Derandomization and Nearly Logarithmic Space Laplcian Solvers

Devansh Shringi, Harry Sha

CSC2240 Final Report - Fall 2022

1 Context: Space Bounded Complexity

1.1 L, RL, NL

An essential resource for computation is space. Like in the time-bounded computation, we can define several interesting complexity classes. The following are analogs for P, RP, and NP, but restricted to logarithmic space instead of polynomial time. Let n be the length of the input.

Definition 1.1 (L)**.** L is the set of decision problems decidable by a **deterministic** Turing Machine using at most $O(\log(n))$ cells on the work tape.

Definition 1.2 (RL)**.** NL is the set of decision problems decidable by a **probabilistic** Turing Machine with one-sided error using at most $O(\log(n))$ cells on the work tape.

Definition 1.3 (NL)**.** NL is the set of decision problems decidable by a **non-deterministic** Turing Machine using at most $O(\log(n))$ cells on the work tape.

The classes are summarized in the following table.

	$x \in L$	$x \notin L$
L	1	0
RL	$\geq 2/3$	0
NL	> 0	0

Figure 1: Acceptance Probabilities

In this view, it is straightforward to see $L \subseteq RL \subseteq NL$. However, just as in the poly-time complexity setting, it is unknown whether or not these containments are strict.

Open Problem 1.4. *Is L equal to RL? Is RL equal to NL?*

The current conjecture is that $L = RL$ but $L \neq NL$. If $L = RL$, then it would mean that we could **derandomize** any randomized algorithm that runs in logarithmic space using logarithmic space. Thus, the derandomization we'll study here provides some limited evidence for $L = RL$.

1.2 Connectivity

Define the following decision problems:

$$STCON = \{\langle G, s, t\rangle : G \text{ is a } \textbf{directed} \text{ graph and there is a path from } s \text{ to } t \text{ in G}\}$$
$$USTCON = \{\langle G, s, t\rangle : G \text{ is an } \textbf{undirected} \text{ graph and there is a path from } s \text{ to } t \text{ in G}\}$$

Fact 1.5. STCON *is complete for NL. I.e.,* STCON $\in NL$ *and* STCON *is hard for NL with respect to logspace reductions.*

It is currently unknown if STCON $\in RL$. However, there is a very natural *randomized* logspace algorithm for USTCON, i.e., USTCON $\in RL$. In fact, a simple random walk, Algorithm 2, does the job.

The random walk uses at most logarithmic space since you only keep track of a single vertex, requiring $\log(n)$ bits to identify. [Ale+79], showed that if s and t are indeed connected by an undirected path, then this algorithm returns true with high probability, so this is indeed an RL algorithm for USTCON.

Algorithm 2: Random Walk

Input: G, s, t

$c = s$;
for n^4 times **do**
  if $c = t$ **then**
    return True
  end
  $c \xleftarrow{R} \Gamma(c)$
end
return False

1.3 Solving Laplacian System

In the later half of this report, we will focus on finding solution for the following equation for a graph G on n vertices and given $b \in \mathbb{R}^n$

$$\boldsymbol{L}x = b$$

where $\boldsymbol{L}$ denotes the Laplacian matrix of the graph. The Laplacian of a graph is the matrix $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$ where $\boldsymbol{D}$ is a diagonal matrix $n \times n$ with weighted degree of v as the (v, v) entry and $\boldsymbol{A}$ is the adjacency matrix. It can also be seen that solving a Laplacian system allows you to solve connectivity in undirected setting, and therefore is at least as hard as USTCON, and so is interesting if it can be done in L. Solving the directed case of Laplacian also implies $RL = L$. The solution for the undirected case in [PS14] used random sampling and was later found to have a solution running in deterministic logspace, using derandomized squaring in [Mur+21]. Thus, this also further supports the argument of $RL = L$.

As we saw in class, solving Laplacian systems has applications in various parts of complexity and algorithms, calculating Effective Resistance, finding Max Flow, Sampling Spanning Trees, and Graph Sparsification being a few.

2 Derandomization of The Random Walk

In a breakthrough result, [Rei05] showed that this algorithm could be derandomized - i.e., that USTCON $\in L$. We will present a proof by [RV05].

2.1 Intuition

The idea of the proof is to improve the connectivity of G, so much so that if s and t are connected, they are connected by an edge and then examine the neighbors of s. The measure of connectivity we will use is expansion.

2.2 Technicalities

Regularlity. First, to make our lives easier, we'll assume that all graphs are regular. To do so, replace a single vertex, v, with a cycle of length $\deg(v)$ such that each vertex in the cycle is adjacent to one of v's neighbors. Then, we can add a self-loop to each vertex. Thus, without loss of generality, it suffices to consider input graphs that are 4-regular and contains a loop on each vertex.

Labelling. A consistent labeling of a K-regular directed graph is an assignment of edges to $[K]$ such that for each vertex, all the outgoing edges have unique labels, and all the incoming edges have unique labels. For a vertex v and an edge label $i \in [K]$, let $v[i]$ be the neighbor of v on the edge labeled i.

We'll also assume that all the graphs we work with have a consistent labeling and that all operations preserve the consistent labeling (you can check this yourself!).

2.3 Notation

For any integer N,

- I_N is the $N \times N$ identity matrix,
- J_N is the $N \times N$ matrix with each entry equal $1/N$,
- $\mathbb{1}_N$ is the $N \times 1$ all ones matrix, and
- $u_N = \frac{1}{N}\mathbb{1}_N$ is the uniform distribution on $[N]$.

2.4 Expansion

Definition 2.1 (Expander)**.** Let X be a K-regular graph on N vertices with random walk matrix M. X is called an (N, K, λ)-expander if the second largest (in absolute value) of M, denoted $\lambda(X)$, is at most λ.

Using the fact that M is real and symmetric, we know that the eigenvectors of the M are orthogonal, and by the Courant-Fisher Theorem,

$$\lambda(X) = \max_{v \perp \mathbb{1}} \frac{\|Mv\|}{\|v\|}$$

Intuition. $\lambda \in [0, 1]$, and the smaller $\lambda(X)$ is, the better the connectivity of the graph and the faster the convergence of a random walk to the uniform distribution.

Another useful characterization of (N, D, λ) graphs is the following

Lemma 2.2. *Let M be the transition matrix of a K-regular graph on N vertices.*

$$\lambda(M) \leq \lambda \iff M = (1-\lambda)J_N + \lambda C$$

for some matrix C with $\|C\| \leq 1$.

The following lemma relates the expansion of a graph and the diameter.

Lemma 2.3. *Let X be a $(N, D, \frac{1}{2N^{1.5}})$-expander. Then X contains a pair between any pair of vertices. The probability that a random neighbor of v is equal w is at least $\frac{1}{N} - \frac{1}{N^2} > 0$.*

Thus, when the expansion of a graph is very small, it contains a clique.

Proof. For a given vertex, v, the distribution of random neighbors is Me_v. I.e., the probability that a random neighbor of v equals w is $(Me_v)_w$. Thus, we want to show that every entry of Me_v is at least $\frac{1}{N} - \frac{1}{N^2}$. To do so, we'd like to use expansion, so we will transform e_v into one that is orthogonal to $\mathbb{1}$. Let $u = \frac{1}{N}\mathbb{1}$ be the uniform distribution on vertices. We'll consider $e_v - u$. This has several nice properties:

1. $e_v - u$ is orthogonal to $\mathbb{1}$.
2. $Mu = u$, so every element of $Me_v \geq \frac{1}{N} - \frac{1}{N^2}$ iff every element of $M(e_v - u)$ has is value at least $-\frac{1}{N^2}$. Clearly, it is sufficient to show that every element of $M(e_v - u)$ has absolute value at most $\frac{1}{N^2}$.
3. $\|e_v - u\| = \sqrt{(N-1)\frac{1}{N^2} + \frac{(N-1)^2}{N^2}} \leq 2$

Thus, since the expansion is at most $\frac{1}{2N^{1.5}}$, $\|M(e_v - u)\|^2 \leq \frac{1}{N^3}$. Let m be the smallest element of $M(e_v - u)$ in absolute value. Then, $\|M(e_v - u)\|^2 \geq Nm^2$. Putting the two together, we get $m \leq \frac{1}{N^2}$ as required. □

The following fact shows that all regular graphs have some expansion to start with - the goal will be to reduce this down to the value in Lemma 2.3.

Lemma 2.4. *Let X be a D-regular graph on N vertices with a loop on each vertex. Then $\lambda(X) \leq 1 - \frac{1}{2D^2N^2}$*

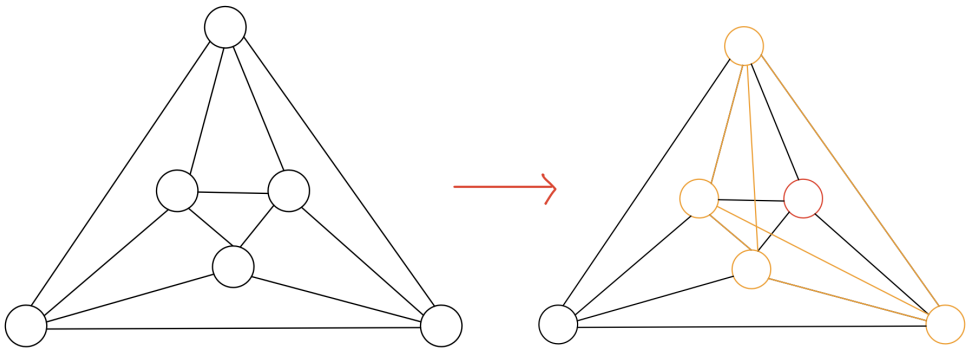

Figure 2: Placing a clique on the neighborhood of a vertex

2.5 Squaring

Definition 2.5 (Squaring)**.** Let X be a graph with adjacency matrix A. Then, the square of X denoted X^2 is the graph with adjacent matrix A^2. I.e., if there is a path of length 2 from u to v in X, there is now an edge from u to v.

Remark 1. One can also view the square as placing a clique on the neighborhood of each vertex. For example, see Figure 2.

Remark 2. Since the transition matrix simply squares, the $\lambda(X^2) = \lambda(X)^2$, so the expansion squares (making it smaller than before).

The downside of this operation is that the degree also squares - if X was D-regular, X is now D^2-regular. To ameliorate this, instead of placing a clique on every neighborhood, place an auxiliary graph (with fewer edges), thereby adding only a subset of the paths of length two in X. This is the definition of the derandomized square.

2.6 Derandomized Squaring

Definition 2.6 (Derandomized Square [RV05])**.** Let X be a labeled K-regular graph on the vertex set $[N]$, and let G be a D-regular graph on the vertex set $[K]$, the derandomized square, denoted $X \circledS G$ has vertex set $[N]$ and is KD regular. For $a \in [K]$, and $b \in [D]$, the (a, b)th neighbor of v is the $(v[a])[a[b]]$.

Note $a[b] \in [K]$ and is viewed as an edge label for $v[a]$. The edge (a, b) captures the path of length two $v \to v[a] \to v[a][a[b]]$. See Figure 3.

Lemma 2.7 (Derandomized Square is almost as good as Squaring [RV05])**.** *If X is a consistently labelled (N, K, λ)-graph and G is a (K, D, μ)-graph, then $X \circledS G$ is an $(N, KD^2, f(\lambda, \mu))$-graph, where*

$$f(\lambda, \mu) = \lambda^2 + \mu - \lambda^2\mu$$

Remark 3. $f(\lambda, \mu) \leq \lambda^2 + \mu$

By the remark, we see that the derandomized square has expansion close to the regular square when μ is small (that is, when G, the auxiliary graph, is also a good expander). This crucial fact makes the derandomized square useful - it has expansion almost as good as the square but smaller degree.

Proof of Lemma 2.7. Let M be the transition matrix of $X \circledS G$. We'll show that any vector perpendicular to $\mathbb{1}$ shrinks by a factor of at least $f(\lambda, \mu)$. The proof will bound the matrix norm of M. Let A be the transition matrix of X and B be the transition matrix of G.

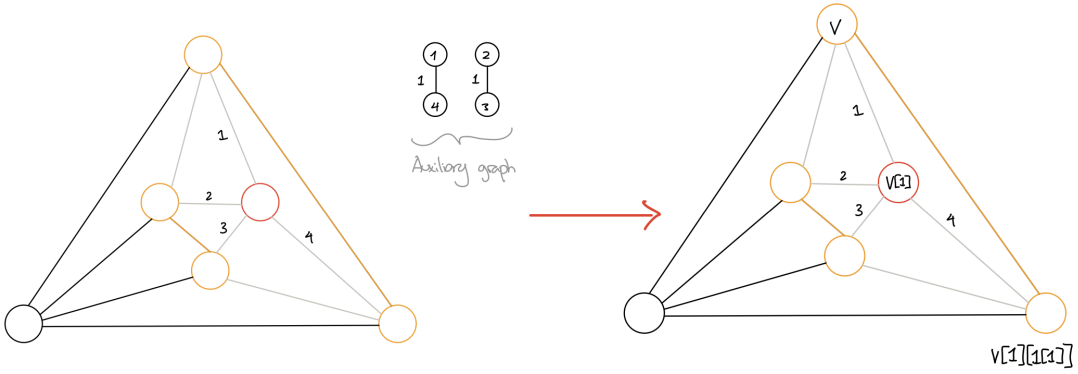

Figure 3: Placing an auxiliary graph on the neighborhood of a vertex. Shows the (1,1)th neighbor of v.

To define the M, it will be useful to consider a random step on $X \circledS G$. Recall that the (a, b)th neighbor of u for $a \in [K], b \in [D]$, is the $a[b]$th neighbor of $u[a]$. For a random step in $X \circledS G$ starting at u, think of it as happening in the following process.

1. Sample $a \xleftarrow{R} [K]$. Move to state (u, a).
2. Go to state $(u[a], a)$.
3. Sample $b \xleftarrow{R} \Gamma(a)$. Go to $(u[a], b)$.
4. Go to $(u[a][b], b)$.
5. Output $u[a][b]$.

To encode the states algebraically, we'll use the tensor product - i.e., $u \otimes v$ encodes (u, v). The transformations that take each state to the next are illustrated in Figure 4.

The first step corresponds to mapping $L = I_N \otimes u_K$. This makes a vector u to (u, u_K). For step, 2, we apply a permutation matrix $\tilde{A}$ such that $\tilde{A}_{(u,a),(u',a')}$ is 1 iff $a' = a$, and $u' = u[a]$. Note that this fixes the second coordinate and moves the first coordinate according to the second. In the third step, we fix the first coordinate and go to a random neighbor of a in G, which corresponds to applying the identity to the first coordinate and the random walk matrix of G to the second. This is captured by the matrix $\tilde{B} = I_N \otimes B$. Step 4 applies $\tilde{A}$ again. Finally, step 5 sums up all the possible ways we could have reached $u[a][b]$. This corresponds to a linear map $P = I_N \otimes \mathbb{1}_K^T$ that projects from probability distributions on $[N] \times [K]$ to probability distributions on $[N]$.

Altogether, we have

$$M = P\tilde{A}\tilde{B}\tilde{A}L.$$

Using Lemma 2.2, we can decompose $B = (1-\mu)J_K + \mu C$ where $||C|| \leq 1$. Thus, $\tilde{B} = (1-\mu)(I_N \otimes J_K) + \mu(I_N \otimes C) = (1-\mu)(I_N \otimes J_K) + \mu\tilde{C}$. Substituting this back into the definition of M, we have

$$M = (1-\mu)P\tilde{A}(I_N \otimes J_K)\tilde{A}L + \mu P\tilde{A}\tilde{C}\tilde{A}L.$$

For the first term, note that $LP = (I_N \times u_k)(I_N \times 1_k^T) = I_N \otimes J_K$. Additionally, note that $P\tilde{A}L = A$ Thus, $P\tilde{A}(I_N \otimes J_K)\tilde{A}L = A^2$, and has norm at most λ^2. For the second, note that $\|L\| = K^{-1/2}$, $\|P\| = K^{1/2}$, $||\tilde{A}|| = 1$ (since $\tilde{A}$ is a permutation matrix), and $\|\tilde{C}\| \leq 1$. Therefore, $\left\|P\tilde{A}\tilde{C}\tilde{A}L\right\| \leq 1$. Thus, we can write

$$M = (1-\mu)A^2 + \mu D$$

Hence for a vector $v \perp \mathbb{1}_N$, we have

$$\|Mv\| \leq (1-\mu)\|A^2 v\| + \mu\|Dv\| = (\lambda^2 + \mu - \lambda^2\mu)\|v\|,$$

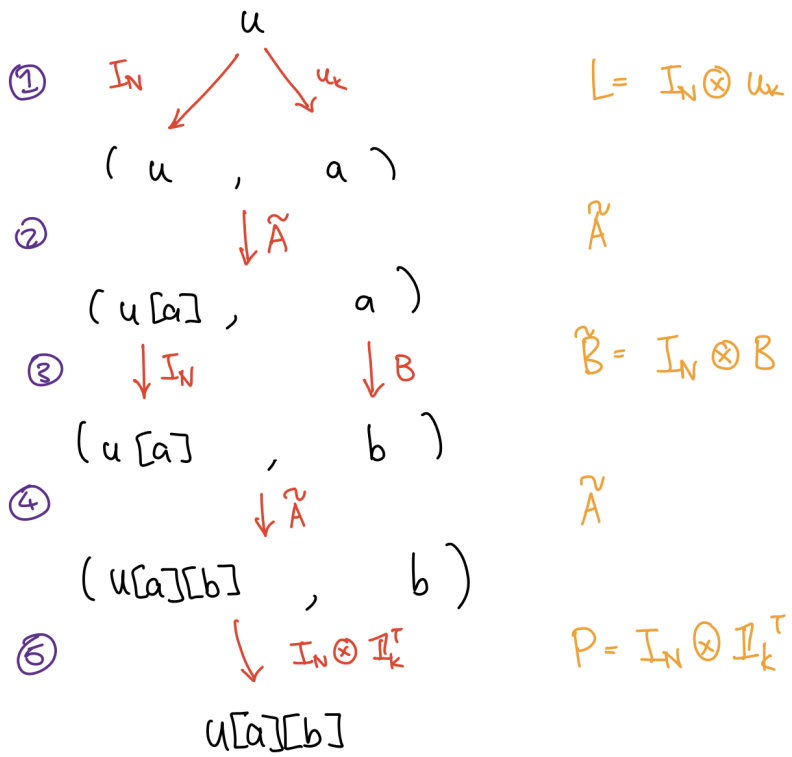

Figure 4: Sequence of transformations

as required.

□

2.7 Sketch of the rest of the proof

Here is an outline of the rest of the proof

- Assume the input graph is 4-regular with a self-loop on each vertex.
- Such a graph starts with some expansion by Lemma 2.4
- Iteratively shrink the expansion using the derandomized square with a sequence of carefully chosen auxiliary graphs and applying Lemma 2.7.
- Repeat until the expansion is small enough that we can apply Lemma 2.3, and argue that the degree is still small enough to do this in logarithmic space.

3 Solving Laplacians in Small Space

For a d-regular undirected multigraph G on n vertices, we will be using $\boldsymbol{M} = \frac{1}{d}A$ to denote the $n \times n$ transition matrix for the random walk on G. Normalized Laplacian of such a graph G is defined as

$$\boldsymbol{L}(G) = \boldsymbol{I} - \boldsymbol{M}(G)$$

And the target of our report is to find a solution for the following equation.

$$\boldsymbol{L}x = b$$

From class, we learned that it is equivalent to finding the Moore-Penrose pseudo-inverse of $\boldsymbol{L}$ denoted by $\boldsymbol{L}^+$, which is an inverse of $\boldsymbol{L}$ in all space except the kernel of $\boldsymbol{L}$.

It can be shown that using a Laplacian solver in an undirected graph, we can solve for st-connectivity in the undirected graph. We saw in previous sections how undirected connectivity is in L, and therefore it is a natural question to ask if we can solve the tougher problem of finding solution to Laplacian equations also in L. It should also be noted that creating a Directed Laplacian solver working in LogSpace will show $RL = L$.

3.1 Peng-Spielman Solver

Before we look at the solution for undirected Laplacian in LogSpace in [Mur+21], we will look at the solution known previously given in [PS14], and why it doesn't work in L.

If we have a simplifying assumption that $\boldsymbol{L}^+ = \boldsymbol{L}^{-1}$, we can look at a simple solution like

$$\boldsymbol{L}^{-1} = \frac{1}{\boldsymbol{I} - \boldsymbol{M}} = \sum_{i=0}^{\infty} \boldsymbol{M}^i = \prod_{k=0}^{\infty} (\boldsymbol{I} + \boldsymbol{M}^{2^k})$$

and we can get a simple solution if we truncate the product at a suitable k. As all eigenvalues of $\boldsymbol{M}$ are ≤ 1, the higher terms contribute less and thus can be approximated. But this identity doesn't work well under spectral approximation, so Peng-Spielman use the following identity in this lemma.

Lemma 3.1. *For a graph G, with random walk transition matrix $\boldsymbol{M}$, Laplacian $\boldsymbol{L}$ and $\boldsymbol{J}$ being the matrix with all entries $\frac{1}{n}$, we have*

$$\boldsymbol{L}(G)^+ = \frac{1}{2} \cdot (\boldsymbol{I} - \boldsymbol{J} + (\boldsymbol{I} + \boldsymbol{M}(G))\boldsymbol{L}(G^2)^+(\boldsymbol{I} + \boldsymbol{M}(G)))$$

Proof. The idea is to show that LHS and RHS have the same eigenvalues and eigenvectors. Let $\Psi_1 = \mathbf{1}, \ldots, \Psi_n$ be the orthonormal eigenvectors of $\boldsymbol{L}$ with eigenvalues $0, \gamma_2, \ldots, \gamma_n$. We know we can write $\boldsymbol{L}$ as $\sum_{i=1}^{n} \gamma_i \Psi_i \Psi_i^\top$. Similarly, we have $\boldsymbol{L}^+ = \sum_{i=2}^{n} \frac{1}{\gamma_i} \Psi_i \Psi_i^\top$.

Consider Ψ_i for $i \neq 1$, and therefore $\Gamma_i \neq 0$. Ψ_i will also be an eigenvector for $\boldsymbol{I}(\boldsymbol{I}\Psi_i = \Psi_i)$ with eigenvalue 1. It will also be an eigenvector for $\boldsymbol{M} = \boldsymbol{I} - \boldsymbol{L}$ ($\boldsymbol{M}\Psi_i = \boldsymbol{I}\Psi_i - \boldsymbol{L}\Psi_i = (1-\gamma_i)\Psi_i$) with eigenvalue $(1 - \gamma_i)$. As we have $\boldsymbol{L}^+ = \sum_{i=2}^{n} \frac{1}{\gamma_i} \Psi_i \Psi_i^\top$, we have RHS multiplied with Ψ_i as

$$\boldsymbol{L}^+ \Psi_i = \left(\sum_{j=2}^{n} \frac{1}{\gamma_j} \Psi_j \Psi_j^\top \right) \Psi_i = \frac{1}{\gamma_i} \Psi_i \Psi_i^\top \Psi_i = \frac{1}{\gamma_i} \Psi_i$$

It similarly will also be an eigenvector of $\boldsymbol{M}^2((1-\gamma_i)^2$ eigenvalue) and $\boldsymbol{I} - \boldsymbol{M}^2(1-(1-\gamma_i)^2$ eigenvalue). Using the definition of Pseudoinverse above, it will also be eigenvector of $\boldsymbol{L}(G^2)^+$ with eigenvalue $\frac{1}{1-(1-\gamma_i)^2}$. Finally, as we know Ψ_i is orthogonal to $\mathbf{1}$, we have Ψ_i as eigenvector of $\boldsymbol{J}(\boldsymbol{J}\Psi_i = 0)$ with eigenvalue 0. So multiplying LHS with Ψ_i we see

$$\begin{aligned}
\frac{1}{2} \cdot (\boldsymbol{I} - \boldsymbol{J} + (\boldsymbol{I} + \boldsymbol{M}(G))\boldsymbol{L}(G^2)^+(\boldsymbol{I} + \boldsymbol{M}(G)))\Psi_i &= \frac{1}{2} \left(1 - 0 + (1 + 1 - \gamma_i) \frac{1}{1 - (1-\gamma_i)^2} (1 + 1 - \gamma_i) \right) \Psi_i \\
&= \frac{1}{2} \cdot \left(1 + \frac{2 - \gamma_i}{\gamma_i} \right) \Psi_i \\
&= \frac{1}{\gamma_i} \Psi_i
\end{aligned}$$

Therefore, Ψ_i is an eigenvector for both LHS and RHS with eigenvalue $\frac{1}{\gamma_i}$, and this is true for all $i \neq 1$.

For eigenvector $\mathbf{1}$, we see that it is an eigenvector for $\boldsymbol{I}, \boldsymbol{J}, \boldsymbol{M}$ with eigenvalue 1. For $\boldsymbol{L}^+\mathbf{1}$ we know it be 0 as all Ψ_i are orthonormal to $\mathbf{1}$. Therefore, $\mathbf{1}$ is eigenvector of LHS with eigenvalue 0. We also know $\boldsymbol{M}^2\mathbf{1} = 1$, and therefore $\boldsymbol{L}(G^2)\mathbf{1} = \boldsymbol{L}^+(G)^2\mathbf{1} = 0$. So we see for RHS multiplied with $\mathbf{1}$

$$\begin{aligned}\frac{1}{2} \cdot (\boldsymbol{I} - \boldsymbol{J} + (\boldsymbol{I} + \boldsymbol{M}(G))\boldsymbol{L}(G^2)^+(\boldsymbol{I} + \boldsymbol{M}(G)))\mathbf{1} &= \frac{1}{2}\left(1 - 1 + (1+1)0(1+1)\right)\mathbf{1} \\ &= 0\end{aligned}$$

Thus, LHS and RHS both have $\mathbf{1}$ as eigenvector with eigenvalue 0. Thus, LHS and RHS have all n eigenvectors and eigenvalues same, and therefore are equal. □

This identity works well under spectral approximation, and thus it's computation can be truncated after $O(\log n)$ iteration of recursion, as we know $\boldsymbol{L}^+(G^{2^{\log n}}) = \boldsymbol{L}^+(G^{2^n}) \approx \boldsymbol{L}^+(\boldsymbol{K}_n) = \boldsymbol{I} - \boldsymbol{J}$.

The problem with repeating the recursion several times is that the degree grows very fast d^{2^k} after k steps, and thus the computation is not feasible.

Peng-Spielman solve this issue using the spectral sparsifiers of [ST11] we studied in class. In every step, you replace G^2 with a sparsified version of it that retains $\frac{\tilde{O}(n)}{\epsilon^2}$ edges giving G' such that $L(G') \approx_\epsilon L(G^2)$. We can easily see in the proof of Lemma 3.1 that replacing $\boldsymbol{L}^+(G^2)$ with $\boldsymbol{L}^+(G')$ will give an ϵ spectral approximation of $\boldsymbol{L}^+(G)$. Since, we repeat these steps $O(\log n)$ times, the final error in $\boldsymbol{L}^+(G_0)$ will be $\epsilon \cdot \log n$ and therefore to keep it constant, we will use $\epsilon = \frac{1}{O(\log n)}$ for constant error approximation, which will keep the number of edges at $\tilde{O}(n)$ and thus computation at each level can be done efficiently.

Problems wrt LogSpace Computation

The main 2 problems that prevent this solution from working in LogSpace are the following:

1. Spectral Sparisifier as we studied in class require random sampling, which cannot be used if we require an algorithm that runs in deterministic LogSpace.
2. In the above computation, even if each level of computation can be done to use $\tilde{O}(\log n)$ space, as there are $O(\log n)$ levels of recursive computation, the total space used will be $O(\log^2 n)$.

To get a solution in L, both these problems need to be solved, and we will use derandomized squaring similar to what we saw earlier for the USTCON problem.

3.2 Using Derandomized Squaring

As we saw in the case of USTCON, we can replace G^2 with derandomized square of G with an expander H, denoted by $G \circledS H$, and have spectral properties remain the same while ensuring the degree doesn't blow up too quickly. This was used in [Mur+21] to give a Laplacian solver using $\tilde{O}(\log n)$ space.

Therefore, the idea will be to replace $\boldsymbol{L}(G^2)^+$ with $\boldsymbol{L}(G \circledS H)^+$, where H is a c-regular graph on d-vertices with absolute value of second-largest eigenvalue ϵ. First, we show that $\boldsymbol{L}(G \circledS H)$ will be an ϵ approximation of $\boldsymbol{L}(G^2)$, and hence it can be used in identity.

Lemma 3.2.

$$\boldsymbol{L}(G^2) \approx_\epsilon \boldsymbol{L}(G \circledS H)$$

Proof. As discussed in proof of 2.7, we have that

$$\boldsymbol{M}(G \circledS H) = P\tilde{A}\tilde{B}\tilde{A}L$$

Recall from the proof that we have $PL = \boldsymbol{I}_n$ and $LP = \boldsymbol{I}_{n\cdot d} \otimes \boldsymbol{J} = \boldsymbol{I}_n \otimes \boldsymbol{J}_d$, and $\tilde{A}$ is a symmetric $(n \cdot d) \times (n \cdot d)$ matrix with property $\tilde{A}^2 = \boldsymbol{I}_n$. Since, H is an ϵ expander, we have that $\boldsymbol{L}_H \approx_\epsilon \boldsymbol{L}_{K_d}$ where K_d denotes the clique on d-vertices. As B was the transition matrix for H, we have $\boldsymbol{L}_H = \boldsymbol{I} - B$, and $\boldsymbol{I}_n \otimes \boldsymbol{L}_H = \boldsymbol{I}_{n\cdot d} - \tilde{B}$. We also have $\boldsymbol{M}(G) = P\tilde{A}L$

So we start with

$$\boldsymbol{L}(G \circledS H) = \boldsymbol{I}_n - \boldsymbol{M}(G \circledS H)$$

Using the above identities, we can write $\boldsymbol{I}_n$ as $P\tilde{A}\boldsymbol{I}_{n\cdot d}\tilde{A}L$, while we know $\boldsymbol{M}(G \circledS H) = P\tilde{A}\tilde{B}\tilde{A}L$. Therefore, we have

$$\begin{aligned}
\boldsymbol{L}(G \circledS H) &= P\tilde{A}\boldsymbol{I}_{n\cdot d}\tilde{A}L - P\tilde{A}\tilde{B}\tilde{A}L \\
&= P\tilde{A}(\boldsymbol{I}_{n\cdot d} - \tilde{B})\tilde{A}L \\
&= P\tilde{A}(\boldsymbol{I}_n \otimes \boldsymbol{L}_H)\tilde{A}L
\end{aligned}$$

As we have $\boldsymbol{L}_H \approx_\epsilon \boldsymbol{L}_K$, we can replace $\boldsymbol{I}_n \otimes \boldsymbol{L}_H$ with $\boldsymbol{I}_n \otimes \boldsymbol{L}_K$ approximately. Also, we know $\boldsymbol{L}_K = \boldsymbol{I}_d - \boldsymbol{J}_d$, so $\boldsymbol{I}_n \otimes \boldsymbol{L}_K = \boldsymbol{I}_n \otimes \boldsymbol{I}_d - \boldsymbol{I}_n \otimes \boldsymbol{J}_d = \boldsymbol{I}_{n\cdot d} - LP$. Therefore, we have

$$\begin{aligned}
\boldsymbol{L}(G \circledS H) &= P\tilde{A}(\boldsymbol{I}_n \otimes \boldsymbol{L}_H)\tilde{A}L \\
&\approx_\epsilon P\tilde{A}(\boldsymbol{I}_n \otimes \boldsymbol{L}_K)\tilde{A}L \\
&= P\tilde{A}(\boldsymbol{I}_{n\cdot d} - \boldsymbol{I}_n \otimes \boldsymbol{J}_d)\tilde{A}L \\
&= P\tilde{A}\boldsymbol{I}_{n\cdot d}\tilde{A}L - P\tilde{A}(\boldsymbol{I}_n \otimes \boldsymbol{J}_d)\tilde{A}L \\
&= \boldsymbol{I}_n - P\tilde{A}LP\tilde{A}L \\
&= \boldsymbol{I}_n - (\boldsymbol{M}(G))^2 \\
&= \boldsymbol{I}_n - \boldsymbol{M}(G^2) \\
&= \boldsymbol{L}(G^2)
\end{aligned}$$

Therefore, we have $\boldsymbol{L}(G^2) \approx_\epsilon \boldsymbol{L}(G \circledS H)$.

□

Sequence of Derandomized Square

We start with $G_0 = G$ as our original graph. We use a sequence of c-regular Expanders $H_0, H_1, \cdots, H_k$ for k derandomized squares and use of Lemma 3.1. H_i is an ϵ-expander on $d \cdot c^i$ vertices. WE define G_{i+1} as follows

$$G_{i+1} = G_i \circledS H_i$$

It is easy to see that G_i is a $d \cdot c^i$-regular multigraph on n vertices. We go up to $k = O(\log n)$ iterations, at which points $G_k \approx K_n$ and $\boldsymbol{L}_{G_k} \approx \boldsymbol{I} - \boldsymbol{J}$. To prevent the degree from growing exponentially, so that we can do the computations in logspace, we choose $c = poly(\log n)$ and therefore all degree remain small. We also have the new version of Lemma 3.1, which we use for computation:

$$L(G_i)^+ \approx_\epsilon \frac{1}{2} \cdot (I - J + (I + M(G_i))L(G_{i+1})^+(I + M(G_i)))$$

Now, we are left with the task of showing how to do this whole computation using only Log space.

3.3 Computation in Log Space

We have the sequence $G_{i+1} = G_i \circledS H_i$ as described above and we have to compute till $k = O(\log n)$ iteration the following identity using only Log space

$$L(G_i)^+ \approx_\epsilon \frac{1}{2} \cdot (I - J + (I + M(G_i))L(G_{i+1})^+(I + M(G_i)))$$

So to obtain the solution of $\boldsymbol{L}x = b$ as $\boldsymbol{L}^+b$, we don't compute and store $\boldsymbol{L}^+$ as it will take $O(n^2)$ space, rather we compute each element of $\boldsymbol{L}^+$ on need to be computed basis, and thus compute $\boldsymbol{L}^+b$. We can open the above identity for k iterations as the following

$$L(G_0)^+ \approx \frac{1}{2}(\boldsymbol{I} - \boldsymbol{J}) + \left(\sum_{i=0}^{k-1} \frac{1}{2^{i+2}} W_i\right) + \frac{1}{2^{k+1}} W_k$$

with $W_i = (\boldsymbol{I} + \boldsymbol{M}_0) \dots (\boldsymbol{I} + \boldsymbol{M}_i)(\boldsymbol{I} - \boldsymbol{J})(\boldsymbol{I} + \boldsymbol{M}_i) \dots (\boldsymbol{I} + \boldsymbol{M}_0)$, where $\boldsymbol{M}_i$ denotes $\boldsymbol{M}(G_i)$.

Therefore, to compute the $(L(G_0)^+)_{ij}$, we need to be able to compute $(\boldsymbol{M}_l)_{ij}$ for all $l \in [k]$, and ij-th element of the product of k matrices.

From the computation for USTCON solution, we already know we can compute $(\boldsymbol{M}_l)_{ij}$, using only $\log(n)$ space, for derandomized squaring up to $k = O(\log n)$ iterations.

Thus, if we show we can multiply $k = O(\log n)$ matrices using only log space, we can compute $\boldsymbol{L}^+$ in $\tilde{O}(\log n)$ space. Thus, proving the following lemma completes the proof.

Lemma 3.3. *Given matrices $M^{(1)}, \dots, M^{(k)}$ and indices i, j $(M^{(1)} \cdot M^{(2)} \cdot \dots \cdot M^{(k)})_{ij}$ can be computed using $O(\log n \cdot \log k)$ space where n is the dimension of the input matrices.*

Proof. To show this, we first argue that we can compute the ij-th entry of the product of 2 matrices $M^{(1)}$ and $M^{(2)}$ in log space. The basic identity is as follows:

$$(M^{(1)} \cdot M^{(2)})_{ij} = \sum_{l=1}^{n} M^{(1)}_{il} M^{(2)}_{lj}$$

To compute this, we will only need to store the index l, which uses $O(\log n)$ space, and the current sum (which we assume can be stored in $O(\log n)$ space as elements of M are small). We also assume we can multiply 2 elements of the matrix in $O(\log n)$ space.

To multiply k of such matrices, we use the following identity made using divide and conquer:

$$(M^{(1)} \dots M^{(k)})_{ij} = \sum_{l=1}^{n} (M^{(1)} \dots M^{(\lfloor k/2 \rfloor)})_{il} (M^{(\lfloor k/2 \rfloor + 1)} \dots M^{(k)})_{lj}$$

As the number of matrices multiplied at each level decreases by half at each step, there will be $O(\log k)$ recursion levels. In this we see that at each recursion level, we will be storing a l which will require $O(\log n)$ space, and the current sum. Thus, the total space used by this algorithm will be $O(\log k \log n)$. □

As we have $k = O(\log n)$, we have that we can compute the above product using $\tilde{O}(\log n)$ space, and thus compute the ij-th element each of the W_i's. Thus, we can compute the ij-th element of $\boldsymbol{L}^+$ as

$$L(G_0)^+ \approx \frac{1}{2}(\boldsymbol{I} - \boldsymbol{J}) + \left(\sum_{i=0}^{k-1} \frac{1}{2^{i+2}} W_i\right) + \frac{1}{2^{k+1}} W_k$$

in $\tilde{O}(\log n)$ space, and therefore the solution $\boldsymbol{L}^+ b$ as well. This completes the proof of us computing the solution of Laplacian in $\tilde{O}(\log n)$ space.

References

[Ale+79] Romas Aleliunas et al. "Random Walks, Universal Traversal Sequences, and the Complexity of Maze Problems". In: 20th Annual Symposium on Foundations of Computer Science (Sfcs 1979). 20th Annual Symposium on Foundations of Computer Science (Sfcs 1979). Oct. 1979, pp. 218–223. DOI: 10.1109/SFCS.1979.34 (cit. on p. 1).

[Rei05] Omer Reingold. "Undirected Connectivity in Log-Space". In: J. ACM 55.4 (2005), pp. 1–24. ISSN: 0004-5411, 1557-735X. DOI: 10.1145/1391289.1391291. URL: https://dl.acm.org/doi/10.1145/1391289.1391291 (cit. on p. 2).

[RV05] Eyal Rozenman and Salil Vadhan. "Derandomized Squaring of Graphs". In: APPROX'05/RANDOM'05. Berlin, Heidelberg: Springer-Verlag, Aug. 22, 2005, pp. 436–447. ISBN: 978-3-540-28239-6. DOI: 10.1007/11538462_37. URL: https://doi.org/10.1007/11538462_37 (cit. on pp. 2, 4).

[ST11] Daniel A. Spielman and Shang-Hua Teng. "Spectral Sparsification of Graphs". In: SIAM J. Comput. 40.4 (Jan. 2011), pp. 981–1025. ISSN: 0097-5397. DOI: 10.1137/08074489X. URL: https://epubs.siam.org/doi/10.1137/08074489X (cit. on p. 8).

[PS14] Richard Peng and Daniel A. Spielman. "An Efficient Parallel Solver for SDD Linear Systems". In: Proceedings of the forty-sixth annual ACM symposium on Theory of computing. STOC '14 (May 31, 2014), pp. 333–342. DOI: 10.1145/2591796.2591832. URL: https://doi.org/10.1145/2591796.2591832 (cit. on pp. 2, 7).

[Mur+21] Jack Murtagh et al. "Derandomization beyond Connectivity: Undirected Laplacian Systems in Nearly Logarithmic Space". In: SIAM J. Comput. 50.6 (Jan. 2021), pp. 1892–1922. ISSN: 0097-5397. DOI: 10.1137/20M134109X. URL: https://epubs.siam.org/doi/abs/10.1137/20M134109X (cit. on pp. 2, 7, 8).