Basics of Graph

Adopted from slides by B. Prabhakaran

Topics Covered

- Definitions
- Types
- Terminology
- Representation
- Sub-graphs
- Connectivity
- Trees

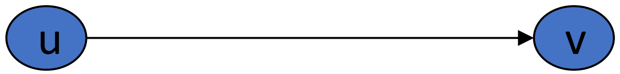
Definitions - Graph

A generalization of the simple concept of a set of dots, links, edges or arcs.

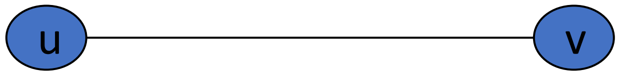
Representation: Graph G =(V, E) consists set of vertices denoted by V, or by V(G) and set of edges E, or E(G)

Definitions – Edge Type

Directed: Ordered pair of vertices. Represented as (u, v) directed from vertex u to v.

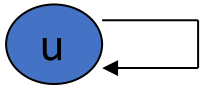


Undirected: Unordered pair of vertices. Represented as {u, v}. Disregards any sense of direction and treats both end vertices interchangeably.



Definitions – Edge Type

- **Loop:** A loop is an edge whose endpoints are equal i.e., an edge joining a vertex to itself is called a loop. Represented as (u, u) = (u)

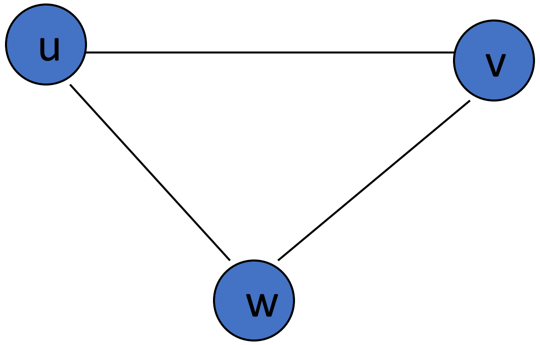


- **Multiple Edges:** Two or more edges joining the same pair of vertices.

Definitions – Graph Type

Simple (Undirected) Graph: consists of V, a nonempty set of vertices, and E, a set of unordered pairs of distinct elements of V called edges (undirected)

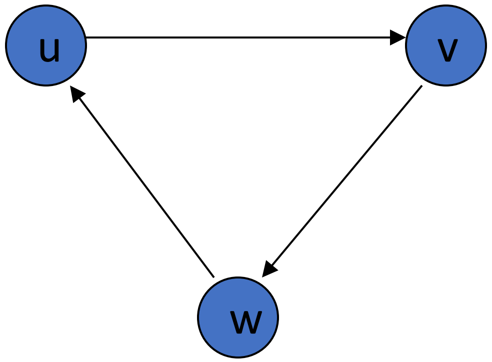
Representation Example: G(V, E), V = {u, v, w}, E = {{u, v}, {v, w}, {u, w}}



Definitions – Graph Type

Directed Graph: G(V, E), set of vertices V, and set of Edges E, that are ordered pair of elements of V (directed edges)

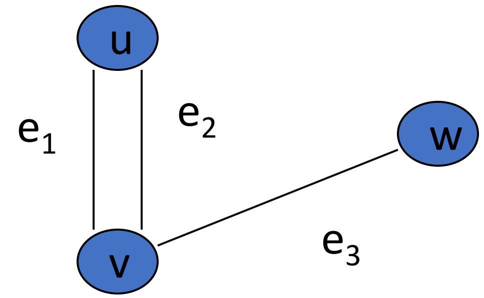
Representation Example: G(V, E), V = {u, v, w}, E = {(u, v), (v, w), (w, u)}



Definitions – Graph Type

Multigraph: G(V,E), consists of set of vertices V, set of Edges E and a function f from E to {{u, v}| u, v V, u ≠ v}. The edges e1 and e2 are called multiple or parallel edges if f (e1) = f (e2).

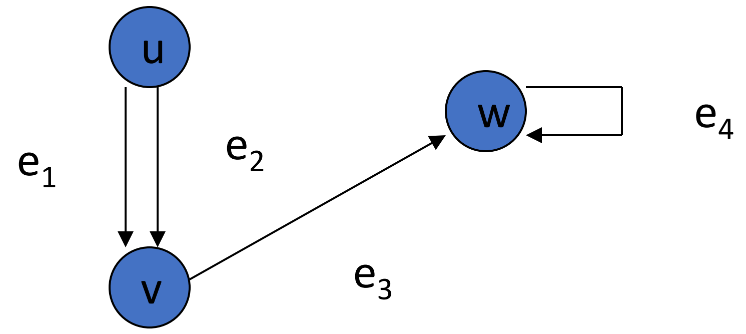
Representation Example: V = {u, v, w}, E = {e_1, e_2, e_3}



Definitions – Graph Type

Directed Multigraph: G(V,E), consists of set of vertices V, set of Edges E and a function f from E to {{u, v}| u, v V}. The edges e1 and e2 are multiple edges if f(e1) = f(e2)

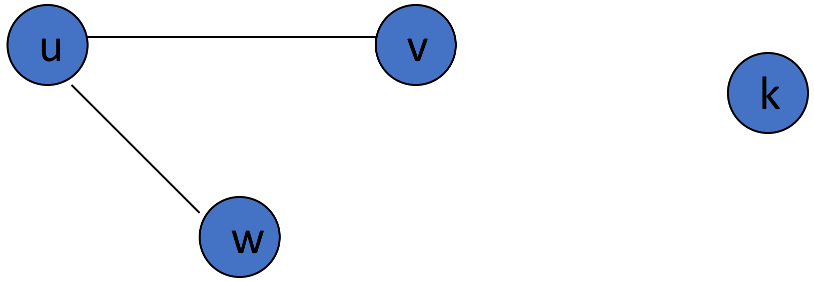
Representation Example: V = {u, v, w}, E = {e_1, e_2, e_3, e_4}



Terminology – Undirected graphs

- u and v are **adjacent** if {u, v} is an edge, e is called **incident** with u and v. u and v are called **endpoints** of {u, v}
- **Degree of Vertex (deg (v)):** the number of edges incident on a vertex.
- **Pendant Vertex:** deg (v) =1
- **Isolated Vertex:** deg (k) = 0

Representation Example: For V = {u, v, w} , E = { {u, w}, {u, v} }, deg (u) = 2, deg (v) = 1, deg (w) = 1, deg (k) = 0, w and v are pendant , k is isolated

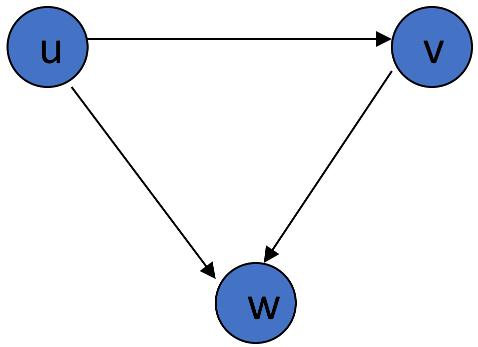


Terminology – Directed graphs

- For the edge (u, v), u is **adjacent to** v OR v is **adjacent from** u, u – **Initial vertex**, v – **Terminal vertex**
- **In-degree (deg^- (u)):** number of edges for which u is terminal vertex
- **Out-degree (deg^+ (u)):** number of edges for which u is initial vertex

Note: A loop contributes 1 to both in-degree and out-degree

Representation Example: For V = {u, v, w} , E = { (u, w), (v, w), (u, v) }, deg^- (u) = 0, deg^+ (u) = 2, deg^- (v) = 1, deg^+ (v) = 1, and deg^- (w) = 2, deg^+ (u) = 0



Theorems: Undirected Graphs

Theorem 1

The Handshaking theorem:

$$2e = \sum_{v \in V} \deg(v)$$

(why?) Every edge connects 2 vertices

Theorems: Undirected Graphs

Theorem 2:

An undirected graph has even number of vertices with odd degree

$\Pr oof\ V1$ is the set of even degree vertices and V2 refers to odd degree vertices

$$2e = \sum_{v \in V} \deg(v) = \sum_{u \in V_1} \deg(u) + \sum_{v \in V_2} \deg(v)$$

$\Rightarrow$ deg (v) is even for $v \in V_1$,

$\Rightarrow$ The first term in the right hand side of the last inequality is even.

$\Rightarrow$ The sum of the last two terms on the right hand side of

the last inequality is even since sum is 2e.

Hence second term is also even

$\Rightarrow$ second term $\sum_{v \in V_2} \deg(u) = even$

Theorems: Directed Graphs

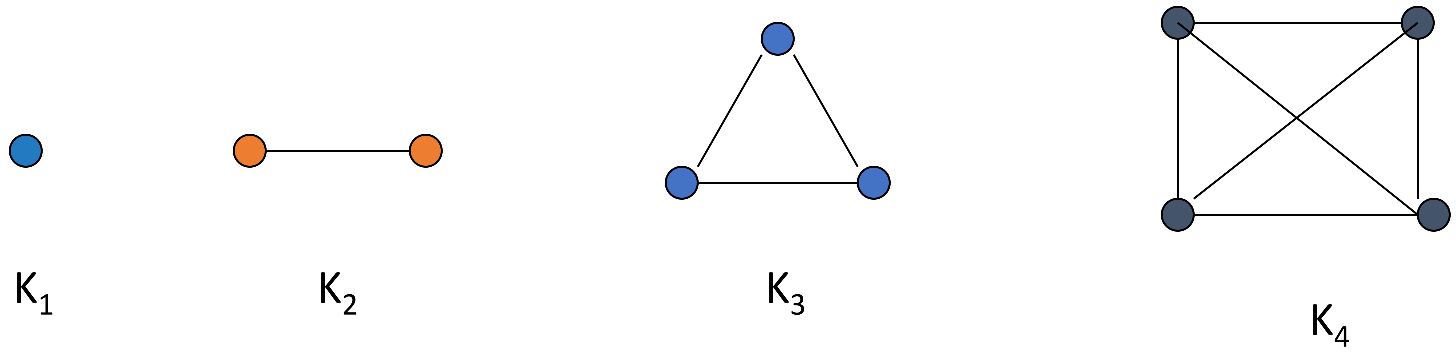
<u>Theorem 3</u>

$$\sum \deg^{+}(u) = \sum \deg^{-}(u) = |\mathrm{E}|$$

Simple Graphs – Special Cases

- **Complete graph:** K_n, is the simple graph that contains exactly one edge between each pair of distinct vertices.

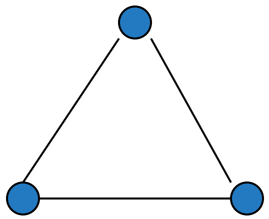
  Representation Example: K_1, K_2, K_3, K_4



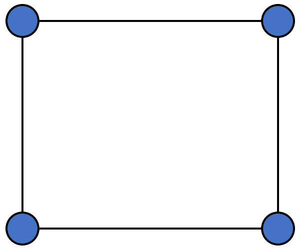
Simple Graphs – Special Cases

- **Cycle:** C_n, $n \geq 3$ consists of n vertices $v_1, v_2, v_3 \dots v_n$ and edges $\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\} \dots \{v_{n-1}, v_n\}, \{v_n, v_1\}$

  Representation Example: C_3, C_4



C_3



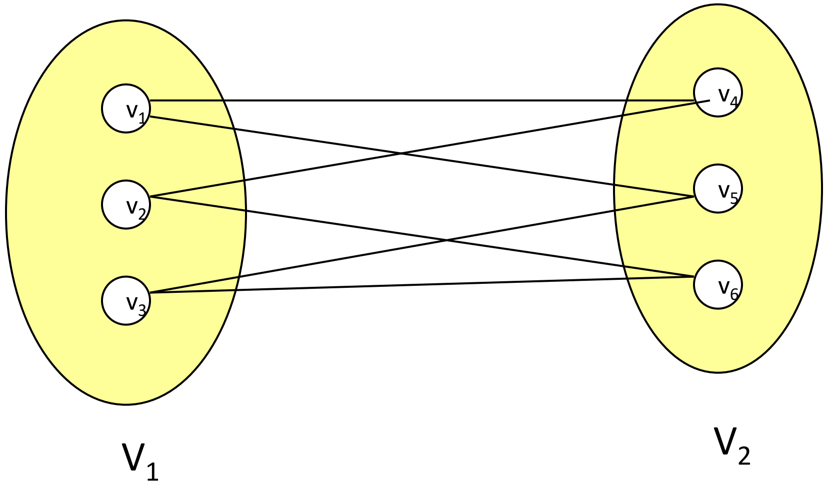
C_4

Bipartite Graphs

- In a simple graph G, if V can be partitioned into two disjoint sets V_1 and V_2 such that every edge in the graph connects a vertex in V_1 and a vertex V_2 (so that no edge in G connects either two vertices in V_1 or two vertices in V_2)

  Application example: Representing Relations

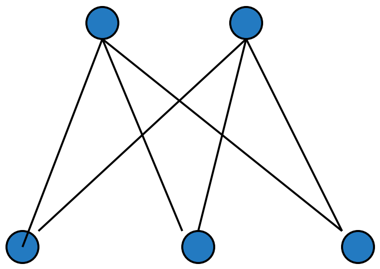
  Representation example: $V_1 = \{v_1, v_2, v_3\}$ and $V_2 = \{v_4, v_5, v_6\}$,



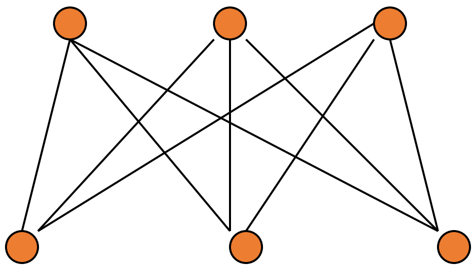
Complete Bipartite Graphs

- $K_{m,n}$ is the graph that has its vertex set portioned into two subsets of m and n vertices, respectively There is an edge between two vertices if and only if one vertex is in the first subset and the other vertex is in the second subset.

  Representation example: $K_{2,3,}$ $K_{3,3}$



$K_{2,3}$



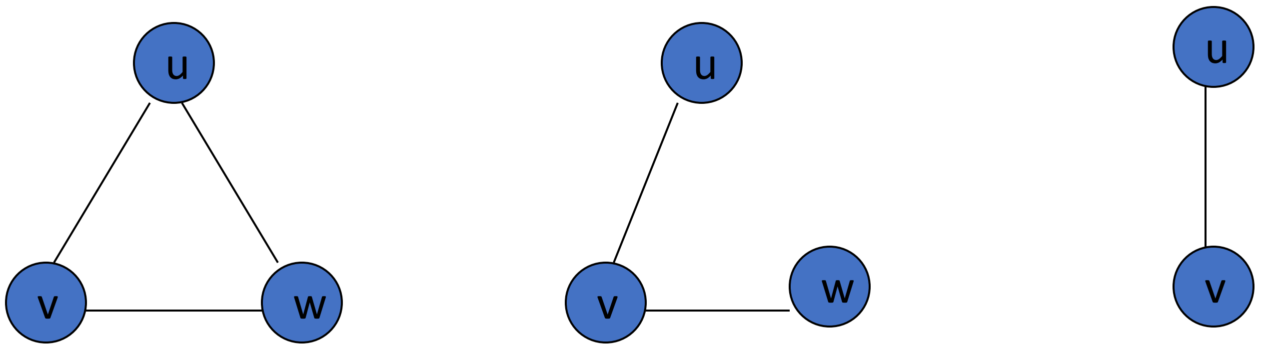
$K_{3,3}$

Subgraphs

- A subgraph of a graph G = (V, E) is a graph H =(V', E') where V' is a subset of V and E' is a subset of E

  Application example: solving sub-problems within a graph

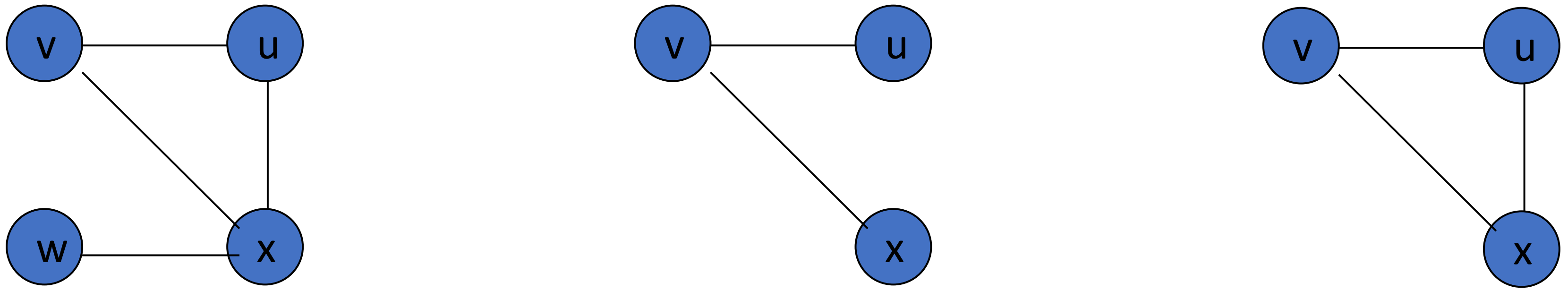
  Representation example: V = {u, v, w}, E = ({u, v}, {v, w}, {w, u}}, H_1 , H_2



G H_1 H_2

Induced Subgraph

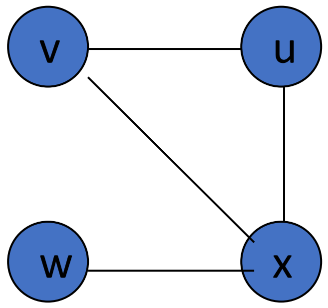
- Graph H=(V’,E’) is an induced subgraph of G=(V,E) if E’ consists of all edges in E with endpoints in V’.



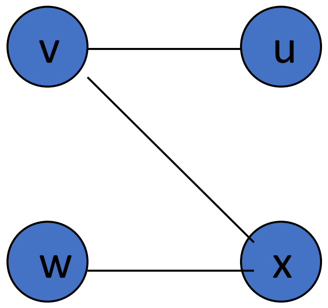
graph subgraph induced subgraph

Spanning Subgraph

- Graph H=(V',E') is a spanning subgraph of G=(V,E) if V'=V.



graph



spanning graph

Representation

- **Incidence (Matrix):** Most useful when information about edges is more desirable than information about vertices.

- **Adjacency (Matrix/List):** Most useful when information about the vertices is more desirable than information about the edges. These two representations are also most popular since information about the vertices is often more desirable than edges in most applications

Representation- Incidence Matrix

- G = (V, E) be an undirected graph. Suppose that $v_1, v_2, v_3, \ldots, v_n$ are the vertices and $e_1, e_2, \ldots, e_m$ are the edges of G. Then the incidence matrix with respect to this ordering of V and E is the $n \times m$ matrix M = [m_{ij}], where

$$m_{ij} = \begin{cases} 1 & \text{when edge } e_j \text{ is incident with } v_i \\ 0 & \text{otherwise} \end{cases}$$

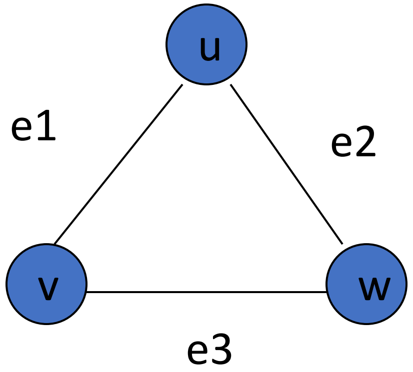
Can also be used to represent :

Multiple edges: by using columns with identical entries, since these edges are incident with the same pair of vertices

Loops: by using a column with exactly one entry equal to 1, corresponding to the vertex that is incident with the loop

Representation- Incidence Matrix

- Representation Example: G = (V, E)



	e_1	e_2	e_3
v	1	0	1
u	1	1	0
w	0	1	1

Representation- Adjacency Matrix

- There is an N x N matrix, where |V| = N , the Adjacenct Matrix (NxN) $A = [a_{ij}]$

For undirected graph

$$a_{ij} = \begin{cases} 1 & \text{if } \{v_i, v_j\} \text{ is an edge of G} \\ 0 & \text{otherwise} \end{cases}$$

For directed graph

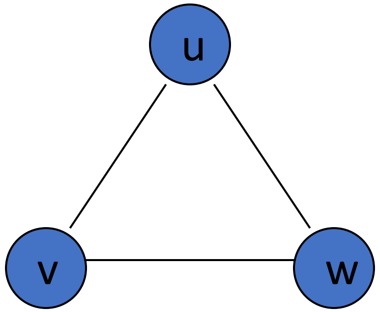
$$a_{ij} = \begin{cases} 1 & \text{if } (v_i, v_j) \text{ is an edge of G} \\ 0 & \text{otherwise} \end{cases}$$

Representation- Adjacency Matrix

- Adjacency is chosen on the ordering of vertices. Hence, there as are as many as n! such matrices.
- The adjacency matrix of simple graphs are symmetric ($a_{ij} = a_{ji}$)
- When there are relatively few edges in the graph the adjacency matrix is a **sparse matrix**
- Directed Multigraphs can be represented by using aij = number of edges from v_i to v_j

Representation- Adjacency Matrix

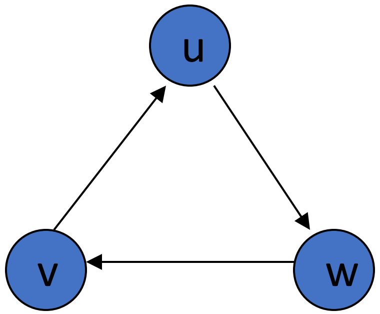
- Example: Undirected Graph G (V, E)



	v	u	w
v	0	1	1
u	1	0	1
w	1	1	0

Representation- Adjacency Matrix

- Example: directed Graph G (V, E)

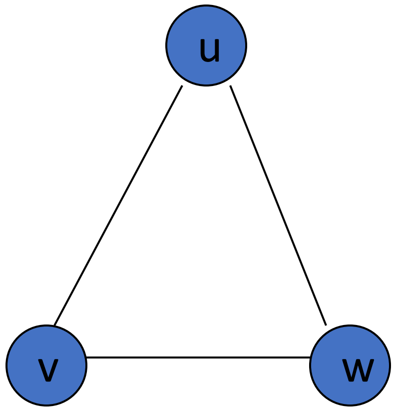


	v	u	w
v	0	1	0
u	0	0	1
w	1	0	0

Representation- Adjacency List

Each node (vertex) has a list of which nodes (vertex) it is adjacent

Example: undirectd graph G (V, E)



node	Adjacency List
u	v , w
v	w, u
w	u , v

Graph - Isomorphism

- G1 = (V1, E2) and G2 = (V2, E2) are isomorphic if:
- There is a one-to-one functional mapping f from V1 to V2 with the property that
  - a and b are adjacent in G1 if and only if f (a) and f (b) are adjacent in G2, for all a and b in V1.
- Function f is called isomorphism

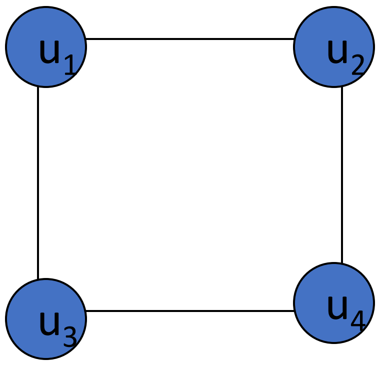
Application Example:

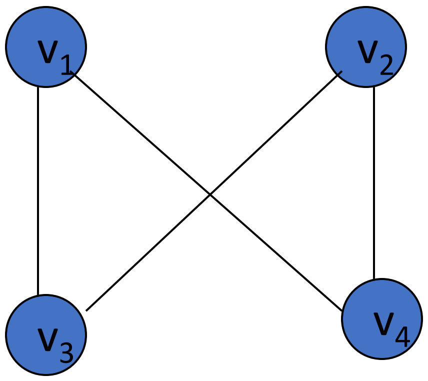
In chemistry, to find if two compounds have the same structure

Graph - Isomorphism

Representation example: G1 = (V1, E1) , G2 = (V2, E2)

$f(u_1) = v_1$, $f(u_2) = v_4$, $f(u_3) = v_3$, $f(u_4) = v_2$,





Connectivity

- Basic Idea: In a Graph Reachability among vertices by traversing the edges

  Application Example:

  - In a city-to-city road-network, if one city can be reached from another city.

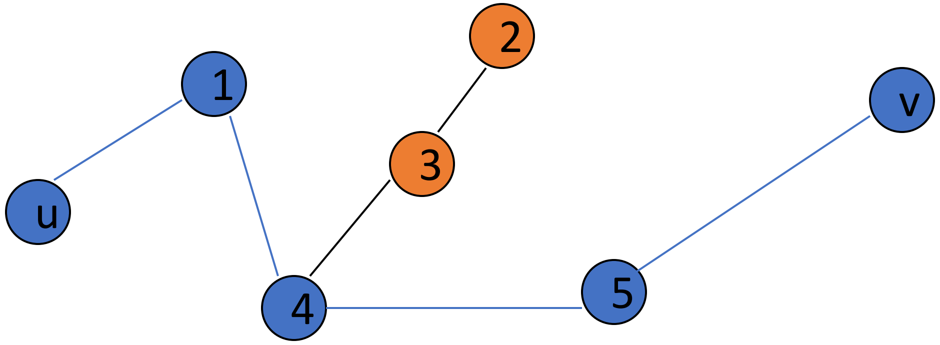
  - Problems if determining whether a message can be sent between two computer using intermediate links

  - Efficiently planning routes for data delivery in the Internet

Connectivity – Path

A **Path** is a sequence of edges that begins at a vertex of a graph and travels along edges of the graph, always connecting pairs of adjacent vertices.

Representation example: G = (V, E), Path P represented, from u to v is {{u, 1}, {1, 4}, {4, 5}, {5, v}}



Connectivity – Path

Definition for Directed Graphs

A **Path** of length n (> 0) from u to v in G is a sequence of n edges $e_1, e_2, e_3, \dots, e_n$ of G such that f $(e_1) = (x_o, x_1)$, f $(e_2) = (x_1, x_2)$, …, f $(e_n) = (x_{n-1}, x_n)$, where $x_0 = u$ and $x_n = v$. A path is said to pass through $x_0, x_1, \dots, x_n$ or traverse $e_1, e_2, e_3, \dots, e_n$

Circuit/Cycle: u = v, length of path > 0

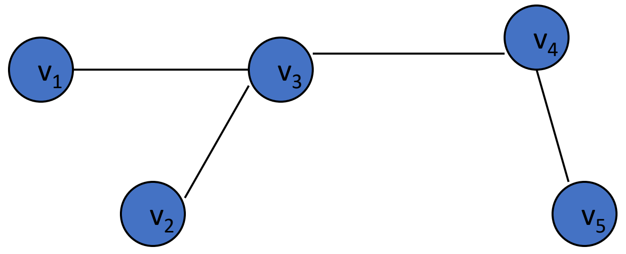
Simple Path: does not contain an edge more than once

Connectivity – Connectedness

Undirected Graph

An undirected graph is connected if there exists is a simple path between every pair of vertices

Representation Example: G (V, E) is connected since for V = {v_1, v_2, v_3, v_4, v_5}, there exists a path between {v_i, v_j}, 1 ≤ i, j≤ 5

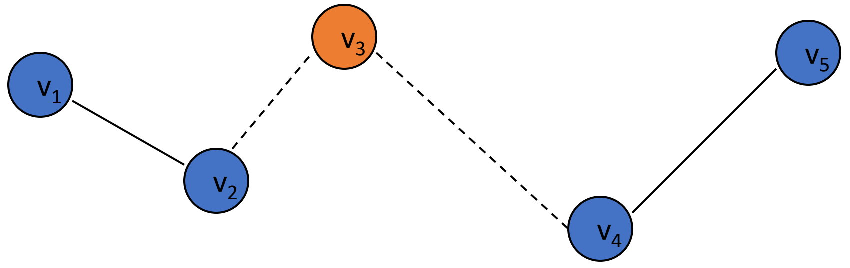


Connectivity – Connectedness

Undirected Graph

- **Articulation Point (Cut vertex):** removal of a vertex produces a subgraph with more connected components than in the original graph. The removal of a cut vertex from a connected graph produces a graph that is not connected
- **Cut Edge:** An edge whose removal produces a subgraph with more connected components than in the original graph.

  Representation example: G (V, E), v_3 is the articulation point or edge $\{v_2, v_3\}$, the number of connected components is 2 (> 1)



Counting Paths

- **Theorem:** Let G be a graph with adjacency matrix A with respect to the ordering v_1, v_2, …, v_n (with directed or undirected edges, with multiple edges and loops allowed). The number of different paths of length r from vi to vj, where r is a positive integer, equals the $(i, j)^{th}$ entry of (adjacency matrix) $A^{r.}$

  Proof: By Mathematical Induction.

  Base Case: For the case N = 1, a_{ij} =1 implies that there is a path of length 1. This is true since this corresponds to an edge between two vertices.

  We assume that theorem is true for N = r and prove the same for N = r +1. Assume that the $(i, j)^{th}$ entry of A^r is the number of different paths of length r from v_i to v_j. By induction hypothesis, b_{ik} is the number of paths of length r from v_i to v_k.

Counting Paths

<u>Case r +1:</u> In $A^{r+1} = A^r . A$,

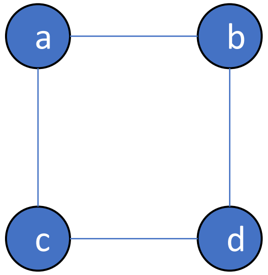
The $(i, j)^{th}$ entry in A^{r+1} , $b_{i1}a_{1j} + b_{i2}\, a_{2j} + \ldots + b_{in}\, a_{nj}$

where b_{ik} is the $(i, j)^{th}$ entry of A^r.

By induction hypothesis, b_{ik} is the number of paths of length r from v_i to v_k.

The $(i, j)^{th}$ entry in A^{r+1} corresponds to the length between i and j and the length is r+1. This path is made up of length r from v_i to v_k and of length from v_k to vj. By product rule for counting, the number of such paths is $b_{ik*}\, a_{kj}$ The result is $b_{i1}a_{1j} + b_{i2}\, a_{2j} + \ldots + b_{in}\, a_{nj\,,}$the desired result.

Counting Paths

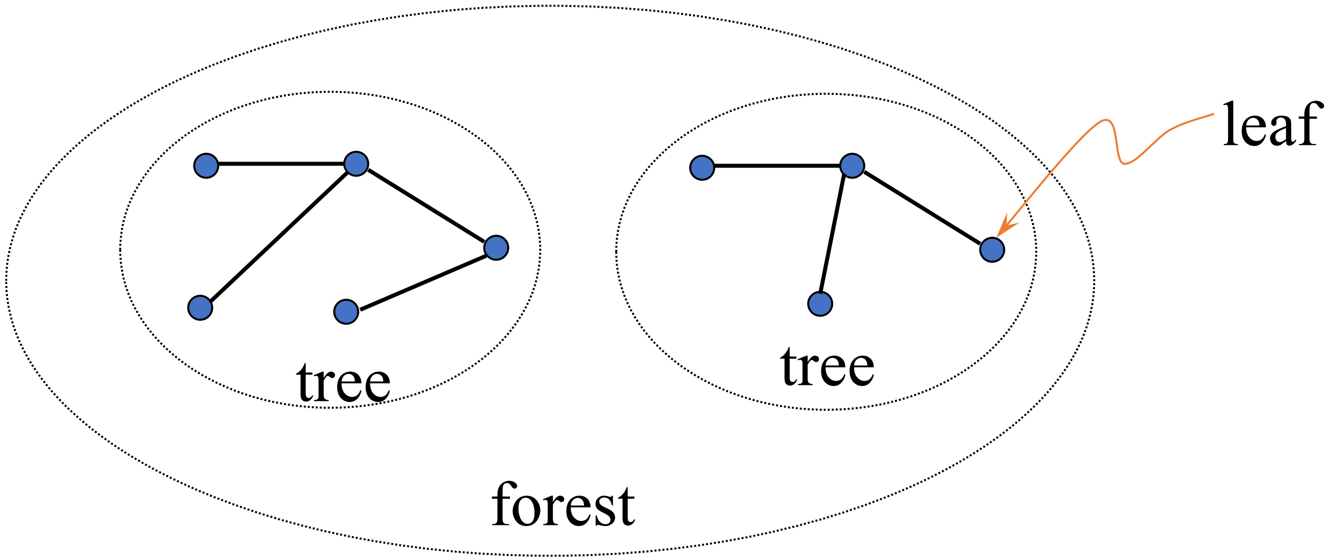


$$A = \begin{matrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{matrix} \qquad A^4 = \begin{matrix} 8 & 0 & 0 & 8 \\ 0 & 8 & 8 & 0 \\ 0 & 8 & 8 & 0 \\ 8 & 0 & 0 & 8 \end{matrix}$$

Number of paths of length 4 from a to d is (1,4)th entry of $A^4 = 8$.

Trees and Forests

- A graph with no cycle is *acyclic*
- A *forest* is an acyclic graph
- A *tree* is a connected acyclic graph
- A *leaf* (or *pendant vertex*) is a vertex of degree 1



Trees

- Theorem: For an n-vertex graph G (with n>=1), the following are equivalent
  - G is a tree
  - G is connected and has no cycles
  - G is connected and has n-1 edges
  - G has n-1 edges and no cycles
  - For $u, v \in V(G)$, G has exactly one u, v-path

Root

- Sometimes it is convenient to consider one vertex of a tree as special. Such a vertex is then called the **root** of this tree.
- With the root it defines a partial ordering on all vertices. We can think of this ordering as expressing the ‘height’.