Graphs

Amotz Bar-Noy

CUNY
**Definition:** A graph is a collection of edges and vertices. Each edge connects two vertices.
Different Drawings of the Same Graph
Graph Isomorphism

Graph $G_1$ and graph $G_2$ are isomorphic if there is a one-one correspondence between their vertices such that the number of edges joining any two vertices of $G_1$ is equal to the number of edges joining the corresponding vertices of $G_2$.

\[a \leftrightarrow A \quad b \leftrightarrow B \quad c \leftrightarrow C \quad d \leftrightarrow D \quad e \leftrightarrow E \quad f \leftrightarrow F\]
Famous Graph Problems

- The 7 bridges of Königsberg
  https://www.youtube.com/watch?v=n2wSo4vfw6c (4:39 min)

- The 4 color map problem
  https://www.youtube.com/watch?v=NgbK43jB4rQ (14:17 min)

- The Traveling Salesperson Problem
  https://www.youtube.com/watch?v=l8KBKitQ3T4 (1:15 min)
  https://www.youtube.com/watch?v=SC5CX8drAtU (2:22 min)
Notations

- \( G = (V, E) \) – graph.
- \( V = \{1, \ldots, n\} \) – set of vertices.
- \( E \subseteq V \times V \) – set of edges.
- \( e = (u, v) \in E \) – edge.
- \( n = |V| = V \) – number of vertices.
- \( m = |E| = E \) – number of edges.
Directed and Undirected Graphs

- In **undirected graphs** \((u, v) = (v, u)\).

- In **directed graphs (D-graphs)** \((u \rightarrow v) \neq (v \rightarrow u)\).

- The **underlying** undirected graph \(G' = (V', E')\) of a directed graph \(G = (V, E)\):
  - Has the same set of vertices: \(V = V'\).
  - Has all the edges of \(G\) without their direction:
    - \((u \rightarrow v)\) becomes \((u, v)\).
Undirected Edges

- Vertices $u$ and $v$ are the **endpoints** of the edge $(u, v)$.
- Edge $(u, v)$ is **incident** with vertices $u$ and $v$.
- Vertices $u$ and $v$ are **neighbors** if edge $(u, v)$ exists.
  - $u$ is **adjacent** to $v$ and $v$ is **adjacent** to $u$.
- Vertex $u$ has **degree** $d$ if it has $d$ neighbors.
- Edge $(v, v)$ is a **(self) loop** edge.
- Edges $e_1 = (u, v)$ and $e_2 = (u, v)$ are **parallel** edges.
Directed Edges

- Vertex $u$ is the **origin** (initial) and vertex $v$ is the **destination** (terminal) of the directed edge $(u \rightarrow v)$.

- Vertex $v$ is the **neighbor** of vertex $u$ if the directed edge $(u \rightarrow v)$ exists.
  - $v$ is **adjacent** to $u$ (but $u$ is not adjacent to $v$).

- Vertex $u$ has
  - **out-degree** $d$ if it has $d$ neighbors.
  - **in-degree** $d$ if it is the neighbor of $d$ vertices.
In **Weighted graphs** there exists a weight function: \( w : E \to \mathbb{R} \).

- Weights could be negative.

\[
 w(AC) \leq w(AB) + w(BC)
\]

- Sometimes weights obey the **triangle inequality**. E.g., Distances in the plane.
Simple Graphs

- A **simple** directed or undirected graph is a graph with no parallel edges and no self loops.

- In a simple directed graph both edges: \((u \rightarrow v)\) and \((v \rightarrow u)\) could exist (they are not parallel edges).

Number of Edges in Simple Graphs:

- A simple undirected graph has at most \(m = \binom{n}{2}\) edges.
- A simple directed graph has at most \(m = n(n - 1)\) edges.
- A **dense** simple (directed or undirected) graph has “many” edges: \(m = \Theta(n^2)\).
- A **sparse** (shallow) simple (directed or undirected) graph has “few” edges: \(m = \Theta(n)\).
Labelled and Unlabelled Graphs

- In a **labelled** graph each vertex has a unique label (ID).
  - Usually the labels are: $1, \ldots, n$.

**Observation:** There are $2^{\binom{n}{2}}$ non-isomorphic labelled graphs with $n$ vertices.

**Proof:** Each possible edge exists or does not exist.
The 8 Labelled Graphs with $n = 3$ vertices.
The 4 Unlabelled Graphs with $n = 3$ Vertices
An undirected or directed path $P = \langle v_0, v_1, \ldots, v_k \rangle$ of length $k$ is an ordered list of vertices such that $(v_i, v_{i+1})$ or $(v_i \rightarrow v_{i+1})$ exists for $0 \leq i \leq k - 1$ and all the edges are different.

An undirected or directed cycle $C = \langle v_0, v_1, \ldots, v_{k-1}, v_0 \rangle$ of length $k$ is an undirected or directed path that starts and ends with the same vertex.

In a simple path, directed or undirected, all the vertices are different.

In a simple cycle, directed or undirected, all the vertices except $v_0 = v_k$ are different.
Special Paths and Cycles

- An undirected or directed **Euler path (tour)** is a path that traverses all the edges.

- An undirected or directed **Euler cycle (circuit)** is a cycle that traverses all the edges.

- An undirected or directed **Hamiltonian path (tour)** is a simple path that visits all the vertices.

- An undirected or directed **Hamiltonian cycle (circuit)** is a simple cycle that visits all the vertices.
Connected Graphs and Strongly Connected Directed Graphs

- **Connectivity:** In connected undirected graphs there exists a path between any pair of vertices.

- **Observation:** In a simple connected undirected graph there are at least $m = n - 1$ edges.

- **Strong connectivity:** In a strongly connected directed graph there exists a directed path from $u$ to $v$ for any pair of vertices $u$ and $v$.

- **Observation:** In a simple strongly connected directed graph there are at least $m = n$ edges.
A connected sub-graph $G'$ is a **connected component** of an undirected graph $G$ if there is no connected sub-graph $G''$ of $G$ such that $G'$ is also a subgraph of $G''$.

A connected component $G'$ is a **maximal** sub-graph with the connectivity property.

A connected graph has exactly one connected component.
A strongly connected directed sub-graph $G'$ is a **strongly connected component** of a directed graph $G$ if there is no strongly connected directed sub-graph $G''$ of $G$ such that $G'$ is also a subgraph of $G''$.

A strongly connected component $G'$ is a **maximal** sub-graph with the strong connectivity property.

A strongly connected graph has exactly one strongly connected component.
In the **WEB graph**, a hyper-link from page \( p \) to page \( q \) is modeled by the directed edge \( (p \rightarrow q) \).

Broder et. al (Graph Structure of the Web, 2000)
Examined a large web graph (200M pages, 1.5B links)
Theorem: Let $G$ be a simple undirected graph with $n$ vertices and $k$ connected components then:

$$n - k \leq m \leq \frac{(n - k)(n - k + 1)}{2}.$$ 

Corollary: A simple undirected graph with $n$ vertices is connected if it has $m$ edges for:

$$m > \frac{(n - 2)(n - 3)}{2}.$$
Assumptions

Unless stated otherwise, usually a graph is:

- Simple.
- Undirected.
- Unlabelled.
- Unweighted.
- Connected.
Forests and Trees

- **Forest**: A graph with no cycles.
- **Tree**: A connected graph with no cycles.

**By definition:**
- A tree is a connected forest.
- Each connected component of a forest is a tree.
Theorem: An undirected and simple graph is a tree if:
- It is connected and has no cycles.
- It is connected and has exactly $m = n - 1$ edges.
- It has no cycles and has exactly $m = n - 1$ edges.
- It is connected and deleting any edge disconnects it.
- Any 2 vertices are connected by exactly one path.
- It has no cycles and any new edge forms one cycle.

Corollary: The number of edges in a forest with $n$ vertices and $k$ trees is $m = n - k$. 
Trees

- **Theorem:** An undirected and simple graph is a tree if:
  - It is connected and has no cycles.
  - It is connected and has exactly $m = n - 1$ edges.
  - It has no cycles and has exactly $m = n - 1$ edges.
  - It is connected and deleting any edge disconnects it.
  - Any 2 vertices are connected by exactly one path.
  - It has no cycles and any new edge forms one cycle.

- **Corollary:** The number of edges in a forest with $n$ vertices and $k$ trees is $m = n - k$. 
**Theorem:**
There are $n^n - 2^n$ distinct labelled $n$ vertices trees.
Theorem: There are $n^{n-2}$ distinct labelled $n$ vertices trees.
Null graphs are graphs with no edges.

The null graph with $n$ vertices is denoted by $N_n$.

In null graphs $m = 0$. 

Null Graphs
Complete graphs (cliques) are graphs with all possible edges.

The complete graph with $n$ vertices is denoted by $K_n$.

In complete graphs $m = \binom{n}{2} = \frac{n(n-1)}{2}$.
Cycles (rings) are connected graphs in which all vertices have degree 2 \((n \geq 3)\).

The cycle with \(n\) vertices is denoted by \(C_n\).

In cycles \(m = n\).
Paths are cycles with one edge removed (paths are trees).

The path with $n$ vertices is denoted by $P_n$.

In paths $m = n - 1$. 
Stars are graphs with one root and $n - 1$ leaves (stars are trees).

- The star with $n$ vertices is denoted by $S_n$.
- In stars $m = n - 1$. 
Wheels are stars in which all the $n - 1$ leaves form a cycle.

The wheel with $n$ vertices is denoted by $W_n$.

In wheels $m = 2n - 2$ for $n \geq 4$. 

Bipartite Graphs

Bipartite graphs \( V = A \cup B \): Each edge is incident to one vertex from \( A \) and one vertex from \( B \).

Observation: A graph is bipartite iff each cycle is of even length.
Complete bipartite graphs $K_{r,c}$: All possible $r \cdot c$ edges exist.
There are $n = 2^k$ vertices representing all the $2^k$ binary sequences of length $k$.

Two vertices are adjacent if their corresponding sequences differ by exactly one bit.
Observation: Hyper-Cubes are bipartite graphs.

Proof:

- **A**: The vertices with even number of 1 in their binary representation.
- **B**: The vertices with odd number of 1 in their binary representation.
- Any edge connects two vertices one from the set A and one from the set B.
**Planar Graphs**

- **Definition:** Planar graphs are graphs that can be drawn on the plane such that edges do not cross each other.

- **Theorem:** A graph is planar iff it does not have sub-graphs homeomorphic to $K_5$ and $K_{3,3}$.

- **Theorem:** Every planar graph can be drawn with straight lines.
$K_5$: the complete graph with 5 vertices.

$K_{3,3}$: the complete $\langle 3, 3 \rangle$ bipartite graph.
In \( \Delta \)-regular graphs the degree of each vertex is exactly \( \Delta \).

In \( \Delta \)-regular graphs \( m = \frac{\Delta \cdot n}{2} \).

The Petersen Graph is a 3-regular graph.
Random Graphs

**Definition I:**
- Each edge exists with probability $0 \leq p \leq 1$.
- **Observation:** Expected number of edges is $E(m) = p \binom{n}{2}$.

**Definition II:**
- A graph with $m$ edges that is selected *randomly* with a uniform distribution over all graphs with $m$ edges.

**Remark I:** Both definition are not equivalent.

**Remark II:** There are many other random graphs models.
Social Graphs

- **Definition:** The social graph contains all the friendship relations (edges) among $n$ people (vertices).

- **I:** In any group of $n \geq 2$ people, there are 2 people with the same number of friends in the group.

- **II:** There exists a group of 5 people for which no 3 are mutual friends and no 3 are mutual strangers.

- **III:** Every group of 6 people contains either three mutual friends or three mutual strangers.
Data structure for Graphs

- **Adjacency lists**: $\Theta(n + m)$ memory.

- **An adjacency Matrix**: $\Theta(n^2)$ memory.

- **An incident matrix**: $\Theta(n \cdot m)$ memory.
The Adjacency Lists Representation

- Each vertex is associated with a linked list consisting of all of its neighbors.
- In a directed graph there are two lists: an incoming list and an outgoing list.
- In a weighted graph each record in the list has an additional field for the weight.
The Adjacency Lists Representation

- Each vertex is associated with a linked list consisting of all of its neighbors.

- In a directed graph there are two lists: an incoming list and an outgoing list.

- In a weighted graph each record in the list has an additional field for the weight.

**Memory:** $\Theta(n + m)$.

- Undirected graphs: $\sum_v \text{Deg}(v) = 2m$

- Directed graphs: $\sum_v \text{OutDeg}(v) = \sum_v \text{InDeg}(v) = m$
Example – Adjacency Lists

A → (B, C, D)
B → (A, C, E)
C → (A, B, F)
D → (A, E, F)
E → (B, D, F)
F → (C, D, E)
The Adjacency Matrix Representation

A matrix $A$ of size $n \times n$:
- $A[u, v] = 1$ if $(u, v)$ or $(u \rightarrow v)$ is an edge.
- $A[u, v] = 0$ if $(u, v)$ or $(u \rightarrow v)$ is not an edge.

In simple graphs: $A[u, u] = 0$


In weighted graphs: $A[u, v] = w(u, v)$
The Adjacency Matrix Representation

- A matrix $A$ of size $n \times n$:
  - $A[u, v] = 1$ if $(u, v)$ or $(u \rightarrow v)$ is an edge.
  - $A[u, v] = 0$ if $(u, v)$ or $(u \rightarrow v)$ is not an edge.

- In simple graphs: $A[u, u] = 0$
- In weighted graphs: $A[u, v] = w(u, v)$

- **Memory:** $\Theta(n^2)$.
  - Independent of $m$ that could be $o(n^2)$ and even $O(n)$. 
Example – Adjacency Matrix

\[
\begin{array}{ccccccc}
A & B & C & D & E & F \\
A & 0 & 1 & 1 & 1 & 0 & 0 \\
B & 1 & 0 & 1 & 0 & 1 & 0 \\
C & 1 & 1 & 0 & 0 & 0 & 1 \\
D & 1 & 0 & 0 & 0 & 1 & 1 \\
E & 0 & 1 & 0 & 1 & 0 & 1 \\
F & 0 & 0 & 1 & 1 & 1 & 0 \\
\end{array}
\]
The Incident Matrix Representation

- A matrix $A$ of size $n \times m$:
  - $A[v, e] = 1$ if undirected edge $e$ is incident with $v$.
  - Otherwise $A[v, e] = 0$.

- In simple graphs all the columns are different and each contains exactly two non-zero entries.

- In weighted undirected graphs: $A[v, e] = w(e)$ if edge $e$ is incident with vertex $v$.

Memory: $\Theta(n \cdot m)$. 

Amotz Bar-Noy (CUNY)  Graphs  46 / 72
A matrix $A$ of size $n \times m$:

- $A[v, e] = 1$ if undirected edge $e$ is incident with $v$.
- Otherwise $A[v, e] = 0$.

In simple graphs all the columns are different and each contains exactly two non-zero entries.

In weighted undirected graphs: $A[v, e] = w(e)$ if edge $e$ is incident with vertex $v$.

**Memory**: $\Theta(n \cdot m)$. 
Example – Incident Matrix

<table>
<thead>
<tr>
<th></th>
<th>(A, B)</th>
<th>(A, C)</th>
<th>(A, D)</th>
<th>(B, C)</th>
<th>(B, E)</th>
<th>(C, F)</th>
<th>(D, E)</th>
<th>(D, F)</th>
<th>(E, F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Which Data Structure to Choose?

- Adjacency matrices are simpler to implement and maintain.
- Adjacency matrices are better for dense graphs.
- Adjacency lists are better for sparse graphs.
- Adjacency lists are better for algorithms whose complexity depends on $m$.
- Incident matrices are not efficient for algorithms.
The **degree** $d_v$ of vertex $v$ in graph $G$ is the number of neighbors of $v$ in $G$.

**The hand-shaking Lemma:** $\sum_{i=1}^{n} d_i = 2m$.
- Each edge "contributes" exactly 2 to the sum.

**Corollary:** Number of odd degree vertices is even.

**The degree sequence** of $G$ is $S = (d_1, \ldots, d_n)$. 

Graphic Sequences

- **The degree** $d_v$ of vertex $v$ in graph $G$ is the number of neighbors of $v$ in $G$.

- **The hand-shaking Lemma**: $\sum_{i=1}^{n} d_i = 2m$.
  - Each edge "contributes" exactly 2 to the sum.

- **Corollary**: Number of odd degree vertices is even.

- **The degree sequence** of $G$ is $S = (d_1, \ldots, d_n)$.

- **Definition**: A sequence $S = (d_1, \ldots, d_n)$ is **graphic** if there exists a graph with $n$ vertices whose degree sequence is $S$. 
Theorem: A sequence \((d_1 \geq d_2 \cdots \geq d_n)\) is graphic if the following two conditions hold:

- \(d_1 + d_2 + \cdots + d_n\) is even.
- for \(1 \leq k \leq n\):
  \[
  \sum_{i=1}^{k} d_i \leq k(k - 1) + \sum_{i=k+1}^{n} \min\{d_i, k\}.
  \]

Complexity: Can be done with \(\Theta(n)\) operations.
Non-Graphic Sequences

- $(3, 3, 3, 3, 3, 3, 3)$ is not graphic (equivalently, there is no 7-vertex 3-regular graph).
  - Since $\sum_{i=1}^{7} d_i$ is odd.

- $(5, 5, 4, 4, 0)$ is not graphic.
  - Since there are 5 vertices and therefore the maximum degree could be at most 4.

- $(3, 2, 1, 0)$ is not graphic.
  - Since there is a vertex with degree 3 and only two additional vertices with a positive degree.
I: The sequence \((0, 0, \ldots, 0)\) of length \(n\) is graphic. Since it represents the null graph \(N_n\).

II: In a graphic sequence \(S = (d_1 \geq \cdots \geq d_n)\) \(d_1 \leq n - 1\).

III: \(d_{d_1+1} > 0\) in a graphic sequence of a non-null graph \(S = (d_1 \geq \cdots \geq d_n)\).
Transformation

- Let $S = (d_1 \geq \cdots \geq d_n)$, then
  - $f(S) = (d_2 - 1 \geq \cdots \geq d_{d_1+1} - 1, d_{d_1+2} \geq \cdots \geq d_n).$
Transformation

Let $S = (d_1 \geq \cdots \geq d_n)$, then

$$f(S) = (d_2 - 1 \geq \cdots \geq d_{d_1+1} - 1, d_{d_1+2} \geq \cdots \geq d_n).$$

**Example:**

- $S = (5, 4, 3, 3, 2, 1, 1, 1)$
- $f(S) = (3, 2, 2, 1, 0, 1, 1)$
Lemma

\[ S = (d_1 \geq \cdots \geq d_n) \text{ is graphic iff } f(S) \text{ is graphic.} \]
Lemma

- \( S = (d_1 \geq \cdots \geq d_n) \) is graphic \textbf{iff} \( f(S) \) is graphic.

\( \Leftarrow \) To get a graphic representation for \( S \), add a vertex of degree \( d_1 \) to the graphic representation of \( f(S) \) and connect this vertex to all vertices whose degrees in \( f(S) \) are smaller by 1 than those in \( S \).

\( \Rightarrow \) To get a graphic representation for \( f(S) \), omit a vertex of degree \( d_1 \) from the graphic representation of \( S \). Make sure (how?) that this vertex is connected to the vertices whose degrees are \( d_2, \ldots, d_1+1 \).
Lemma

\[ S = (d_1 \geq \cdots \geq d_n) \text{ is graphic iff } f(S) \text{ is graphic.} \]

\[
\begin{align*}
\iff & \text{ To get a graphic representation for } S, \text{ add a vertex of degree } d_1 \text{ to} \\
& \text{ the graphic representation of } f(S) \text{ and connect this vertex to all} \\
& \text{ vertices whose degrees in } f(S) \text{ are smaller by 1 than those in } S. \\
\Rightarrow & \text{ To get a graphic representation for } f(S), \text{ omit a vertex of degree } d_1 \\
& \text{ from the graphic representation of } S. \text{ Make sure (how?) that this} \\
& \text{ vertex is connected to the vertices whose degrees are } d_2, \ldots, d_{d_1+1}. 
\end{align*}
\]
Algorithm

\texttt{Graphic}(S = (d_1 \geq \cdots \geq d_n \geq 0))
\begin{enumerate}
\item case \( d_1 = 0 \) return \textsc{true} (* Obs. I *)
\item case \( d_1 \geq n \) return \textsc{false} (* Obs. II *)
\item case \( d_{d_1+1} = 0 \) return \textsc{false} (* Obs. III *)
\item otherwise return \texttt{Graphic}(\texttt{Sort}(f(S))) (* Lemma *)
\end{enumerate}
Implementation Outline

- Maintain $n$ sets of vertices $B_{n-1}, B_{n-2}, \ldots, B_1, B_0$.
- $B_i$ contains all the vertices that are “looking” for $i$ more neighbors.
- Initially $v_i$ is placed in bin $B_{d_i}$.
- In each round,
  - Let the degree of the highest degree vertex $u$ be $d$.
  - Let $u_1, u_2, \ldots, u_d$ be the new neighbors of $u$ whose degrees are $c_1, c_2, \ldots, c_d$ respectively.
  - Move $u$ from $B_d$ to $B_0$.
  - For all $1 \leq j \leq d$, move $u_j$ from $B_{c_j}$ to $B_{c_j-1}$.

Complexity: $\Theta(m)$ for all rounds since $\sum_{i=1}^{n} d_i = 2m$. 
Maintain $n$ sets of vertices $B_{n-1}, B_{n-2}, \ldots, B_1, B_0$.

$B_i$ contains all the vertices that are “looking” for $i$ more neighbors.

Initially $v_i$ is placed in bin $B_{d_i}$.

In each round,

- Let the degree of the highest degree vertex $u$ be $d$.
- Let $u_1, u_2, \ldots, u_d$ be the new neighbors of $u$ whose degrees are $c_1, c_2, \ldots, c_d$ respectively.
- Move $u$ from $B_d$ to $B_0$.
- For all $1 \leq j \leq d$, move $u_j$ from $B_{c_j}$ to $B_{c_j-1}$.

**Complexity:** $\Theta(m)$ for all rounds since $\sum_{i=1}^{n} d_i = 2m$. 
Call the vertices of the graphic sequence \( v_1, v_2, \ldots, v_n \) where the degree of \( v_i \) is \( d_i \).

Initially there are no edges in the graph.

In each round,

- Let the degree of the highest degree vertex \( v_i = u \) be \( d \).
- Let \( v_{i_1} = u_1, v_{i_2} = u_2, \ldots, v_{i_d} = u_d \) be the new neighbors of \( v_i = u \).
- For all \( 1 \leq j \leq d \), add the edge \((v_i, v_{i_j}) = (u, u_j)\) to the graph.

Complexity: \( \Theta(m) \) for all rounds since \( \sum_{i=1}^{n} d_i = 2m \).
Call the vertices of the graphic sequence \( v_1, v_2, \ldots, v_n \) where the degree of \( v_i \) is \( d_i \).

Initially there are no edges in the graph.

In each round,

- Let the degree of the highest degree vertex \( v_i = u \) be \( d \).
- Let \( v_{i_1} = u_1, v_{i_2} = u_2, \ldots, v_{i_d} = u_d \) be the new neighbors of \( v_i = u \).
- For all \( 1 \leq j \leq d \), add the edge \((v_i, v_{i_j}) = (u, u_j)\) to the graph.

**Complexity:** \( \Theta(m) \) for all rounds since \( \sum_{i=1}^{n} d_i = 2m \).
Example

Initial sequence: \((A, B, C, D, E, F, G, H) = (4, 4, 3, 2, 2, 2, 2, 1)\)
Example

After Round 1: \((A, B, C, D, E, F, G, H) = (0, 3, 2, 1, 1, 2, 2, 1)\)
Example

- After Round 2: \((A, B, C, D, E, F, G, H) = (0, 0, 1, 1, 1, 1, 1, 1)\)
After Rounds 3, 4, 5: \((A, B, C, D, E, F, G, H) = (0, 0, 0, 0, 0, 0, 0, 0)\)
The realizing graph: \((A, B, C, D, E, F, G, H) = (4, 4, 3, 2, 2, 2, 2, 1)\)
Call the vertex that is selected in each round the **pivot** vertex.

The algorithm works for any vertex being the **pivot** vertex as long as it is connected to the highest degree vertices.

Different selections of **pivot** vertices may lead to different non-isomorphic realizations.

However, not all the graphs can be realized by this algorithm.
Initial sequence: \((A, B, C, D, E, F, G, H) = (4, 4, 3, 2, 2, 2, 2, 1)\)
After Round 1: \((A, B, C, D, E, F, G, H) = (3, 4, 3, 2, 2, 2, 2, 0)\)
Example

After Round 2: \((A, B, C, D, E, F, G, H) = (2, 3, 3, 2, 2, 2, 0, 0)\)
Example

After Round 3: \((A, B, C, D, E, F, G, H) = (2, 2, 2, 2, 2, 0, 0, 0)\)
Example

- After Round 4: \((A, B, C, D, E, F, G, H) = (1, 1, 2, 2, 0, 0, 0, 0)\)
Example

After Round 5: \((A, B, C, D, E, F, G, H) = (0, 1, 1, 0, 0, 0, 0, 0)\)
Example

- After Round 6: \((A, B, C, D, E, F, G, H) = (0, 0, 0, 0, 0, 0, 0, 0)\)
The realizing graph: \((A, B, C, D, E, F, G, H) = (4, 4, 3, 2, 2, 2, 2, 1)\)
The Two Realizations Are Not Isomorphic