CS640 Report: USTCON is in L

Alan Nair Abhyuday Pandey Pravar Deep Singh

Indian Institute of Technology, Kanpur

December 29, 2020

1 Introduction

The undirected s, t connectivity problem aims to check if there is path between a designated pair of vertices $s, t \in V$ in a given undirected graph $G = (V, E)$. The corresponding language, known as USTCON can be defined as follows,

$$\text{USTCON}=\{\langle G, s, t\rangle | \exists \text{ a path between } s, t \in V \text{ in undirected graph } G\}$$

It is simple to observe that USTCON $\in$ P, because it can be realized by simply doing a BFS/DFS from vertex s and checking if vertex t is ever encountered. However, this trivial algorithm takes linear space. The question now is – what is the optimal space-bound we can achieve for this problem?

Previous Results There exists a simple NL algorithm for USTCON. We can start from vertex s, initialize a counter to zero and non-deterministically move to a neighbor and increment the counter. We continue this procedure until the counter reaches n or we encounter t. Clearly, if there is a path between s and t, there must be at least one accepting path. Note that this algorithm also works for STCON i.e. the directed variant of this problem.

A simple implication of Savitch's theorem [9] is that USTCON $\in$ SPACE($\log^2 n$). A major breakthrough [1] was achieved when USTCON was shown to be in RL. The algorithm crucially used the fact that a random walk of a given polynomial length will reach t from s with a high probability. Later, Saks and Zhou [8] showed that USTCON $\in$ SPACE($\log^{3/2} n$) using derandomization techniques. Further, Armoni, Ta-Shma, Wigderson and Zhou [3] just 2 years later showed that USTCON $\in$ SPACE($\log^{4/3} n$). For long, it was believed that USTCON may have a complexity intermediate between L and NL. In 2005, Omer Reingold [5] showed that USTCON $\in$ L.

2 Motivation

The motivation underlying Reingold's algorithm is quite intuitive. If one wishes to find the connectivity of two vertices, first one should try to improve the "connectivity" of graph G. A common technique of doing so is *powering*. The k^{th} power of G, i.e. G^k contains an edge (u, v) for every path of length k in graph G from u to v. Clearly, connectivity between two vertices does not change upon powering the graph. However, the problem with powering the graph is that it leads to an exponential blow up in the degree of the graph (if G is a D-regular graph, then G^k is a D^k-

regular graph) and we want to avoid that in order to determine the connectivity in log-space. If we are able to improve the connectivity of the graph such that the graph has a logarithmic diameter and constant degree, then we can enumerate all paths starting from s of length $\ell = O(\log n)$ by storing ℓ edges and checking if any of these paths visits t. Also, since each edge would require $O(\log D)$ bits, and D is held constant we'll get a log-space algorithm for USTCON.

3 Algorithm

3.1 Notations

The notations we use is as follows. A graph G is an (N, D, λ)-graph if it has N vertices, degree D and spectral expansion λ. Note that G in this case is a D-regular graph, i.e. all vertices have degree D.

Definition 3.1. Suppose A is the adjacency matrix representation of the D-regular graph G and M is the normalized adjacency matrix $\frac{A}{D}$. It is known that M has n real eigenvalues $\lambda_1 \geq \lambda_2 \ldots \geq \lambda_n$ where $\lambda_1 = 1$. Moreover, the spectral expansion is defined as $\lambda = \max\{|\lambda_2|, |\lambda_n|\}$.

Proposition 3.2 ([5])**.** If G is an (N, D, λ)-graph, then G^k is an (N, D^k, λ^k)-graph

It is well known that the second largest eigenvalue is a really good measure of expansion properties of a graph. In particular, for every $\lambda < 1$, there exists $\epsilon > 0$ such that for every (N, D, λ)-graph G (called an *expander*) and a given set of vertices S such that $|S| \leq \frac{N}{2}$, atleast $(1 + \epsilon).|S|$ vertices of G are connected by an edge to some vertex in S. This immediately implies that if $\lambda < 1$ and is a *constant*, G has a diameter which is $\mathcal{O}(\log N)$.

Using this fact, we infer that connectivity in a constant degree expander with $\lambda < 1$ and a constant can be computed in log-space. A high-level description of the algorithm that does so for G, s, t is as follows. Enumerate all paths D^ℓ where $\ell = \mathcal{O}(\log N)$ from s. Accept if t is present in any of the paths. Enumerating all these paths requires $\mathcal{O}(\log N. \log D)$ space. We summarize the result as follows.

Theorem 3.3. Connectivity in an (N, D, λ)-expander where D is a constant and $\lambda < 1$ is a constant can be computed in $\mathcal{O}(\log N)$ space.

We will now discuss an important property of any D-regular, connected, non-bipartite graph. We later show how to transform an arbitrary connected graph to a $D-$regular non-bipartite connected graph.

Lemma 3.4 ([2])**.** For every D-regular, connected, non-bipartite graph G on N vertices it holds that $\lambda(G) \leq 1 - \frac{1}{DN^2}$.

Thus, the aim now is to transform a given graph to a constant degree expander with diameter $\mathcal{O}(\log N)$. The tools used to accomplish this are *powering* and *zig-zag products.*

3.2 Zig-Zag Products

We shall briefly discuss the powerful concept of zig-zag product which is the elementary tool for "reducing" the degree from blowing up due to powering.

The construction of zig-zag product for an (N, D, λ)-graph G and (D, d, α)-graph H proceeds as follows. Assume that $V(H) = [D] = \{1, 2, \ldots, D\}$. Define a mapping, called *rotation mapping*, Rot_G on $V(G) \times V(H)$ by $\text{Rot}_G(u, i) = (v, j)$ where u is the j^{th} neighbour of v and v is the i^{th} neighbour of u. The construction was given by Reingold, Vadhan and Wigderson [7].

Proposition 3.5 ([7])**.** The *zig-zag product* of G and H denoted by $G \textcircled{z} H$ is a graph with the following properties.

1. The vertex set of $G \textcircled{z} H$ is $V(G) \times V(H)$. Hence the total number of vertices are ND.
2. There exists an edge between vertices (u, i) and (v, j), where $(u, i), (v, j) \in V(G) \times V(H)$ if and only if $\exists i', j' \in V(H)$ such that $(i, i'), (j, j') \in E(H)$ and $\text{Rot}_G(u, i') = (v, j')$. A little analysis shows that $G \textcircled{z} H$ is d^2-regular.

We shall now state the spectral property of zig-zag product which will be crucially used in Reingold's algorithm [6].

Lemma 3.6 (spectral property of zig-zag product [6])**.** Suppose G and H are (N, D, λ)-graph and (D, d, α)-graph respectively, then $G \textcircled{z} H$ is an $(ND, d^2, f(\lambda, \alpha))$-graph such that,

$$1 - f(\lambda, \alpha) \geq \tfrac{1}{2}(1 - \alpha^2)(1 - \lambda)$$

Observe that powering an (N, D, λ)-graph G yields a (N, D^2, λ^2)-graph G^2. Intuitively, this operation increases the degree of graph but improves the spectral expansion. Zig-zag product on the other hand controls the degree of graph but worsens the spectral expansion. Interestingly, we can get the best of both worlds by a careful combination of both the operations.

3.3 Transforming the Graph to Expander

In this section, we assume that G is connected. This may look strange, but we do so because Reingold's algorithm independently takes each connected component of the graph and checks if t exists in the connected component that contains s.

Before diving into the transformation of graph to an expander, we show how to convert graph G to a D^{16}-regular graph. The conversion procedure is as follows. Suppose, G is the given graph. Replace any vertex of degree $d > 3$ with a cycle on d vertices, and connect the edges incident on this vertex to each of these d vertices present in the cycle. It is evident that after this procedure, the number of vertices will only suffer a quadratic blowup and the degree of each of the vertex will be ≤ 3. Finally, add self loops to each of the vertices until it becomes a D^{16}-regular graph. Interestingly, the construction ensures that the resultant graph is non-bipartite. Henceforth, we assume that G is a D^{16}-regular graph which is non-bipartite.

The reader might be tempted to know if there is something special in the number D^{16}. The reason of using this number becomes clear from the following fact.

Fact 3.7. There exists some constant D and a $(D^{16}, D, \frac{1}{2})$-expander.

The existence of such an expander can be shown through probabilistic methods. However, a constant D also implies that such an expander can be found by exhaustive search within constant space bounds. Henceforth, we shall denote this expander by H.

With the above background, we are in a position to describe the Reingold's algorithm. The fundamental idea is to use zig-zag product and powering alternately to improve the spectral expansion while keeping the degree in check. The key transformation is as follows.

Proposition 3.8 (Key Transformation [6]). On input G and H, where G is a D^{16}-regular graph on $[N]$ and H is a D-regular graph on $[D^{16}]$, both given by their rotation maps, the transformation $\mathcal{T}$ outputs the rotation map of a graph G defined as follows:

- Set $\ell = 2\lceil \log DN^2 \rceil$
- Set $G_0 = G$ and for $i > 0$ define G_i recursively as $(G_{i-1} ⓩ H)^8$

Denote by $\mathcal{T}_i(G, H)$ the graph G_i, and $\mathcal{T}(G, H) = G_\ell$. The transformation itself can be carried out in log-space.

It is important to highlight that if we use rotation map representation of graphs, each powering requires $\mathcal{O}(\log deg(G))$ additional space where G is the graph undergoing powering. A brief description for the same can be found in Appendix A. Since, there are only ℓ rounds, additional space used will only be $\mathcal{O}(\ell . \log D) = \mathcal{O}(\log N)$ as the degree remains constant throughout.

From the properties of powering and zig-zag product it should be evident that after each step G_i will have degree D^{16}, which is the same as that of the input graph G. Thus, $\mathcal{T}(G, H)$ also has the same degree as G. It follows from Lemma 3.5(1) that G_i is a graph on $N \times D^{16i}$ vertices. Fortunately, since $\ell = \mathcal{O}(\log N)$ therefore G_ℓ has only $O(poly(n))$ number of vertices. This won't be a roadblock as we show that G_ℓ is an expander with $\lambda(G_\ell) \leq \frac{1}{2}$.

Lemma 3.9. *G_ℓ is a constant degree expander with $\lambda(G_\ell) \leq \frac{1}{2}$.*

Proof. Since $G = G_0$ is a $D-$regular, connected and non-bipartite graph on N vertices, by lemma 3.4, $\lambda(G_0) \leq 1 - 1/DN^2$. We claim that it is enough to show that $\forall i > 0,\ \lambda(G_i) \leq \max\{\lambda(G_{i-1})^2, 1/2\}$ because then $\lambda(G_\ell) \leq \max\{\lambda(G_0)^{2^\ell}, 1/2\}$ and by our choice of $\ell = 2\lceil \log DN^2 \rceil$, we can verify by elementary calculations that $(1 - 1/DN^2)^{2^\ell} \leq 1/2$. So let $\lambda(G_{i-1}) = \lambda$. Recall from Fact 3.7 that $\lambda(H) = \frac{1}{2}$. So, from Proposition 3.2 and Lemma 3.6, we have $\lambda(G_i) = (\lambda(G_{i-1} ⓩ H))^8 \leq (1 - \frac{3}{8}(1-\lambda))^8 < (1 - \frac{1}{3}(1-\lambda))^8$. If $\lambda < 1/2$, then $\lambda(G_i) < (5/6)^8 < 1/2$. If $\lambda \geq 1/2$, we can verify by elementary calculations that $(1 - \frac{1}{3}(1-\lambda))^4 \leq \lambda$, which gives $\lambda(G_i) < (1 - \frac{1}{3}(1-\lambda))^8 \leq \lambda^2$. □

Using Theorem 3.3, the connectivity of vertices corresponding to s and t in G_ℓ can be computed in log-space. Since the transformation outlined in Proposition 3.8 can also be accomplished in log-space, we get a log-space algorithm for USTCON. Thus, USTCON $\in$ L.

4 Conclusion and Future Work

Reingold showed that SL, the class of problems reducible to USTCON, also lies in L. However, the following problems still remain open [6].

1. It remains open if any randomized log-space machine can be simulated with a deterministic log-space machine i.e. whether RL=L.
2. A common conjecture is that Savitch's algorithm [9] is optimal deterministic algorithm for STCON, the counterpart of USTCON for directed graphs. While this conjecture may be true, to the best of our knowledge there exists no evidence for the same.

References

[1] Romas Aleliunas, Richard M. Karp, Richard J. Lipton, László Lovász, and Charles Rackoff. Random walks, universal traversal sequences, and the complexity of maze problems. In *20th Annual Symposium on Foundations of Computer Science, San Juan, Puerto Rico, 29-31 October 1979*, pages 218–223. IEEE Computer Society, 1979. doi:10.1109/SFCS.1979.34.

[2] Noga Alon and Benny Sudakov. Bipartite subgraphs and the smallest eigenvalue. *Comb. Probab. Comput.*, 9(1):1–12, 2000. URL: http://journals.cambridge.org/action/displayAbstract?aid=46757.

[3] Roy Armoni, Amnon Ta-Shma, Avi Wigderson, and Shiyu Zhou. An $O(\log(n)^{4/3})$ space algorithm for $(s,\ t)$ connectivity in undirected graphs. *J. ACM*, 47(2):294–311, 2000. doi:10.1145/333979.333984.

[4] Cynthia Dwork and Prahladh Harsha. Lecture 8: Undirected connectivity is in logspace. 2005.

[5] Omer Reingold. Undirected st-connectivity in log-space. In Harold N. Gabow and Ronald Fagin, editors, *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, pages 376–385. ACM, 2005. doi:10.1145/1060590.1060647.

[6] Omer Reingold. Undirected connectivity in log-space. *J. ACM*, 55(4):17:1–17:24, 2008. doi:10.1145/1391289.1391291.

[7] Omer Reingold, Salil P. Vadhan, and Avi Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors. *Electron. Colloquium Comput. Complex.*, 8(18), 2001. URL: http://eccc.hpi-web.de/eccc-reports/2001/TR01-018/index.html.

[8] Michael E. Saks and Shiyu Zhou. BP $_{\text{h}}$space(s) subseteq dspace(s$^{3/2}$). *J. Comput. Syst. Sci.*, 58(2):376–403, 1999. doi:10.1006/jcss.1998.1616.

[9] Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *J. Comput. Syst. Sci.*, 4(2):177–192, 1970. doi:10.1016/S0022-0000(70)80006-X.

A Powering requires $\mathcal{O}(\log deg(G))$ additional space

Although Reingold gave a detailed proof for the same in his seminal work [6], we shall restrict ourselves to a high-level description. Parts of this description are borrowed from lecture notes [4] of Dwork and Harsha as part of a course offered in 2005 on expanders at Stanford.

If G is a D-regular graph, then G^2 is a D^2-regular graph. We can assume that the edge label incident on each vertex are from set $[D] \times [D]$. We compute $\mathrm{Rot}_{G^2}(u, (i_1, i_2))$ as follows. Suppose $(u, (i_1, i_2))$ is already written on the tape, we first compute $\mathrm{Rot}_G(u, i_1) = (w, j_2)$ and replace the content by $(w, (j_2, i_2))$. Observe that this step includes the additional space required to compute Rot_G. Similarly, $\mathrm{Rot}_G(w, i_2) = (v, j_1)$ is computed and the contents are replaced by $(v, (j_2, j_1))$. Finally, j_1 and j_2 are swapped to get $(v, (j_1, j_2))$ which is indeed the value of $\mathrm{Rot}_{G^2}(u, (i_1, i_2))$. Suppose $\mathrm{SPACE}(G)$ denote the additional space to compute Rot_G, it follows that –

$$\mathrm{SPACE}(G^2) = \mathrm{SPACE}(G) + \mathcal{O}(\log deg(G))$$

Observe that if the powering is done $\ell = \mathcal{O}(\log N)$ time the space complexity achieved will be $\sum_{i=1}^{i=\ell} \mathcal{O}(\log deg(G^{2i}))$ which results in $\mathcal{O}(\log^2 N)$ space algorithm as the degree can grow quite large. However, the Reingold's algorithm keeps degree constant after each round resulting in the total space complexity to be only $\sum_{i=1}^{i=\ell} \mathcal{O}(\log D)$ which is $\mathcal{O}(\log D. \log N)$ or simply log-space.