# CS289a: Great Theory Hits of 21st Century

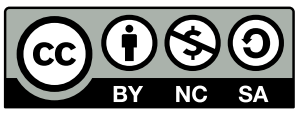
Tingfeng Xia

Winter 2023

Tingfeng Xia.

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





# Contents


**1 SL = L: Undirected connectivity in Logspace** **5**
- 1.1 Computing Resources . . . 5
- 1.2 Problem Statement . . . 5
  - 1.2.1 Log Space USTCON - Results . . . 6
- 1.3 Randomized Algorithm for Connectivity . . . 6
- 1.4 Spectral Graph Theory . . . 7
- 1.5 Path Enumeration . . . 11
  - 1.5.1 Reingold's Idea . . . 12
  - 1.5.2 Reducing Degree . . . 12
  - 1.5.3 Improving Spectral Gap . . . 12
- 1.6 Zig Zag Analysis . . . 16
- 1.7 Undirected s-t Connectivity in Log Space . . . 21
  - 1.7.1 The Procedure . . . 21
  - 1.7.2 Path Enumeration Path Length Analysis . . . 21
  - 1.7.3 Space Complexity . . . 22
- 1.8 Expander Graphs . . . 23
  - 1.8.1 History of Expander Graphs . . . 24

**2 Sensitivity Conjecture** **25**
- 2.1 Classic Combinatorial Measures . . . 25
  - 2.1.1 Decision-Tree Complexity . . . 25
  - 2.1.2 Certificate Complexity . . . 25
- 2.2 Sensitivity Conjecture . . . 28

# 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 $O(|V| + |E|)$ and $O(|V|)$ 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 $Pr[\text{Algorithm wrong}] < \frac{1}{3}$

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 f 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 $\leq 1$.

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

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

Thus, $\lambda \leq 1$. ■

**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 $|\lambda| \leq 1$. With $x_j = \max(\mathbf{x})$ as the largest entry in the eigenvector, we recall (and abstract the inequality used back then as

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

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

$$x_i = x_j, \quad \forall v_i \in N(v_j) \tag{1.4.7}$$

This suffices as a proof to every eigenvector with eigenvalue 1 to $M(G)$ is the $\mathbf{1}$ 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

$$1 = \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N \tag{1.4.8}$$

*Proof:* This follows from the proof for Prop. 1.5 where we proved that all eigenvalues of $M(G)$ have magnitude $\leq 1$. Since the one vector $\mathbf{1}$ 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 $M(G)$

$$\max(|\lambda_2|, |\lambda_3|, \ldots, |\lambda_N|) \leq 1 \tag{1.4.9}$$

*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

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

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 $\lambda$ 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 $\mathbf{v}$ 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 $(N, D, \lambda)$ 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 $= \infty$, 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] $\lambda$ 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 $(N, D, \lambda)$ graph with self loops, then $G^2$ is a $(N, D^2, \lambda^2)$ 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

$$(\log D) \cdot \log_{\frac{1}{\lambda}} N \rightsquigarrow \left(\log D^2\right) \log_{\frac{1}{\lambda^2}} N \tag{1.5.15}$$

$$= 2 \cdot \log D \cdot \frac{1}{2} \cdot \log_{\frac{1}{\lambda}} N \tag{1.5.16}$$

$$= (\log D) \cdot \log_{\frac{1}{\lambda}} N \tag{1.5.17}$$

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

$$G_{i+1} = G_i^2 ⓩ H \tag{1.5.18}$$

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

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

**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**

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

**Rotations**

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

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

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

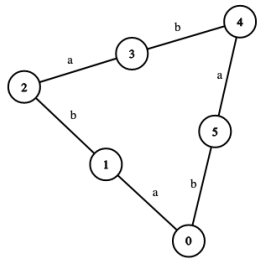



Figure 1.1: Illustration of consistent labelling.

and

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

**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 $F = G \textcircled{z} H$,

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

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

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 $M_G$,

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

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

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

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

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

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

$$M_F = \left( (1-\lambda_H)\mathbb{I}_N \otimes \frac{J_D}{D} + \lambda_H \mathbb{I}_N \otimes E_H \right) \times P \times \left( (1-\lambda_H)\mathbb{I}_N \otimes \frac{J_D}{D} + \lambda_H \mathbb{I}_N \otimes E_H \right) \tag{1.6.20}$$

$$= (1-\lambda_H)^2 \left( \mathbb{I}_N \otimes \frac{J_D}{D} \right) \cdot P \cdot \left( \mathbb{I}_N \otimes \frac{J_D}{D} \right) \tag{1.6.21}$$

$$+ (1-\lambda_H)\lambda_H \underbrace{\left( \mathbb{I}_N \otimes \frac{J_D}{D} \right) \cdot P \cdot (\mathbb{I}_N \otimes E_H)}_{E^{(1)}} \tag{1.6.22}$$

$$+ \lambda_H(1-\lambda_H) \underbrace{(\mathbb{I}_N \otimes E_H) \cdot P \cdot \left( \mathbb{I}_N \otimes \frac{J_D}{D} \right)}_{E^{(2)}} \tag{1.6.23}$$

$$+ \lambda_H^2 \cdot \underbrace{(\mathbb{I}_N \otimes E_H) \cdot P \cdot (\mathbb{I}_N \otimes E_H)}_{E^{(3)}} \tag{1.6.24}$$

$$= (1-\lambda_H)^2 \left( \mathbb{I}_N \otimes \frac{J_D}{D} \right) P \left( \mathbb{I}_N \otimes \frac{J_D}{D} \right) + \underbrace{(1-\lambda_H)\lambda_H E^{(1)} + \lambda_H(1-\lambda_H)E^{(2)} + \lambda_H^2 E^{(3)}}_{E^{(4)}} \tag{1.6.25}$$

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.

$$\|E^{(1)}\| = \left\| \left( \mathbb{I}_N \otimes \frac{J_D}{D} \right) \cdot P \cdot (\mathbb{I}_N \otimes E_H) \right\| \tag{1.6.26}$$

$$\leq \left\| \mathbb{I}_N \otimes \frac{J_D}{D} \right\| \cdot \|P\| \cdot \|\mathbb{I}_N \otimes E_H\| \tag{1.6.27}$$

$$\leq 1 \cdot 1 \cdot 1 \tag{1.6.28}$$

$$\leq 1 \tag{1.6.29}$$

Then,

$$\|E^{(4)}\| \leq \|(1-\lambda_H)\lambda_H + \lambda_H(1-\lambda_H) + \lambda_H^2\| \tag{1.6.30}$$

$$= (1-\lambda_H)\lambda_H + \lambda_H(1-\lambda_H) + \lambda_H^2 \tag{1.6.31}$$

$$= 2\lambda_H(1-\lambda_H) + \lambda_H^2 \tag{1.6.32}$$

$$= 1 - (1-\lambda_H)^2 \tag{1.6.33}$$

**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**

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

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]
$$G_k = G_{k-1}^2 ⓩ H \tag{1.7.2}$$
- **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 $a_\cdot$ is a vertex name in $H$.

### 1.7.2 Path Enumeration Path Length Analysis

Let $\lambda_k$ 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 $\lambda_{k-1}^2$ 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 $L = O(\log N_0)$, 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

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

*Proof:* By invariant stated above,

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

then the path length bound becomes

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

### 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

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

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 $G_L$, $(e_1, e_2, \ldots, e_\Delta)$ 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 $G_k = G_{k-1}^2 ⓩ H$.

**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 $Rot_{G_{k-1}^2 ⓩ H}$ 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 $\lambda$-expander if $G$ is a $(N, D, \lambda)$-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 $\lambda$ values

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

$$\vee_n = \neg(\wedge(\neg x_1, \dots, \neg x_n)) \tag{2.1.13}$$
$$= 1 - (1 - x_1)(1 - x_2) \dots (1 - x_n) \tag{2.1.14}$$

**Proposition 2.4**

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

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

*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,

$$f(x) = \sum_{\ell \in L} (x_{i1} == a_1)(x_{i2} == a_2) \dots (x_{id} == a_d) \cdot (\ell.\text{output}) \tag{2.1.17}$$
$$= \sum_{\ell \in L} (a_1 x_{i1} + (1 - a_1)(1 - x_{i1})) \dots (d x_{id} + (1 - a_d)(1 - x_{id})) \tag{2.1.18}$$
$$\implies degree(f) \leq DT(f) \tag{2.1.19}$$

**Proposition 2.5: Summary**

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

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,

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

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 $\Delta$ through out this chapter.

*Proof:* [TODO] will be added later

**Theorem 2.4: Hao Huang, 2019**

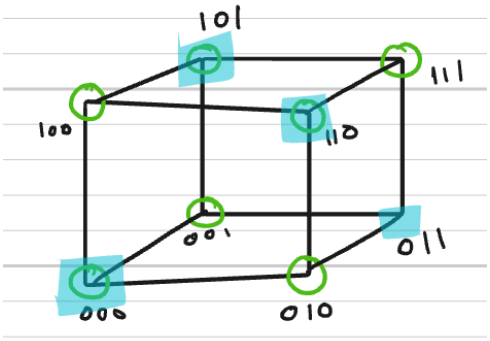



Figure 2.1: N3 Parity Sub-graph.

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

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

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 with interlacing eigenvalues

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

♡