1 Introduction

The relationship between memory use, nondeterminism, and randomization is central to space-bounded computation. Whether or not nondeterminism provides computation power that is not present in deterministic machines remains an open question. Whether or not randomness can save computation space, or conversely that all randomized algorithms can be derandomized at a small memory cost also remains a open problem. Our current understanding is that

$$L \subseteq SL \subseteq RL \subseteq NL \subseteq L^c$$

where

$$\begin{aligned} L &= \text{ problems solvable in deterministic logspace} \\ SL &= \text{ problems solvable in nondeterministic symmetric logspace} \\ RL &= \text{ problems solvable in polytime randomized logspace} \\ NL &= \text{ problems solvable in nondeterministic logspace} \\ L^c &= \text{ problems solvable in deterministic polylog space} \end{aligned}$$

Determining which of these containments are actually equalities form a basis of problems which computer scientists have made progress towards since the advent of the field. This survey will show one of these particular equalities, $L = SL$, a result proved by Omer Reingold in 2004 [1]. While the majority of this paper focuses on this result and summarizes the key points of his paper, we discuss briefly extensions of this proof in progress towards $L = RL$ as a consequence.

2 Background

This project explores the problem of determining $s - t$ reachability in deterministic log space in an undirected graph (an SL-complete problem). There are many simple algorithms that solve this problem; for instance, classical graph traversal algorithms such as depth-first search can solve this problem in polynomial time (putting $USTCON$ in P). The trouble with such an algorithm is that the stack depth it required is at worst $O(n)$ in the number of nodes of the graph, thus making it not a logarithmic-space algorithm. We will show that while it is possible to reduce this space complexity from linear to logarithmic, the process to do so is not at all trivial and requires careful consideration of the properties of the given graph. Thus, we will initially formalize how this problem is presented and how we analyze it. First, we make the following assumption: although subsequent analysis will deal with higher-level abstractions than Turing Machines, we can imagine our input graph as given to us on a read-only input tape. Therefore, our measure of space will be a measure of the auxiliary space we use on a work tape.

In addition, it turns out that we can place $USTCON$ in RL. Namely,

Lemma 2.1. *If s and t are connected in d-regular graph G, then a random walk of length $O(d^2n^3\log(n))$ from s reaches t with probability greater than or equal to $\frac{1}{2}$. Proof in Appendix.*

While neither DFS nor this probabilistic algorithm solve the question we're interested in, they do offer some valuable intuition: namely, that this question is easier on sparse (not very many choices to make at every node) and well connected graphs (if there exists a path from s to t then there exists a short path). Graphs which have these nice properties of being sparse yet well connected are known as **expander graphs** and are integral in this proof.

3 Expander Graphs and Their Properties

We begin with some basic definitions regarding graphs [2]:

Definition 3.1. *A graph is **undirected** iff its adjacency matrix is symmetric.*

Definition 3.2. *A graph is d-**regular** if each vertex has exactly d edges incident to it.*

For the following, let $G = (V, E)$ be an undirected, d-regular graph on n vertices. For every node, we fix an arbitrary indexing of edges. That is,

Definition 3.3. *For a node v, its edge assigning function is*

$$f_v : [d] \to V$$

where $f_v(i)$ is the ith edge adjacent to v (assigned arbitrarily).

In traditional representations of graphs, traversing an edge from u to v does not track which edge (index) was taken, and instead specifies with respect to the two nodes themselves. We define another graph representation, which instead defines edges with respect to a single node and an index according to its edge assigning function:

Definition 3.4. *Define the **Rotation Map** of G as*

$$Rot_G : [n] \times [d] \to [n] \times [d]$$

where

$$Rot_G(v, i) \mapsto (w, j)$$

if $f_v(i) = w$ and $f_w(j) = v$.

For the rest of this survey, it is convenient to consider non-bipartite, d-regular graphs for some $d \geq 3$ constant. While these are certainly not properties of every graph, it is easy to convert an arbitrary graph to one with these properties while preserving $s - t$ connectivity between every pair of nodes. Namely,

1. To transform a bipartite graph into a non-bipartite graph, simply add a self-loop at every vertex. This clearly does not change pairwise connectivity, and also causes any graph to no longer be bipartite.
2. To transform an arbitrary graph into a d-regular graph, simply replace every vertex with a cycle of length equal to its degree, giving a 3-regular graph. Then, add as many self-loops as necessary to reach degree d. An example is illustrated below.

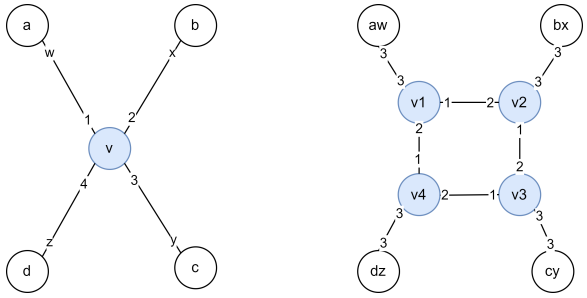


Figure 1: Degree 3 transformation on a single node of degree 4

A key fact for our analysis is that neither of these transformations requires the new graph to be explicitly stored (which is important for space-bounded computation). Instead, we can just modify the rotation map: to traverse (v, i), simply traverse $(v_i, 3)$ where we assume 'non-cycle' edges to be indexed as 3. Therefore, from here on out we can assume without loss of generality that any graph we consider is d-regular, and non-bipartite.

Definition 3.5. *An expander graph is a graph which is simultaneously well connected and sparse. Over the following definitions, we'll explore different metrics which measure how 'good' of an expander a graph is.*

Definition 3.6. *The **normalized adjacency matrix** or **transition probability matrix** M_G of G is the adjacency matrix representation of G normalized by d. Equivalently,*

$$M_{u,v} = \frac{1}{d}\left|(i,j) \in [d]^2 \mid Rot_G(u,i) = (v,j)\right|$$

For undirected, d-regular graphs, M_G will be regular and symmetric, so it has a set of eigenvectors which is a basis for $\mathbb{R}^V$. The eigenvector $\mathbf{v}_1 = [1 \ \cdots \ 1]^T$ has eigenvalue $\lambda_1 = 1$. Define $\lambda(G) = \lambda_2$.

Lemma 3.7. *If G is a connected, d-regular graph, then the spectrum of M_G is $\lambda_1 > \lambda_2 \geq \cdots \geq \lambda_n$ where $\lambda_1 = 1$. Larger spectral gaps correspond to better expansion properties of the graph.*

Proof. We've argued that 1 is always an eigenvalue. Suppose there exists some other $\lambda > 1$. Then, there exists corresponding eigenvector v such that $M_G v = \lambda v$. Suppose the entry v_k is the largest in v. Then,

$$(M_G v)_k = \frac{1}{d}\sum_{\{j,k\}\in E} v_j \leq \frac{1}{d}(dv_k) = v_k$$

However, $\lambda v_k > v_k$, a contradiction. Therefore, every eigenvalue is at most 1.

Now, we use the connectedness of G to argue that there is only one eigenvector. The quickest way to do this is to use Perron-Frobenius: M_G has no invariant coordinate subspaces (i.e. every proper subset S of the vertices has neighbors not in S), so it is an irreducible matrix and thus the multiplicity of the eigenvalue is 1. □

Definition 3.8 (spectral expansion γ)**.** *A graph G has spectral expansion γ if $\lambda(G) \leq 1 - \gamma$.*

The spectral gap can be hard to conceptualize. Here is some intuition: Suppose we are finding the probability distribution for the end of a random walk, i.e. $M^\ell \boldsymbol{\pi}$ for starting position $\boldsymbol{\pi}$.

$$\boldsymbol{\pi} = \sum_{i=1}^{n} c_i \mathbf{v}_i \implies M^\ell \boldsymbol{\pi} = \sum_{i=1}^{n} \lambda_i^\ell c_i \mathbf{v}_i$$

Since λ_1 is always 1, the value of this sum is most impacted by λ_2.

Definition 3.9. *An undirected, d-regular graph G on n vertices with $\lambda(G) = \lambda$ is referred to as an (n, d, λ)-expander graph from this point forwards. We note that it is possible to think of any graph as an expander (with potentially zero or very poor expansion).*

Definition 3.10 ((K, A) vertex expansion)**.** *A graph G is a (K, A) vertex expander if for every set S of size at most K, the neighborhood $N(S) = \bigcup_{v\in S} N(v)$ has size at least $A \cdot |S|$.*

Theorem 3.11 (spectral expansion $\implies$ vertex expansion)**.** *If G is a regular digraph with $\gamma = 1 - \lambda$ spectral expansion for some $\lambda \in [0, 1]$, then for every $\alpha \in [0, 1]$, G is an $\left(\alpha n, \frac{1}{(1-\alpha)\lambda^2 + \alpha}\right)$ expander. In particular G is a $\left(\frac{n}{2}, 1 + \gamma\right)$ expander.*

Proof. Adapted from [3]. To prove this, we first give some definitions and lemmas which will help us get to the result of spectral expansion implies vertex expansion.

Definition 3.12. *For some probability distribution π, the collision probability, denoted $CP(\pi)$, is the probability that two independent samples from π are equal.*

Definition 3.13. *The support of some probability distribution π, denoted $Supp(\pi)$, is $\{x \mid \pi_x > 0\}$.*

Lemma 3.14. *If π is a probability distribution over the vertices, and u is the uniform probability distribution over the vertices, then*

$$CP(\pi) = \|\pi\|^2 = \|\pi - u\|^2 + \frac{1}{n}$$

Proof. The first equality is, by definition, of collision probability. Then, we note that $(\pi - u) \perp u$ so

$$\|\pi\|^2 = \|u\|^2 + \|\pi - u\|^2$$

Finally, we note that $\|u\|^2$ is just $\frac{1}{n}$ since u is the uniform distribution. □

Lemma 3.15.

$$CP(\pi) \geq \frac{1}{|Supp(\pi)|}$$

Furthermore,

$$CP(\pi) = \frac{1}{|Supp(\pi)|} \iff \pi \text{ is uniform on } Supp(\pi)$$

Proof.

$$\begin{aligned}
1 &= \sum_{x \in Supp(\pi)} \pi_x && \text{by the definition of Supp} \\
&\leq \sqrt{|Supp(x)|} * \sqrt{\sum_x \pi_x^2} && \text{by Cauchy-Schwartz (with equality iff uniform)} \\
&= \sqrt{|Supp(\pi)} * \sqrt{CP(\pi)} && \text{by definition of CP}
\end{aligned}$$

We note that the proof can be made somewhat simpler by applying the method of the previous proof, and comparing π with the uniform distribution over the support of π. □

Finally, we can finish up the proof of Theorem 3.11. First, note that since M_G is a real symmetric matrix, its eigenvectors are orthogonal. Thus, letting $u = \begin{bmatrix} \frac{1}{n} & \cdots & \frac{1}{n} \end{bmatrix}^T$ be the eigenvector of eigenvalue $\lambda_1 = 1$, we have $\lambda_2 = \max_{v : v \perp u} \frac{\|Mv\|}{\|v\|}$. For each such v, we can find a probability distribution π over the vertices such that $\pi - u$ is a multiple of v (and vice versa); thus, $\lambda_2 = \max_{\pi} \frac{M_G(\pi - u)}{\pi - u}$. Using this alternate definition for λ, we have from the previous lemmas that

$$\begin{aligned}
CP(M_G\pi) - \frac{1}{n} &= \|M_G\pi - u\|^2 \\
CP(M_G\pi) - \frac{1}{n} &= \frac{\|M_G(\pi - u)\|^2}{\|\pi - u\|^2} \cdot \|\pi - u\|^2 \\
&\leq \lambda^2 \|\pi - u\|^2 \\
&= \lambda^2 \left[CP(\pi) - \frac{1}{n} \right]
\end{aligned}$$

Consider any subset S of up to αn vertices. If we take π to be the uniform distribution on these vertices and use the above equation, we can learn about $N(S)$: in particular, $\frac{1}{|N(S)|} \leq CP(M_G\pi)$ so

$$\begin{aligned}\left(\frac{1}{|N(S)|} - \frac{1}{n}\right) &\leq \lambda^2 \cdot \left(\frac{1}{|S|} - \frac{1}{n}\right) \\ \frac{1}{N(S)} &\leq \frac{\lambda^2}{|S|} + \frac{1-\lambda^2}{n} \\ \frac{1}{N(S)} &\leq \frac{\lambda^2}{|S|} + \frac{\alpha}{|S|n} \\ |S| \cdot \frac{1}{\lambda^2 + \alpha(1-\lambda^2)} &\leq |N(S)|,\end{aligned}$$

where we lower bound n by $|S|/\alpha$ because the blue quantity is positive.

□

Lemma 3.16. *For every d-regular, connected, non-bipartite graph G on $[n]$, we have that $\lambda(G) \leq 1 - \frac{1}{dn^2}$.*

Proof. [4], [5] Suppose v is an eigenvector with eigenvalue $\lambda(G)$. Then, we know it is orthogonal to the first eigenvector $v_1 = [1 \cdots 1]^T$ so $\sum_{i=1}^{n} v_i = 0$. By a basic counting argument, we see that

$$\begin{aligned}v^T(1-\lambda_2)v &= v^T(I - M_G)v \\ &= \sum_{i=1}^{n} v_i^2 - \frac{2}{d}\sum_{\{i,j\}\in E} v_i v_j \\ &= \frac{1}{d}\sum_{\{i,j\}\in E}(v_i - v_j)^2,\end{aligned}$$

Normalize v so that $\|v\|^2 = 1$, $v_i > 0$ and $v_j < 0$ are the largest and smallest entries–note that they must have opposite sign because all the entries sum to 0. Then, find a path $i, u_1, \ldots, u_\ell, j$ in G and keep only the terms in the previous summation which correspond to edges along the path:

$$\begin{aligned}1 - \lambda_2 &\geq \frac{1}{d}\left[(v_i - v_{u_1})^2 + \sum_{k=1}^{\ell-1}(v_{u_k} - v_{u_{k+1}})^2 + (v_{u_\ell} - v_j)^2\right] \\ &\geq \frac{1}{d(\ell+1)}\left[(v_i - v_{u_1}) + \sum_{k=1}^{\ell-1}(v_{u_k} - v_{u_{k+1}}) + (v_{u_\ell} - v_j)\right]^2 \\ &= \frac{1}{d(\ell+1)}(v_i - v_j)^2 = \frac{1}{d(\ell+1)}(|v_i| + |v_j|)^2 > \frac{1}{dn^2},\end{aligned}$$

where we used Cauchy-Schwarz on $\mathbf{a} = [1 \ \cdots \ 1], \mathbf{b} = [v_i - v_{u_1} \ \cdots \ v_{u_\ell} - v_j]$ to get the second line and used that $\|v\|^2 \leq n(\max(|v_i|, |v_j|)^2 = 1 \implies (|v_i| + |v_j|)^2 > \frac{1}{n}$ and $\ell + 2 \leq n \implies \frac{1}{\ell+1} > \frac{1}{n}$ in the last line. □

Remark 3.17. *In the above lemma we used a trivial upper bound for ℓ, so that the result matches lemma 2.5 in the paper. A similar technique can be used to show that $\lambda_n \geq -1 + \frac{1}{nd(D+1)}$, and this result is cited in the paper. But the proof of the original lemma is not given, perhaps it is too trivial or standard.*

Suppose G a graph with N vertices is a $\left(\frac{N}{2}, A\right)$ vertex expander for some $A > 1$. In this specific case, we note some additional properties. Namely,

Lemma 3.18. *G is connected.*

Proof. Suppose for a contradiction that G is not connected. Let S be the smallest connected component (by number of vertices) of G. We know by assumption that $|S| \leq \frac{N}{2}$. By the definition of vertex expander, $|N(S)| \geq A|S|$. However, by assumption, $N(S) = S$, a contradiction. Therefore, G must be connected. □

Lemma 3.19. *For any two $u, v \in V$ there is a path of $O(\log N)$ edges from u to v.*
More concisely, G has logarithmic diameter.

Proof. Consider arbitrary vertices s, t in G. We argue that they are at most $O(\log N)$ apart. By the definition of vertex expander, the number of vertices within i of S is at least $A^i = (1+\epsilon)^i$. That is, there are more than $\frac{N}{2}$ vertices within $O(\log(n))$ distance of s. Make the analogous argument from t (since the expansion only applies for sets at most size $\frac{N}{2}$), and find a vertex which is within $O(\log(n))$ hops from both s and t. Finally, stitch the paths together to conclude that the distance between s and t is $O(\log N)$. □

Theorem 3.20. *There exists some constant D_e and a $((D_e)^{16}, D_e, \frac{1}{2})$-graph. Proof sketch given in Appendix.*

3.1 The Powering Graphs and Zig-Zag Product

It is not guaranteed that our input graph G is a "good" expander; thus, we need to apply transformations to it in order to increase its expansion while not drastically increasing its degree (and, of course, while maintaining connectivity). The transformations we will consider here are graph powering and the zigzag product. Powering a graph is relatively straightforward:

Definition 3.21. *For any graph G, G^k is the kth power of G, determined by taking the normalized adjacency matrix to the kth power. Interpreted visually, the kth power of the graph adds edges between s and t in G if there is a path of length k between s and t in G.*

This operation is helpful because it improves connectivity properties. However, the increased connectivity does come at a cost:

Lemma 3.22. *If G is an (N, D, λ) graph, then G^k is an (N, D^k, λ^k) graph.*

Proof. The normalized adjacency matrix of G^k is simply the normalized adjacency matrix of G raised to the kth power. Thus, the degrees and eigenvalues all get raised to the kth power. □

That is, while powering a graph improves connectivity, it simultaneously increases degree. Recall that we'd like expander graphs to be relatively sparse. Thus, we introduce the zigzag product: it is a product on two graphs which is designed to reduce degree without affecting expansion too much. More specifically, given some (N, D, λ)-graph G and some other (D, d, α)-graph H (these are importantly the same D), we can define the zigzag product as follows:

- $G ⓩ H$ is a graph with nodes in $V(G) \times V(H)$

- Its edges, defined in terms of a rotation map, are as follows:

Algorithm 1 Computing the Rotation Map of the Zigzag Product

procedure $Rot_{G\textcircled{z}H}((a,x),(i,j))$
  $(a',i') \leftarrow (Rot_H(a,i))$
  $(w,b') \leftarrow (Rot_G(v,a'))$
  $(b,j') \leftarrow (Rot_H(b',j))$
  return $((w,b),(j',i'))$
end procedure

Qualitatively, we can imagine making a copy of H_v for each node $v \in V(G)$ and connected each "cloud" to every other cloud. Then, the edges in $G\textcircled{z}H$ are a subset of paths of length 3 where the first and third edges are within a single H-cloud, and the second edge is between H-clouds. That is, a single edge in our zigzag product graph, is of the form short-long-short. We take enough of such paths such that the rotation map as defined above is well and uniquely defined.

Lemma 3.23. *If H has degree d, then $G\textcircled{z}H$ has degree d^2.*

Proof. Edges in the zigzag product are defined by two edge indices in H: that is, it is two 'short' edges. Since each 'short' edge can be identified by a value in d, it follows that degree is d^2. Critically, for constant sized d, d^2 is also constant. □

Lemma 3.24. *Given the restrictions on G and H earlier, we have that $\lambda(G\textcircled{z}H) \leq 1 - \frac{1}{2}\left(1-\alpha^2\right)(1-\lambda)$. Namely, the expansion properties of $G\textcircled{z}H$ are not that much worse than those of G or H.*

Proof. In [6] show the following theorem:

Theorem 3.25. *If G is an (N,D,λ)-graph and H is a (D,d,α) graph, then $G\textcircled{z}H$ is an $(ND,d^2,f(\lambda,\alpha))$ graph, where*

$$f(\lambda,\alpha) = \frac{1}{2}(1-\alpha^2)\lambda + \frac{1}{2}\sqrt{(1-\alpha^2)^2\lambda^2 + 4\alpha^2}$$

.

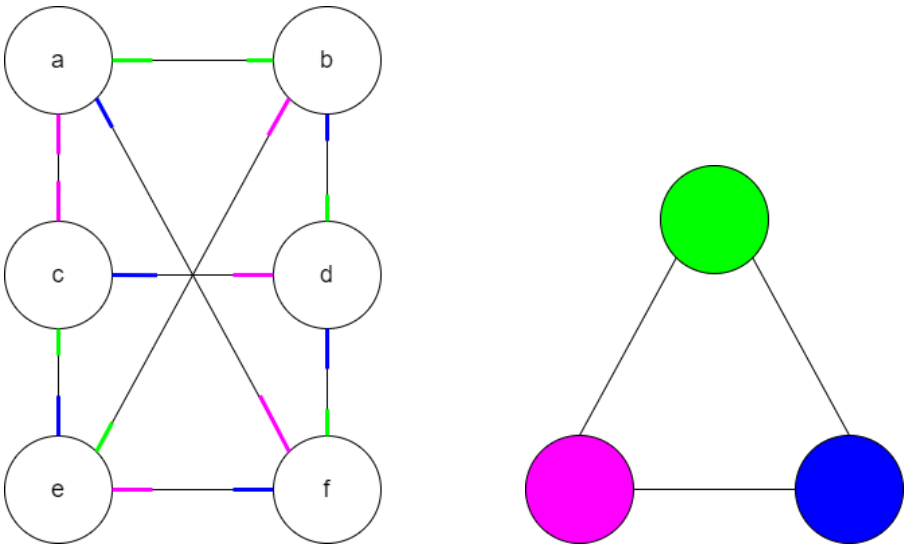
From this, we note that $\lambda \leq 1$, and write

$$\begin{aligned}\frac{1}{2}(1-\alpha^2)\lambda + \frac{1}{2}\sqrt{(1-\alpha^2)^2\lambda^2 + 4\alpha^2} &\leq \frac{1}{2}(1-\alpha^2)\lambda + \frac{1}{2}\sqrt{(1-\alpha^2) + 4\alpha^2} \\ &= \frac{1}{2}(1-\alpha^2)\lambda + \frac{1}{2}(1+\alpha^2) = 1 - \frac{1}{2}(1-\alpha^2)(1-\lambda).\end{aligned}$$

From this, we can see that the spectral gap of $G\textcircled{z}H$, or $1 - f(\lambda,\alpha)$, is bounded from below by $\frac{1}{2}(1-\alpha^2)(1-\lambda)$; as $(1-\lambda)$ is the spectral gap of G, and $(1-\alpha^2)$ is bounded from below by $(1-\alpha)$ for $\alpha \in [0,1]$, we have that this quantity is also not much worse than a constant factor times the spectral gap of H. □

3.2 Examples

Consider the following two graphs G and H:

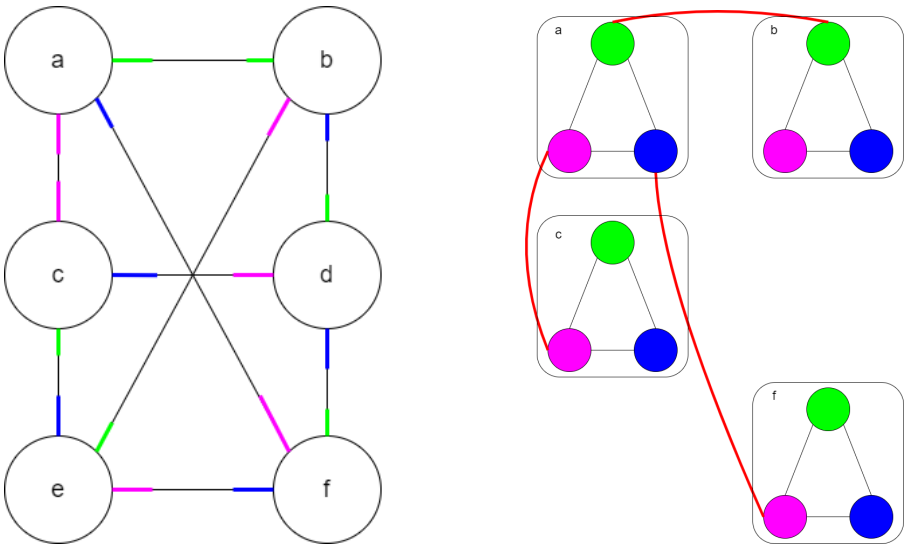


(a) Graph G with constant degree 3

(b) Expander H with 3 nodes

Figure 2: 2 graphs for which the zigzag product is well defined.

The colors are simply a visual way of representing edge indices which may be easier for our reader to manually verify. Notably, the colored edges correspond to the colors of the nodes in the expander. Consider the effect of the zigzag product on node a and its neighbors (the rest of the graph is transformed analogously):



(a) Graph G with constant degree 3

(b) Subedges in the zigzag of G and H around node a

Figure 3: Zigzag product applied on node a, showing subedges

We note that every node (i.e. a and its neighbors in this local example) has been replaced by a copy of H. Also, the edge between cloud a and cloud b is connected via the green node in the expander, since that corresponds to the edge coloring shown on the left. In the resulting graph on the right, the

black edges are short edges and the red edges are long edges. Recall that the shown edges here are not actually edges in the zigzag product: the edges of the zigzag product are actually a subset of all paths of length 3 in this graph of the form short-long-short. Visually:

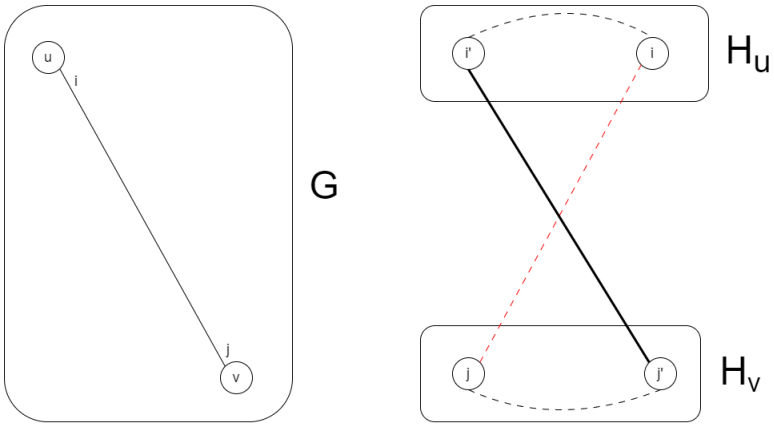


Figure 4: Edges in the zigzag product

The dotted lines above show the red and black edges we pictured earlier. The edges in the final zigzag product will, however, correspond to the solid black line. As we consider the zigzag product in the proof of *USTCON*, it will often be helpful to think of edges of the zigzag product as paths of length 3 in the underlying graphs.

4 Connectivity Algorithms

4.1 Connectivity in Log-Diameter Graphs

Definition 4.1. *Let the diameter of some graph G be the length of the longest shortest path between any 2 nodes.*

Then, the following lemma follows easily:

Lemma 4.2. *If G is a d-regular undirected graph over n vertices for $d = O(1)$ and diameter of $\ell = O(\log n)$, then we can determine $s - t$ connectivity in $\log^2(n)$ space. This follows from just running a DFS, which will have at most logarithmically many recursive calls.*

It is actually quite easy to improve this algorithm to run in $O(\log(n))$ space. In a normal DFS exploration, we keep track of our path of visited nodes, each of which takes $\log(n)$ space to store. However, we have the handy feature of d-regularity which we do not adequately exploit. Therefore, if we simply store indices of forwards and backwards pointers, rather than the nodes themselves, we have reduce the stack space of each recursive call down to $O(1)$.

Lemma 4.3. *If G is a d-regular undirected graph over n vertices for $d = O(1)$ and diameter of $\ell = O(\log n)$, then we can determine $s - t$ connectivity in $\log(n)$ space.*

Proof. We make use of the rotation map definition of graphs. We also define the inverse of the rotation map to take in u, v and return the index δ for which $Rot(v, \delta) = u$. Then,

Algorithm 2 DFS on Undirected Graph of Log Diameter in Logspace

```
i ← L                                          ▷ i is a global variable in memory
u ← s                                          ▷ u is a global variable in memory
procedure DFS
    if i = 0 then return false
    else if u = t then return true
    else
        i ← i − 1
        for δ ∈ {1, ..., d} do
            (v, δ) ← Rot(u, δ)
            u ← v
            DFS()                              ▷ DFS is called recursively at most ℓ times
            (u, _) ← Rot(u, δ)
        end for
        i ← i + 1
    end if
end procedure
```

Now, in every stack frame, we just need to store δ, which is constant sized. Therefore, we have 1 log-space element in global memory, combined with log stack-space for the recursive procedure, for a total of $O(\log(n))$ as desired. □

Therefore, if we can transform our graph into this structure: d-regular for constant sized d and logarithmic in diameter, then Algorithm 2 will give us a procedure for deterministic undirected reachability. We previously showed in Figure 1 that we can achieve constant degree. Therefore, the rest of this paper focuses on how to achieve log diameter. That is, we define some transformation τ such that $\tau(G)$ is c-regular for some constant c and logarithmic in diameter, and also achieves precisely the same connectivity properties as G, and we can implicitly compute $\tau(G)$ is logarithmic auxiliary space.

4.2 Transforming an Arbitrary Graph into an Expander

Definition 4.4. *Given some* $\left(D^{16}, D, \lambda\right)$ *graph* G *and* $\left(D, d, \frac{1}{2}\right)$ *expander* H *for* $D = O(1)$*, we define the following family of graphs inductively:* $G_0 = G$, $G_{t+1} = (G_t ⓩ H)^8$.

Lemma 4.5. *Given some graph* G *that is an* (N, D, λ) *expander, for any* $s, t \in V(G)$*,* s, t *are connected if and only if* $(s, 1), (t, 1)$ *are connected in* $G ⓩ H$*. That is, the zigzag product preserves connectivity on expander graphs.*

We omit the proof of the above lemma as it is shown directly in [1].

Lemma 4.6. G_ℓ *has spectral gap of at least* $\frac{1}{2}$ *for some* $\ell = 2\lceil \log DN^2 \rceil \in O(\log N)$.

Proof. Recall Lemma 3.24 with $\alpha = \frac{1}{2}$. Then, we have that the expansion of $G ⓩ H$ is bounded by

$$1 - \frac{1}{2}\left(1 - (\frac{1}{2})^2\right)(1 - \lambda) = \frac{3}{8}(1 - \lambda)$$

Intuitively, the above bound shows that taking the zig-zag product with H does not greatly decrease the expansion properties of G. We also note that powering a graph by a factor of t also raises the bound of the magnitude of the second-largest eigenvalue by a power of t.

We also have from Lemma 3.16 that for every D-regular, connected, non-bipartite graph on $[N]$, a bound for the magnitude of its second largest eigenvalue is $1 - 1/DN^2$.

From these facts, we now lower bound the spectral gap of G_i (by upper bounding the magnitude of its second-largest eigenvalue). We first note that $(1 - 1/DN^2)^2\ell < \frac{1}{2}$; thus, all we have to show that is that for a given transformation, $\lambda(G_i) \leq \max(\lambda(G_{i-1})^2, \frac{1}{2})$, as in the first case, the transformation will make the second largest eigenvalue closer to $\frac{1}{2}$ at least as quickly as squaring it would (and squaring it ℓ times will always get it under $\frac{1}{2}$), and in the second case, the second largest eigenvalue is already $\frac{1}{2}$.

To show this, note that $\lambda(G_{i-1} ⓩ H) \leq 1 - \frac{3}{8}(1 - \lambda) < 1 - \frac{1}{3}(1 - \lambda)$, where $\lambda = \lambda(G_{i-1})$. Raising both sides to the power of 8 (as graph powering also raises all its eigenvalues to that power), we get $\lambda((G_{i-1} ⓩ H)^8) = \lambda(G_i) < (\frac{1}{3}(1 - \lambda))^8$. Finally, we can verify that if $\lambda < \frac{1}{2}$, then $\lambda(G_i) < \left(\frac{5}{6}\right)^8 < \frac{1}{2}$, and otherwise, that $(\frac{1}{3}(1 - \lambda))^4 \leq \lambda$ and thus that $\lambda(G_i) \leq \lambda^2$. The lemma then indeed follows. □

4.3 Undirected Connectivity Implementation in Logspace

We'd like to simply compute G_L as defined above, and then run Algorithm 2. However, we cannot actually compute out the graph G_L explicitly since storing it will take too much space. We are only able to store the complete original graph because it is given to us on the read-only input tape, which we cannot overwrite. Thus, any explicitly computed variation of G will take more space than what we have. We therefore give a procedure which computes transitions in G_L on the fly; in particular, we define an algorithm which can compute specific values of Rot_t in log space.

Suppose we have a log-space algorithm $Rot_t(\gamma)$. For some global variable u in memory, it traverses from u along the γth edge in the graph G_t. If u's γth neighbor is some node v, then this procedure reassigns the spot in memory for u to v and recurses. Finally, it returns γ', the index for which u is v's γ'th neighbor. Then, we can modify our previous algorithm to work on G_L rather than G by computing values of Rot_t on the fly:

Algorithm 3 DFS on an Implicit Graph

$i \leftarrow L$ ▷ i is a global variable in memory
$u \leftarrow (s, 1^{\ell+1})$ ▷ u is a global variable in memory
procedure DFS ▷ Note that we no longer take an argument node
  if $i = 0$ **then return** false
  end if
  if $u = (t, 1^{\ell+1})$ **then return** true
  end if
  $i \leftarrow i - 1$
  for $\gamma \in \{1, \ldots, D_e^{16}\}$ **do**
    $\gamma' \leftarrow ROT_\ell(\gamma)$ ▷ Discard all but constant space after subroutine call
    DFS() ▷ DFS is called recursively at most ℓ times
    $ROT_\ell(\gamma')$ ▷ This backtracks back to original u
  end for
  $i \leftarrow i + 1$
end procedure

Since we are now traversing on G_L rather than G, we cannot just start with s and look for t as these are not nodes in G_L. If we consider the definition of G_L, we know

$$V(G_L) = V((G_{L-1} ⓩ H)^8) = V(G_{L-1} ⓩ H) = V(G_{L-1} \times H) = V(G_0 \times \cdots \times H) = (v, x)$$

where $v \in V(G_0), x \in [D_e]^{\ell+1}$. Therefore, our starting node we fix to be $(s, 1^{\ell+1})$ and our ending node is $(t, 1^{\ell+1})$. We note that since ℓ is logarithmic, 1^ℓ takes logarithmic space: therefore, storing the identifier of a node in this derived graph is still log-space.

Since i is initialized to be $\ell \in O(\log(n))$, this algorithm makes at most $\log(n)$ recursive calls. At each depth level, we need to store γ across the recursive call. γ is by definition a constant. At each depth level, we also need to do all the work associated with making a call to ROT_t. However, this is stack space that is reusable. Therefore, at any one instant, this procedure uses at most $\log(n) + \log(n) * O(1)$ (space used to compute ROT plus log depth levels times constant space per level), which is indeed logarithmic.

Therefore, it remains to show the procedure to compute ROT.

Algorithm 4 Computing Values of $Rot_t(u, \delta)$ for Implicit Graph G_t

procedure $ROT_t(\gamma)$ ▷ Returns only the backwards index: takes too many bits to store nodes
  if t = 0 **then**
    Look up $Rot(u, \gamma)$ for G_0
  end if
  $u \in V((G_{t-1})ⓩH)^8$
  $\gamma = (\gamma_1, \ldots \gamma_8) \in [D_e^2]^8$
  Allocate memory for $\gamma' \in [D_e^2]^8$
  for $i = 1 \ldots 8$ **do**
    $u = (v_i, a_i) \in V(G_{t-1}) \times V(H)$
    $\gamma_i = (\alpha_i, \beta_i) \in [D_e]^2$
    ▷ Compute $Rot_{G_{t-1}ⓩH}(u_i, \gamma_i)$
    $(a_i', \alpha_i') = Rot_H(a_i, \alpha_i)$ ▷ SHORT: Read directly from H
    $u \leftarrow v_i$ ▷ Traverse one node forwards
    $b_i' = ROT_{t-1}(a_i')$ ▷ LONG: Recursive call to procedure
    $(b_i, \beta_i') = Rot_H(b_i', \beta_i)$ ▷ SHORT: Read directly from H
    $\gamma_i' = (\beta_i', \alpha_i')$ ▷ Use previously allocated memory to store γ_i'
    $u \leftarrow (u, b_i)$
  end for
  return γ' ▷ This procedure DFSes from u. γ' returns the backpointer.
end procedure

A single edge in G_L is a series of 8 edges in $(G_{L-1}ⓩH)$, which we represent with u_i's. Similarly, γ is really a sequence of indices in H. From there our computations are exactly those of the definition of the zigzag product, only when we go to evaluate Rot_{L-1}, we do not have this value anywhere and must recursively compute it using this procedure. We also can't pass around nodes as arguments and returns, as they take up too much memory, so we allocate a single global variable that we overwrite using saved pointers (which are constant space). We note that since L is logarithmic, this procedure will recurse $O(\log(n))$ times. At each level of recursion, we track γ, γ' which are both constant space. We also have auxiliary variables such as i, t which are clearly constant. Therefore, this recursive procedure can be completed in log space.

5 Consequences, & Extensions

5.1 SL = L

Recall that SL is the space of problems solvable with a nondeterministic symmetric Turing Machine. Therefore, we can think of this as a problem of determining whether there exists a path between start and accepting configurations in a symmetric Turing machine, making $USTCON$ a complete problem for the complexity class. As $USTCON$, an SL-complete problem, has now been shown to be solvable in deterministic log space, this has the implication that $SL = L$, thus rendering the SL complexity class, which had been developed specifically for the purposes of analyzing $USTCON$ and its equivalent problems, effectively useless. All these problems were now also simply in L.

5.2 Progress Towards RL = L

Reingold's result regarding $USTCON$ can be extended towards showing derandomization of RL. Towards the beginning of this survey, we saw an RL algorithm for $USTCON$ - therefore, the proof we've presented was actually a specific case of RL derandomization. It turns out that we can generalize this

proof more. In particular, we can show a deterministic log-space algorithm for directed s-t connectivity in graphs such that every node has the property where its in-degree equals its out-degree. The below is a high level summary of Reingold's approach [7] to this problem - as it is not the focus of our survey, we omit proofs and low level details.

Before discussing the approach, it is helpful to consider the structure undirected graphs give (as opposed to directed graphs). The obvious difference is that edges are reversible. Another property is the fact that in undirected graphs, the stationary distribution of a walk is such that the probability of a vertex is proportional to its degree. This well-behaved-ness turns out to be central, and indeed directed graphs with the additional structure of equal in- and out-degree also have this property. The key idea with the well-behaved stationary distribution is that such graphs can be reduced to regular graphs where the stationary distribution is uniform. The study of directed graphs requires us to modify/expand the relevant terminology we use. In particular:

Definition 5.1. *An $n \times n$ matrix M is called a Markov chain on the state space $[n]$ if for every $u \in [n]$, it holds that $\sum_v M(u,v) = 1$. The underlying graph is $G = ([n], E)$ where $(u,v) \in E$ if and only if $M(u,v) > 0$*

Definition 5.2. *A distribution $\pi \in \mathbb{R}^n$ is stationary for Markov chain M if $M\pi = \pi$.*

Definition 5.3. *A Markov chain M is time reversible with respect to π stationary if for every pair of $u, v \in [n]$ it holds that $\pi(u)M(v,u) = \pi(v)M(u,v)$. Notably, symmetric markov chains (i.e. d-regular undirected graphs) are time reversible. Random walks on directed graphs are therefore not typically reversible.*

Central to this proof is the rate at which Markov chains converge to a stationary distribution. For a time-reversible Markov chain, this is characterized by the second largest eigenvalue, which is why it was of interest in our study of undirected graphs. In the directed case, we need to make a generalization of the second largest eigenvalue for situations in which all our eigenvalues may not be real.

Definition 5.4.

$$\langle x, y \rangle_\pi = \sum_{v \in supp(\pi)} \frac{x(v) \cdot y(v)}{\pi(v)}$$

Norms w.r.t. this inner product are as expected. That is, π becomes a unit vector.

With this modified inner product, we can define our generalization of spectral expansion in the directed case:

Definition 5.5.

$$\lambda_\pi(M) = \max_{x \in \mathbb{R}^n | \langle x, \pi \rangle_\pi = 0} \frac{\|Mx\|_\pi}{\|x\|_\pi}$$

This definition of spectral expansion captures the same key idea as the second largest eigenvalue in the undirected case: namely, when λ_π is small, the Markov chain converges quickly to π. Therefore, both in the *USTCON* algorithm previously presented as well as this one, we're really interested in the long-term behavior of random walks.

It turns out that most of the *USTCON* algorithm may be translated to this new setting. However, since the previous analysis on the zig-zag product and spectral expansion relied on adjacency matrices being symmetric, they do not extend and it is necessary to bound the spectral gap of the zig-zag product of two regular digraphs (here, we define regular to be such that every vertex has the same the same in-degree, and every vertex has the same out-degree). For a regular digraph, the uniform distribution is always stationary - we are therefore generally interested in spectral expansion w.r.t. π being the uniform distribution and omit it from our notation.

First, we note that powering also improves expansion in directed graphs. Equivalently, you can think of this as reducing mixing time (rate of convergence to uniform distribution).

Lemma 5.6. *For any stationary distribution π of G, $\lambda_\pi(G^t) \leq \lambda_{pi}(G)^t$. Proof omitted.*

Unfortunately, the directed version suffers from the same problem as in the undirected case: namely, that degree increases. Fortunately, the zigzag product on directed graphs is also a degree reducing operation, producing a product graph with degree the square of the expander. Therefore, it remains to show that the zigzag product does not decrease the spectral gap too badly.

First, we note that regular digraphs have a similar bound on their second-largest eigenvalue:

Lemma 5.7. *Let G be a connected, D-regular digraph on N vertices in which every vertex has at least αD self-loops (aperiodic). Then, $\lambda(G) \leq 1 - \Omega\left(\frac{\alpha}{DN^2}\right)$. This lemma may be thought of as the directed analogy to Lemma 3.16. Proof omitted.*

Using this lemma, we can prove the bounds we'd like on the zig-zag product in the directed setting. That is,

Theorem 5.8. *If $\lambda(G_1) \leq 1 - \gamma_1$ and $\lambda(G_2) \leq 1 - \gamma_2$ then $\lambda(G_1 ⓩ G_2) \leq 1 - \gamma_1 * \gamma_2^2$. Proof omitted.*

Having characterized spectral expansion in a generalization for directed graphs and proven relevant bounds on powering and the zigzag product, we can give present an identical algorithm. Namely,

1. Transform our regular graph into a new graph such that each connected component is an expander. As before, we can define this transformation recursively as $G_i = (G_{i-1} ⓩ H)^{40}$
2. Then, in our log diameter graph, we can give the same implicit DFS as previously argued.

While this algorithm only applies to regular digraphs, a fairly strong restriction, it has some interesting consequences regarding arbitrary graphs.

Definition 5.9. *A graph and associated edge labelling are considered consistently labelled if all vertices have equal in-degree and out-degree, and our labelling is such that $ROT(u, i) = (v, i)$. That is, the labellings of any edge are the same from the perspective of either endpoint.*

As it turns out, every regular digraph can be consistently labelled. We can create a pseudorandom walk generator on consistently labelled graphs. In our pathfinding algorithm, we created a graph G_L. Due to its expansion properties, it holds that a short random walk will converge to the uniform distribution. That is if we start at some vertex and follow the pseudorandom walk, we'll end at an almost uniformly distributed vertex. Since we assumed that the labels of an edge (u, v) was identical both as an outgoing edge from u and an incoming edge of v, it holds that actually the edge labels associated with the walk are independent of G and the starting vertex. Therefore, the edge labels taken can be computed without G_L itself, and therefore require only a small amount of space.

An important consequence is that if we could generalize this pseudorandom walk generator to all regular graphs (even ones without consistent labelling), that would imply $RL = L$. This is due to the fact that the so called poly-mixing s-t connectivity problem is RL complete. This question asks whether the random walk on input graph G has a stationary distribution π such that $\lambda_\pi(G) \leq 1 - \frac{1}{k}$ and $\pi(s), \pi(t) \geq \frac{1}{k}$ where k is another input parameter.

6 Appendix

6.1 Proof of $USTCON \in RL$

Perhaps the first major breakthrough regarding understanding *USTCON* was made in 1979 [8] which showed that it is solvable in randomized logspace with onesided error (RL). In particular, they were able to show that a random walk from an arbitrary vertex of a connected component will visit the other vertices in a polynomial number of steps. The consequence of this idea was that the algorithm which simply takes a random walk (storing current vertex and a counter) executes with only logspace use. We give an overview of the proof for this below:

Consider G an undirected graph with n vertices and e edges. Then, we can characterize a random walk on G as a Markov chain where states are probabilities and the transition probabilities give neighboring vertices equal probability $P_{i,j} = \frac{1}{d(i)}$ for all j such that (i, j) an edge). We also define $T(i, j)$ to represent the expected number of transitions until j is reached, starting at i.

Lemma 6.1. *$T(i,i) = \frac{2e}{d(i)}$. Let $\pi(i)$ be the stationary distribution of i. Then, $\pi(i) = \frac{d(i)}{2e}$. Mean recurrence time is the reciprocal of the stationary probability, giving the desired result.*

Corollary 6.2. *A consequence of Lemma 6.1 is that for any (i, j) edge, the frequency with which this edge is traversed $\frac{1}{2e}$.*

Lemma 6.3. *For adjacent vertices (i, j) it holds that $T(i, j) + T(j, i) \leq 2e$, with equality iff the removal of (i, j) edge increases the number of connected components (i.e. is a bridge). Since all transitions have the same long-run frequency, the sum can be thought of as expected number of transitions in a round trip. In addition, the expected number of iterations between i and j is only 1 for a bridge. The result follows.*

Lemma 6.3 can be extended inductively for arbitrary i, j not necessarily adjacent. Then, if we overload notation and define $T(i)$ to be the expected time to visit all vertices starting at i, we have that

Theorem 6.4. *The expected time to visit all vertices of a connected component beginning at some vertex i is polynomial in the number of nodes and edges. In particular, $T(i) \leq 2e(n-1)$.*

Proof. Fix a spanning tree H. DFS traversal of H starting at i gives a traversal which uses each edge exactly twice ending at i. Denote this order of traversal $i_0, \ldots, i_{2n-1}$. This gives a lower bound on the expected time to visit all vertices.

$$\begin{aligned} T(i_0, i_1) + T(i_1, i_2) + \cdots + T(i_{2n-3}, i_{2n-2}) &= \sum T(i, j) + T(j, i) \\ &\leq 2e(n-1) \end{aligned}$$

□

The result we'd like follows from Markov: if we run an algorithm which takes a random walk of length $4e(n-1)$, then

$$\Pr[X \geq 2 * 2e(n-1)] \leq \frac{2e(n-1)}{4e(n-1)} = \frac{1}{2}$$

for X a random variable denoting the number of steps until every vertex has been traversed. Since we clearly have no false positives, this places $USTCON \in RL$.

6.2 Explicit Construction of Expanders

Throughout this proof of *USTCON*, we assumed the existence of some constant degree expander H. In particular, we assumed the existence of an expander with parameters $\left(D^{16}, D, \frac{1}{2}\right)$. We show an explicit construction [6] of a graph of such parameters.

This construction is based on the affine plane. We begin with a graph on D^2 nodes, then use the zigzag product to obtain a graph over more nodes. For q some prime power, we let $\mathbb{F}_q$ be the finite field of order q. Let AP_q be a graph with vertex set $\mathbb{F}_q^2$ and edges $\{((a,b),(c,d)) \mid ac = b + b\}$. Equivalently, for each point (a,b), we have the (a,b) is connected to all the points on the line $L_{a,b} = \{(x,y) \mid y = ax - b\}$.

Lemma 6.5. *AP_q is an $\left(q^2, q, \frac{1}{\sqrt{q}}\right)$ graph. Moreover, the rotation map can also be computed in $poly(logq)$ time given a field representation. Since we just need a constant sized expander, for the purposes of the USTCON proof, it is not necessary to know the time complexity associated with creating such a graph and we focus our attention on showing a this graph has the desired properties.*

Proof. (Adapted from [6]) Let M be the $q^2 \times q^2$ adjacency matrix of AP^q. For row (a,b) and column (a',b'), we note that the entry of M^2 is the number of common neighbors of (a,b) and (a',b') in AP^q normalized by q^2. For $a \neq a', b \neq b'$, we know that $L_{a,b}, L_{a',b'}$ intersect at a unique point (axiom of affine space). If exactly one coordinate is the same, the intersection is empty. Finally, for $a' = a, b' = b$ we have an intersection of size q. That is

$$M^2 = \frac{1}{q^2}\begin{pmatrix} qI_q & J_q & \dots & J_q \\ J_q & qI_q & & J_q \\ \vdots & & \ddots & J_q \\ J_q & J_q & \dots & qI_q \end{pmatrix} = \frac{I_q \otimes qI_q + (J_q - I_q) \otimes J_q}{q^2}$$

where I_q is the identity, and J_q is an all-ones matrix. We can calculate the eigenvalues of M^2. In particular, we note that J_q has eigenvalues q (multiplicity 1) and 0 (multiplicity $q-1$). Therefore, $(J_q - I_q \otimes J_q$ has eigenvalues $(q-1)q, -q, 0$. Adding to $I_q \otimes qI_q$ increases all eigenvalues by q, and then finally, we normalized them each by q^2, resulting in eigenvalues of 1 (multiplicity 1) and 0 (multiplicity $q-1$) and $\frac{1}{\sqrt{q}}$ (multiplicity $(q-1)q$). This shows the spectral expansion is indeed $\frac{1}{\sqrt{q}}$. □

Corollary 6.6. *We can inductively define*

$$AP_q^1 = AP_q \otimes AP_2$$
$$AP_q^{i+1} = AP_q^1 ⓩ AP_q$$

From the same bounds on the zigzag product previously proved about the zigzag product and bounds on the tensor product (out of scope of this survey, we have that AP_q^i is $\left(q^{2(i+1)}, q^2, O\left(\frac{i}{\sqrt{q}}\right)\right)$. We omit the proof of this. Then, for $i = 7$, we have a $\left(q^{16}, q^2, O\left(\frac{7}{\sqrt{q}}\right)\right)$. So selection of some q such that $O\left(\frac{7}{\sqrt{q}}\right) \leq \frac{1}{2}$ is sufficient for our needs.

References

[1] O. Reingold, "Undirected st-connectivity in log-space," *Electronic Colloquium on Computational Complexity (ECCC)*, Jan. 2004. DOI: 10.1145/1060590.1060647.

[2] S. Hoory, N. Linial, and A. Widgerson, "Expander graphs and their application," *Bulletin (New Series) of the American Mathematical Society*, vol. 43, Oct. 2006. DOI: 10.1090/S0273-0979-06-01126-8.

[3] S. Vadhan, *Pseudorandomness* (Foundations and trends in theoretical computer science). Now Publishers, 2012, ISBN: 9781601985941. [Online]. Available: https://books.google.com/books?id=iam4lAEACAAJ.

[4] N. Alon and B. Sudakov, "Bipartite subgraphs and the smallest eigenvalue," *Combinatorics Probability and Computing*, vol. 9, Feb. 1970. DOI: 10.1017/S0963548399004071.

[5] L. Lovász, *Combinatorial Problems and Exercises* (AMS/Chelsea publication). North-Holland Publishing Company, 1979, ISBN: 9780821869475. [Online]. Available: https://books.google.com/books?id=e99fXXYx9zcC.

[6] O. Reingold, S. Vadhan, and A. Wigderson. "Entropy waves, the zig-zag graph product, and new constant-degree expanders." (2001), [Online]. Available: https://www.math.ias.edu/~avi/PUBLICATIONS/MYPAPERS/SALIL/EXPANDERS/ANNALS/RVW01.pdf.

[7] O. Reingold, L. Trevisan, and S. Vadhan, "Pseudorandom walks on regular digraphs and the rl vs. l problem," vol. 2006, May 2006, pp. 457–466. DOI: 10.1145/1132516.1132583.

[8] R. Aleliunas, "Random walks, universal traversal sequences, and the complexity of maze problems," in *20th Annual Symposium on Foundations of Computer Science (Sfcs 1979)*, IEEE Xplore, 1979, pp. 218–223. [Online]. Available: https://doi.org/10.1109/SFCS.1979.34.