Week 4

Lecture 7 & 8*

4.1 Vertex degrees and Counting

Definition 4.1. The **degree sequence** of a graph is the list of vertex degrees, usually written in nonincreasing order, as $d_1, d_2 \geq ... \geq d_n$

How to check whether a given degree sequence is a valid or not? For graphs which allows multiple edges and self loops sufficient condition is to check whether sum of degrees is even or not. If the sum is even then sequence is valid otherwise not. But a degree sequence with sum as a even number is not a sufficient condition for simple graphs.

Definition 4.2. A **graphic sequence** is a list of nonnegative numbers that is the degree sequence of some simple graph. A simple graph with degree sequence d "realizes" d.

Havel-Hakimi Algorithm

Havel-Hakimi is an algorithm that answer the question whether a given sequence of non-negative integers is can form a degree sequence with corresponding simple graph.
The steps of the algorthms:
Input: A list of non negative integers
Output: True if list represent the degree sequence for a simple graph else false

1. Sort the list in descending order.
    - 1.1 If any element in list is negative
        - 1.1.1 return false
    - 1.2 If all element in list are zero
        - 1.2.1 return true
2. Remove first element of the list, let it be n.
3. If remaining element in list is less then n.

*Lecturer: Dr. Anand Mishra. Scribe: Yogesh Kumar.

   3.1 return false

4. Else decrease next n elements by one
5. repeat step 1.

Example 1:
Input: 5, 5, 5, 4, 2, 1, 1, 1
Step 1: 5, 5, 5, 4, 2, 1, 1, 1
Step 2: 5, 5, 4, 2, 1, 1, 1
Step 3: 5 < 7
Step 4: 4, 4, 3, 1, 0, 1, 1
Step 5: Repeat step 1

Step 1: 4, 4, 3, 1, 1, 1, 0
Step 2: 4, 3, 1, 1, 1, 0
Step 3: 4 < 6
Step 4: 3, 2, 0, 0, 1, 0
Step 5: Repeat step 1

Step 1: 3, 2, 1, 0, 0, 0
Step 2: 2, 1, 0, 0
Step 3: 3 < 4
Step 4: 1, 0, -1, 0, 0, 0
Step 5: Repeat step1

Step 1: 1, 0, 0, 0, 0, -1
Step 1.1: One element is negative
Step 1.1.1: Return False
Output: False

Example 2:
Input: 5, 5, 4, 3, 2, 2, 2, 1
Step 1: 5, 5, 4, 3, 2, 2, 2, 1
Step 2: 5, 4, 3, 2, 2, 2, 1
Step 3: 5 < 7
Step 4: 4, 3, 2, 1, 1, 2, 1
Step 5: Repeat step 1

Step 1: 4, 3, 2, 2, 1, 1, 1
Step 2: 3, 2, 2, 1, 1, 1
Step 3: 4 < 6
Step 4: 2, 1, 1, 0, 1, 1
Step 5: Repeat step 1

Step 1: 2,1,1,1,1,0
Step 2: 1,1,1,1,0
Step 3: $2 < 6$
Step 4: 0,0,1,1,0
Step 5: Repeat step 1

Step 1: 1,1,0,0,0
Step 2: 1,0,0,0
Step 3: $1 < 4$
Step 4: 0,0,0,0
Step 5: Repeat step 1

Step 1: 0,0,0,0
Step 1.2: All element are zero
step 1.2.1 return true
Output: True

Q. Prove or disprove: If u and v are the only vertices of odd degree in a graph G , then G contains a u, v path.
Proof by contradiction: Let there are two component of graph G, G_1 and G_2. Where, $u \epsilon G_1$ and $v \epsilon G_2$. Now $G1$ has only one vertices with odd degree. Then sum of degrees in G_1 will be odd. And according to hand shaking lemma it is not possible.
Hence there must a path between u and v.

Q. Let l, m, and n be nonnegative integers with $l + m = n$. Find necessary and sufficient conditions on l, m, n such that there exists a connected simple n-vertex graph with l vertices of even degree and m vertices of odd degree.
Solution: For $n \geq 1$ such that $l + m = n$ there exist a exists a connected simple n-vertex graph with l vertices of even degree and m vertices of odd degree if and only if m is even except $(m, n, p) = (2, 0, 2)$. Necessary condition, For every graph we have an even number of vertices of odd degree, and the only simple connected graph with two vertices has both degrees odd. So m must be even. Sufficient condition, let assume a cycle of length l add m vertices of degree one with a common neighbor on the cycle. That vertex of the cycle has even degree because m is even.

Q. Determine the maximum number of edges in a bipartite subgrap of P_n, of C_n, and of K_n.
Solution For a path of n vertices i.e. P_n, maximum number of edges will be $n - 1$ as P_n is itself a bipartite subgraph of itself. For cycle graph with n vertices i.e. C_n, if n is even then C_n is a bipartite graph so there will we n number of maximum edges. If n is odd then we have to remove an edge to make it bipartite hence number of maximum edges will be $n - 1$. For a complete graph i.e. K_n the largest possible bipartite graph is $K_{\lfloor n/2 \rfloor, \lceil n/2 \rceil}$ with $\lfloor n^2/4 \rfloor$

edges.

4.2 Definitions

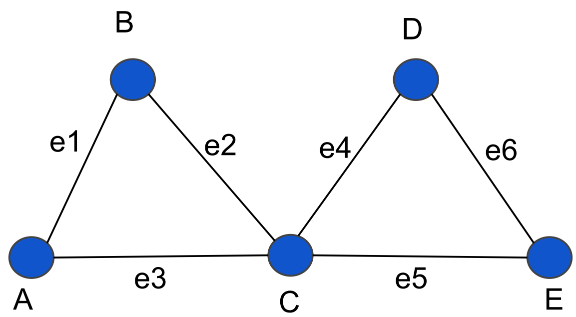

Figure 4.1: A Graph with five nodes

Definition 4.3. A **Walk** is an ordered sequence of edges and vertices. In walk edges and vertices can repeat. E.g. in Figure 4.1 $B, e2, C, e5, E, e6, D, e4, C, e3, A$ represents a walk.

Definition 4.4. A **path** is a walk where vertices can not repeat. E.g. in Figure 4.1 $A, e1, B, e2, C, e4, E$ represents a path.

Definition 4.5. A **trail** is an open walk where edges can not repeat, but vertices can repeat. E.g. in Figure 4.1 $B, e2, C, e5, E, e6, D, e4, C, e3, A$ represents a trail.

Definition 4.6. A **cycle** is a closed walk where all vertices except end-point vertices are distinct. E.g. in Figure 4.1 $A, e1, B, e2, C, e3, A$ represents a cycle.

Definition 4.7. A **circuit** is a closed walk where all edges are distinct. E.g. in Figure 4.1 $A, e3, C, e5, E, e6, D, e4, C, e2, B, e1, A$ represents a circuit.

4.3 Directed Graphs

Definition 4.8. A **directed graph or digraph** G is a triple consisting of a vertex set $V(G)$, an edge set $E(G)$, and a function assigning each edge an ordered pair of vertices. The first vertex of the ordered pair is the tail of the edge, and the second is the head; together, they are the endpoints. We say that an edge is an edge from its tail to its head.

E.g. in Figure 4.2 for edge $1 \to 3$, 1 is head and 2 is tail. Also 1 is predecessor of 3 and 3 is successor of 1. Similar to undirected graph a cycle in directed graph is a close path expect last edge repetition.

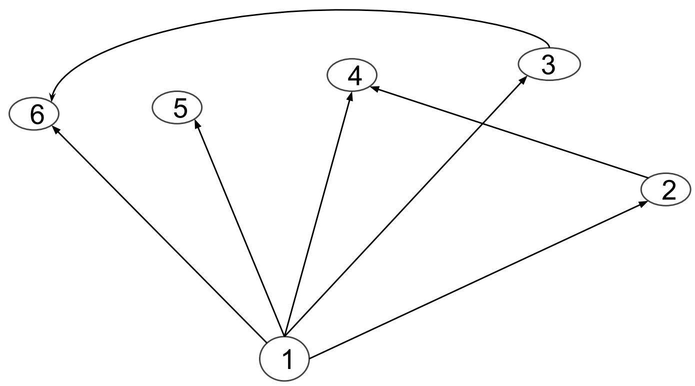

Figure 4.2: A Directed Graph with five nodes

Definition 4.9. The **underlying graph** of a digraph D is the graph G obtained by treating the edges of D as unordered pairs; the vertex set and edge set remain the same, and the endpoints of an edge are the same in G as in D, but in G they become an unordered pair.

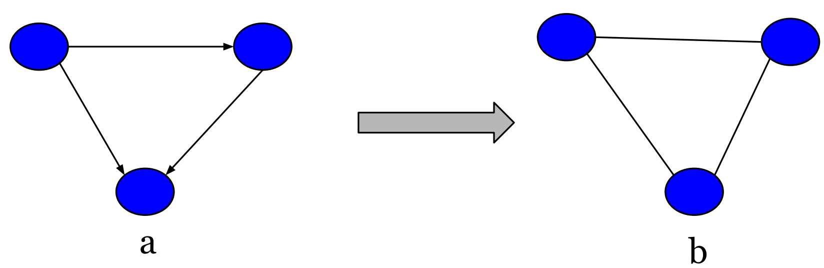

Figure 4.3: a, b represent directed and it's underlying graph respectively

In Figure 4.3 part a, represent the directed graph while part b represent it's underlying graph.

Representation of Directed Graphs

Adjacency Matrix

Adjacency matrix of graph represented by Figure 4.4 is:

$$A(G) = \begin{array}{c} \\ w \\ x \\ y \\ z \end{array} \begin{array}{c} \begin{array}{cccc} w & x & y & z \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \end{array} \tag{4.1}$$

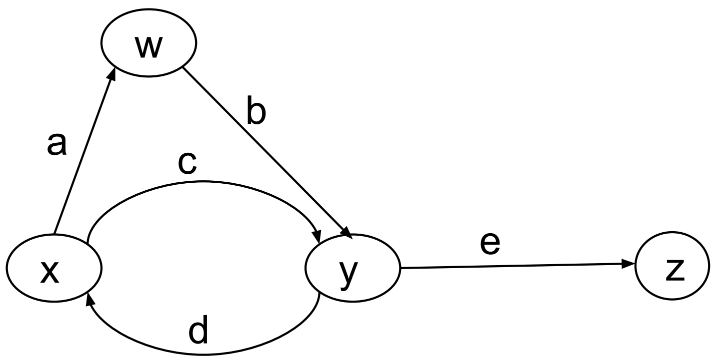

Figure 4.4: a, b represent directed and it's underlying graph respectively

here $A_{x,y}$ represent an edge between vertices x and y.

Incident Matrix

Incident matrix of graph represented by Figure 4.4 is:

$$M(G) = \begin{array}{c} \\ w \\ x \\ y \\ z \end{array} \begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{pmatrix} -1 & +1 & 0 & 0 & 0 \\ +1 & 0 & +1 & -1 & 0 \\ 0 & -1 & -1 & +1 & +1 \\ 0 & 0 & 0 & 0 & -1 \end{pmatrix} \end{array} \tag{4.2}$$

where $-1, +1, 0$ represent incoming, outgoing and no edge respectively.

Weakly and Strongly Connected Graphs

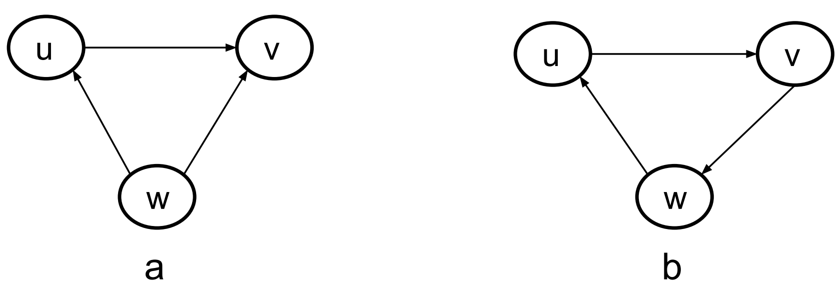

Figure 4.5: a, b represent weakly and strongly connected graph respectively

Definition 4.10. A digraph is **weakly** connected if it's underlying graph is connected. In Figure 4.5 a represent a weakly connected graph

Definition 4.11. A digraph is **strongly** connected if for each pair of vertices there is a path. In Figure 4.5 b represent a strongly connected graph.

Kernel of a Digraph

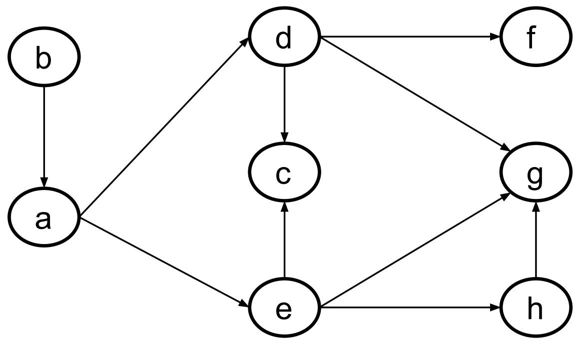

Figure 4.6: A Digraph

Definition 4.12. A **kernel** in the digraph D is a set $S \subseteq V(D)$ such that S induces no edges and every vertex outside S has a successor in S. For e.g. in Figure 4.6 $K = a, f, g, c$ is a kernel.