Lecture 01
Graph Theory: Introduction

Dr. Rajiv Misra
Associate Professor
Dept. of Computer Science & Engg.
Indian Institute of Technology Patna
rajivm@iitp.ac.in

Preface

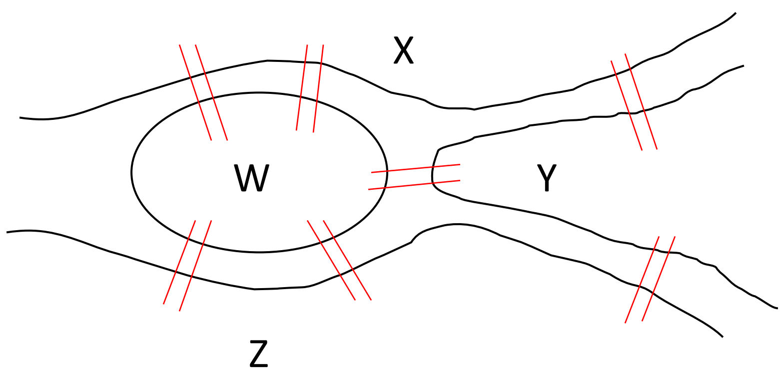
Content of this Lecture:

- Graph theory is a delightful playground for the exploration of proof techniques in discrete mathematics, and its results have applications in many areas of the computing, social and natural sciences.

- In this lecture, we will discuss a brief introduction to the fundamentals of graph theory and how graphs can be used to model the real world problems.

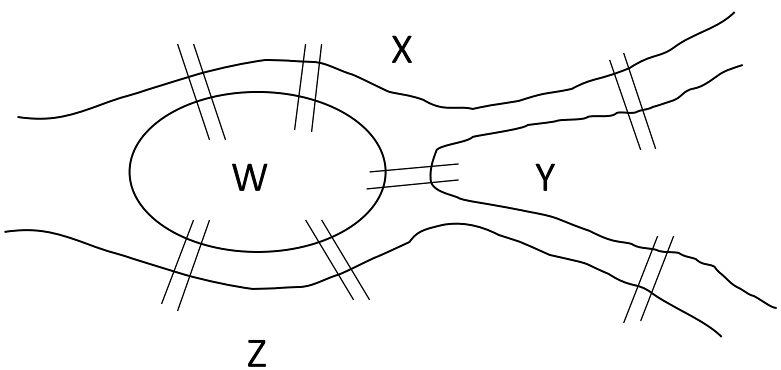
The Königsberg Bridge Problem (1736)

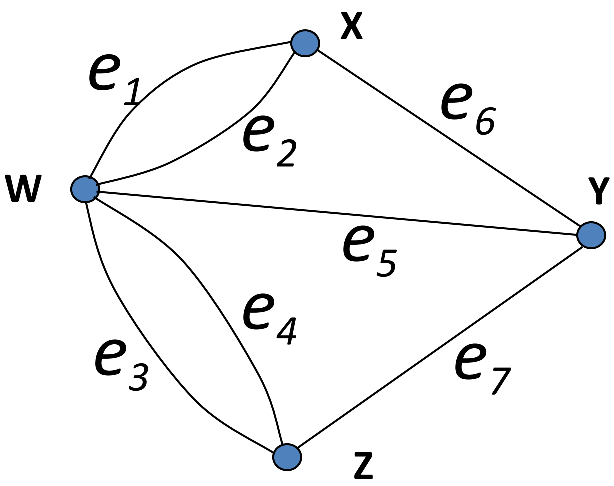
- Königsberg is a city on the Pregel river in Prussia
- The city occupied two islands plus areas on both banks
- Problem:
  - Whether they could leave home, cross every bridge exactly once, and return home.



General Model

- A ***vertex*** : a region
- An ***edge*** : a path(bridge) between two regions





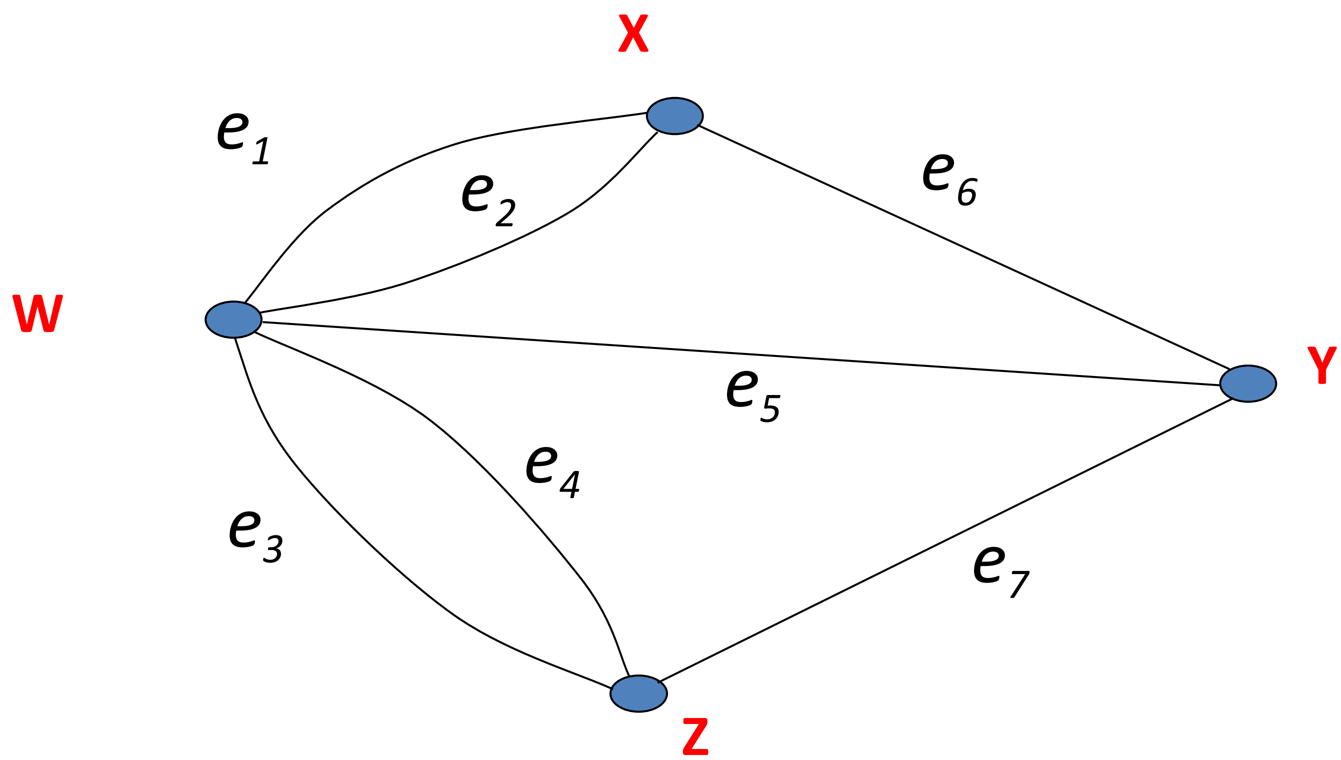
What is a Graph?

- *G = (V, E)*
- *V = nodes (or vertices).*
- *E = edges (or arcs) between pairs of nodes.*
- Captures pairwise relationship between objects.
- Graph size parameters:

The ***order*** of a graph ***G***, written ***n (G)***, is the number of vertices in ***G***. i.e. $n(G) = |V|$

The ***size*** of a graph ***G***, written ***e (G)***, is the number of edges in ***G***. i.e. $e(G) = |E|$

Example



$V = \{x, y, w, z\}$

$E = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7\}$

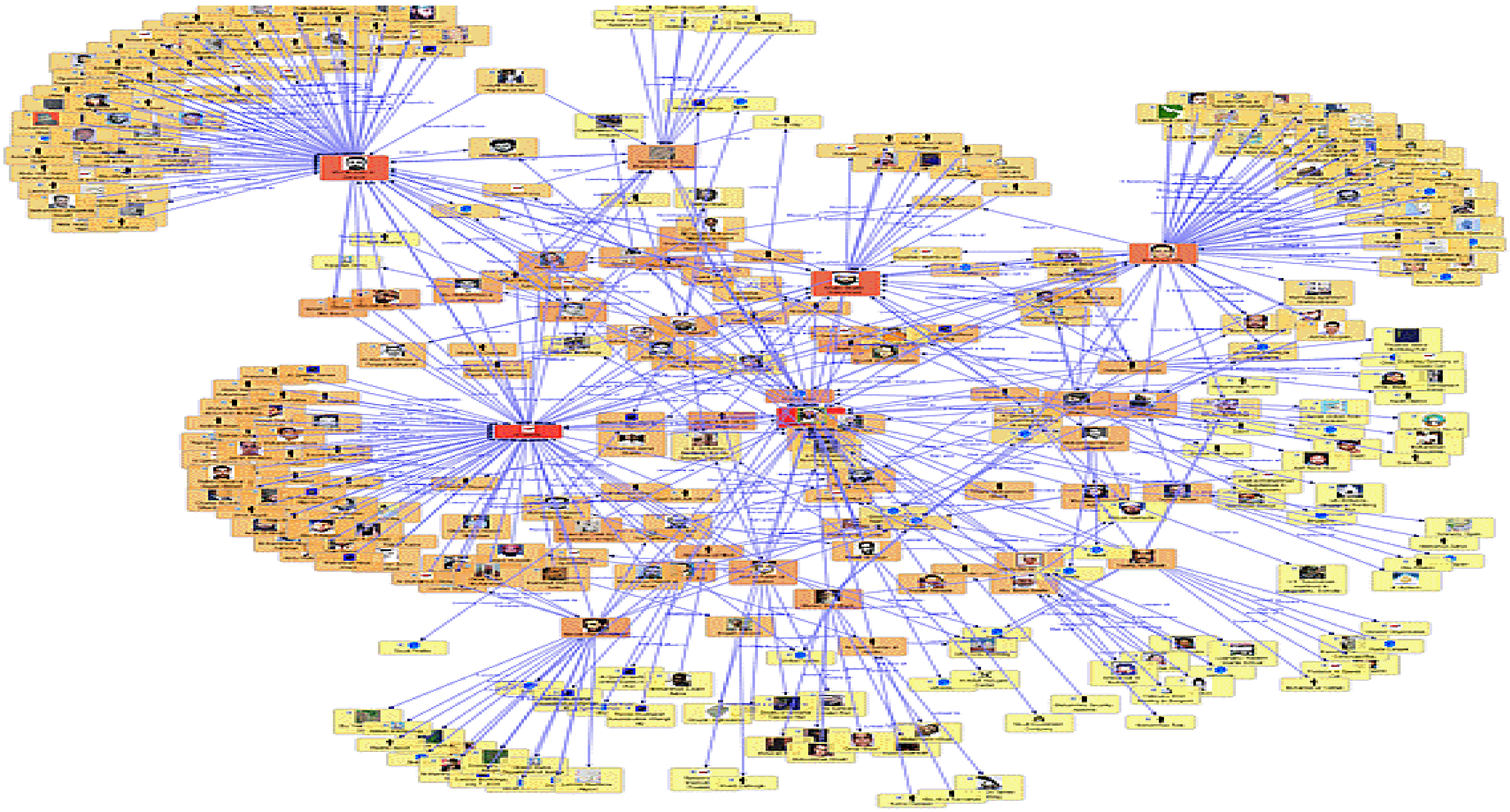
$n(G) = 4$, $e(G) = 7$

Graphs used in Applications

graph	node	edge
communication	telephone, computer	fiber optic cable
circuit	gate, register, processor	wire
mechanical	joint	rod, beam, spring
financial	stock, currency	transactions
transportation	street intersection, airport	highway, airway route
internet	class C network	connection
game	board position	legal move
social relationship	person, actor	friendship, movie cast
neural network	neuron	synapse
protein network	protein	protein-protein interaction
molecule	atom	bond

Social Network: Graph

- Node: Person, Edge: Link

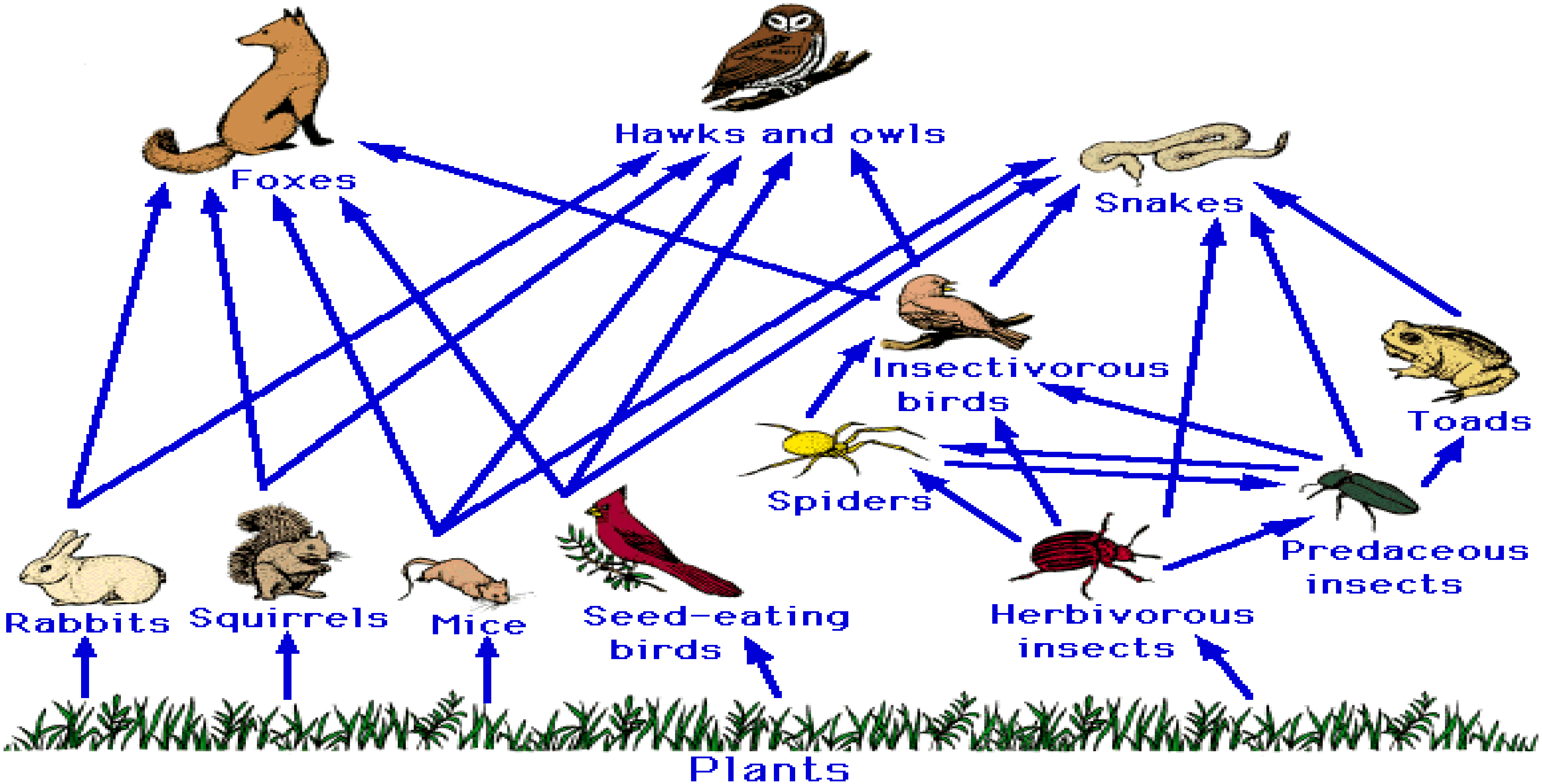


Road Network: Graph

- Node: intersection; Edge: one-way street.

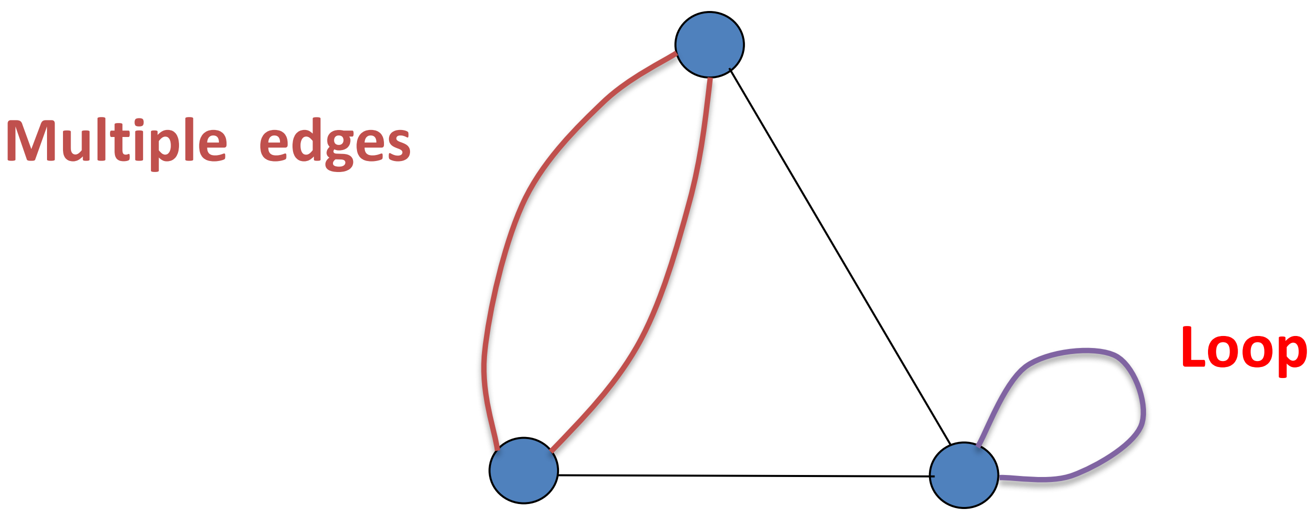
Ecological Food Web: Graph

- Food web graph.
- Node: species, Edge: from prey to predator.



Loop, Multiple edges

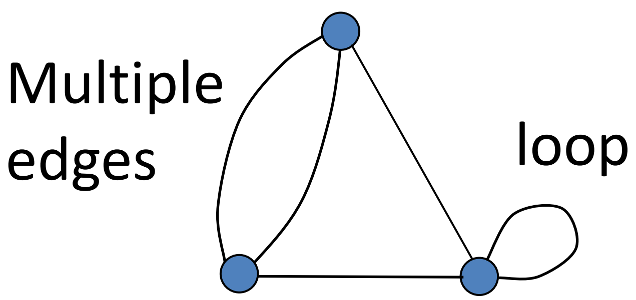
- ***Loop* :** An edge whose endpoints are equal
- ***Multiple edges* :** Edges have the same pair of endpoints
- Example:



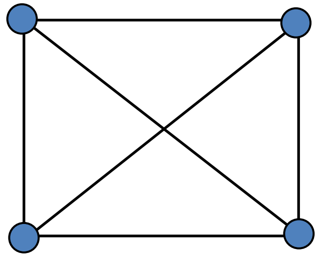
Simple Graph

- ***Simple graph* :** A graph has no loops or multiple edges

- Example:



It is **not simple**.



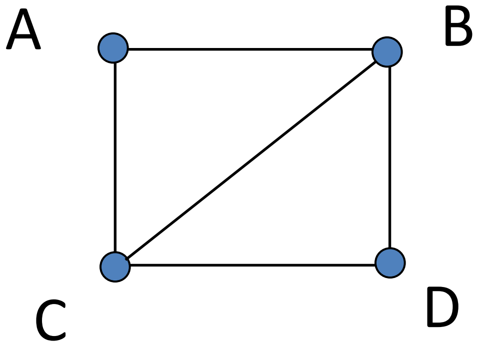
It is a **simple** graph.

Adjacent, neighbors

- Two vertices are ***adjacent*** and are ***neighbors*** if they are the endpoints of an edge.

- Example:

- A and B are adjacent
- A and D are not adjacent



Finite Graph, Null Graph

- ***Finite graph*** : The graph whose vertex set and edge set are finite

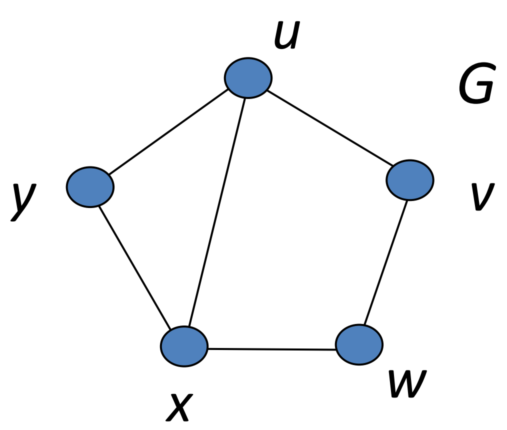
- ***Null graph*** **:** The graph whose vertex set and edges are empty

Clique and Independent set

- A ***Clique*** in a graph: a set of pairwise adjacent vertices (a complete subgraph)
- An ***independent set*** in a graph: a set of pairwise nonadjacent vertices

Example:

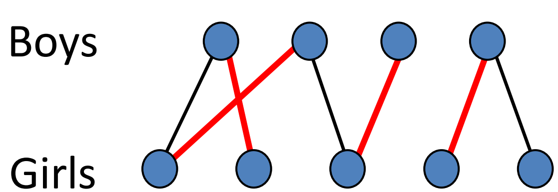
- {x, y, u} is a clique in G
- {u, w} is an independent set

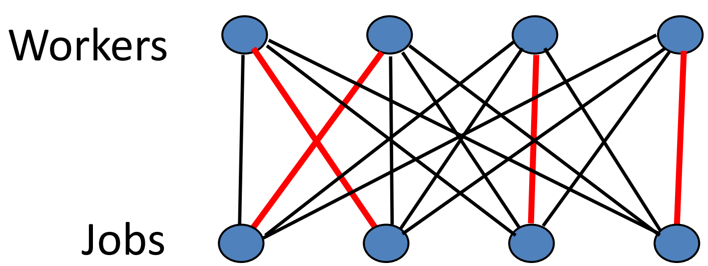


Bipartite Graphs

- A graph ***G*** is ***bipartite*** if ***V*(*G*)** is the union of two disjoint independent sets called ***partite sets of G***
- Also: The vertices can be partitioned into two sets such that each set is independent

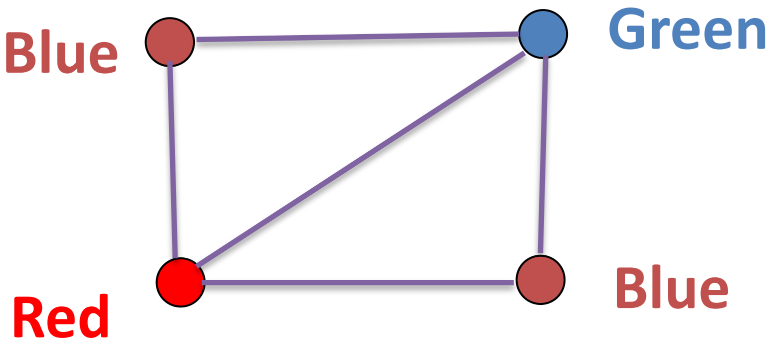
- Example:
  (i) Matching Problem, (ii) Job Assignment Problem





Chromatic Number

- The ***chromatic number*** of a graph ***G***, written $\chi(G)$, is the ***minimum number of colors*** needed to label the vertices so that adjacent vertices receive different colors



$\chi(G)$ **= 3**

Maps and Coloring

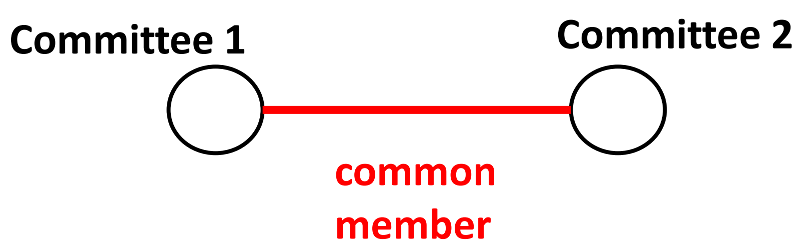
- A ***map*** is a partition of the plane into connected regions
- Can we color the regions of every map using at most **four colors** so that neighboring regions have different colors?

- Map Coloring $\rightarrow$ graph coloring
  - A region $\rightarrow$ A vertex
  - Adjacency $\rightarrow$ An edge

Scheduling and Graph Coloring

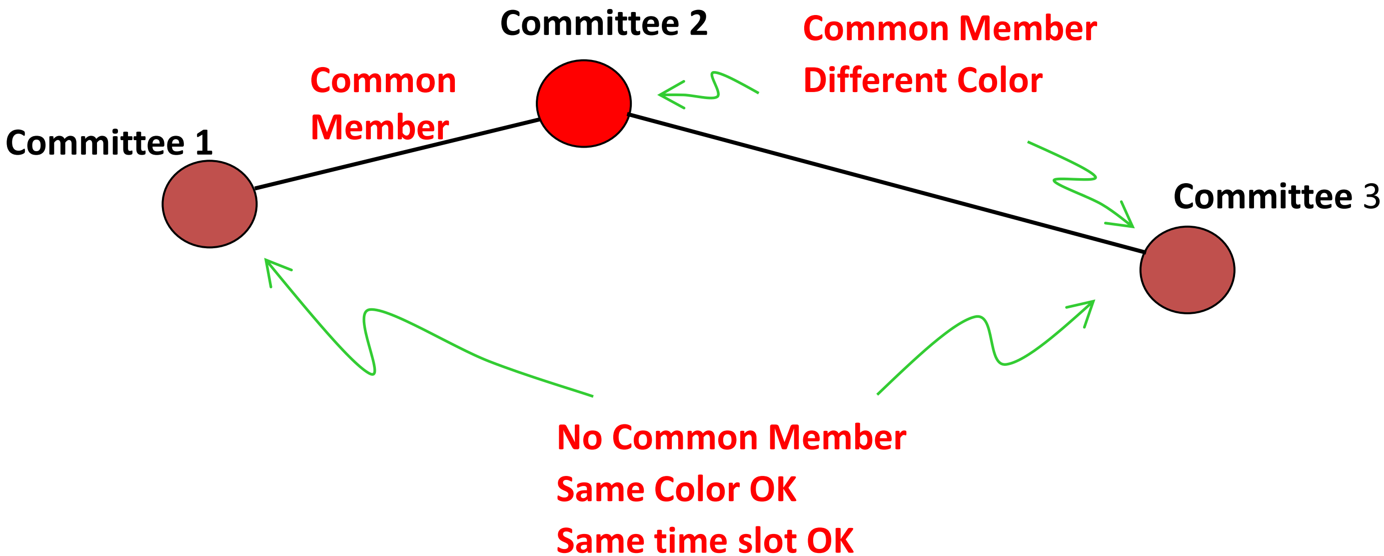
Model:

- One committee being represented by a vertex
- An edge between two vertices if two corresponding committees have common member
- Two adjacent vertices can not receive the same color
- Two committees can not hold meetings at the same time if two committees have common member



Scheduling and Graph Coloring

- Scheduling problem is equivalent to graph coloring problem



Path, Cycle, Walk and Trails

- ***Path* :** a sequence of **distinct** vertices such that two consecutive vertices are adjacent

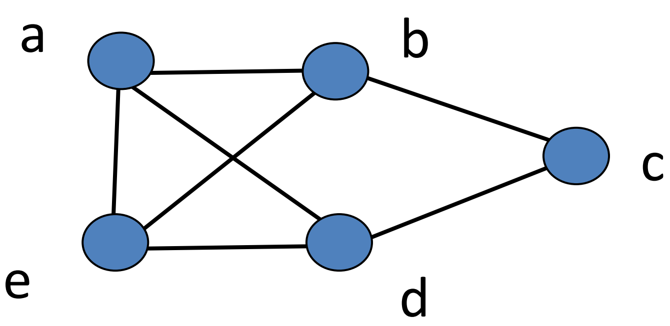
- ***Cycle* :** a closed Path

- ***Walk* :** A ***walk*** is a list of vertices and edges $v_0, e_1, v_1, \ldots., e_k, v_k$ such that, for $1 \leq i \leq k$, the edge e_i has endpoints v_{i-1} and v_i.

- ***Trail* :** A ***trail*** is a walk with **no repeated edge**.

Example

- (a, d, c, b, e) is a path
- (a, b, e, d, c, b, e, d) is not a path; it is a walk
- (*a, d, c, b, e, a*) is a cycle



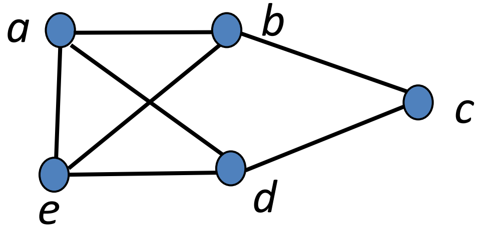
Subgraphs

- A ***subgraph*** of a graph $\boldsymbol{G}$ is a graph $\boldsymbol{H}$ such that:
- $\boldsymbol{V(H) \subseteq V(G)}$ and $\boldsymbol{E(H) \subseteq E(G)}$ and
- The assignment of endpoints to edges in $\boldsymbol{H}$ is the same as in $\boldsymbol{G}$.

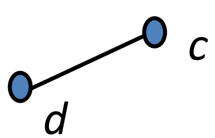
Example

- Example: $\boldsymbol{H_1}$, $\boldsymbol{H_2}$, and $\boldsymbol{H_3}$ are subgraphs of $\boldsymbol{G}$

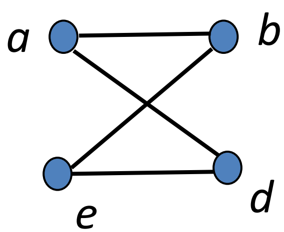
G



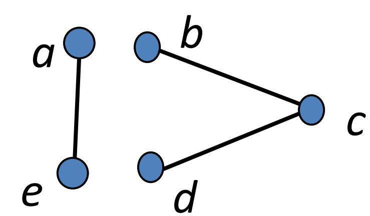
H₁



$\boldsymbol{H_2}$



$\boldsymbol{H_3}$



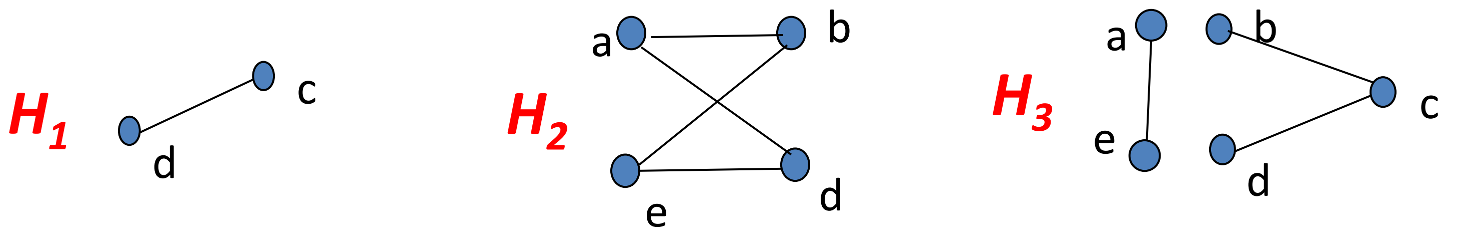
Connected and Disconnected

- ***Connected*** : There exists at least one path between two vertices
- ***Disconnected*** : Otherwise

- Example:

H_1 and H_2 are connected

H_3 is disconnected

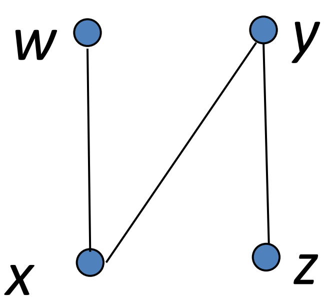


Isomorphism

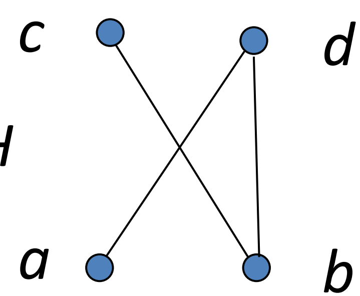
- An ***isomorphism*** from a simple graph *G* to a simple graph *H* is a bijection $f: V(G) \to V(H)$ such that $uv \in E(G)$ if and only if $f(u)f(v) \in E(H)$
- We say "***G is isomorphic to H***", written $G \cong H$

- *Example:*

G



H

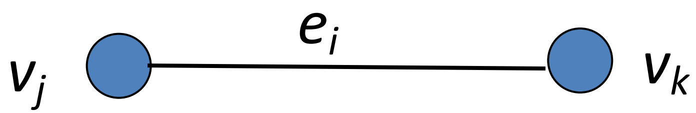


f_1:	*w*	*x*	*y*	*z*
	c	*b*	*d*	*a*

f_2:	*w*	*x*	*y*	*z*
	a	*d*	*b*	*c*

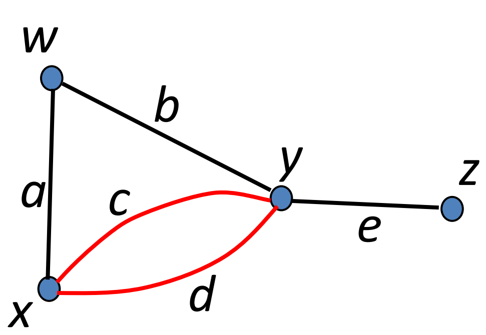
Adjacency, Incidence, and Degree

- Assume e_i is an edge whose endpoints are (v_j, v_k)
- The vertices v_j and v_k are said to be ***adjacent***
- The edge e_i is said to be ***incident upon*** v_j
- ***Degree*** of a vertex v_k is the number of edges incident upon v_k. It is denoted as $d(v_k)$



Adjacency Matrix

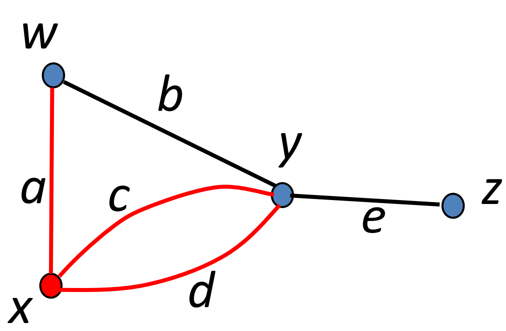
- Let $G = (V, E)$, $|V| = n$ and $|E| = m$
- The ***adjacency matrix*** of G written $A(G)$, is the n-by-n matrix in which entry $a_{i,j}$ is the number of edges in G with endpoints $\{v_i, v_j\}$.
- Example:



$$\begin{array}{c} \\ w \\ x \\ y \\ z \end{array} \begin{array}{c} \begin{array}{cccc} w & x & y & z \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 2 & 0 \\ 1 & 2 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

Incidence Matrix

- Let $G = (V, E)$, $|V| = n$ and $|E| = m$
- The ***incidence matrix*** $M(G)$ is the n-by-m matrix in which entry $m_{i,j}$ is 1 if v_i is an endpoint of e_i and otherwise is 0.
- Example:

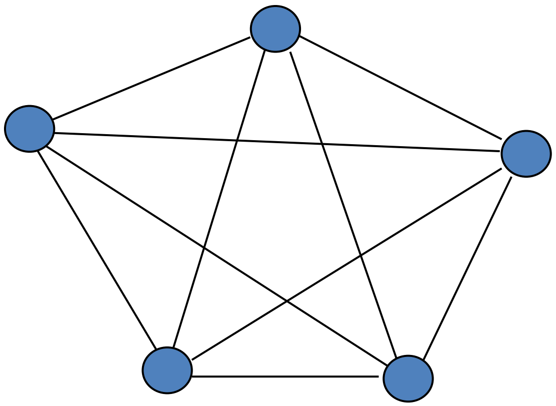


$$
\begin{array}{c c}
 & \begin{array}{ccccc} a & b & c & d & e \end{array} \\
\begin{array}{c} w \\ x \\ y \\ z \end{array} & \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}
\end{array}
$$

Complete Graph

- ***Complete Graph*** : a simple graph whose vertices are pairwise adjacent

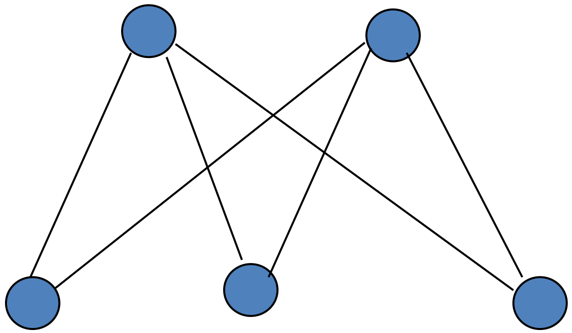
- Example



Complete Graph

Complete Bipartite Graph or Biclique

- ***Complete bipartite graph*** (biclique) is a simple bipartite graph such that two vertices are adjacent if and only if they are in different partite sets.



Complete Bipartite Graph

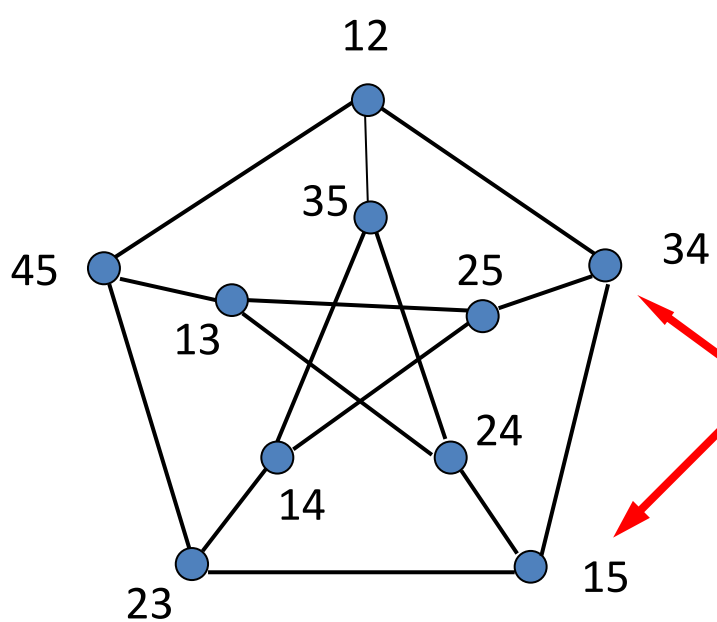
Petersen Graph

- The ***petersen graph*** is the simple graph whose vertices are the 2-element subsets of a 5-element set and whose edges are pairs of disjoint 2-element subsets

Example

- Assume: the set of 5-element be (1, 2, 3, 4, 5)
- Then, 2-element subsets:

(1,2) (1,3) (1,4) (1,5) (2,3) (2,4) (2,5) (3,4) (3,5) (4,5)

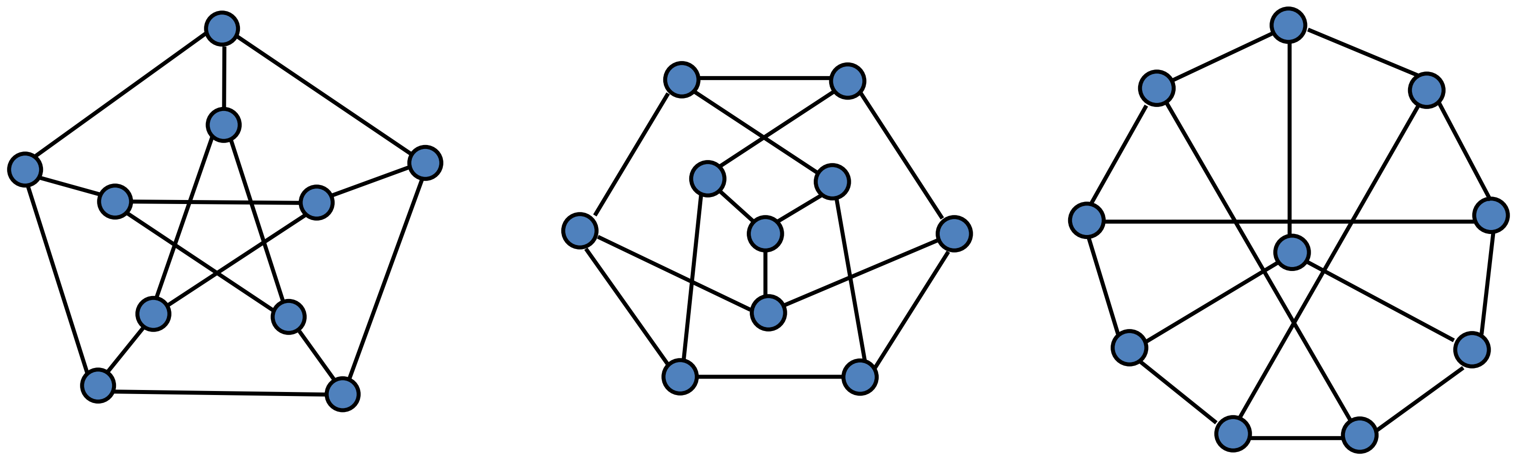


45: (4, 5)

Disjoint, so connected

Example

- Three drawings



Conclusion

- This lecture introduces the basic concepts and formal model of graph theory and how graphs can be used to model problems.
- In the field of computer science, it is mainly used to solve problems or to represent scenarios related with networks.
- In upcoming lectures, we will try to give an insight on its detailed concepts that will give a good understanding of the further details.