CS289a: Great Theory Hits of 21st Century

Tingfeng Xia

Winter 2023

Tingfeng Xia.

Notes reorganized from https://hackmd.io/@raghum/greathits.

Contents

Chapter 1

SL = L: Undirected connectivity in Logspace

1.1 Computing Resources

Four main computing resources that we consider as limited (and measure the performance of our algorithms against)

- Time
- Memory
- Randomness
- Communication

1.2 Problem Statement

- **Input**: Graph $G = (V, E)$; with source and target marked as s, t
- **Output**: YES iff s and t are connected, NO otw.

Above is the "traditional" definition of $s - t$ connectivity which we can solve with a vanilla BFS or DFS. This will take $\mathcal{O}(|V| + |E|)$ and $\mathcal{O}(|V|)$ extra bits of space / memory. The question is then, can we solve the same problem with sub-linear extra memory usage.

Proposition 1.1

There is a randomized algorithm with $5 \log |V|$ bits of additional memory (directed and undirected graphs).

Proposition 1.2: Omer Reingold, 2005

There is a log space ($O(\log |V|)$) algorithm **(deterministic)** for undirected graphs. [a]

[a] first great hit ...

It is yet unknown if we can achieve log space for directed graphs (with deterministic algorithm). The best known algorithms runs with $O(\log |V|)^{3/2}$ bits of memory. Why is this so challenging?

Proposition 1.3

If divided $s - t$ connectivity can be solved with $O(\log |V|)$ extra bits of memory (without randomness), then any randomized algorithm can be made deterministic at the expenses of a constant factor increase in memory.

1.2.1 Log Space USTCON - Results

Here we highlight the progression in space complexity in various papers

- **Nisan, 92**: Space $O(\log^2 N)$, time $N^{O(1)}$ algorithm... improved to $O(\log^{4/3} N)$ in space.
- **Reingold, 05**: Space $O(\log N)$, time $N^{O(1)}$ algorithm.
- **Trifornov, 05**: Space $O((\log N)(\log \log N))$ algorithm.

1.3 Randomized Algorithm for Connectivity

Algorithm 1.1: Random Walk Algorithm for Connectivity

Here is the algorithm

```
• $steps \leftarrow 0$
• $current \leftarrow s$; $target \leftarrow t$
• while $steps < T$
    – $current \leftarrow$ random neighbor of current
    – if $current == target$ return $YES$
• return $NO$
```

The total memory for this algorithm is

$$2 \log N + \log T \leq 5 \log N \tag{1.3.1}$$

extra bits, assuming we can get random neighbor.

Proposition 1.4: Alenilaus, 80s

If $T = 100N^3$ steps, then Pr[Algorithm wrong] $< \frac{1}{3}$

which can improved to arbitrary accuracy by repeating the algorithm. Algorithms of this nature can perform bad on graphs known as "Lollipop Graphs" and even worse a "Dumbell Graph"

1.4 Spectral Graph Theory

Consider an undirected graph $G = (V, E)$,

Definition 1.1: Degree

Degree of a vertex v is the number of edges v is connected to.

Definition 1.2: Regular

Graphs is "regular" if all vertices have same degree.

Definition 1.3: Adjacency Matrix

$A(G)$ is a symmetric matrix where $A(G)_{ij} = 1$ if $\{i, j\}$ is an edge, 0 otw.

Definition 1.4: Normalized Adj Matrix

If G is regular and has degree D, then the normalized adjacency matrix is defined as

$$M(G) \equiv \frac{A(G)}{D} \tag{1.4.1}$$

Lemma 1.1

If G is regular, then 1 is an eigenvalue of $M(G)$. And $\mathbf{v}_1 = \begin{bmatrix} 1 & 1 & \dots & 1 \end{bmatrix}^\top$ is an eigenvector with eigenvalue 1.

Proposition 1.5: Eigenvalues of Regular Graphs

If G is regular, then all eigenvalues of $M(G)$ have magnitude ≤ 1.

Proof: WLOG assume x_3 is the largest entry in the vector $\mathbf{x}$, then

$$\begin{aligned}
\lambda|x_3| &= |M_{31}x_1 + M_{32}x_2 + \ldots + M_{3N}x_N| && (1.4.2)\\
&\le M_{31}|x_3| + M_{32}|x_3| + \ldots + M_{3N}|x_3| && (1.4.3)\\
&= (M_{31} + \ldots + M_{3N})|x_3| && (1.4.4)\\
&= 1|x_3| && (1.4.5)
\end{aligned}$$

Thus, $\lambda \le 1$. ■

Proposition 1.6: Connectedness and Matrices

Regular $G = (V, E)$ is connected if and only if the only eigenvector with eigenvalue 1 for $M(G)$ is the all 1 vector. [a]

[a] i.e., eigenvalue 1 has an multiplicity of 1.

Proof: [Regular $G = (V, E)$ is connected implies $\lambda = 1$ has multiplicity of 1 for $M(G)$.] From proof to Prop. 1.5 we already know that $|\lambda| \le 1$. With $x_j = \max(\mathbf{x})$ as the largest entry in the eigenvector, we recall (and abstract the inequality used back then as

$$|\lambda||x_j| = |(M(G)\mathbf{x})_j| = \left| \sum_{v_i \in N(v_j)} x_i \right| /D \le |x_j| \qquad (1.4.6)$$

We are now interested in the condition of when $\lambda = 1$, $|\lambda||x_j| = |x_j|$, in which case we need

$$x_i = x_j, \quad \forall v_i \in N(v_j) \qquad (1.4.7)$$

This suffices as a proof to every eigenvector with eigenvalue 1 to $M(G)$ is the **1** vector. ■

Proof: [$\lambda = 1$ has multiplicity of 1 for $M(G)$ implies regular $G = (V, E)$ is connected.] todo …

Proposition 1.7: Eigenvalues of a Regular Graph

If G is regular, then the eigenvalues of $M(G)$ are

$$1 = \lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_N \qquad (1.4.8)$$

Proof: This follows from the proof for Prop. 1.5 where we proved that all eigenvalues of $M(G)$ have magnitude ≤ 1. Since the one vector **1** is an eigenvector of $M(G)$ with eigenvalue one, we know that $\lambda_1 = 1$ is attainable. This suffices as a proof. ■

Proposition 1.8

G is connected and regular if and only if on $M(G)$

$$\max(|\lambda_2|, |\lambda_3|, \ldots, |\lambda_N|) < 1 \qquad (1.4.9)$$

Proof: todo ...

Proposition 1.9: Eigenvalues of D-Regular Graphs

If G is a D-regular graph, then

- 1 is an eigenvalue of $M(G)$, and
- all eigenvalues of $M(G)$ are at most 1 in absolute value

Definition 1.5: Self Loops

We add connections from each node in the graph to themselves. In the matrix representation, we set $G_{ii} = 1, \forall i$.

Definition 1.6: Second Largest Eigenvalue

... denoted as $\lambda(G)$ or $\lambda_2(G)$.

Lemma 1.2

If G is D-regular and **has self loops**, then G is connected if and only if $\lambda(G) < 1$.

Proof: [G is disconnected implies $\lambda(G) = 1$ (via contrapositive).] Consider a graph G such that it is comprised of two clouds of disjoint graphs G_1 and G_2. Then the adjacency matrix of G will take a block matrix form

$$M_G = \begin{bmatrix} M_{G_1} & [\mathbf{0}] \\ [\mathbf{0}] & M_{G_2} \end{bmatrix} \tag{1.4.10}$$

From linear algebra, we know that the eigenvalues of M_G will be the union of eigenvalues of M_{G_1} and M_{G_2}. Now, consider

$$\mathbf{x}^{(1)} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 0 \\ 0 \end{bmatrix} \quad \text{and} \quad \mathbf{x}^{(2)} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ 1 \end{bmatrix} \tag{1.4.11}$$

are both eigenvectors of M_G with eigenvalues of 1. Hence, there are two orthogonal eigenvectors with eigenvalue 1, and $\lambda(G) = 1$ as wanted. ■

Proof: [If G is connected, then $\lambda(G) < 1$.] We already know that 1 is an eigenvalue of M_G with the $\mathbf{1}$ vector as eigenvector. Suppose λ is also an eigenvalue with $\mathbf{v}$ as an eigenvector and $\mathbf{v}$ is perpendicular to $\mathbf{1}$. Now,

$$\mathbf{1} \perp \mathbf{v} \implies \langle \mathbf{v}, \mathbf{1} \rangle = v_1 + v_2 + \cdots + v_N = 0 \tag{1.4.12}$$

The vector **v** must contain some positive entries and some negative entries, we separate them into two sets

$$P = \{i : v_i \geq 0\} \quad \text{and} \quad N = \{i : v_i < 0\} \tag{1.4.13}$$

where both sets are non-empty by Eq. 1.4.12. Taking a step back and reorganize the goal into matrix form

$$M_G \begin{bmatrix} + \\ + \\ \vdots \\ - \\ - \end{bmatrix} = \lambda \begin{bmatrix} + \\ + \\ \vdots \\ - \\ - \end{bmatrix} \qquad \text{where} \qquad \begin{bmatrix} + \\ + \\ \vdots \\ - \\ - \end{bmatrix} = \begin{bmatrix} \mathbf{P} \\ - \\ \mathbf{N} \end{bmatrix} = \mathbf{v} \tag{1.4.14}$$

Per element,

$$\sum_{j=1}^{N} M_G[i,j] \cdot v_j = \lambda \cdot v_i, \quad \forall i \tag{1.4.15}$$

By the connectedness assumption, there must always be some edge connecting P and N the two sets, so

$$\lambda \left(\sum_{i \in P} v_i \right) = \sum_{i \in P} \left(\sum_{j=1}^{N} M_G[i,j] \cdot v_j \right) \tag{1.4.16}$$

$$= \sum_{j=1}^{N} v_j \sum_{i \in P} M_G[i,j] \tag{1.4.17}$$

$$= \sum_{j \in P} v_j \left(\sum_{i \in P} M_G[i,j] \right) + \sum_{j \in N} v_j \left(\sum_{i \in P} M_G[i,j] \right) \tag{1.4.18}$$

$$\leq \sum_{j \in P} v_j (1) + \sum_{j \in N} v_j \left(\sum_{i \in p} M_G[i,j] \right) \tag{1.4.19}$$

$$< \sum_{j \in P} v_j \tag{1.4.20}$$

where in the last two steps we utilized the facts that M_G's columns add up to 1 and we have at least 1 non-zero entry in each row and col of M_G.

In summary, we obtained

$$\lambda \left(\sum_{i \in P} v_i \right) < \left(\sum_{j \in P} v_j \right) \quad \Longrightarrow \quad \lambda < 1 \tag{1.4.21}$$

■

Definition 1.7: Spectral Gap

Spectral Gap of a D-regular graph G is defined as

$$\text{SpectralGap}(G) \equiv 1 - \lambda(G) \tag{1.4.22}$$

Lemma 1.3

If G is a D-regular connected graph with self-loops, then

$$\lambda(G) \leq 1 - \frac{1}{2D^2 \cdot N^2} \tag{1.4.23}$$

Definition 1.8

We say a graph G is (N, D, λ) if it has N vertices, D regular and $\lambda(G) \leq \lambda$.

1.5 Path Enumeration

The simplest case is wen the shortest path between s, t is short. Then, we can enumerate all paths of some length and see if t is reached.

The algorithm goes as follows

Algorithm 1.2

1, Explore all paths of length less than or equal to T from s. 2, If you reach t in these explorations, output YES. If not, output NO.

This takes $O(\log D) \cdot T$ extra space, where D is the degree of the graph and T is the loop times.

Definition 1.9: Graph Diameter

Diameter of a graph is defined as the length of the longest shortest path for any pair of vertices. By convention,

- G disconnected, diameter = ∞, and
- G connected, diameter = $\max_{i \neq j} (ShortestPath(i, j))$

Proposition 1.10: Extra Space for Path Enumeration

Path enumeration will solve the $s - t$ connectivity in with max extra space

$$(\log D) \cdot \Delta(G) \tag{1.5.1}$$

bits, where $\Delta(G)$ is the max diameter of connected components of G.

Proposition 1.11

If G is connected, D-regular, has self-loops, then[a]

$$Diameter(G) \leq \lceil \log_{\frac{1}{\lambda}} N \rceil + 1 \tag{1.5.2}$$

[a] λ is the second largest eigenvalue, N is the matrix size (number of nodes).

1.5.1 Reingold's Idea

We see from the proposition above that the bigger the spectral gap, the smaller the number of extra bits we need in space for the algorithm. The problem then is how we can transform the graph enlarging the spectral gap while not hurting the degree too much. Formally, we want to transform (G, s, t) to $(\bar{G}, \bar{s}, \bar{t})$ such that

- s, t connected in G if and only if $\bar{s}, \bar{t}$ connected in $\bar{G}$, and
- $\lambda(\bar{G}) < \lambda(G)$, and
- $Degree(\bar{G})$ is not much worse then $Degree(G)$

1.5.2 Reducing Degree

For the first part of Eq. 1.5.1, we can reduce the degree of any graph with

Algorithm 1.3: Degree Reduce Procedure

The procedure,

- Break each edge into two vertices, and
- Add local edges at each "old" vertices, and
- Add self loops to make graph

Proposition 1.12

The procedure outlined above generates a degree 4 graph.

1.5.3 Improving Spectral Gap

Definition 1.10: Multi-graphs

A multi-graph is a superset of our old definition of a graph, except we allow repeated edges between nodes. This is represented as values larger than 1 in the adjacency matrix. All definitions are carried over without change: degree, normalized adjacency matrix, and $\lambda(G)$.

With the degree reducing algorithm in Sec. 1.5.1, we can reduce any graph to a degree of 4. This means Eq. 1.5.1 is now transformed into

$$(\log 4) \cdot \Delta(G) = 2 \cdot \Delta(G) \tag{1.5.3}$$

extra bits of storage. How should we improve

$$\Delta(G) \leq \log_{\frac{1}{\lambda}} N \tag{1.5.4}$$

which is the largest diameter of any connected component?

Idea: Input G, s, t where G has self-loops and transform that into G', s', t' where

$$\lambda(G') \ll \lambda(G) \tag{1.5.5}$$

Goal: Operations to improve (decrease) the second largest eigenvalue.

Definition 1.11: Squaring the Graph

Add new edges: if (u, v) and (v, w) are edges, then add an edge (u, w).

Proposition 1.13: Adjacency of Squared Graph

$$A_{G^2} = (A_G)^2 \tag{1.5.6}$$

in matrix representation, and we allow multi-graph in this setting.

Proof:

$$(A_G)^2_{[i,j]} = \sum_{k=1}^{N} (A_G)_{[i,k]} (A_G)_{[k,j]} \tag{1.5.7}$$

■

Proposition 1.14: Squared Graph Spectral Gap

If G is a (N, D, λ) graph with self loops, then G^2 is a (N, D^2, λ^2) graph with self loop. Since connected $\implies \lambda < 0, \lambda^2 < \lambda$.

Theorem 1.1: Squared Matrix Spectral Decomposition

M is a symmetric matrix with eigenvalues

$$\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \cdots \geq \lambda_N \tag{1.5.8}$$

then, M^2 is a symmetric matrix with the same eigenvectors but with eigenvalues

$$\lambda_1^2, \lambda_2^2, \dots, \lambda_N^2 \tag{1.5.9}$$

Proof: For M, we have

$$M\mathbf{x} = \lambda\mathbf{x} \tag{1.5.10}$$

Then,

$$M^2\mathbf{x} = \lambda M\mathbf{x} = \lambda^2\mathbf{x} \tag{1.5.11}$$

This concludes the proof. ■

Corollary 1.1.1: Squared Graph Eigenvalues

It follows from Thm. 1.1 directly that if $\lambda_1, \dots, \lambda_N$ are eigenvalues for the original graph matrix M, then the new squared M^2 matrix has the same eigenvectors but with eigenvalues $\lambda_1^2, \lambda_2^2, \dots, \lambda_N^2$ instead.

Proposition 1.15: Normalized Adjacency of Squared Graph

The normalized graph matrix of G^2, is such that

$$M_{G^2} = (M_G)^2 \tag{1.5.12}$$

Proof: Recall that

$$M_G = \frac{A_G}{D} \tag{1.5.13}$$

Then,

$$M_{G^2} = \frac{A_{G^2}}{D^2} = \frac{(A_G)^2}{D^2} = \left(\frac{A_G}{D}\right)^2 = (M^G)^2 \tag{1.5.14}$$

This concludes the proof. ■

Proposition 1.16: Square Graph Does Not Save Memory

Recall that our initial goal was to save extra memory used. Here with squaring, though we enlarged the spectral gap as desired ($(1-\lambda) \to (1-\lambda^2)$), the degree got larger ($D \to D^2$). In total, extra bits is

$$\begin{aligned} (\log D) \cdot \log_{\frac{1}{\lambda}} N &\rightsquigarrow \left(\log D^2\right) \log_{\frac{1}{\lambda^2}} N && (1.5.15) \\ &= 2 \cdot \log D \cdot \frac{1}{2} \cdot \log_{\frac{1}{\lambda}} N && (1.5.16) \\ &= (\log D) \cdot \log_{\frac{1}{\lambda}} N && (1.5.17) \end{aligned}$$

which is exactly what we had before. This suffices as a proof for squaring matrices alone does not bring any memory savings. ■

Goal Taking a step back, we can see that we need to find a powering operation that improves the second largest eigenvalue **while not increasing degree too much**. This leads to the following algorithm:

Algorithm 1.4: Reingold, 2005

For a graph specified as (G, s, t) where G is 4-regular and has self-loops, define a recursive relationship

$$G_{i+1} = G_i^2 \text{ⓩ} H \tag{1.5.18}$$

where H is a special graph. This recursion covers the transformation

$$(G, s, t) \rightsquigarrow (G_1 = G^2 \text{ⓩ} H, \bar{s}, \bar{t}) \rightsquigarrow (G_2 = G_1^2 \text{ⓩ} H, \bar{\bar{s}}, \bar{\bar{t}}) \rightsquigarrow \ldots \tag{1.5.19}$$

Remark G_i^2 part decreases the second largest eigenvalue, and the ⓩH part brings down the degree while not hurting second largest eigenvalue.

Definition 1.12: Consistent Labelling

G is a D-regular graph. A consistent labelling is a mapping

$$L : \mathbb{E} \to [D] \tag{1.5.20}$$

such that at each vertex all edges of the vertex have distinct labels.

Example Figure 1.1 depicts a consistent edge labelling of the graph.

Definition 1.13: Zig Zag Product

Input & Output

$$\left.\begin{aligned} G &: (N, D, -) \\ H &: (D, D_1, -) \end{aligned}\right\} \to G \text{ⓩ} H : (ND, D_1^2, -) \tag{1.5.21}$$

Rotations

$$\left.\begin{aligned} Rot_G &: [N] \times [D] \to [N] \times [D] \\ Rot_H &: [D] \times [D_1] \to [D] \times [D_1] \end{aligned}\right\} \tag{1.5.22}$$

$$\to Rot_{G \text{ⓩ} H} : [N \cdot D] \times \left([D_1^2]\right) \to [N \cdot D] \times \left([D_1^2]\right) \tag{1.5.23}$$

$$\equiv Rot_{G \text{ⓩ} H} : [N \cdot D] \times ([D_1] \times [D_1]) \to [N \cdot D] \times ([D_1] \times [D_1]) \tag{1.5.24}$$

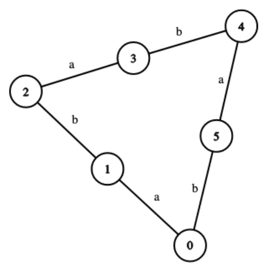

Figure 1.1: Illustration of consistent labelling.

and

$$Rot_{G \textcircled{z} H}\left((v,a),(k_1,k_2)\right): \tag{1.5.25}$$
$$\rightarrow (a',i') \leftarrow Rot_H(a,k_1) \tag{1.5.26}$$
$$\rightarrow (w,b') \leftarrow Rot_G(v,a') \tag{1.5.27}$$
$$\rightarrow (b,i'') \leftarrow Rot_H(b',k_2) \tag{1.5.28}$$
$$\rightarrow \text{output}\left((w,b),(k_2,k_1)\right) \tag{1.5.29}$$

English Explanation The Zig-Zag product $G \textcircled{z} H$ replaces each vertex of G with a copy (cloud) of H, and connects the vertices by moving a small step (zig) inside the cloud, followed by a big step between two clouds, and finally performs another small step (zag) inside the destination cloud.

1.6 Zig Zag Analysis

In the previous section, we highlighted a combinatorial product between two graphs called Zig Zag product. Here we present properties and analysis of the product.

Goal Consider the definition, where we are given graphs

$$G:(N,D,\lambda_G) \qquad \text{and} \qquad H:(D,D_1,\lambda_H) \tag{1.6.1}$$

What can we tell about $G \textcircled{z} H$? In particular, it will be a $(ND, D_1^2, ??)$ graph?

Definition 1.14: Tensor Product

For $A \in \mathbb{R}^{d_1 \times 1}$ and $B \in \mathbb{R}^{d_2 \times d_2}$,

$$\mathbb{R}^{(d_1^2)\times(d_2^2)} \ni A \otimes B \quad \text{where} \quad (A \otimes B)_{ij} = [A_{ij}B] \tag{1.6.2}$$

Proposition 1.17: Adjacency of Zig Zag Product

For $F = G Ⓩ H$,

$$A_F = (\mathbb{I}_N \otimes A_H) \cdot P_G \cdot (\mathbb{I}_N \otimes A_H) \tag{1.6.3}$$

$$= \begin{bmatrix} [A_H] & [\mathbf{0}] & \dots & [\mathbf{0}] \\ [\mathbf{0}] & [A_H] & \dots & [\mathbf{0}] \\ \vdots & \vdots & \ddots & [\vdots] \\ [\mathbf{0}] & [\mathbf{0}] & \dots & [A_H] \end{bmatrix} \begin{bmatrix} [\dots] & [\dots] & \dots & [\dots] \\ [\dots] & [\dots] & \dots & [\dots] \\ \vdots & \vdots & \ddots & [\vdots] \\ [\dots] & [\dots] & \dots & [\dots] \end{bmatrix} \begin{bmatrix} [A_H] & [\mathbf{0}] & \dots & [\mathbf{0}] \\ [\mathbf{0}] & [A_H] & \dots & [\mathbf{0}] \\ \vdots & \vdots & \ddots & [\vdots] \\ [\mathbf{0}] & [\mathbf{0}] & \dots & [A_H] \end{bmatrix} \tag{1.6.4}$$

where each [...] inside is of size $(D \times D)$, and thus the outer matrices are all of sizes $(D \times N) \times (D \times N)$. Graphically, we note that the two $(\mathbb{I}_N \otimes A_H)$ parts represent zig and zag steps respectively in the product while the P transition is the inter-cloud big step.[a]

Normalized Adjacency

$$M_F = A_F/D = (\mathbb{I}_N \otimes M_H) \cdot P_G \cdot (\mathbb{I}_N \otimes M_H) \tag{1.6.5}$$

[a]A permutation transition P guarantees that P has only one 1 in each row and column.

Lemma 1.4: Linear Algebra: LA1

For G is a $(N, D, -)$ graph,

$$\lambda_G \leq \lambda \iff M_G = (1-\lambda)\frac{J_N}{N} + \lambda \cdot E \tag{1.6.6}$$

[a] where [b]

$$\|E\| \leq 1 \tag{1.6.7}$$

[a]J_N is a box of $N \times N$ matrix, filled with 1's in every entry.
[b]$\|\cdot\|$ is the spectral norm of a matrix, **and is equal to the largest absolute eigenvalue for a symmetric matrix.**

Remark Consider spectral decomposition of M_G,

$$M_G = 1 \cdot \begin{bmatrix} 1/\sqrt{n} \\ 1/\sqrt{n} \\ \vdots \\ 1/\sqrt{n} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{n}} & \frac{1}{\sqrt{n}} & \cdots & \frac{1}{\sqrt{n}} \end{bmatrix} + \lambda_2 \mathbf{v}_2 \mathbf{v}_2^\top + \lambda_3 \mathbf{v}_3 \mathbf{v}_3^\top + \cdots + \lambda_N \mathbf{v}_N \mathbf{v}_N^\top \tag{1.6.8}$$

$$= 1 \left(\frac{J_N}{N} \right) + \lambda_2 \mathbf{v}_2 \mathbf{v}_2^\top + \lambda_3 \mathbf{v}_3 \mathbf{v}_3^\top + \cdots + \lambda_N \mathbf{v}_N \mathbf{v}_N^\top \tag{1.6.9}$$

$$= (1-\lambda) \left(\frac{J_N}{N} \right) + \underbrace{\left(\lambda \left(\frac{J_N}{N} \right) + \lambda_2 \mathbf{v}_2 \mathbf{v}_2^\top + \lambda_3 \mathbf{v}_3 \mathbf{v}_3^\top + \cdots + \lambda_N \mathbf{v}_N \mathbf{v}_N^\top \right)}_{E} \tag{1.6.10}$$

If fact this sum that we called E is the Eigen decomposition of E itself!

Lemma 1.5: Linear Algebra LA 2

For $A, B \in \mathbb{R}^{N \times N}$,

- $\|A + B\| \leq \|A\| + \|B\|$
- $\|A \cdot B\| \leq \|A\| \cdot \|B\|$
- $\|A \otimes B\| = \|A\| \cdot \|B\|$

Lemma 1.6: Linear Algebra LA 3

For any permutation matrix (each row and column has exactly one non-zero) P,

$$\|P\| = 1 \tag{1.6.11}$$

Theorem 1.2: Rozenmann-Vadhan 05, RVW 01

For

$$G : (N, D, \lambda_G) \quad \text{and} \quad H : (D, D_1, \lambda_H) \tag{1.6.12}$$

$F = G \textcircled{z} H$ is a (ND, D_1^2, λ_F)-graph, where

$$\lambda_F \leq 1 - (1-\lambda_H)^2 (1-\lambda_G) \tag{1.6.13}$$

Alternatives

$$\lambda_F \leq 1 - (1-\lambda_H)^2 (1-\lambda_G) \tag{1.6.14}$$

$$\iff (1-\lambda_H)^2 (1-\lambda_G) \leq 1 - \lambda_F \tag{1.6.15}$$

$$\iff Gap(H)^2 \cdot Gap(G) \leq Gap(F) \tag{1.6.16}$$

Proof: We start with writing out a complete form of M_F. Recall from earlier (Prop. 1.17) that

$$M_F = (\mathbb{I}_N \otimes M_H) \cdot P_G \cdot (\mathbb{I}_N \otimes M_H) \tag{1.6.17}$$

By Lemma 1.4, we can derive

$$M_H = \left((1-\lambda_H)\frac{J_D}{D} + \lambda_H E_H\right) \qquad \text{where} \quad \|E_H\| \leq 1 \tag{1.6.18}$$

Then,

$$\mathbb{I}_N \otimes M_H = (1-\lambda_H)\mathbb{I}_N \otimes \frac{J_D}{D} + \lambda_H \mathbb{I}_N \otimes E_H \tag{1.6.19}$$

and thus

$$\begin{aligned}
M_F &= \left((1-\lambda_H)\mathbb{I}_N \otimes \frac{J_D}{D} + \lambda_H \mathbb{I}_N \otimes E_H\right) \times P \times \left((1-\lambda_H)\mathbb{I}_N \otimes \frac{J_D}{D} + \lambda_H \mathbb{I}_N \otimes E_H\right) && (1.6.20)\\
&= (1-\lambda_H)^2 \left(\mathbb{I}_N \otimes \frac{J_D}{D}\right) \cdot P \cdot \left(\mathbb{I}_N \otimes \frac{J_D}{D}\right) && (1.6.21)\\
&\quad + (1-\lambda_H)\lambda_H \underbrace{\left(\mathbb{I}_N \otimes \frac{J_D}{D}\right) \cdot P \cdot (\mathbb{I}_N \otimes E_H)}_{E^{(1)}} && (1.6.22)\\
&\quad + \lambda_H(1-\lambda_H) \underbrace{(\mathbb{I}_N \otimes E_H) \cdot P \cdot \left(\mathbb{I}_N \otimes \frac{J_D}{D}\right)}_{E^{(2)}} && (1.6.23)\\
&\quad + \lambda_H^2 \cdot \underbrace{(\mathbb{I}_N \otimes E_H) \cdot P \cdot (\mathbb{I}_N \otimes E_H)}_{E^{(3)}} && (1.6.24)\\
&= (1-\lambda_H)^2 \left(\mathbb{I}_N \otimes \frac{J_D}{D}\right) P \left(\mathbb{I}_N \otimes \frac{J_D}{D}\right) + \underbrace{(1-\lambda_H)\lambda_H E^{(1)} + \lambda_H(1-\lambda_H)E^{(2)} + \lambda_H^2 E^{(3)}}_{E^{(4)}} && (1.6.25)
\end{aligned}$$

Here, $\|E^{(1)}\|, \|E^{(2)}\|, \|E^{(3)}\| \leq 1$ by applying Lemma 1.5 multiple times. I present the proof for $\|E^{(1)}\| \leq 1$ here.

$$\begin{aligned}
\|E^{(1)}\| &= \left\|\left(\mathbb{I}_N \otimes \frac{J_D}{D}\right) \cdot P \cdot (\mathbb{I}_N \otimes E_H)\right\| && (1.6.26)\\
&\leq \left\|\mathbb{I}_N \otimes \frac{J_D}{D}\right\| \cdot \|P\| \cdot \|\mathbb{I}_N \otimes E_H\| && (1.6.27)\\
&\leq 1 \cdot 1 \cdot 1 && (1.6.28)\\
&\leq 1 && (1.6.29)
\end{aligned}$$

Then,

$$\begin{aligned}
\|E^{(4)}\| &\leq \|(1-\lambda_H)\lambda_H + \lambda_H(1-\lambda_H) + \lambda_H^2\| && (1.6.30)\\
&= (1-\lambda_H)\lambda_H + \lambda_H(1-\lambda_H) + \lambda_H^2 && (1.6.31)\\
&= 2\lambda_H(1-\lambda_H) + \lambda_H^2 && (1.6.32)\\
&= 1 - (1-\lambda_H)^2 && (1.6.33)
\end{aligned}$$

Summary of Above

$$M_F = (\mathbb{I}_N \otimes M_H) \cdot P_G \cdot (\mathbb{I}_N \otimes M_H) \tag{1.6.34}$$

$$= (1-\lambda_H)^2 \underbrace{\left(\mathbb{I}_N \otimes \frac{J_D}{D}\right) \cdot P \cdot \left(\mathbb{I}_N \otimes \frac{J_D}{D}\right)}_{\alpha} + E^{(4)} \tag{1.6.35}$$

where

$$\|E^{(4)}\| \leq 1 - (1-\lambda_H)^2 \tag{1.6.36}$$

We can massage and transform this result with

$$\alpha = \frac{1}{D^2}(\mathbb{I}_N \otimes J_D) \cdot P \cdot (\mathbb{I}_N \otimes J_D) \tag{1.6.37}$$

$$= \frac{1}{D^2} \cdot A_G \otimes J_D \tag{1.6.38}$$

$$= \frac{A_G}{D} \otimes \frac{J_D}{D} \tag{1.6.39}$$

$$= M_G \otimes \frac{J_D}{D} \tag{1.6.40}$$

and plug back into Eq. 1.6.35 to get

$$M_F = (1-\lambda_H)^2 \left(M_G \otimes \frac{J_D}{D}\right) + E^{(4)} \tag{1.6.41}$$

Recall from Lemma 1.4 that

$$M_G = (1-\lambda_G) \cdot \frac{J_N}{N} + \lambda_G \cdot E_G \qquad \text{where} \quad \|E_G\| \leq 1 \tag{1.6.42}$$

plugging this back into Equation 1.6.41, we get

$$M_F = (1-\lambda_H)^2(1-\lambda_G)\left(\frac{J_N \otimes J_D}{ND}\right) + (1-\lambda_H)^2 \cdot \lambda_G \left(E_G \otimes \frac{J_D}{D}\right) + E^{(4)} \tag{1.6.43}$$

$$= (1-\lambda_H)^2(1-\lambda_G)\frac{J_{ND}}{ND} + \underbrace{(1-\lambda_H)^2 \cdot \lambda_G \cdot \left(E_G \otimes \frac{J_D}{D}\right) + E^{(4)}}_{E^{(5)}} \tag{1.6.44}$$

We can quantify $\|E^{(5)}\|$ as

$$\|E^{(5)}\| \leq (1-\lambda_H)^2 \cdot \lambda_G \cdot \left\|E_G \otimes \frac{J_D}{D}\right\| + \|E^{(4)}\| \tag{1.6.45}$$

$$\leq (1-\lambda_H)^2 \cdot \lambda_G \cdot 1 + 1 - (1-\lambda_H)^2 \tag{1.6.46}$$

$$= 1 - (1-\lambda_G) \cdot (1-\lambda_H)^2 \tag{1.6.47}$$

Summary of Above

$$M_F = (1-\mu)\cdot\frac{J_{ND}}{ND} + E^{(5)} \qquad \text{where} \quad \|E^{(5)}\| \le 1-(1-\lambda_G)\cdot(1-\lambda_H)^2 \tag{1.6.48}$$

i.e., the second largest eigenvalue of M_F is at most $1-(1-\lambda_G)\cdot(1-\lambda_H)^2$ (by Lemma 1.4).

This concludes the proof. ■

1.7 Undirected s-t Connectivity in Log Space

1.7.1 The Procedure

Algorithm 1.5: USTCONN Log Space

This is a four step procedure, with input $G=(V,E)$ and two nodes s,t in the graph. We wish to output if s,t are connected with each other.

- **Step 1.** Transform
$$(G,s,t) \rightsquigarrow (G_0,s_0,t_0) \tag{1.7.1}$$
where G_0 is a B^2 regular graph, for some constant B. [a]
- **Step 2.** Fix H, a $(B^4,B,1/4)$ graph.
- **Step 3.** For $k \leftarrow 1,\dots,L$, compute[bc]
$$G_k = G_{k-1}^2 \textcircled{z} H \tag{1.7.2}$$
- **Step 4.** Solve s_L, t_L connectivity on G_L by path enumeration.

[a]For example, we have the power to transform any graph into a 4-regular graph using the algorithm detailed previously.

[b]Invariant: G_k is always a B^2 regular graph.

[c]Remark: $|G_k| = N\cdot B^{4k}$. We than think of vertices in G_k as $\bar{v} \equiv (v,a_1,a_2,\dots,a_k)$ where v is a vertex in G_0 and $a_.$ is a vertex name in H.

1.7.2 Path Enumeration Path Length Analysis

Let λ_k denote the second largest eigenvalue of a connected component of G_k. We know from Thm. 1.2 that

$$\lambda_k \le 1-(1-\lambda_H)^2\cdot(1-\lambda_{k-1}^2) \tag{1.7.3}$$

where we have λ_{k-1}^2 because $G_k = G_{k-1}^2 ⓩ H$. By assumption on H that $\lambda_H \leq 1/4$),

$$\lambda_k \leq 1 - \left(\frac{3}{4}\right)^2 \cdot (1 - \lambda_{k-1}^2) \tag{1.7.4}$$

$$\implies \left(\frac{3}{4}\right)^2 (1 - \lambda_{k-1}^2) \leq 1 - \lambda_k \tag{1.7.5}$$

$$\implies \left(\left(\frac{3}{4}\right) \cdot (1 + \lambda_{k-1})\right)(1 - \lambda_{k-1}) \leq (1 - \lambda_k) \tag{1.7.6}$$

$$\implies (1 - \lambda_k) \geq \min\left(\frac{1}{18}, \frac{35}{32}(1 - \lambda_{k-1})\right) \tag{1.7.7}$$

Proposition 1.18

If $L = \mathcal{O}(\log N_0)$, then we will have

$$1 - \lambda_L \geq \frac{1}{18} \qquad \iff \qquad \lambda_L \leq \frac{17}{18} \tag{1.7.8}$$

Proposition 1.19

Path enumeration on G_L results in paths of length

$$\Delta = \mathcal{O}\left(\log_{\frac{1}{\lambda_L}} N_L\right) = \mathcal{O}\left(\log N_0\right) \tag{1.7.9}$$

Proof: By invariant stated above,

$$N_L = N_0 \cdot B^{4L} \tag{1.7.10}$$

then the path length bound becomes

$$\log_{\frac{1}{\lambda_L}} N_L \leq \log_{\frac{18}{17}} N_L \leq \mathcal{O}(\log N_0) + (L) \tag{1.7.11}$$

1.7.3 Space Complexity

Input We are given the raw input graph G_0, original source and target s_0, t_0. Recall from earlier that we can label vertices in G_L as

$$(v \in G_0, a_1 \in V_H, a_2 \in V_H, \dots, a_L \in V_H) \tag{1.7.12}$$

which is a path that we took from a vertex in G_0 via a series of steps labeled with vertices in H.

Goal Given a sequence of edge labels in G_L, $(e_1, e_2, \dots, e_\Delta)$ we have to check where we end in G_L.

Idea Recall that the original goal is to compute this query space efficiently. In particular, that means we don't have the luxury of computing out the entire G_L graph. We can instead compute the rotation maps

$$Rot_{G_k} : [N_k] \times [B^2] \to [N_k] \times [B^2] \tag{1.7.13}$$

recursively, following $G_k = G_{k-1}^2 \textcircled{z} H$.

Recursive Computation Suppose we are to compute

$$Rot_{G_k}\left((v, a_1, a_2, \dots, a_k), e \in [B^2]\right) \tag{1.7.14}$$

given $Rot_{G_{k-1}}$. We can compute $Rot_{G_{k-1}^2 \textcircled{z} H}$ using the steps outlined in Definition 1.13. Unwinding,

$$Rot_{G_{k-1}^2}(\bar{v}, (f_1, f_2)) = Rot_{G_{k-1}}\left(Rot_{G_{k-1}}(\bar{v}, f_1), f_2\right) \tag{1.7.15}$$

where f_1, f_2 are edge labels in G_{k-1}.

Space Complexity With some careful book keeping, we can implement the recursive procedure outlined above with a total of

$$O(\log N) \tag{1.7.16}$$

extra memory.

1.8 Expander Graphs

Definition 1.15: Spectral Expander

A D-regular graph is a λ-expander if G is a (N, D, λ)-graph.

Example Suppose that we have a $(N = 10^6, D = 10, \lambda \leq 1/2)$-graph. Then, every two nodes in this graph are connected by a path of length $\leq \log D \log_{1/\lambda} N = \log 10 \log_2 10^6 \leq 24$.

Proposition 1.20: Expander Graphs Properties

If $\lambda \leq \frac{1}{2}$, then for any set of nodes S in this expander graph , the number of edges between S and $\bar{S}$ is such that

$$E(S, \bar{S}) \geq \Omega(1) \cdot |S| \tag{1.8.1}$$

Definition 1.16: Expander Graph Intuitively

Expander Graphs = extremely well-connected graphs with few edges.

1.8.1 History of Expander Graphs

The guiding question can be summarized as follows: (with example D and λ values

> Given N, degree $D = 4$ and $\lambda = 0.9$. Can we find a (M, D, λ)-graph where $M \in (N, 20)$.

...

Chapter 2

Sensitivity Conjecture

2.1 Classic Combinatorial Measures

Consider a family of functions with signature

$$f : \{0,1\} \equiv \{1,-1\}^n \to \{0,1\} \tag{2.1.1}$$

the question is then how can we measure the complexity of f.

2.1.1 Decision-Tree Complexity

Definition 2.1: Decision Tree Complexity

$DT(f)$ is the min-depth of a decision tree that computes f.

As two easy examples,

$$DT(\vee) = n \qquad \text{and} \qquad DT(\wedge) = n \tag{2.1.2}$$

for $f = \cdot(x_1, x_2, \dots, x_n)$. Examples here are such that they can be of linear depth or logarithmic. Regardless, these trees depend on all input bits for computation.

2.1.2 Certificate Complexity

Definition 2.2: Certificate Complexity

For a specific input

$$CC(f,x) = \min\{|S| : S \text{ is a certificate of } x\} \tag{2.1.3}$$

where we say $S \subseteq [n]$ is a certificate for f at x if all inputs that agree with x on S have same f value. Then, for functions, we take

$$CC(f) = \max_x CC(f,x) \tag{2.1.4}$$

Example Consider $f = \wedge$. Then $CC(\wedge, x) = 1$ except for when $x = (1, \dots, 1)$ where

$$CC(\wedge, (1, \dots, 1)) = n \tag{2.1.5}$$

Thus,

$$CC(\wedge) = n \tag{2.1.6}$$

Proposition 2.1

$$CC(f) \leq DT(f) \tag{2.1.7}$$

Example For the majority operation, $MAT_3 : \{0,1\}^3 \to \{0,1\}$. Then, $DT(MAT_3) = 3$ and $CC(MAT_3) = 2$.

Proposition 2.2

$$DT(f) \leq CC(f)^2 \tag{2.1.8}$$

Definition 2.3: Sensitivity

$S(f,x) = \#$ of neighbors y of x with $f(y) \neq f(x)$. Where 'neighbors' mean that y and x differs in exactly one bit. Then, as expected

$$S(f) = \max_x S(f,x) \tag{2.1.9}$$

Definition 2.4: Hyper Cube

A hyper cube H_n has (bit strings) vertices $\{0,1\}^n$. Two vertices are adjacent if they differ in exactly one coordinate. A hypercube is a regular graph with degree n.

Definition 2.5: Sensitivity with Hyper Cube

With help of hyper cube, sensitivity = max over all vertices x, # neighbors of opposite color.

Proposition 2.3: Function as Polynomial

For any function $f(x_1, \dots, x_n) : \{0,1\}^n \to \{0,1\}$, there exists a equivalent representation

$$f(x_1, \dots, x_n) \sum_{I \subseteq [n]} C_I \prod_{i \in I} x_i \tag{2.1.10}$$

Definition 2.6: Degree

$$degree(f) : f : \{0,1\}^n \to \{0,1\} \tag{2.1.11}$$

is equal to the degree of the polynomial representing f, i.e.

$$\max\{|I| : C_I \neq 0\} \tag{2.1.12}$$

Examples

- $\wedge_n$, $degree(\wedge_n) = n$ since $\wedge_n = x_1 \cdot x_2 \dots x_n$
- $\vee_n$, $degree(\vee_n) = n$ since

$$\begin{aligned} \vee_n &= \neg(\wedge(\neg x_1, \dots, \neg x_n)) && (2.1.13)\\ &= 1 - (1 - x_1)(1 - x_2)\dots(1 - x_n) && (2.1.14) \end{aligned}$$

Proposition 2.4

Degree and decision tree depth for a function $f : \{0,1\}^n \to \{0,1\}$ satisfies

$$degree(f) \le DT(f) \tag{2.1.15}$$

Proof: We can write the function as a decision tree, then

$$f(x) = \sum_{\ell \in L} (\text{if } x \text{ leads to } \ell) \cdot (\ell.\text{output}) \tag{2.1.16}$$

where L is the set of all leaves in the decision tree. Then,

$$\begin{aligned} f(x) &= \sum_{\ell \in L} (x_{i1} == a_1)(x_{i2} == a_2)\dots(x_{id} == a_d) \cdot (\ell.\text{output}) && (2.1.17)\\ &= \sum_{\ell \in L} (a_1 x_{i1} + (1 - a_1)(1 - x_{i1}))\dots(dx_{id} + (1 - a_d)(1 - x_{id})) && (2.1.18)\\ &\implies degree(f) \le DT(f) && (2.1.19) \end{aligned}$$

Proposition 2.5: Summary

$$\left.\begin{matrix} S(f) \le CC(f) \\ degree(f) \end{matrix}\right\} \le DT(f) \le CC(f)^2 \tag{2.1.20}$$

Note that here we don't know how $degree(f)$ relates to things on the left. This is called sensitivity conjecture.

Theorem 2.1

$degree(f), DT(f), CC(f), RDT(f), QDT(f)$ are all within polynomial factors of each other.

$$\begin{aligned} DT(f) &\leq CC(f)^2 \\ degree(f) &\leq CC(f)^{C_1} \\ DT(f) &\leq degree(f)^{C_2} \end{aligned} \tag{2.1.21}$$

2.2 Sensitivity Conjecture

Proposition 2.6: Sensitivity Conjecture (Nisan-Szegedy, 1989)

$$S(f) \geq degree(f)^{C_1}, \quad C_1 \in \mathbb{R} \tag{2.2.1}$$

or equivalently,

$$S(f) \geq DT(f)^{C_2}, \quad C_2 \in \mathbb{R} \tag{2.2.2}$$

Theorem 2.2: Hao Huang, 2019

$$degree(f) \leq S(f)^2 \tag{2.2.3}$$

Theorem 2.3: Graph Conjecture Equivalence (Gotsman-Linial, 1992)

Statement If for every subgraph of H of H_n of $2^{n-1}+1$ vertices,

$$\Delta(H) \geq g(n) \qquad \Longrightarrow \qquad \forall f, S(f) \geq g(degree(f)) \tag{2.2.4}$$

where $\Delta(H)$ for a graph H is equal to the max degree of any vertex in H. [a]

Equivalence Sensitivity Conjecture (1989) is equivalent to "Graph Conjecture" proposed by Gotsman-Linial (1992). Huang proved this graph conjecture in 2019.

[a] We will use this meaning of Δ through out this chapter.

Proof: [TODO] will be added later

Theorem 2.4: Hao Huang, 2019

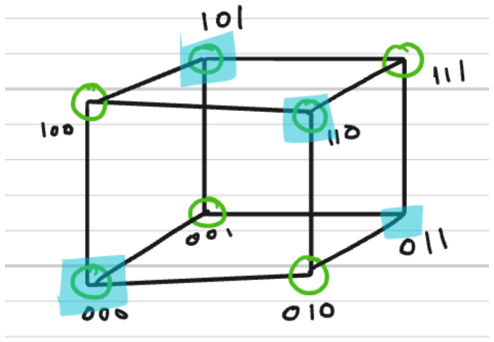

Figure 2.1: N3 Parity Sub-graph.

For any subgraph H of H_n of $\geq 2^{n-1}+1$ vertices has

$$\Delta(H) \geq \sqrt{n} \tag{2.2.5}$$

The extra "+1" part in the formulation plays a significant role. As an example, consider $H =$ the subgraph with vertices

$$\{x \mid parity(x) = 0\} \tag{2.2.6}$$

If $|H| = 2^{n-1}$, i.e. exactly half of the graph, then in this case we can construct $\Delta(H) = 0$. See Figure 2.1 for an example.

Proposition 2.7: Observation 1

Take any graph G, let A_G be its adjacency matrix. Then, [a]

$$\Delta(G) \geq |\lambda_1(A_G)| \tag{2.2.7}$$

Recall that if A_G was regular, we showed that largest eigenvalue = degree of the graph.

[a] Here $\lambda_1(G)$ = largest eigenvalue in magnitude.

Proof: Suppose $A_G\mathbf{v} = \lambda_i\mathbf{v}$. Let v_{i^*} be the largest entry of $\mathbf{v}$ in absolute value.

$$|\lambda_i||v_{i^*}| = \left|\sum_{j=1}^{n} A_{i^*j} \cdot v_j\right| \tag{2.2.8}$$

$$\leq \sum_{j=1}^{n} |A_{i^*j}| \cdot |v_j| \tag{2.2.9}$$

$$\leq \sum_{j:A_{i^*j}\neq 0} |v_j| \tag{2.2.10}$$

$$\leq \sum_{j:A_{i^*j}\neq 0} |v_{i^*}| \tag{2.2.11}$$

$$= degree(i^*) \cdot |v_{i^*}| \tag{2.2.12}$$

i.e., $|\lambda_1| \leq degree(i^*)$ as wanted. ■

Definition 2.7: Signed-Adjacency Matrix

A matrix "B" is called a ***signing*** of a graph G if

$$B_{ij} = (i,j) \text{ edge } ? \in \{0,1\} : 0 \tag{2.2.13}$$

Proposition 2.8: Observation 2

Take any graph G, let B be a signed adjacency matrix of G. Then,

$$\Delta(G) \geq |\lambda_1(B)| \tag{2.2.14}$$

Theorem 2.5: Cauchy-Interlacing Theorem

For a symmetric matrix $M \in \mathbb{R}^{n\times n}$, with eigenvalues

$$\lambda_1(M) \geq \lambda_2(M) \geq \lambda_3(M) \geq \cdots \geq \lambda_N(M) \tag{2.2.15}$$

its sub-matrix M_{-1} is a $(n-1)\times(n-1)$ matrix, obtained by deleting i-th row and i-th column, with interlacing eigenvalues

$$\lambda_i(M) \geq \lambda_i(M_{-1}) \geq \lambda_{i+1}(M) \tag{2.2.16}$$

Corollary 2.5.1: General Form, Repeating Interlacing

Let A be a symmetric matrix in $\mathbb{R}^{n\times n}$, and $B \in \mathbb{R}^{M\times M}$ be a principal sub-matrix, with

$$\lambda_N(A) \leq \lambda_{N-1}(A) \leq \cdots \leq \lambda_1(A) \tag{2.2.17}$$

and

$$\lambda_M(B) \leq \lambda_{M-1}(B) \leq \cdots \leq \lambda_1(B) \tag{2.2.18}$$

Then,

$$\lambda_{N-M+i}(A) \leq \lambda_i(B) \leq_i (A) \tag{2.2.19}$$

Theorem 2.6: Hao Huang, 2019

For any subgraph H of H_n of $\geq 2^{n-1}+1$ vertices, it has

$$max - degree(H) \geq \sqrt{n} \tag{2.2.20}$$

Proof: We know that $max - degree(H) \geq$ largest eigenvalue of B where B is the principal matrix corresponding to vertices in the subgraph of B_n. Then, by Cauchy Interlacing Theorem, we have

$$\lambda_1(B) \geq \lambda_{N-M+1}(B_n) \geq \lambda_{2^{n-1}}(B_n) \geq \sqrt{n} \tag{2.2.21}$$

This completes the proof. ■

Proposition 2.9

$\forall n, \exists$ a signing B_n of H_n such that

$$B_n^2 = n \cdot \mathbb{I}_{2^n} \tag{2.2.22}$$

Proposition 2.10

If B_n is as above, then it has 2^{n-1} eigenvalues that are $\sqrt{n}$ and the other 2^{n-1} eigenvalues take value $-\sqrt{n}$.

Proof: We know that $B_n^2 = n \cdot \mathbb{I}_{2^n}$, so every eigenvalue of B_n satisfies $\lambda^2 = n$. This implies

$$\lambda \in \{\sqrt{n}, -\sqrt{n}\} \tag{2.2.23}$$

Recall that the trace of a symmetric matrix is the sum its eigenvalues. Then here,

$$Tr(B_n) = 0 \tag{2.2.24}$$

since B_n must have zero diagonal entries. Hence it must be the case that out of 2^n total eigenvalues, half of them are $-\sqrt{n}$ while the other half have value $\sqrt{n}$. ■

Chapter 3

Sunflower Conjecture

3.1 Sunflower Lemma

Definition 3.1: Sunflowers (Erdös-Rado, 1960's)

Take $f = \{s_1, \dots, s_m\}$ is a sunflower if all pairwise intersections are the same. i.e.,

$$S_i \cap S_j = S_k \cap S_e \qquad \text{if} \quad i \neq j, k \neq e. \tag{3.1.1}$$

Equivalently, $S_1, \dots, S_m$ is a sunflower if the "core"

$$C = S_1 \cap S_2 \cap \dots \cap S_m \tag{3.1.2}$$

are such that

$$S_1 \backslash C, S_2 \backslash C, \dots, S_m \backslash C \tag{3.1.3}$$

are disjoint

Lemma 3.1: Sunflower Lemma (Erdös-Rado, 1962)

Take $f = \{S_1, \dots, S_m\} \subseteq [n]$, [a] where $|S_i| \leq k, \forall i \in [m]$. Then S contains a r-sunflower if

$$m \geq (r-1)^k \cdot k! \tag{3.1.4}$$

English Explanation This essentially says that any large collection of sets (large enough) must contain a sunflower. Interestingly enough, this expression does not depend on the size of universe, n.

[a] This is a bit of abuse of notation. We really meant $S_i \subseteq [n], \forall i$ and f is a set of such subsets.

Proof: The proof is an induction on k.

Base Case $k = 1$. Then S_i, S_j are disjoint and $m > (r-1) \implies m \geq r$. So all of f is a sunflower with empty core.

Inductive Step There are two cases to consider. Suppose the family f had $\geq r$ disjoint sets, then we are done. Otherwise, we choose some maximal collection of disjoint sets

$$S_{i1}, S_{i2}, ..., S_{il} \qquad \text{where} \quad l \leq r-1 \tag{3.1.5}$$

This means that we cannot add another set that is disjoint from these. In particular, every other set must intersect with one of these sets. Now, call

$$A \triangleq S_{i1} \cup S_{i2} \cup \cdots \cup S_{il} \tag{3.1.6}$$

then the total number of elements in A is bounded by

$$\begin{aligned} |A| &\leq k \cdot l && (3.1.7)\\ &\leq k \cdot (r-1) && (3.1.8)\end{aligned}$$

Now, every set S_i must intersect A. Then, some element $a \in A$ must occur in $\geq \alpha = m/|A|$ many sets.

$$\begin{aligned} \alpha &\geq \frac{m}{k \cdot (r-1)} && (3.1.9)\\ &\geq \frac{(r-1)^k \cdot k!}{k \cdot (r-1)} && (3.1.10)\\ &= (r-1)^{k-1} \cdot (k-1)! && (3.1.11)\\ \implies m &\geq (r-1)^k \cdot k! && (3.1.12)\end{aligned}$$

This concludes the proof. ■

Proposition 3.1: Erdös-Rado Conjecture

For every r, there is some constant c_r, such that if $m > c_r^k$, then it contains a r-sunflower.

Theorem 3.1: ALWZ, 2020

There exists a constant C such that if $f = \{S_1, \dots, S_m\} \subseteq [n]$ and $|S_i| \leq k$, and if $m > (c \cdot r \cdot \log k)^k$, then f contains a r-sunflower.

Definition 3.2: Link of a Set System

Take $f = \{S_1, \dots, S_m\} \subseteq [n]$. Suppose $I \subseteq [n]$, then define the link of I in f as

$$f_i = \{S_i : S_i \supseteq I\} \qquad \overline{f_I} = \{S_i \quad I : S_i \supseteq I\} \tag{3.1.13}$$

Definition 3.3: Spread

A set system $f = \{S_1, S_2, \dots, S_m\} \subseteq [n]$, where each $|S_i| \leq k$, is s-spread if

- $|f| \geq s^k$, and
- $\forall I \subseteq [n], |f_I| \leq s^{k-|I|} \cdot |f|$

Lemma 3.2: Main Lemma, ALWZ20

If f is s-spread for

$$s = c \cdot \log(rk) \log\log(rk) \tag{3.1.14}$$

then f contains r disjoint sets.

Proposition 3.2

Main Lemma implies new sunflower lemma bounds.

Proof: Suppose we have

$$|f| \geq C(r,k)^k = s^k = (c \cdot \log(rk) \log\log(rk))^k \tag{3.1.15}$$

If there exists a I such that $|f_I| > s^{-|I|}$, then we are done. We see

$$\begin{aligned} |f_I| &\geq s^{-|I|}(c \cdot \log(rk) \log\log(rk))^k && (3.1.16)\\ &= s^{k-|I|} && (3.1.17)\\ &\geq (c \cdot \log(r(k-|I|)) \cdot \log\log(r \cdot (k-|I|)))^{k-|I|} && (3.1.18) \end{aligned}$$

so we rely on induction for this proof. If $\forall I$, $|f_I| < s^{-|I|} \cdot |f|$. Then, f is s-spread ad by main lemma f contains r disjoint sets. ■

3.2 Robust Sunflower Lemma

Lemma 3.3: Robust Sunflower Lemma (RSL)

Consider f as $s-spread$, and w is a random subset of $[n]$ of size $\lfloor \frac{n}{r} \rfloor$. Then,

$$Pr_w[\exists j, S_j \subseteq W] \geq 1 - \frac{1}{2r} \tag{3.2.1}$$

It is natural to wonder why we need this different form of the Sunflower Conjecture. In fact, this more robust RSL implies the Main lemma.

The idea to the proof of RSL lies in the fact that we have flexibility in choosing W (of size n/r, assuming divisibility), and we don't have to choose it in one shot. We choose W as a sequence of sets

$$W = V_1 \cup V_2 \cup \cdots \cup V_t \tag{3.2.2}$$

where each V_i is a random subset of what's left of size $\frac{n}{rt}$.

TODO add lecture 9 from min 57.

Proposition 3.3: First Refined RSL

If f is s-spread and V is a random subset of $\lfloor \frac{n}{rt} \rfloor$. Then,

$$\mathbb{E}_{j\in[m],V}[|\chi(j,V)|] \leq k \cdot \left(1 - \frac{1}{\log s}\right) \tag{3.2.3}$$

Proposition 3.4: Second Refined RSL

If f is s-spread, U is a set and V is a random subset of size $\lfloor \frac{n}{rt} \rfloor$. Then,

$$\mathbb{E}_{j\in[m],V}[|\chi(j,U\cup V)|] \leq \mathbb{E}_{j,U}[|\chi(j,U)|] \cdot \left(1 - \frac{1}{\log s}\right) \tag{3.2.4}$$

TODO add lecture 9 from min 1:10

Proposition 3.5

RSL $\implies$ MSL.

This stronger form of the conjecture comes in handy when we are doing induction, we can have a more powerful hypothesis in the induction step.

Proof: Intuitively, consider the system to be a "spread" one, as then we can sample random patches W's of the total space $[n]$, each of size $\lfloor \frac{n}{r} \rfloor$ elements. We expect that a patch covers some set S_i with a reasonable chance ($(1 - \frac{1}{2r})$ stated in theorem). If this is the case, then if we take r patches, we immediately know

$$\text{RSL} \implies \begin{cases} Pr[\exists j, S_j \subseteq W_1] = Pr[E_1] \geq 1 - \frac{1}{2r} \\ Pr[\exists j, S_j \subseteq W_2] = Pr[E_2] \geq 1 - \frac{1}{2r} \\ \quad \vdots \\ Pr[\exists j, S_j \subseteq W_r] = Pr[E_r] \geq 1 - \frac{1}{2r} \end{cases} \tag{3.2.5}$$

Then, we can take the union bound of events to bound Pr[everything above happens at the same time] as

$$\begin{aligned} Pr[E_1 \cup E_2 \cup \cdots \cup E_r] &\leq Pr[E_1] + Pr_[E_2] + \cdots + Pr[E_r] && (3.2.6) \\ &= r \cdot \frac{1}{2r} && (3.2.7) \\ &= \frac{1}{2} && (3.2.8) \\ &> 0 && (3.2.9) \end{aligned}$$

So there $\exists$ r disjoint sets. ■

Chapter 4

Information Complexity and Applications

4.1 Communication Complexity

Consider the setup where there are two parties Alice holding some information x and Bob holding some other information y. The goal is to compute a function $f(x, y)$ that possibly depends on information from both parties. How we exchange these information for the sake of computing f is called a protocol. The question now is "what is the best protocol in terms of number of bits exchanged"? Few ways exist

Definition 4.1: Deterministic Protocol

Denoted as *Det*, where we compute $f(x, y)$ exactly.

Definition 4.2: Randomized Protocol

Denoted as *Rand*, where we want to output the correct answer with probability $\geq$, say, $\frac{9}{10}$. Two types of randomized protocols exist, named in terms of if the random bits are shared or kept private. [a]

[a] A good example to think about is fixing random seed for a pseudorandom number generator makes the random bits public.

Definition 4.3: Protocol Complexities

- $Det(f)$ = minimum number of bits needed to compute f, exactly
- $R_{0-1}^{private}(f)$ = minimum number of bits needed to compute f with probability ≥ 0.9 and private random bits.

- $R_{0-1}^{public}(f)$ = minimum number of bits needed to compute f with probability ≥ 0.9 and public random bits.

Proposition 4.1: Protocol Complexities' Relationship

$$Det(f) \geq R_{0-1}^{private}(f) \geq R_{0-1}^{public}(f) \tag{4.1.1}$$

As an example, consider the equality test, where $EQ : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ where $EQ(x, y) = 1$ if $x = y$ and 0 otherwise. We have

- $Det(EQ) \leq n$; further, we claim that $Det(EQ) = n$.
- $R_{0-1}^{public}(EQ) \leq 5 \leq \mathcal{O}(1)$, and
- $R_{0-1}^{private}(EQ) \leq \mathcal{O}(\log n) + \mathcal{O}(1)$

Proposition 4.2

$$R_{0-1}^{private}(EQ) \leq c \cdot \log n \tag{4.1.2}$$

Theorem 4.1: Newman, 91

$$R_{0-1}^{private}(f) \leq R_{0-1}^{public}(f) + \mathcal{O}(\log n) \tag{4.1.3}$$

and

$$Det(f) \leq 2^{\mathcal{O}\left(R_{0-1}^{private}(f)\right)} \tag{4.1.4}$$

4.2 Applications of Communication Complexity

4.2.1 NOF & NIH Models

Number on Forehead (NOF) and Number in Hand (NIH) models were proposed by Chandra, Furst, Lipton in 83. In NIH, each party (say four parties A, B, C, D) holds onto their own piece of information and they communicate and compute $f(x_1, \dots, x_4)$ which is exactly our original model. In NOF, however, each party have access to all the rest information except for their own piece. In either case, communication is defined using a "Blackboard" model where each party can come to the board and write down information to communicate. At the end of the day, communication cost is the total number of bits written on the board.

Let's now take a look at $f = ZERO$ where $ZERO(x_1, \dots x_m) = 1$ if $\sum_{i=1}^{m} x_i = 0$ and 0 otherwise.

Proposition 4.3

$$Det_{NOF}(ZERO) \leq \log N + O(1) \tag{4.2.1}$$

Theorem 4.2: Tighter Version (CFL, 83)

$$Det_{NOF}(ZERO) \leq O\left(\sqrt{\log N}\right) \tag{4.2.2}$$

4.2.2 AP-Free Coloring

Definition 4.4: Ap-Free Coloring

Coloring $\{-N, -N+1, \ldots, N-1, N\}$ with p colors such that there is no monochromatic 3-term arithmetic progression. i.e., no a, b, c are of the same color such that $b - a = c - b$.

4.3 Lower Bounds on Communication

Consider the disjunction operation, defined as

Definition 4.5: Two Party Disjunction

$DIST_n : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ where $DIST_n(x, y)$ is equal to 1 if at all indices i, $x_i \wedge y_i = 0$; and zero otherwise.

Theorem 4.3

$$Det(DIST_n) = n \qquad R_{0-1}(DIST_n) \geq \Omega(n) \tag{4.3.1}$$

4.4 Shannon's Information Theory

We start with a measure for randomness.

Definition 4.6: Entropy of RV & Conditional Entropy

The entropy of a random variable measures how chaotic it is.

$$H(X) = \sum_{x \in Supp(X)} Pr[X = x] \cdot \log_2\left(\frac{1}{Pr[X = x]}\right) \tag{4.4.1}$$

We can also measure entropy for a conditional random variable. In this case

$$H(X|Y) = \mathbb{E}_{y \leftarrow Y}[H(X|Y = y)] = \sum_{y \in Supp(Y)} Pr[Y = y] \cdot H(X|Y = y) \tag{4.4.2}$$

Proposition 4.4: Properties of Entropy

- $H(X,Y) = H(Y) + H(X|Y) \leq H(Y) + H(X)$
- $H(X|Y) \leq H(X)$

Definition 4.7: Mutual Information

Mutual Information quantifies the mutual dependence between two random variables, and is defined as

$$I(X;Y) = H(X) + H(Y) - H(X,Y) = H(X) - H(X|Y) \tag{4.4.3}$$

Proposition 4.5: Independent Mutual Information

If two random variables X, Y are independent, then

$$I(X;Y) = 0 \tag{4.4.4}$$

where knowing information about one tells you zero additional information.

Definition 4.8: Conditional Mutual Information

$$I(X;Y|Z) = \mathbb{E}_{Z=\delta}[I(X|_{Z=\delta}; Y|_{Z=\delta})] \tag{4.4.5}$$

Clearly, the conditional mutual information is equal to zero for independent random variables, i.e. For

$$X,Y,Z \sim Bin(0,1) \qquad I(X;Y) = 0 \quad I(X;Y|Z) = 0 \tag{4.4.6}$$

It is more interesting to look at the case where some variables are dependent on each other. Consider the case where $X, Y \sim Bin(0,1)$ with $Z \in \{0,1\}$ such that $X \oplus Y \oplus Z = 0$.[4.4.1] This means

$$(x,y,z) \in \begin{cases} (0,0,0) \\ (1,1,0) \\ (1,0,1) \\ (0,1,1) \end{cases} \tag{4.4.7}$$

[4.4.1] $\oplus$ means XOR.

where if we discard the z position, the distribution is uniform random across a two dimensional binary space. In this case

$$I(X;Y) = I(X;Z) = I(Y;Z) = 0 \tag{4.4.8}$$

while

$$I(X;Y|Z) = 1 \tag{4.4.9}$$

Definition 4.9: Entropy Chain Rule

$$H(X_1, \dots, X_n) = \sum_{j=1}^{n} H(X_j | X_1, \dots, X_{j-1}) \tag{4.4.10}$$

where each X_j in the expansion is conditioned on everything that comes before it.

Definition 4.10: Mutual Information - Vector RVs and RV

Consider random vector $X = (X_1, \dots, X_n)$. We can measure the mutual information between a random vector and a random variable.

$$I(X;Z) = I(X_1, \dots, X_n; Z) = \sum_{j=1}^{n} I(X_j; Z | X_1, \dots, X_{j-1}) \tag{4.4.11}$$

Note that this formulation does not tell us about the mutual information between the components of X. Rather, it only says about the relationship between X and Y.

Proposition 4.6: Sub-additivity of Entropy

$$H(X_1, \dots, X_n) \leq H(X_1) + H(X_2) + \dots + H(X_n) \tag{4.4.12}$$

4.5 Information Complexity

Definition 4.11: Distributional Communication Complexity

Consider some randomized protocol π such that

$$\forall X, Y \qquad Pr_{\sim\pi}[\pi(X,Y) = DISJ_n(X,Y)] \geq 0.9 \tag{4.5.1}$$

which means we have some protocol that can answer $DISJ_n(X,Y)$ correctly 90% of the time then ($\implies$) any distribution μ on $X \times Y$ has such that

$$Pr_{\sim\pi;(x,y)\eqsim\mu}[\pi(X,Y) = DISJ_n(X,Y)] \geq 0.9 \tag{4.5.2}$$

which means we will be able to answer $DISJ_n(X,Y)$ correctly 90% of the time for ***any random input***.

English For every question that I throw at you, if you have 90% chance answering it correctly, then that means if I ask a random question you will succeed 90% of the time. The implied is a weaker statement but it suffices for our purpose.

Distributional Communicational Complexity

$$R_{0-1,\mu}(DISJ_n) = \text{minimum \# of bits needed to achieve above guarantee.} \tag{4.5.3}$$

Examples Consider this μ. X is a random string where last $n/2$ bits are zero. Y is a random string where the first $n/2$ bits are zero. Then,

$$R_{0-1,\mu}(DISJ_n) = 1 \tag{4.5.4}$$

because they are always disjoint, and they can always output 'yes disjoint' and they will be correct.

As a second example, consider μ such that X and Y are independent completely random strings $\in \{0,1\}^n$. In this case, it is extremely likely that the two random strings have some overlap. Then, $DISJ_n$ becomes very easy to answer, as both parties only have to answer 'not disjoint'. Thus

$$R_{0-1,\mu}(DISJ_n) = 1 \tag{4.5.5}$$

Note that

$$Pr[DISJ_n(X,Y) = 1] = \left(\frac{3}{4}\right)^n \tag{4.5.6}$$

but for reasonably large n this converges to zero.

Definition 4.12: Information Cost of a Protocol

Consider $x, y \leftarrow \mu$.

$$IC(\pi,\mu) = I(X;\pi|Y) + I(Y;\pi|X) \tag{4.5.7}$$

which is the total amount of **new** information that both parties learnt from the protocol. To make sure that the quantification is on new information learnt, we condition out each parties own information.

Proposition 4.7

$$IC(\pi,\mu) \leq |\pi| \tag{4.5.8}$$

which says that the information that both parties learnt is at most the length of the protocol.

Proof:

Intuitive

$$I(X;\pi|Y) \leq H(\pi) \tag{4.5.9}$$

and

$$I(Y;\pi|X) \leq H(\pi) \tag{4.5.10}$$

then

$$IC(\pi,\mu) = I(X;\pi|Y) + I(Y;\pi|X) \leq 2H(\pi) \leq 2 \cdot |\pi| \tag{4.5.11}$$

but since in each individual round only one party can learn new information (the speaker does not learn new information), with some careful book keeping and chain rule, we can derive $IC(\pi,\mu) \leq |\pi|$.

Proof Sketch

$$\begin{aligned}
IC(\pi,\mu) &= I(X;\pi|Y) + I(Y;\pi|X) && (4.5.12)\\
&= O(X;\pi_1,\dots,\pi_r|Y) + I(Y;\pi_1,\dots,\pi_r|X) && (4.5.13)\\
&= \sum_{j=1}^{r}\left[I(X;\pi_j|Y,\pi_,\dots,\pi_{j-1}) + I(Y;\pi_j|Y,\pi_,\dots,\pi_{j-1})\right] && (4.5.14)\\
&\leq \sum_{j=1}^{r} H(\pi_j|\pi_1,\dots,\pi_{j-1}) && (4.5.15)\\
&= H(\pi) && (4.5.16)
\end{aligned}$$

where again we utilize the fact that in each round only one person learns new information.

Definition 4.13: Information Cost of Function

... takes a max-min formulation

$$IC_{0-1}(f) = \max_{\text{all dist } \mu} IC_\mu(f) \tag{4.5.17}$$

where (consider P to be the set of protocols that can compute f with a accuracy of 0.9)

$$IC_\mu(f) = \min_{\pi\in P} IC(\pi,\mu) \tag{4.5.18}$$

To summarize,

$$IC(f) = \max_{\mu}\min_{\pi} IC(\pi,\mu) \tag{4.5.19}$$

Proposition 4.8

$$IC_{0-1}(f) \leq R_{0-1}(f) \tag{4.5.20}$$

Definition 4.14: Functions of Multi-variate Input/Output

We had $f : A \times B \to \{0,1\}$. We now define

$$f^{\otimes n} : A^n \times B^n \to \{0,1\}^n \tag{4.5.21}$$

where $f^{\otimes n}((a_1, \dot{,} a_n), (b_1, \dots, b_n)) =$ compute all answers.

Theorem 4.4

$$IC(f^{\otimes n}) \geq n \cdot IC(f) \tag{4.5.22}$$

and with equality at limit

$$\lim_{n\to\infty} \frac{IC(f^{\otimes n})}{n} = IC(f) \tag{4.5.23}$$

This theorem intuitively says that you cannot make things up: there is no savings for multivariate f, you still have to compute everything.

Theorem 4.5: BBCR10

$$n \cdot R_{0-1}(f) \leq \frac{1}{\sqrt{n}} R_{0-1}(f^{\otimes n}) \tag{4.5.24}$$

4.6 Proof of Thm.: Lower Bounds on Communication

We now take a look at the big picture. Recall that we wanted to show $R_{0-1}(DISJ_n) \geq \Omega(n)$ and we know the following

- $R_{0-1}(DISJ_n) \geq IC(DISJ_n)$
- $(\dagger)$ $IC_\mu(DISJ_n) \geq n \cdot IC_\mu(NAND)$ [4.6.1]
- $(\ddagger)$ $IC_\mu(NAND) \geq 0.01$

We first show $(\dagger)$ for a specific distribution, called Razborov's Distribution

Definition 4.15: Razborov's Distirbution

A tuple shaped random variable (X, Y) such that $(x, y) \sim (X, Y)$ has the following proba-

[4.6.1] Notice that $DISJ_n(X,Y) = \bigwedge_{i=1}^n NAND(x_i, y_i)$, which is why we introduce $NAND$ function in this chain.

bility density function

$$pdf_{Razborov}(x,y) = \begin{cases} 1/3 & (x,y) = (0,0) \\ 1/3 & (x,y) = (0,1) \\ 1/3 & (x,y) = (1,0) \\ 0 & (x,y) = (1,1) \end{cases} \tag{4.6.1}$$

Notice that under this definition, each person's marginal distribution is a bit biased.

Theorem 4.6: An Upper-bound, Piece 1

If there exists a protocol π for $DISJ_n$ that is correct on all x, y with probability 0.9, then there exists a protocol π' for $NAND$ that is correct on all x, y with probability 0.9, and

$$IC(\pi', \mu) \leq \frac{1}{n} \cdot IC(\pi, \mu^n) \tag{4.6.2}$$

Theorem 4.7: Piece 2

If π' is a protocol that is correct on all x, y with prob 0.9, then

$$IC(\pi', \mu) \geq 0.01 \tag{4.6.3}$$

Definition 4.16: Information Cost of Function

Consider function $f : X \times Y \to \{0,1\}$, and some distribution μ over $X \times Y$. Define

$$IC_{\mu,\varepsilon} = \inf_{\text{protocols } \pi \text{ that compute } f \text{ with error } \leq \varepsilon} IC(\pi, \mu) \tag{4.6.4}$$

To be more specific, the protocol π here is such that

$$\forall (x,y), \quad Pr[\pi(x,y) = f(x,y)] \geq 1 - \varepsilon \tag{4.6.5}$$

Finally define information cost of a function as

$$IC_{\varepsilon}(f) = \max_{\mu} IC_{\mu,\varepsilon}(f) \tag{4.6.6}$$

We will use a handy notation for distributions across n composite functions $f^{\otimes n}$. For σ, a distribution over $\{0,1\}$, we consider $\mu \equiv \sigma^n$ as a distribution over $\{0,1\}^n \times \{0,1\}^n$.

Proposition 4.9

- $IC_{\sigma,\varepsilon}(NAND) \leq \frac{1}{n} \cdot IC_{\sigma^n,\varepsilon}(DISJ_n)$

- $\Omega_\varepsilon(1) = IC_{\sigma,\varepsilon}(NAND)$

Chapter 5

Extension Complexity

5.1 LPs & Its Two Views

Extension Complexity pertains how much power lies within linear programs. First, we state the canonical form of Linear Programs (LP).

Definition 5.1: Linear Program

A linear program take the following vectorized canonical form

$$\begin{aligned} \text{maximize} \quad & \mathbf{c}^\top \mathbf{x} && (5.1.1) \\ \text{subject to} \quad & A\mathbf{x} \leq \mathbf{b} && (5.1.2) \end{aligned}$$

We here illustrate the two views of Linear Programs. First, we consider the dimension to optimize over is $n = 2$. Then, if we unwrap the vectorization we have the constraints

$$\begin{bmatrix} a_1^1 x_1 + a_2^1 x_2 \\ \vdots \\ a_1^m x_1 + a_2^m x_2 \end{bmatrix} \leq \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix} \tag{5.1.3}$$

Now, if we plot the lines corresponding to the linear constraints, we will get something like Figure 5.1. Considering the fact that we are optimizing over a linear objective, the maximizer must appear on the vertices of the formed polytope. We call the region bounded in between all the constraints the feasible region of the Linear Program and each edge segments as "facets". Two views arise in this formulation.

Polyhedron View says that the feasible region is the ***intersection*** of the linear inequalities. Notice that we are strictly limiting ourselves to the intersections here, no area included.

Polytope View says that the feasible region can be stated as

$$Convex - Hull\{v_1, \ldots v_6\} \tag{5.1.4}$$

where we define

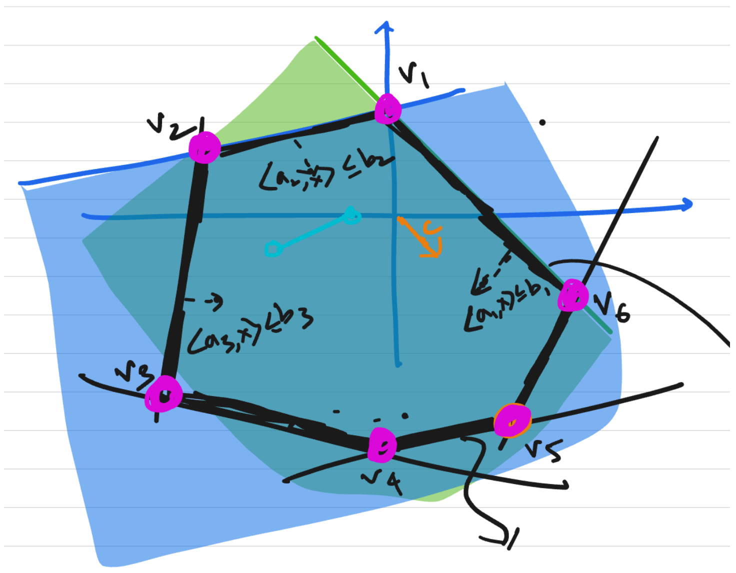

Figure 5.1: Plotted LP constraints

Definition 5.2: Convex Hull

Given $S = \{v_1, \ldots, v_N\}$, then

$$Convex-Hull(S) = \left\{ x : x = \sum_i \lambda_i v_i, \qquad s.t. \quad \lambda_i \geq 0, \sum_i \lambda_i = 1 \right\} \tag{5.1.5}$$

Remark: The formed set is convex, since for all $x, y \in S$, so does $(x+y)/2 \in S$.

Now that we defined the two formulations, we state the equivalence.

$$\left.\begin{array}{r} \max\langle \mathbf{c}, \mathbf{x}\rangle \\ x_i \in \{0,1\}^n \\ \sum x_i \leq n/2 \end{array}\right\} \quad \equiv \quad \left\{\begin{array}{r} \max\langle \mathbf{c}, \mathbf{x}\rangle \\ 0 \leq x_i \leq 1 \\ \sum x_i \leq n/2 \end{array}\right. \tag{5.1.6}$$

With this equivalent, we have only $2n+1$ facets to optimize over, and hence we can solve it efficiently.

5.2 Extension Complexity and Cross Polytope

The Cross Polytope example will help us find and define the extension complexity "gap". We start with the definition

Definition 5.3: Cross Polytope

$$Cross-Polytope \equiv Convex-Hull\{e_1, -e_1, e_2, -e_2, \ldots, e_n, -e_n\} \tag{5.2.1}$$

where $e_i \in \{0,1\}^n$ is the standard indicator vector.

Notice that here we essentially have a hypercube, in n dimensions, meaning it needs 2^n inequalities to specify. But also, from the definition, we see there are only $2n$ vertices. Hence, there is a big gap between number of vertices and number of inequalities is.

We now take a look at the cross polytope in an alternative view

$$Cross - Polytope = \left\{ \mathbf{x} : \sum_{i=1}^{n} |x_i| \leq 1 \right\} \tag{5.2.2}$$

$$= \left\{ \mathbf{x} : \exists \mathbf{y}, \quad s.t. \quad \sum y_i \leq 1 \ \wedge \ -y_i \leq x_i \leq y_i \right\} \tag{5.2.3}$$

where notice that by adding a new variable $\mathbf{y}$, we are able to describe cross-polytope with only $2n+1$ inequalities. We define this gap as the extension complexity. Formally,

Definition 5.4: Extension Complexity (Yannakaskis, 88)

The extension complexity of a convex polytope P is the smallest number of facets among convex polytopes Q that have P as a projection. In this context, Q is called an extended formulation of P; and it may have much higher dimension than P.

In English: we are given the freedom to add variables. Now, with this power, we want to minimize the number of inequalities. What is the minimum number of inequalities needed to describe the initial problem in this setting?

We denote extension complexity as $XC(P)$ = minimum number of facets of Q over all extensions Q of P.

Following the notation appeared in the definition, we consider P a polytope

$$P = Convex - Hull\{v_1, \dots, v_n\} \tag{5.2.4}$$

and $Q \subseteq \mathbb{R}^{n+m}$ is an extension of P if

$$P = \{\mathbf{x} : \exists \mathbf{y}, (\mathbf{x}, \mathbf{y}) \in Q\} \tag{5.2.5}$$

To rephrase from the definition, the idea here is that there are many possible extensions (Q). Maybe some Q can be defined with fewer inequalities and define $XC(P)$ as the minimum number of inequalities needed across all extensions.

Notice that the extended form can easily be written down as a equivalent form of the original optimization goal

$$\begin{array}{c} \max\langle \mathbf{c}, \mathbf{x} \rangle \\ \mathbf{x} \in P \end{array} \equiv \begin{array}{c} \max\langle (\mathbf{c}, 0), (\mathbf{x}, \mathbf{y}) \rangle \\ (\mathbf{x}, \mathbf{y}) \in Q \end{array} \tag{5.2.6}$$

Recall at the cross polytope example we presented at the beginning of this section. We have shown the upper bound

$$XC(Cross - Polytope) \leq 2n + 1 \tag{5.2.7}$$

5.3 Examples

5.3.1 Permutahedron

We start by defining a permutahedron.

Definition 5.5: Permutahedron

As the name suggests, a permutahedron is formed by vertices such that they are permutations of each other.

$$\mathbb{R}^n \supseteq P_n = Cvx\{(1,2,3,\dots,n),(2,1,3,\dots,n),\dots\} \tag{5.3.1}$$

which is a convex hull formed by $n!$ vertices.

For a permutahedron, we find that

Proposition 5.1: Facets and Vertices of Permutahedron

$$\#vertices(P_n) = n! \qquad \#facets(P_n) = 2^n \tag{5.3.2}$$

Now, suppose we wish to optimize over this permutahedron set P_n, we have

$$\begin{aligned} \max \quad & \langle \mathbf{c}, \mathbf{x}\rangle \\ & \langle \mathbf{a}_1, \mathbf{x}\rangle \\ & \vdots \\ & \langle \mathbf{a}_N, \mathbf{x}\rangle \\ \text{where } & x \in P_n \end{aligned} \tag{5.3.3}$$

Proposition 5.2: Bounds on Permutahedron

Birkoff-Von Neuman Theorem states that

$$XC(P_n) \le n^2 \tag{5.3.4}$$

and this is later improved by Goemans to

$$XC(P_n) \le O(n \log n) \tag{5.3.5}$$

5.3.2 Spanning Tree Polytope

Definition 5.6

The spanning tree polytope is defined as follows

$$\mathbb{R}^{C_2^n} \supseteq SP_n = Cvx(\text{indicator vectors of spanning tree}) \tag{5.3.6}$$

The definition above seem rather abstract. Here is an example. Consider a graph with four nodes. Then, there are at most $C_2^n = C_2^4 = 6$ edges in this graph, and then we can represent the edges with indicators. I.e., we have possible edges

$$(a,b) \quad (a,c) \quad (a,d) \quad (b,c) \quad (b,d) \quad (c,d) \tag{5.3.7}$$

and it suffices to use indicators to denote which edges are chosen.

For example, consider the following graph which looks like a chain

$$a - b - c - d \tag{5.3.8}$$

it takes indicator form

$$(1,0,0,1,0,1) \tag{5.3.9}$$

we verify that a three edged graph has three 1's in its corresponding indicator vector.

Handling weighted graph is simple, we only need to replace all the 0's with ∞'s to signify that choosing them incurs infinity penalty; we also replace the 1's with respective weights of the edges.

In the classic Minimum Spanning Tree (MST) problem, we aim to find a set of edges in a graph such that it forms a tree and has the lowest total weight. This problem can be translated into LP with the help of our indicator formulation,

$$MST \quad \equiv \quad \begin{matrix} \min \langle \mathbf{w}_G, \mathbf{x} \rangle \\ \mathbf{x} \in SP_n \end{matrix} \quad \equiv - \begin{pmatrix} \max \ -\langle \mathbf{w}_G, \mathbf{x} \rangle \\ \mathbf{x} \in SP_n \end{pmatrix} \tag{5.3.10}$$

Proposition 5.3: Vertices and Facets in Spanning Tree Polytope

$$\#vertices(SP_n) = \#trees \qquad \#facets(SP_n) = 2^n \tag{5.3.11}$$

Proposition 5.4: Extension Complexity of Spanning Tree Polytope

$$XC(SP_n) = \mathcal{O}(n^3) \tag{5.3.12}$$

5.3.3 TSP Polytope

Definition 5.7: TSP Problem

Given a weighted graph, $G = (V, E)$, find the tour of a least total weight; where tour is defined as a loop inside the graph that visits every vertex exactly once, finishing at the starting position.

Definition 5.8: TSP Polytope

$$\mathbb{R}^{C_2^n} \supseteq Cvx(\text{all valid tours encoded with indicator vectors}) \tag{5.3.13}$$

where the definition utilizes the indicator trick for specifying graphs we say in the previous spanning tree polytope example.

With this definition, our TSP problem turns into the following optimization problem

$$\begin{pmatrix} \min \langle W_G, \mathbf{x} \rangle \\ \mathbf{x} \in TSP_n \end{pmatrix} \quad \equiv \quad \text{find minimum tour of } G \tag{5.3.14}$$

Proposition 5.5: Vertices and Facets of TSP Polytope

$$\#vertices(TSP_n) = n! \qquad \#facets(TSP_n) = 2^n \tag{5.3.15}$$

In the previous examples, we were lucky to find polynomial extension complexities. However, for TSP, a known NP-Complete problem, we are not so lucky. The question then to ask is what is $XC(TSP_n)$?

Theorem 5.1: Y88, Symmetric Extension Complexity

$$XC_{Symm}(TSP_n) \geq 2^{\Omega(n)} \tag{5.3.16}$$

i.e., is at least exponential. If we restrict the TSP_n such that the graphs inside are symmetric, then we cannot get much savings.

The remaining question to ask is what is $XC(TSP_n)$? This was left as an open question in the original paper.

5.4 FMPTW12 & Alternative Proof

Theorem 5.2: Fiorini, Massor, Prokutta, Tiwary, Wolf (FMPTW12)

$$XC(TSP_n) \geq 2^{\Omega(n)} \tag{5.4.1}$$

Definition 5.9: Slack Matrix

Consider the polytope, written in the forms both as a convex hull and as a polyhedron

$$P = Cvx\left(\mathbf{x}_1, \dots, \mathbf{x}_N\right) = \{\mathbf{x} : \langle a_j, \mathbf{x}\rangle \leq b_j\,, j = 1, \dots, N\} \tag{5.4.2}$$

Whenever we have the above, Slack Matrix $S \in \mathbb{R}^{N\times M}$ can then be defined as

$$S[i,j] = b_j - \langle a_j, x_i\rangle \tag{5.4.3}$$

i.e. "how much slack is there in the j-th inequality for vertex i.

Non- negative Property We note that the slack matrix has entries $S[i,j] \geq 0, \forall i, j$.

Definition 5.10: Rank

Recall from LA that the rank of a matrix $rank(S)$ is either of the following

- dimension of the space spanned by the rows of S, the row space
- dimension of the space spanned by the columns of S, the column space
- minimum r such that S can be written as $S = UV^\top$, where $U \in \mathbb{R}^{N\times r}$ and $V^\top \in \mathbb{R}^{r\times M}$.

Definition 5.11: Non Negative Rank (NNR)

If we have a non negative matrix, the NNR is a similar notion. Formally, if we have matrix $S \geq 0$, then

$$nnr(S) = \min_r \left\{S = UV^\top\right. \tag{5.4.4}$$

where $\mathbb{R}^{N\times r} \ni U \geq 0$ and $\mathbb{R}^{r\times M} \ni V^\top \geq 0$. Notice that here we added an additional requirement for the factorized matrices to also be non negative.

Proposition 5.6

$$rank(S) \leq nnr(S) \leq \min(N, M) \tag{5.4.5}$$

Theorem 5.3: NNR and XC, Y88

If P is a polytope and S is a slack matrix of P, then

$$XC(P) = nnr(S) \tag{5.4.6}$$

This theorem relates something geometric (XC) to a very concrete computable value. We will use this to prove Theorem 5.2, i.e. we try to show *nnr* (slack matrix of TSP_n) = $2^{\Omega(n)}$. We will not show this exactly. Instead, we show it for another polytope, which has a reduction to TSP_n. We leave that to later part of this chapter. Let's show Theorem 5.3.

Proof: We show it from two inequalities. First, we show

$$XC(P) \leq nnr(S) \tag{5.4.7}$$

To be exact, we want to show: if $nnr(S) = r$, then there is an extension of P with $\leq r$ inequalities. From definitions, we can write

$$S = UV, \qquad U \geq 0, V \geq 0, U \in \mathbb{R}^{N\times r}, V \in \mathbb{R}^{r\times M} \tag{5.4.8}$$

Call the rows of U as u_i and the columns of V as v_j. Then,

$$b_j - \langle a_j, \mathbf{x}_i \rangle = S[i,j] = \langle u_i, v_j \rangle \tag{5.4.9}$$

where i, j corresponds to indices of vertices and inequalities respectively. We can massage the above equality to form

$$\langle a_j, x_i \rangle + \langle u_i, v_j \rangle = b_j \tag{5.4.10}$$

$$\iff \langle (a_j, v_j), (x_i, u_i) \rangle = b_j \tag{5.4.11}$$

Recall definition of P,

$$P = \{x : Ax \leq b\} \tag{5.4.12}$$

Now we define a new extension

$$Q = \{(x,u) : Ax + Vu = b, u \geq 0\} = \left\{(x,u) : \langle a_j, x \rangle + \langle v_j, u \rangle = b_j, u \geq 0\right\} \tag{5.4.13}$$

♡