Undirected Graph Connectivity

Kishlaya Jaiswal

August 12, 2020

Abstract

We will discuss the problem of st-connectivity in undirected graphs, giving a short proof for solving it in logspace using random walks and then finally we will give a detailed overview of Omer Reingold's result - solving this problem in logspace without using randomness.
Advisor: *Samir Datta*

1 Introduction

Consider the following problem: given a undirected graph G and two vertices $s, t \in G$, determine if t is reachable from s.

$$USTCON = \{(G, s, t) \mid \text{t is reachable from s in G}\}$$

Widely known algorithms for solving this problem are Breadth-First Search and Depth-First Search. The running time of these algorithms is $O(|V| + |E|)$ and requires $O(|V|)$ space, that is $USTCON \in P$.

But can we do better? It is clear that we can't improve on time complexity. So we turn to look at this problem from the point of view of complexity theory.

We know that $USTCON \in NL$, because we can non-deterministically guess the vertices on the path from s to t one-by-one, and description of each vertex requires only $\log |V|$ many bits. Furthermore, using Savitch's theorem we get that $USTCON \in L^2$

Our next hope would be to prove $USTCON \in L$. But in making an attempt to this problem, what might seem more obvious at first is that $USTCON \in RL$, for we can do a random walk on the graph starting at s with the hope that, after not-too-many steps we have a good chance to hit t if t was in the same connected component as s, otherwise it is clear that we will never hit t. Let's try and make this argument more precise.

2 $USTCON \in RL$

The idea is straight-forward - start a random walk at s and continue until $poly(n)$ steps, where a random step is: if at vertex u, then choose a neighbour v of u uniformly at random and go to v. If we do not hit t then we output that there is no path connecting them. We now analyze the probability of failure - not hitting t even though s and t are in the same connected components.

So let's say s and t are in the same component. Denote by E_{uv} the expected time to hit v starting at u for any two vertices u and v. Using Markov's inequality, we get:

$$\mathbb{P}(\text{Time taken to hit } t > K) \leq \frac{E_{st}}{K}$$

The stationary distribution for this walk is π given by $\pi(u) \sim d_u$. Using the fundamental theorem for Markov Chains, $E_{uu} = \frac{1}{\pi_u} = \frac{2m}{d_u}\ \forall u \in V$.

For any edge $(u, v) \in E$, we calculate E_{uv} as follows:

$$E_{uu} = \sum_{v:(u,v)\in E} \frac{1 + E_{uv}}{d_u}$$

$$\implies E_{uv} \leq d_u E_{uu} = 2m$$

Finally, let σ be a path from s to t then $E_{st} \leq \sum_{(u,v)\in\sigma} E_{uv} \leq 2|\sigma|m \leq 2m(n-1)$. Hence it suffices to take $4n^3$ many steps in our random walks so that $\mathbb{P}(\text{failure}) \leq \frac{1}{2}$.

3 $USTCON \in L$

Improving from Savitch's theorem [NSW92] showed that $USTCON \in L^{3/2}$. Later, [Arm+00] improved this to $USTCON \in L^{4/3}$. Finally in 2004, Omer Reingold [Rei04] showed that $USTCON \in L$, whose proof we discuss now.

We will assume that the given graph G is d-regular and each connected component is non-bipartite. This assumption is justified because given a graph replace each vertex v with a cycle of length d_v connecting each neighbour of v to a unique vertex in the cycle, thus making the graph 3-regular. Now add $d-3$ self-loops on each vertex to make it d-regular.

We begin with a simple observation that given a d-regular graph with diameter atmost ρ of each component, if we just did a brute-force search by checking for t on all possible paths from s, it would require us to enumerate all $O(d^\rho)$ paths and hence take $O(\rho \log d)$ space. In general, ρ can be as large as n. But if our graph was well-connected and we had $\rho = O(\log n)$, then we have accomplished our goal.

An intuitive way to make our graph *well-connected* is by connecting each vertex to a vertex which was originally at distance 2. This operation makes it d^2-regular, making our graph even more dense. But if we keep repeating this process only, then after $\log n$ steps, space complexity of enumerating all paths would turn $O(\rho \log d^{2^{\log n}}) = O(\rho n \log d)$, defeating our purpose.

So we need another operation to make our graph *sparse*, keeping the degree constant. Intuitively, we will take a small graph and replace each vertex in our original graph with this small graph and edges are taken to be the "zigzag" lines. Thus, we reduce the degree of the graph at the cost of deteriorating the connectedness slightly. This operation is called zigzag product of graphs.

Both these operations preserve the connected components of G (Squaring preserves the connected components of G because each component contains an odd-cycle). Alternating this process of squaring and zigzaging $O(\log n)$ times, we will find that G becomes a well-connected sparse graph with a diameter $O(\log n)$ and constant degree. Thus, we are done.

Now to make our above intuition precise, we start with the following definitions.

3.1 Expanders

As already hinted above, expanders are well-connected but sparse graphs. Well-connectedness is specified by cuts of atleast some constant size and sparseness using constant small degree.

Given a graph G and $S \subseteq V$, let

$$\Gamma(u) := \{v \mid (u, v) \in E\}$$

$$\Gamma(S) := \bigcup_{u\in S} \Gamma(u)$$

Definition 3.1. G is said to be a $(1+\epsilon)$ vertex-expander if for any subset S of vertices with $|S| \leq N/2$, $|\Gamma(S)| \geq (1+\epsilon)|S|$.

Remark. Let G be $(1+\epsilon)$ vertex-expander for some $\epsilon > 0$ then, G has logarithmic diameter

Vertex expansion is a combinatorial property but it's easy to deal with algebraic properties, like spectrum of a graph and in particular with the following quantity: let $\lambda(G)$ be the largest (in absolute value) eigenvalue of the normalized adjacency matrix $\mathrm{A} = D^{-1/2}AD^{-1/2}$, and since $\mathbf{u} = \frac{1}{n}\mathbf{1}$ is an eigenvector of A with largest eigenvalue 1, we can equivalently say

$$\lambda(G) = \max_{\mathbf{x}\perp\mathbf{u}, \|\mathbf{x}\|=1} \|\mathrm{A}\mathbf{x}\|$$

Definition 3.2. G is said to be a (n, d, λ) spectral-expander if G is d-regular graph on n vertices with $\lambda(G) \leq \lambda$.

Finally we have the result establishing the connection between two notions from algebra and combinatorics: every spectral expander is also a vertex expander.

Theorem 3.3. *Let G be a (n, d, λ) spectral-expander then G is also a $(1+\epsilon)$ vertex-expander, where $\epsilon = \frac{1-\lambda^2}{1+\lambda^2}$*

In fact, both these notions of expansion are equivalent but we don't need the converse relation for our proof.

Remark. Let G be (n, d, λ) spectral-expander where $\lambda < 1$ then G has diameter $O(\log n)$ where the leading constant in our big-O notation depends upon λ (as it depends upon ϵ).

3.2 Graph Squaring

Let A be the adjacency matrix of G then G^2 is the graph defined using the adjacency matrix A^2.

Since G is d-regular and G^2 is d^2-regular, we get that the normalized adjacency matrix of G^2 is the square of normalized adjacency matrix of G. Thus $\lambda(G^2) = \lambda(G)^2$.

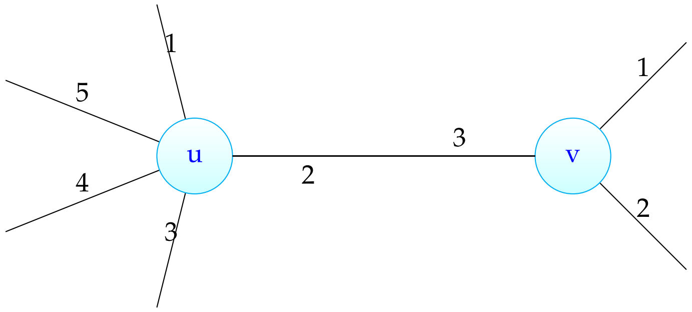
So squaring improves the spectral expansion at the cost of exploding the degree of a graph. In particular, if G is (n, d, λ) expander then G^2 is a (n, d^2, λ^2) expander.

3.3 Zigzag product

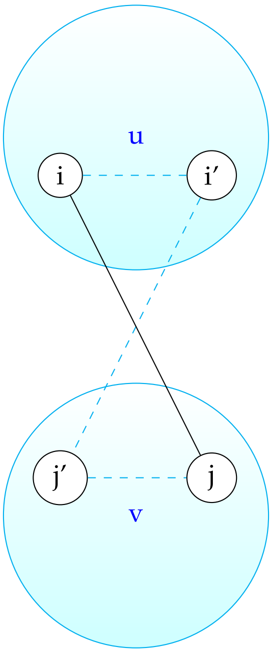
Let G be (n, d_1, λ_1) expander and H be (d_1, d_2, λ_2) expander.

Define $G Ⓩ H$ as a graph on the vertex set $V(G) \times V(H)$.

In G, label each neighbour of a vertex u from $\{1 \ldots d_1\}$ so that $u[i]$ represents the i^{th} neighbour of u and if $(u, v) \in E$ then $u[i] = v$ and $v[j] = u$. For example: here $u[2] = v$ and $v[3] = u$



Now take any d_2-regular graph H on d_1 vertices (we label the vertices from $\{1 \dots d_1\}$), and replace each vertex in G with a *cloud* H, that is consider a graph on vertex set $V(G) \times V(H)$ with the following edges: Suppose you're at a vertex (u, i). Choose any neighbour i' of i and go to (u, i'). Use this index i' to go to a different cloud which is (v, j') where $v = u[i']$ and $u = v[j']$. Finally choose any neighbour j of j' and go to (v, j). Add an edge between (u, i) and (v, j).



Notice that from (u, i) there are d_2 many choices for (u, i'). (v, j') is already determined from (u, i'). And from (v, j') there are d_2 many choices for (v, j). Hence degree of any vertex is d_2^2.

Lemma 3.4. *$G \textcircled{z} H$ is a $(nd_1, d_2^2, f(\lambda_1, \lambda_2))$ expander where $f(\lambda_1, \lambda_2) \leq \lambda_1 + \lambda_2 + \lambda_2^2$*

3.4 Existence of a small expander

To begin our algorithm, we need to start with a good expander for zigzaging our graph with. [RVW01] showed that there exists a constant d_0 and a graph H which is $(d_0^{16}, d_0, 1/2)$ expander.

3.5 Reingold's algorithm

It is now clear that if we fix any two *constants* d and $\lambda < 1$, then for any input graph G such that each component of G is a (n, d, λ) expander, we can solve the connectivity problem (restricted to these class of graphs) in logspace.

Thus, we take these constants to be $d = d_0^{16}$ and $\lambda = \frac{1}{2}$ and then it suffices to show how to convert any graph G into a $(O(n), d_0^{16}, 1/2)$ expander.

Algorithm 1: Transform G into a $(n, d_0^{16}, 1/2)$ expander

```
Input: $G$
Convert $G$ into a $d_0^{16}$ regular graph by replacing each vertex with a cycle and adding
  self-loops
Set $l = 2\lceil \log(d_0^{16} n^2) \rceil$
Let $G_0 = G$
for $i \leftarrow 1$ to $l$ do
    compute $G_i = (G_{i-1} \textcircled{z} H)^8$
end
Output $G_l$
```

We mention once again that both the operations: squaring and zigzag product preserve the connected components, that is s and t are connected in G iff they are connected in G_l.

3.6 Analysis

Since we need to only enumerate paths starting at s, we only work with the component containing s. Hence, for keeping the analysis simple, we assume that G is connected.

Lemma 3.5. *Each G_i $(0 \leq i \leq l)$ is a graph on nd_0^{16i} and is d_0^{16}-regular*

Proof. We proceed by induction on i. For $i = 0$ it follows immediately. Now suppose it is true for for some i, that is G_i is $nd_0^{16i}, d_0^{16}, \lambda)$ expander for some λ. Since H is $(d_0^{16}, d_0, 1/2)$ expander, we get $G_i \textcircled{z} H$ is a $(nd_0^{16(i+1)}, d_0^2, \lambda')$ expander for some λ'.

Therefore, $G_{i+1} = (G_i \textcircled{z} H)^8$ is a graph on $nd_0^{16(i+1)}$ vertices and is d_0^{16}-regular. □

Thus, each of the zigzag products in our algorithm is valid and G_l is a graph on $O(n)$ vertices.

Lemma 3.6. *If $l = 2\lceil \log x \rceil$ then $\left(1 - \frac{1}{x}\right)^{2^l} \leq \frac{1}{2}$*

Proof.

$$2\lceil \log x \rceil \leq 2 \log x + 2 \implies 2^l \leq 4x^2$$

$$\implies \left(1 - \frac{1}{x}\right)^{2^l} \leq \left(1 - \frac{1}{x}\right)^{4x^2} \leq \frac{1}{2}, \forall x \geq 1$$

□

Lemma 3.7. *For any d-regular, connected and non-bipartite graph G on n vertices,*

$$\lambda(G) \leq 1 - \frac{1}{dn^2}$$

Proof. □

Lemma 3.8. $\lambda(G_i) \leq \max\{\lambda(G_{i-1})^2, \frac{1}{2}\}$

Proof. □

Lemma 3.9. $\lambda(G_l) \leq 1/2$

Proof. Suppose for some $1 \leq i \leq l - 1$, $\lambda(G_i) \leq \frac{1}{2}$, then

$$\lambda(G_{i+1}) \leq \max\left(\lambda(G_i)^2, \frac{1}{2}\right) \leq \max\left(\frac{1}{4}, \frac{1}{2}\right) = \frac{1}{2}$$

and hence $\lambda(G_l) \leq \frac{1}{2}$ by induction. Otherwise if $\lambda(G_i) > \frac{1}{2}\ \forall 1 \leq i \leq l-1$, then

$$\lambda(G_l) \leq \lambda(G_{l-1})^2 \leq \lambda(G_{l-2})^{2^2} \leq \cdots \leq \lambda(G_0)^{2^l} \leq \left(1 - \frac{1}{d_0^{16} n^2}\right)^{2^l} \leq \frac{1}{2}$$

□

3.7 Implementation

On a final note, we mention that using the *Rotation Map* representation of graphs, we can implement the above algorithm in logspace.

That is, all the three steps:

- Converting G into d_0^{16}-regular non-bipartite graph
- Squaring a graph
- Taking zigzag product with H (H is a constant graph and can be directly encoded on a TM)

can be implemented in logspace.

References

[Arm+00] Roy Armoni et al. "An $O(\log(n)^{4/3})$ space algorithm for (s, t) connectivity in undirected graphs." In: *J. ACM* 47.2 (2000), pp. 294–311. DOI: 10.1145/333979.333984. URL: https://doi.org/10.1145/333979.333984.

[NSW92] Noam Nisan, Endre Szemerédi, and Avi Wigderson. "Undirected Connectivity in O(log ^1.5 n) Space." In: *33rd Annual Symposium on Foundations of Computer Science, Pittsburgh, Pennsylvania, USA, 24-27 October 1992*. IEEE Computer Society, 1992, pp. 24–29. DOI: 10.1109/SFCS.1992.267822. URL: https://doi.org/10.1109/SFCS.1992.267822.

[Rei04] Omer Reingold. "Undirected ST-Connectivity in Log-Space." In: *Electronic Colloquium on Computational Complexity (ECCC)* 094 (2004). URL: http://eccc.hpi-web.de/eccc-reports/2004/TR04-094/index.html.

[RVW01] Omer Reingold, Salil P. Vadhan, and Avi Wigderson. "Entropy Waves, the Zig-Zag Graph Product, and New Constant-Degree Expanders and Extractors." In: *Electronic Colloquium on Computational Complexity (ECCC)* 8.18 (2001). URL: http://eccc.hpi-web.de/eccc-reports/2001/TR01-018/index.html.

[Vad12] Salil P. Vadhan. "Pseudorandomness." In: *Foundations and Trends in Theoretical Computer Science* 7.1-3 (2012), pp. 1–336. DOI: 10.1561/0400000010. URL: https://doi.org/10.1561/0400000010.