G_{i+1}) \leq (1 - \frac{3}{16})^8 < 1/2$. If $\lambda(G_i) > 1/2$, we know for $x \in [1/2, 1]$ that $(1 - \frac{3}{8}(1-x))^8 \leq x^2$, that means $\lambda(G_{i+1}) \leq \lambda(G_i)^2$.

Therefore, $\lambda(G_{i+1}) \leq \max(\lambda(G_i), 1/2)$ as desired. □

It follows from Lemma 15 and Theorem 16 that for $k = O(\log N)$ that $\lambda(G_k) \leq 1/2$, implying G_k is an expander.

We omit the straightforward proof that the transformation operates separately on each connected component of G, thus maintaining the connected component structure of G, which allows us to perform these operations and still solve $s - t$ connectivity. The proof is presented in Reingold (2008).

Finally, we just need to argue that the rotation map Rot_{G_k} is computable in log space from the rotation map Rot_{G_1}. This is certainly non-obvious, since naive methods would result in $O(\log N \log\log N)$ required space. We omit the proof in the survey, but we'll quickly explain the intuition. The evaluation of $\text{Rot}_{G_{i+1}}$ is composed of a constant number of operations, each of which is either an evaluation of Rot_{G_i}, or requires a constant amount of memory. By reusing this constant memory across operations, and keeping a constant size counter to track the operation we're on, the evaluation of $\text{Rot}_{G_{i+1}}$ only requires a constant amount more space than the evaluation of Rot_{G_i}.

5.2 Solving USTCON on a constant-degree expander

Once each connected component is a constant-degree expander, the rest of the algorithm follows easily. We'll include the algorithm and the brief analysis for completeness.

Proposition 17 *Let $\lambda < 1$ be some constant. Then there exists a space $O(\log D \cdot \log N)$ algorithm $\mathcal{A}$ that takes as input a D-regular undirected graph G on N vertices and two vertices s, t and the*

following holds. If s and t in the same connected component and this component is an (N', D, λ)-graph, then $\mathcal{A}$ outputs 'connected'. Conversely, if $\mathcal{A}$ outputs 'connected', then s and t are connected.

Proof: $\mathcal{A}$ simply enumerates all D^l paths of length $l = O(\log N)$ from s, where the constant factor is the same as the constant factor in Proposition 8. If any of these paths contains t, $\mathcal{A}$ outputs 'connected'. Otherwise, it outputs 'unconnected'.

It suffices to show this algorithm is $O(\log D \cdot \log N)$ space and is correct. Following any path, given as a sequence of edge labels in $[D]$, takes $O(\log N)$ space. Enumerating all D^l paths takes $O(\log D \cdot \log N)$ space. By Proposition 8, we know that if s and t are in the same connected component and this component is an (N', D, λ)-graph, then $\mathcal{A}$ will find a path from s to t with length at most l and output connected. $\mathcal{A}$ also only outputs connected when it finds a path.

□

5.3 Bringing it together

So, we've described how to perform each step of our high-level overview in section 2. As a quick recap, our algorithm takes the input graph and transforms it to a constant degree regular graph. Then, we perform powering and zig-zag product on this graph to transform every connected component to a constant degree expander. To finish, we simply enumerate all logarithmically long paths starting from s and see if they reach t. Each of these steps causes at most polynomial increase in the size of the graph, so each step can be done in space logarithmic in the initial input size. Therefore, our algorithm proves the key result:

Theorem 18 $\mathsf{USTCON} \in L$

Corollary 19 $SL = L$

6 Further discussion

After Reingold (2008), Rozenman and Vadhan (2005) were able to achieve the same result by introducing de-randomized graph squaring. Roughly speaking, de-randomized graph squaring results in a graph as well connected as the traditional graph square, while only increasing the degree by a constant factor as opposed to squaring it. The operation is very similar to the zig-zag product. In fact, in Rozenman and Vadhan (2005) they prove the same bound on the second eigenvalue as Theorem 13 using the same proof method we demonstrated in this survey. The benefits of using de-randomized squaring instead of the zig-zag product is that it doesn't change the vertex set, and the logarithmic space arguments are more straightforward. That being said, both methods have the same general intuition, and have extremely similar analyses. We refer the reader to Rozenman and Vadhan (2005) if they're interested.

There are many open problems based on this work. We'll list some here:

- De-randomizing RL and showing $RL = L$. There was some progress done in Reingold et al. (2005) towards this by extending the ideas presented here for $SL = L$. They reduced $RL = L$

to producing pseudorandom walks on regular digraphs in logarithmic space, with the only barrier being the labeling of the edges.

- Determining if Savitch's Theorem is optimal for $STCON$, the decision problem for s, t connectivity in any directed graph. Since $STCON$ is NL-complete, this could demonstrate L relative to NL. In Chung et al. (2007), they demonstrated an algorithm for solving $STCON$ in deterministic log-space if given the stationary probability distribution of the random walk on the graph.
- Better analysis of the zig-zag product, replacement product, and de-randomized squaring. In particular, the best bounds on λ we have are quite loose, and the constants in the bounds could be optimized. Rozenman and Vadhan (2005) presented an entirely different proof method, but the results are not tight for the zig-zag and replacement products.
- Determining the space and time trade offs for USTCON, and optimizing the running time. In Rozenman and Vadhan (2005), the de-randomized squaring algorithm obtains better performance than the methods using the zig-zag or replacement products. Improvements here also arise from tighter analysis of the zig-zag and replacement products.
- Understanding and devising more efficient graph operations for improving connectivity. The zig-zag product, for example, is relatively unnatural, and due to how complicated it is, is very difficult to analyze in a tight way. The de-randomized squaring operation, for example, is much more motivated, although the construction is still relatively unnatural. There could be more ideas here.

References

Alon, N. (1986). Egienvalues and expanders. In *Combinatorica*.

Alon, N. and Sudakov, B. (2000). Bipartite subgraphs and the smallest eigenvalue. In *Combinatorics, Probability and Computing*.

Chung, K.-M., Reingold, O., and Vadhan, S. (2007). S-t connectivity on digraphs with known stationary distribution. In *Proceedings of the 22nd Annual IEEE Conference on Computational Complexity (CCC)*.

Lewis, H. R. and Papadimitriou, C. H. (1982). Symmetric space-bounded computation. In *Theoretical Computer Science*.

Reingold, O. (2008). Undirected connectivity in log-space. In *Journal of the ACM*.

Reingold, O., Trevisan, L., and Vadhan, S. (2005). Pseudorandom walks in biregular graphs and the RL vs. L problem. In *Electronic Colloquium on Computational Complexity (ECCC)*.

Reingold, O., Vadhan, S., and Wigderson, A. (2002). Entropy waves, the zig-zag graph product, and new constant-degree expanders. In *Annals of Mathematics*.

Rozenman, E. and Vadhan, S. (2005). Derandomized squaring of graphs. In *Approximation, Randomization and Combinatorial Optimization*.

18.405J / 6.841J Advanced Complexity Theory

Spring 2016