Advanced Graph Theory

Instructor

Arobinda Gupta (Post-midsem)

Bivas Mitra (Pre-midsem)

TA

Aman Antil

Course Directives

- Time: Thu (3:00pm-5.00pm), Friday (2.00pm-4.00pm)
- CS 107
- Webpage: ??

Tutorials

- Textbook

Introduction to Graph Theory

- -- Douglas B West

Course Outline

- **Module 1: Fundamentals of Graph Theory**
- Graphs and their representations, Types of graphs (simple, complete, bipartite, regular, etc.), Subgraphs, graph isomorphism, Cliques, independent sets, Vertex degree and counting principles
- **Module 2: Graph Traversal and Structural Properties**
- Walks, trails, paths, and cycles, Connected components, Cut vertices and cut edges, connected and disconnected graphs, maximal connected component, Graphical degree sequences
- **Module 3 : Eulerian Graphs**
- Eulerian paths and Eulerian graphs, Eulerian graph characterization , Chinese Postman Problem, Euler's Theorem , Fleury's Algorithm
- **Module 4: Hamiltonian Graphs**
- Hamiltonian paths and cycles , Sufficient conditions for Hamiltonicity, Applications
- **Module 5: Connectivity**
- Vertex and edge connectivity, (k)-connected and (k)-edge-connected graphs, Menger's Theorem, Harry graph, edge-cut,, bonds

- **Module 6: Bipartite Graphs**
- Bipartite and complete bipartite graphs, Characterization of bipartite graphs , Applications of bipartite graphs
- **Module 7: Network Flow**
- Flow networks, Residual graphs and augmenting paths, Maximum Flow – Minimum Cut Theorem, Applications of network flow
- **Module 8: Planar Graphs**
- Planar embeddings, Euler's Formula, Dual graphs,Characterization of planar graphs, Kuratowski's Theorem
- **Module 9: Matching**
- Matching in graphs, Maximum and perfect matching, Hall's Marriage Theorem, Applications of matching
- **Module 10: Graph Coloring**
- Vertex and edge coloring, Chromatic number, Coloring algorithms, Applications in scheduling, register allocation, and frequency assignment

Evaluation

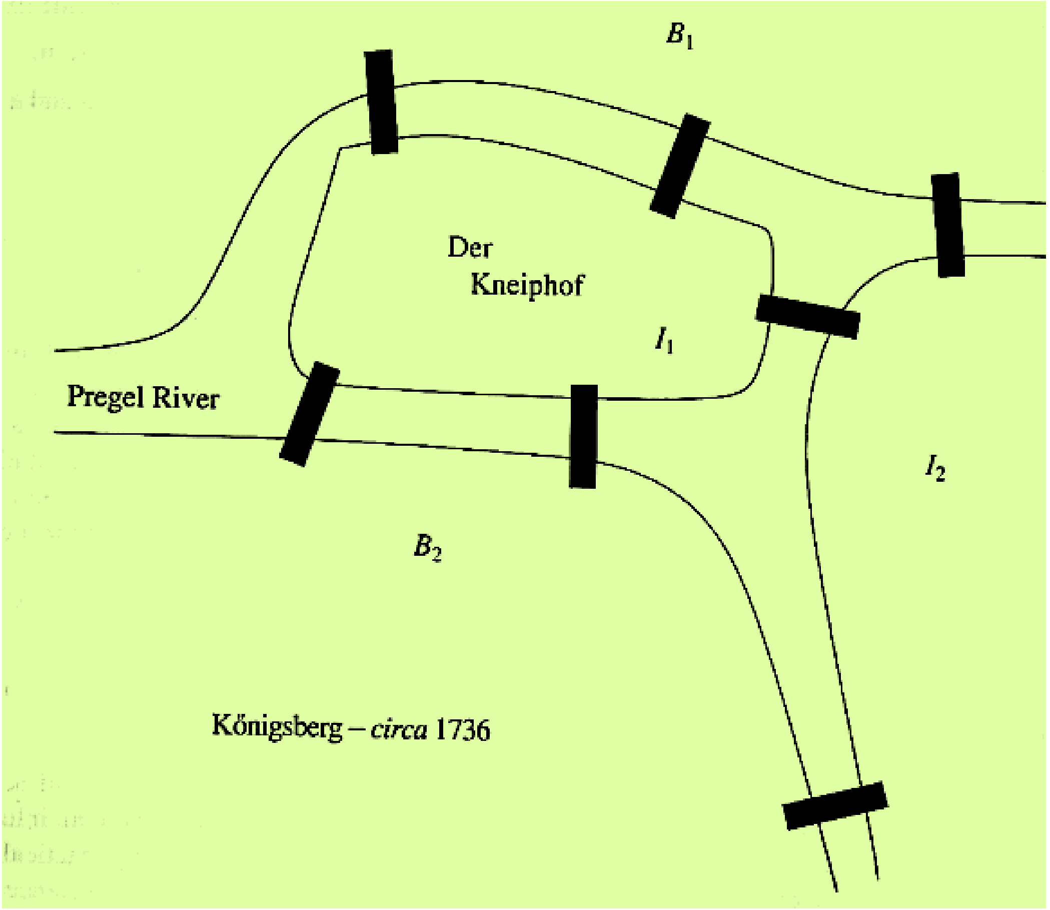
Midsem: 30
Endsem: 30
Class Test 1 (pre-midsem): 10
Class Test 2 (post-midsem): 10
Tutorial/Assignment (pre-midsem): 6
Tutorial/Assignment (post-midsem): 6
Attendance (pre-midsem): 4
Attendance (post-midsem): 4

Introduction and History

Graph theory is considered to have begun in 1736 with the publication of Euler’s solution of Kőnigsberg Bridge Problem (**Leonard Euler**, known to be the Father of Graph Theory).

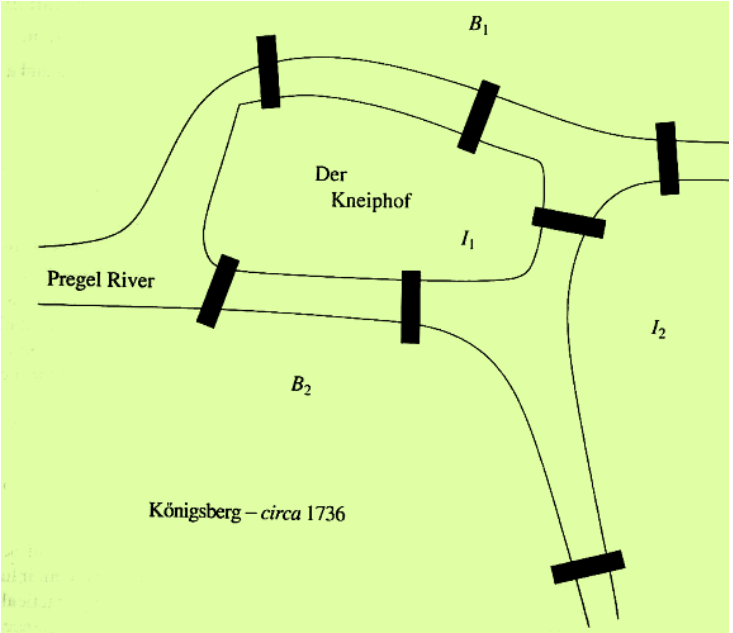
Why Study Graphs?

- **Problem 1: The Bridges of Kőnigsberg (1736 Leonard Euler)**



Is it possible to make a round trip through downtown Kőnigsberg, traversing each bridge exactly once.

First try to model this problem with a graph.



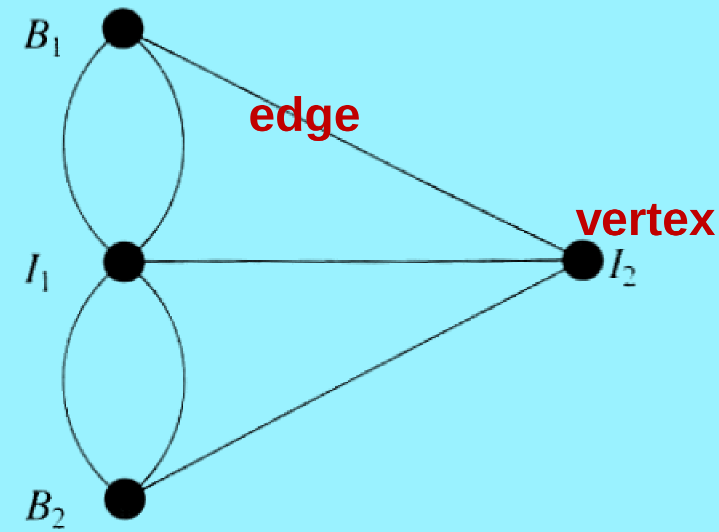
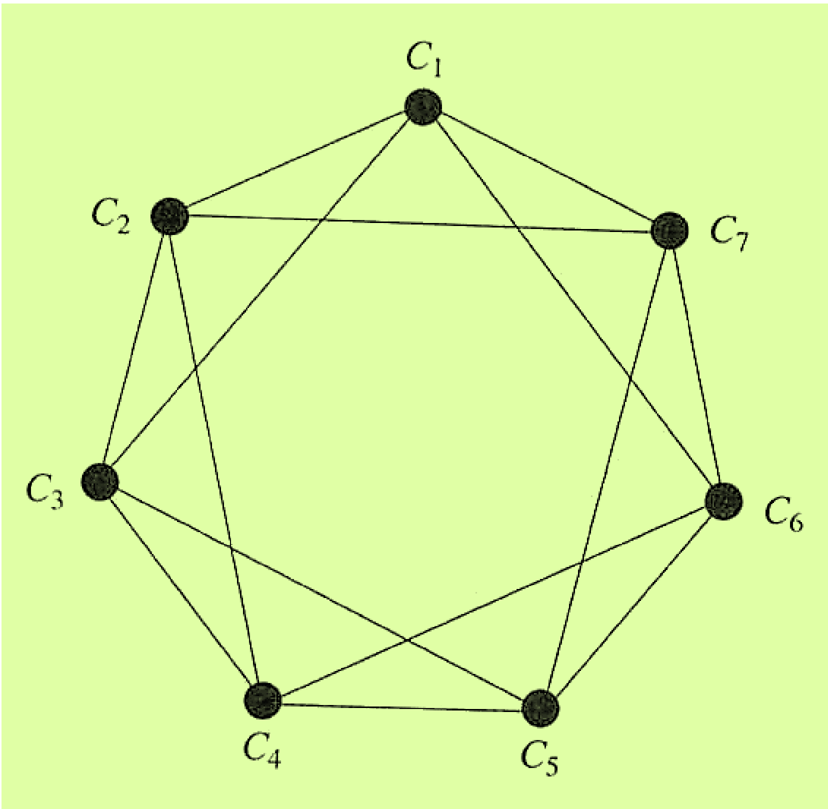


Figure 1.2: The graph of the bridges of Kőnigsberg problem.

Q: Start at any vertex, go along each edge exactly once, and end up at the starting vertex？

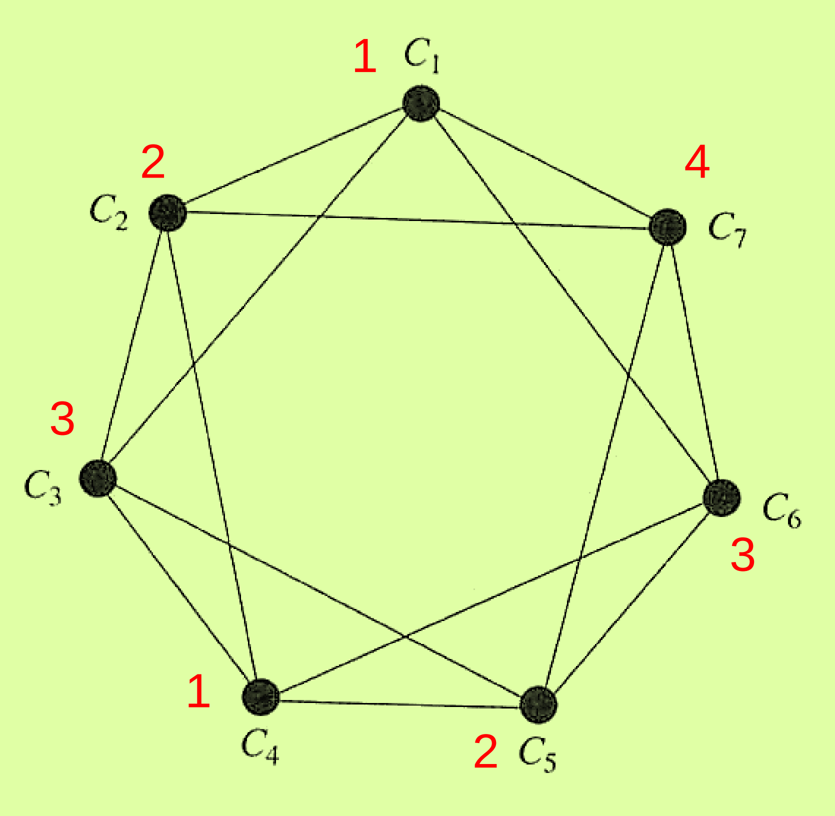
Euler proved that this problem has no solution.

Problem 2: Storing Volatile Chemicals



C_1, C_2, …, C_7 are different kinds of chemicals where an edge between C_i and C_j indicates a grave danger in storing these chemicals in the same warehouse.

Q. What is the minimum number of warehouses the factory needs in order to store its chemical products safely？



Problem: What is the minimum number of warehouses the factory needs in order to store its chemical products safely？

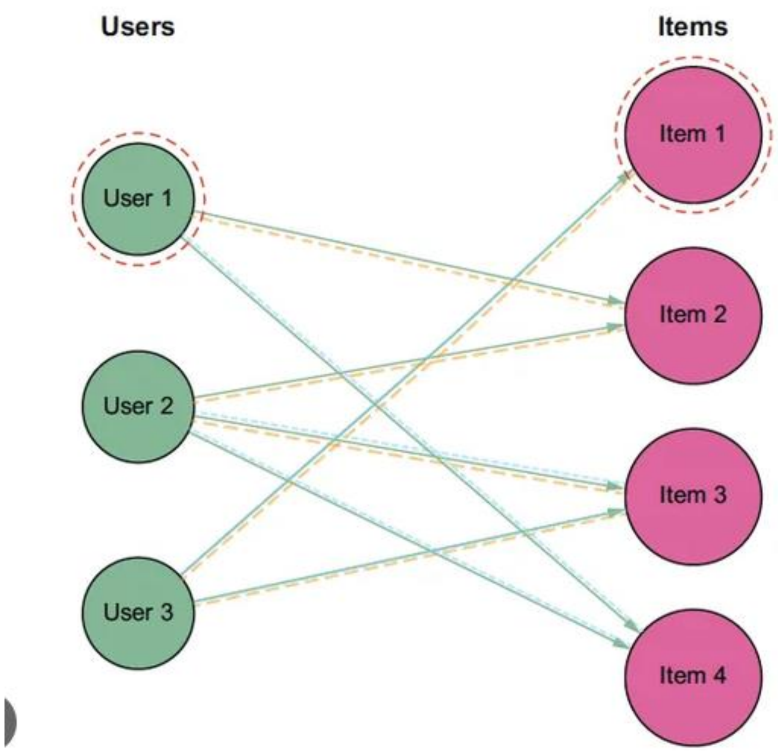
Ans: 4
(Ch8: Graph Coloring)

Graph Theory Powers Modern Technologies

Many real-world problems are graph problems.

Problem	Graph Theory Concept
Google Maps navigation	Shortest Path
Internet routing	Spanning Tree, Routing Algorithms
Facebook friend recommendations	Link Prediction
Course scheduling	Graph Coloring
Project planning	DAG & Topological Ordering
Social influence analysis	Centrality Measures
Recommendation systems	Bipartite Graphs
Web search (Google PageRank)	Directed Graphs
Protein interaction analysis	Cliques & Communities

Recommendation Algorithms



Why Learn Definitions?

- **Precise definitions eliminate ambiguity and lead to correct algorithms.**

- • Walk ≠ Path
- • Tree ≠ General Graph
- • Clique ≠ Complete Graph

Without precise definitions:

- Algorithms become incorrect.
- Proofs become invalid.
- Different people interpret problems differently.

Why Learn Theorems?

Theorems reveal fundamental properties that hold for **every graph**.

- Handshaking Lemma
- Connected graph needs at least $n-1$ edges
- Euler's Theorem
- Menger's Theorem

These results allow us to:

- Design efficient algorithms.
- Avoid unnecessary computations.
- Prove correctness of solutions.
- Predict graph properties before computation.

Why Learn Proofs?

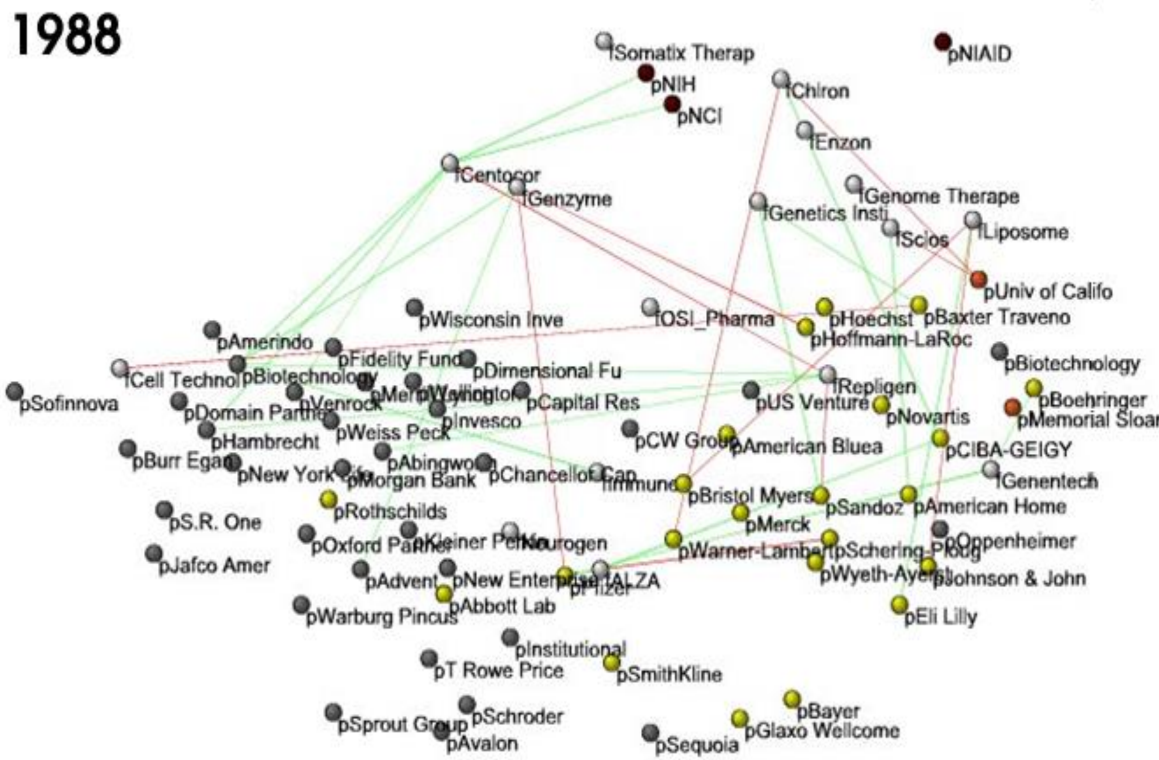
- Proof techniques you will learn:
- • Induction
- • Contradiction
- • Extremal Principle
- • Constructive Proof
- • Case Analysis

- These techniques are invaluable for algorithm design and correctness proofs.

Journey of This Course

- Definitions
- ↓
- Lemmas
- ↓
- Theorems
- ↓
- Proof Techniques
- ↓
- Algorithms
- ↓
- Real-world Applications

Business ties in US biotech-industry

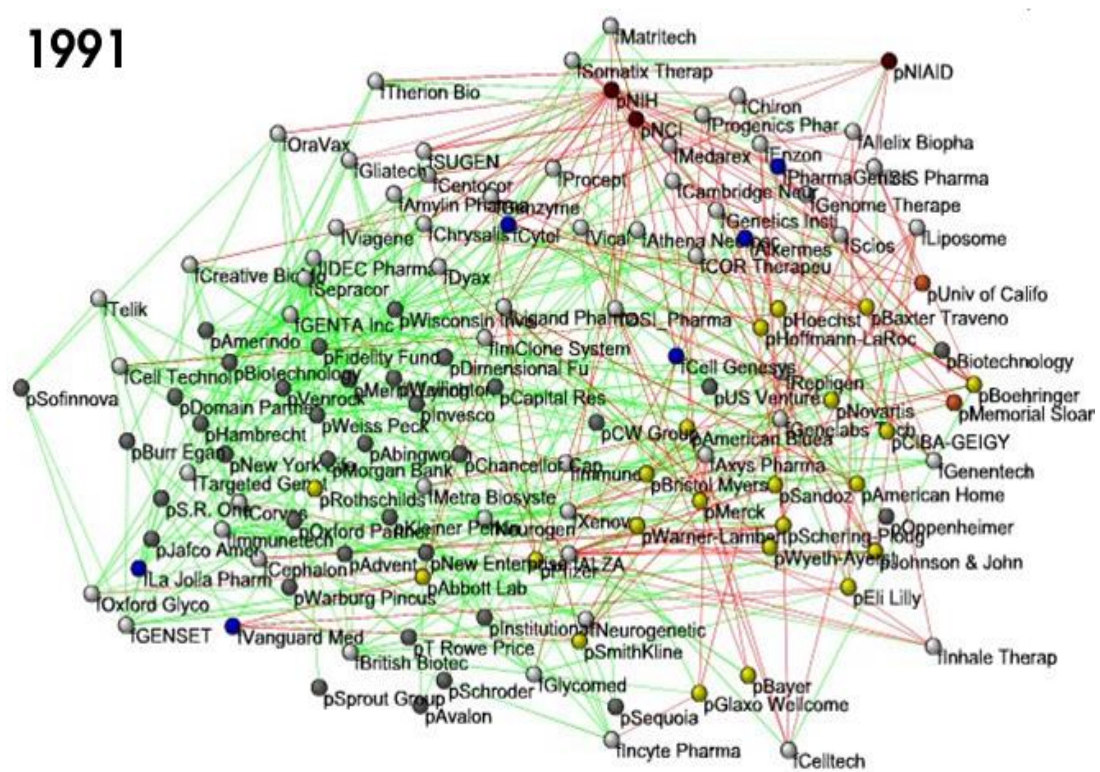


Nodes: companies: investment
pharma
research labs
public
biotechnology

Links: financial
R&D collaborations

http://ecclectic.ss.uci.edu/~drwhite/Movie

Business ties in US biotech-industry



Nodes: companies: investment

pharma

research labs

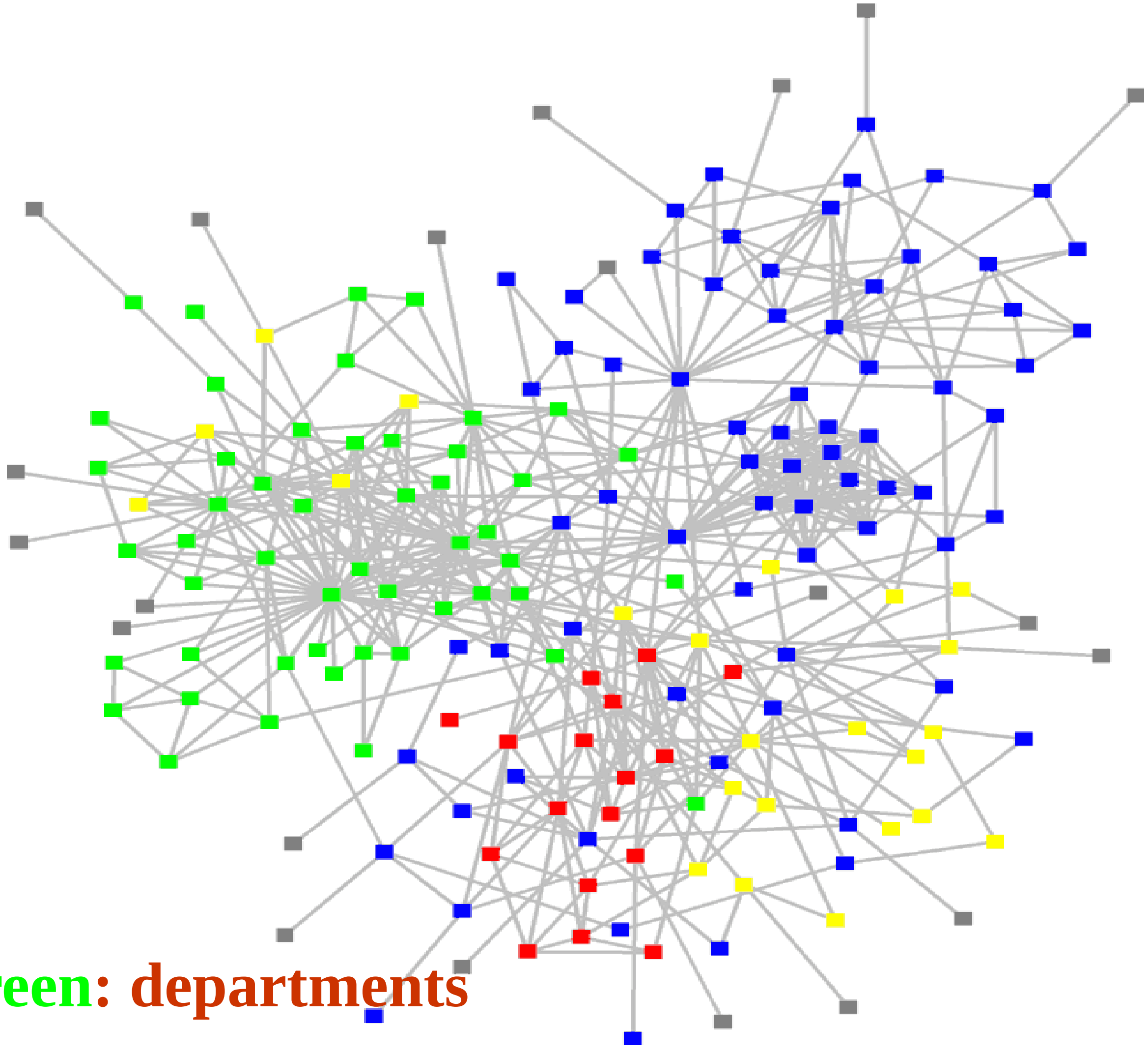
public

biotechnology

Links: financial

R&D collaborations

Structure of an organization



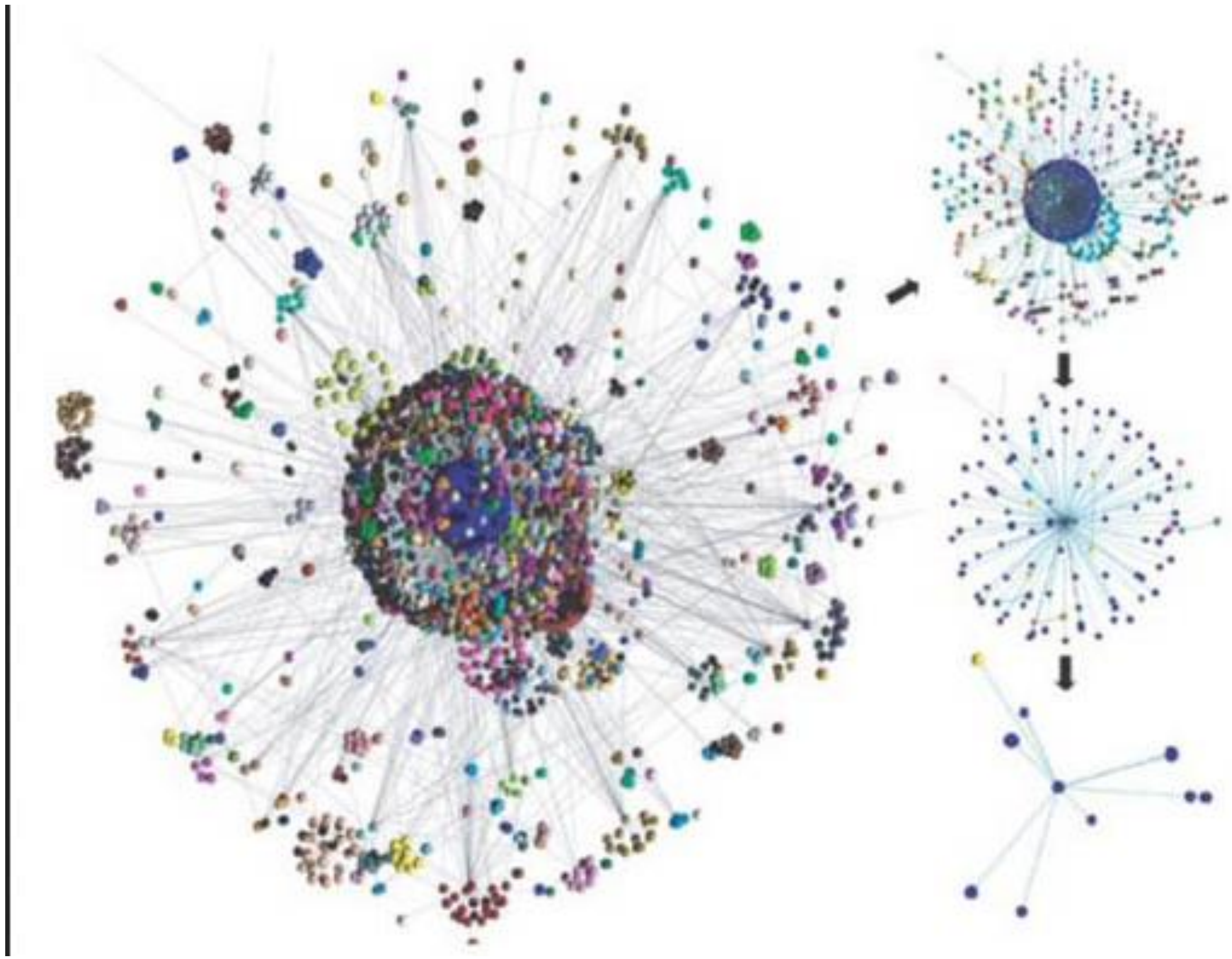
Red, blue, or green: departments

Yellow: consultants

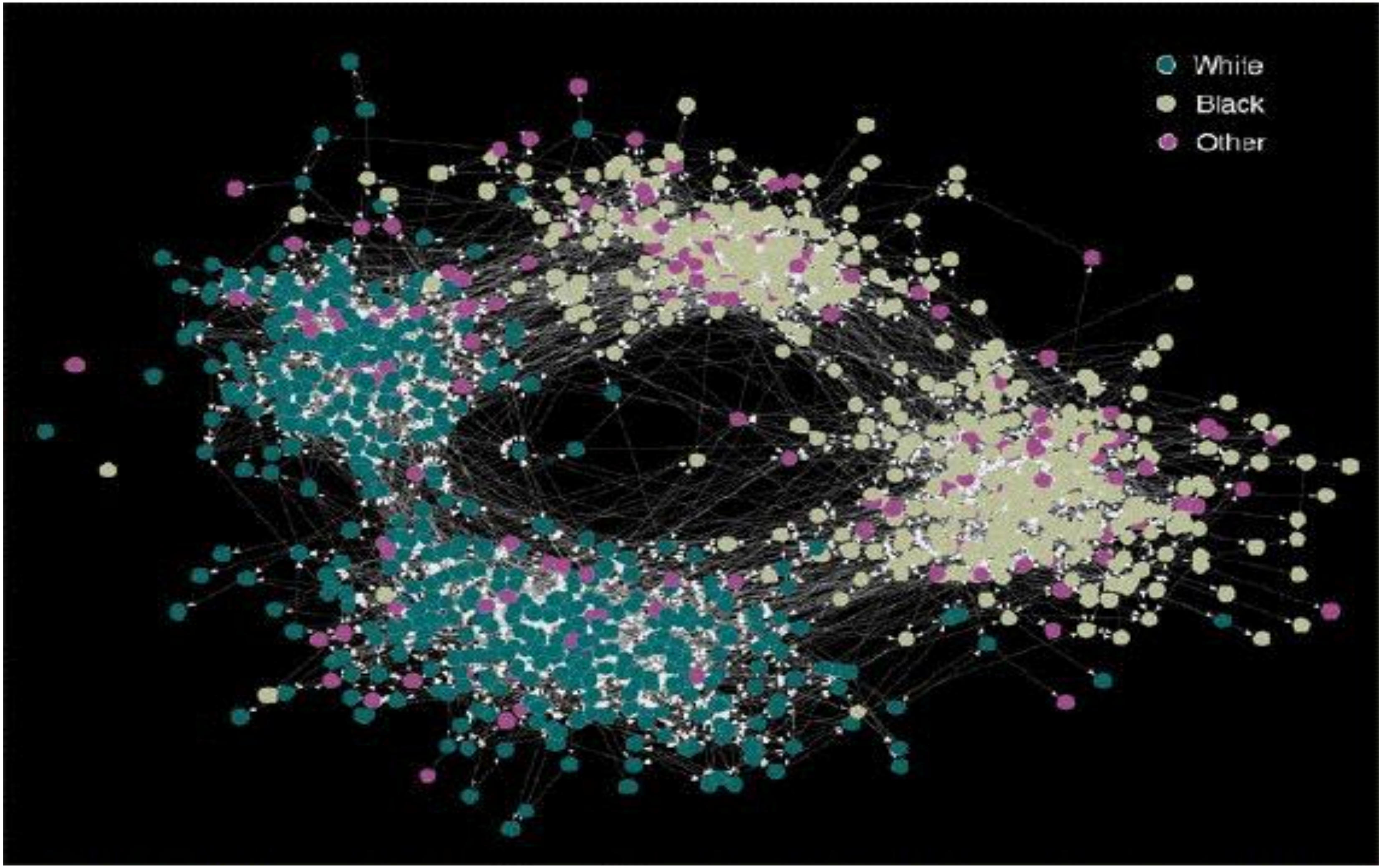
Grey: external experts

www.orgnet.com

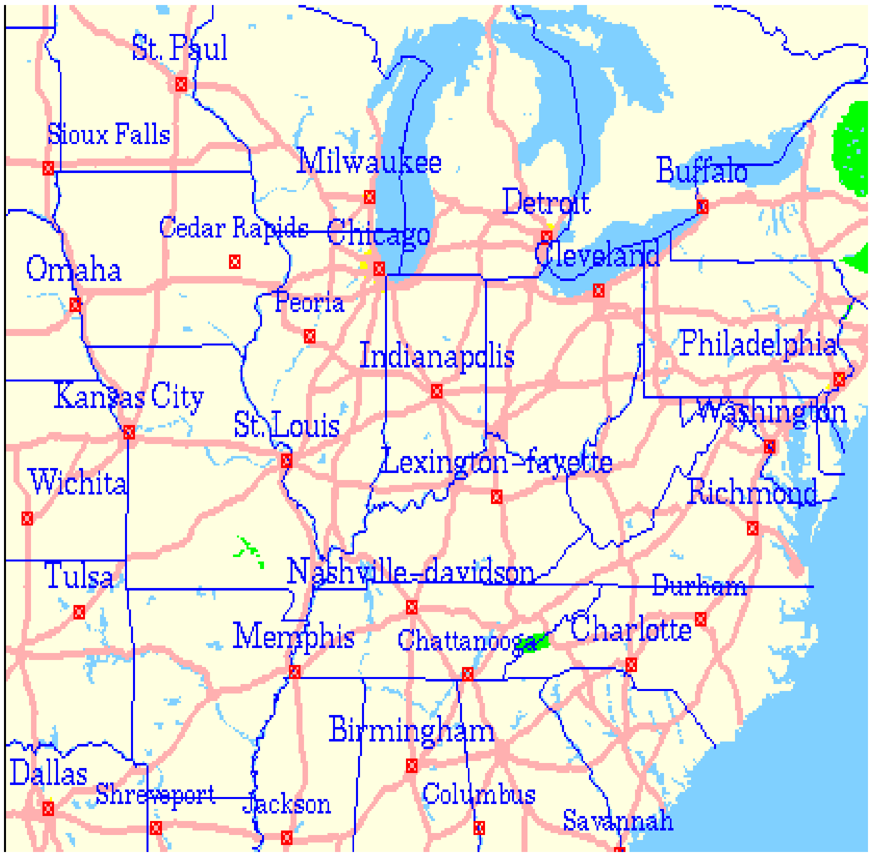
Web network

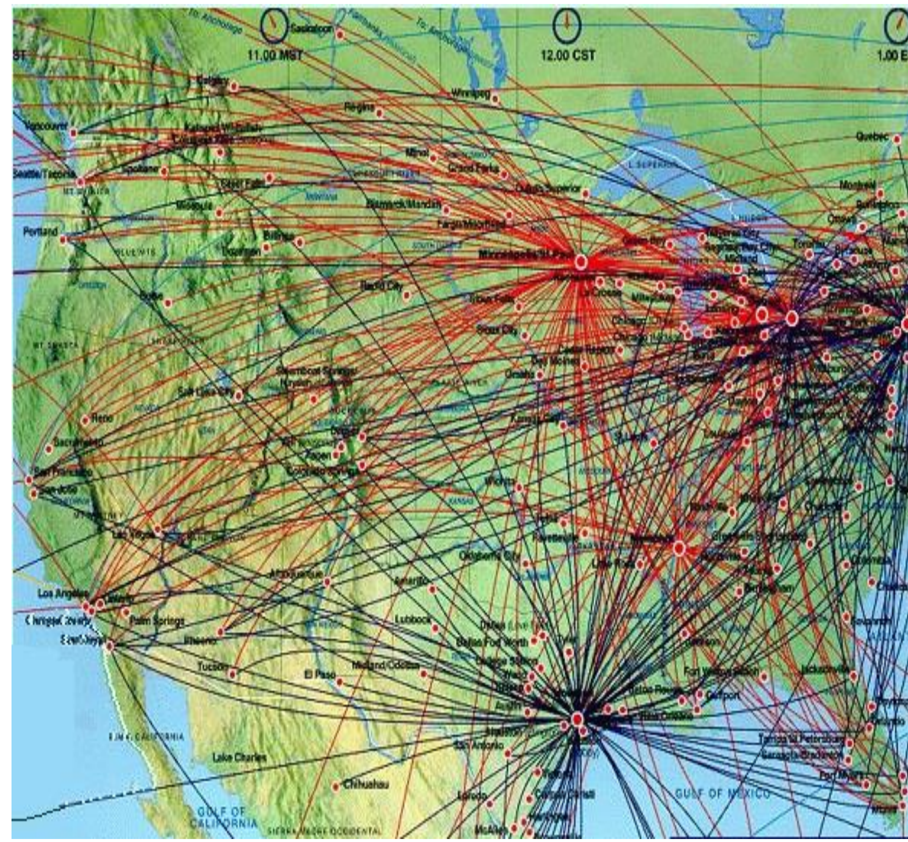


Friendship Network

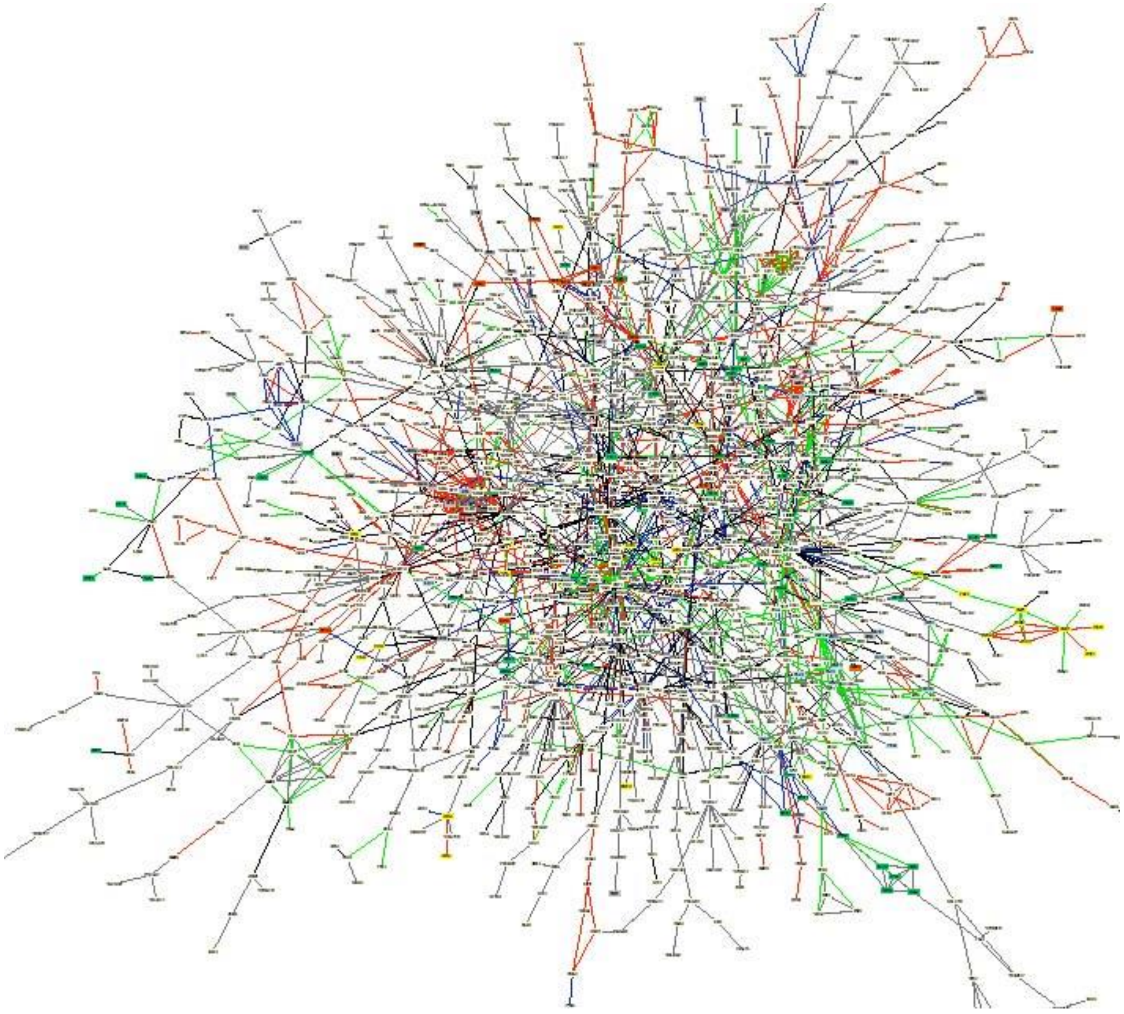


Road and Airlines Network





Yeast protein-protein interaction network

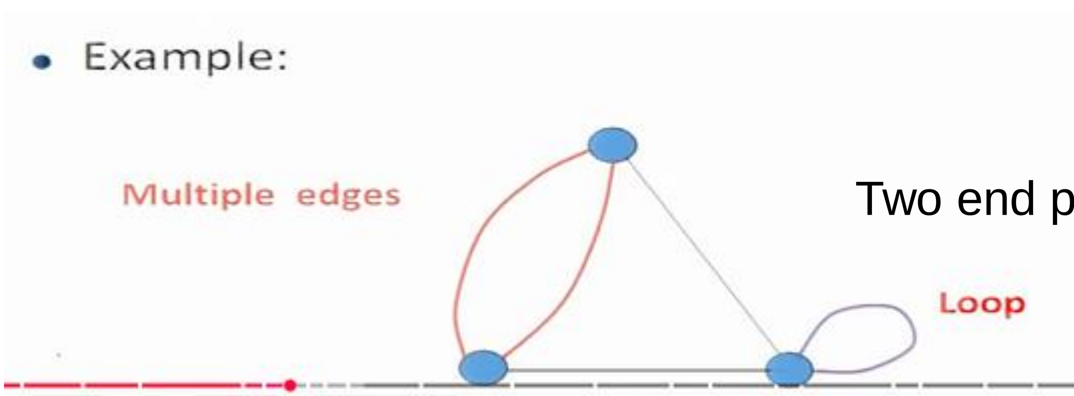


Graph: Definition

- A graph $G = (V,E)$ with n vertices and m edges consists of:
  - a **vertex set** $V(G) = \{v_1, \ldots, v_n\}$, and
  - an **edge set** $E(G) = \{e_1, \ldots, e_m\}$, where each edge consists of two (possibly equal) vertices called its endpoints.
  - Edges capture pairwise relationship between vertices
- The **order** of the graph G, written as n(G), is the number of vertices in G. $n(G)= |V|$
- The **size** of the graph G, written as e(G), is the number of edges in G. $e(G)=|E|$

Loops and multiple edges

- Loop: An edge whose two end points are equal
- Multiple edges: Edges have same pairs of end points.



Two end points

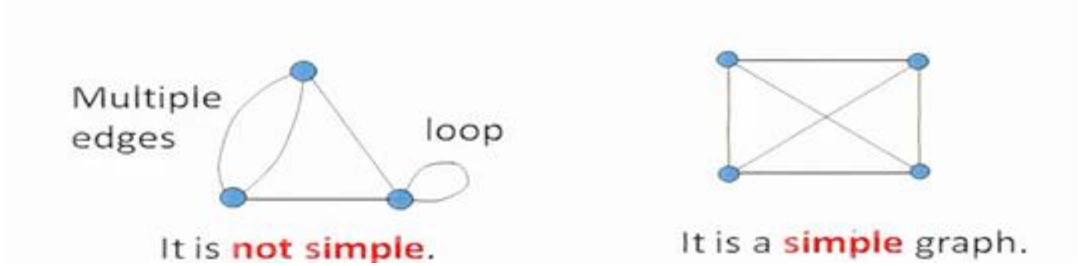
Multigraph

- A loop is an edge whose endpoints are equal.

- Multiple edges are edges having the same set of end-points.

- When we do want to allow multiple edges or loops we say multigraph.
- When we talk about a graph without multiple edges and loops, it says simple graph.

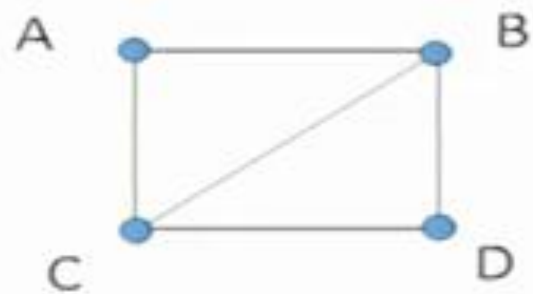
- **Remarks** A multigraph might have no multiple edges or loops. Every (simple) graph is a multigraph, but not every multigraph is a (simple) graph.



Adjacent nodes

- Two vertices are adjacent and are neighbors, if they are the end point of an edge

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



Finite / Infinite / null graphs

- Finite graph: Graph whose vertex and edge set are finite
- Infinite graph: A graph is infinite if its vertex set V or edge set E are infinite
- Empty graph: Graph whose vertex set and edge set are empty
- Null graph: Graph whose edge set are empty

Subgraph

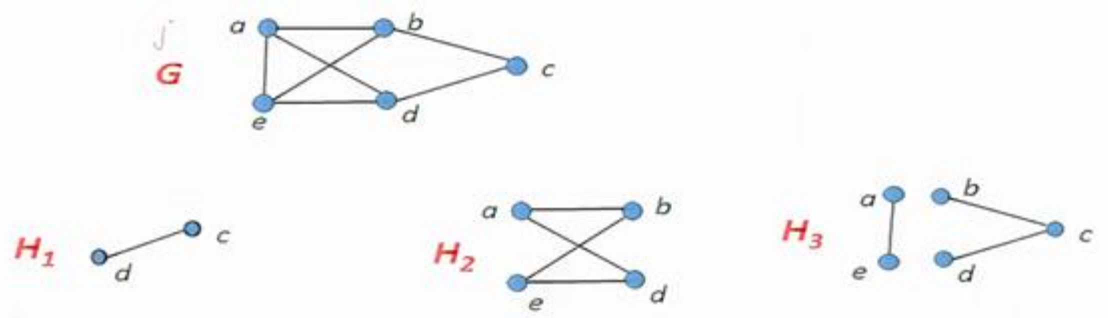
- A *subgraph* of a graph G is a graph H such that:
  - $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$

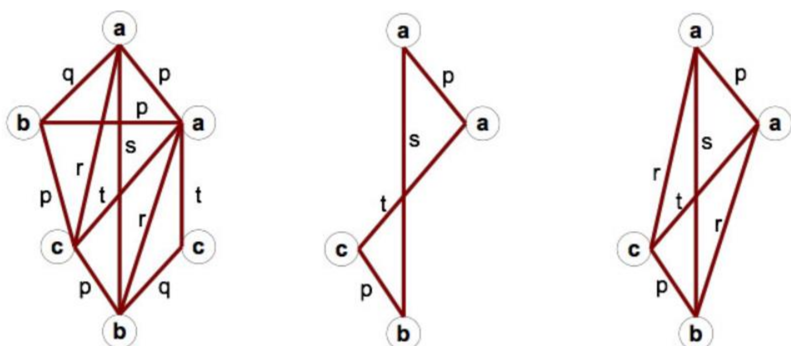
  The assignment of endpoints to edges in H is same as in G.

- An *induced subgraph* of G is a subgraph H of G such that E(H) consists of **all edges of G** whose endpoints belong to V(H)

Subgraph

- Example: H_1, H_2, and H_3 are subgraphs of G



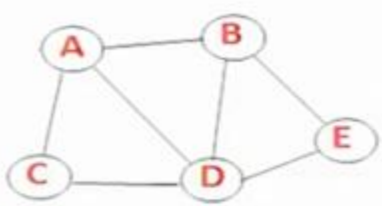


Induced subgraph

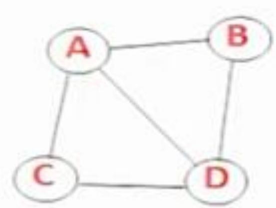
- An ***induced subgraph*** :
- A subgraph obtained by deleting a set of vertices
- We write $\boldsymbol{G}[T]$ for $\boldsymbol{G}\text{-}\overline{T}$, where $\overline{T} = V(G)\text{-}T$

$\boldsymbol{G}[T]$ is the subgraph of $\boldsymbol{G}$ induced by $\boldsymbol{T}$

Example: Assume T: {A, B, C, D}



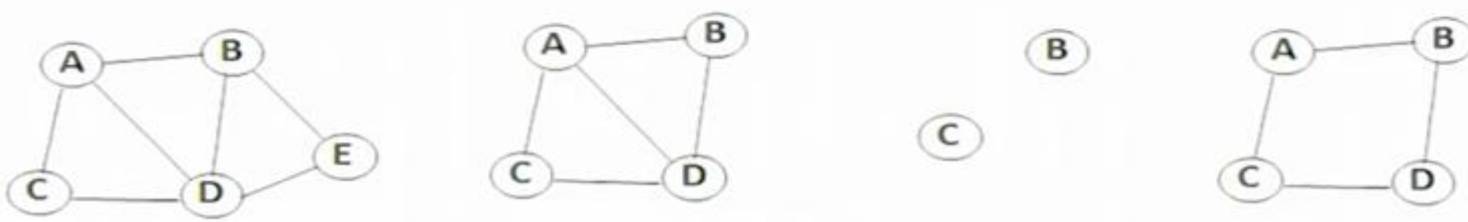
G



G[*T*]

Induced subgraph

G_2 is the subgraph of G_1 induced by (A, B, C, D)
G_3 is the subgraph of G_1 induced by (B, C)
G_4 is **not** the subgraph induced by (A, B, C, D)



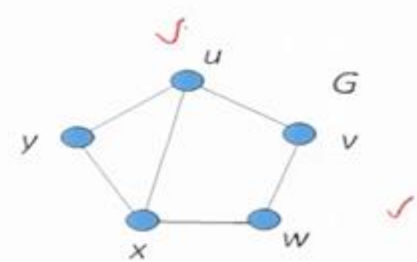
G_1 G_2 $G3$ G_4

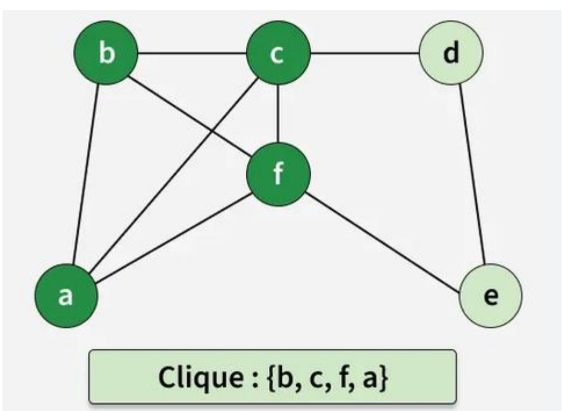
Clique in a graph

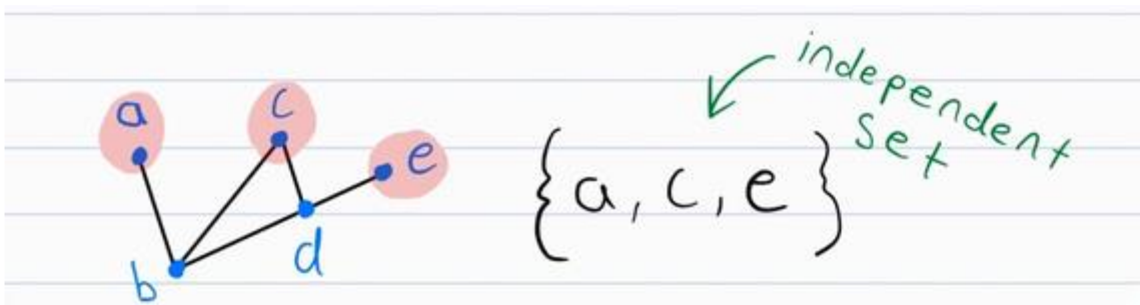
- A clique in the graph: clique in a graph is a subset of vertices such that every pair of distinct vertices in the subset is connected by an edge. (a complete subgraph)
- An Independent set in a graph: A set of pairwise non-adjacent vertices

Example:

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

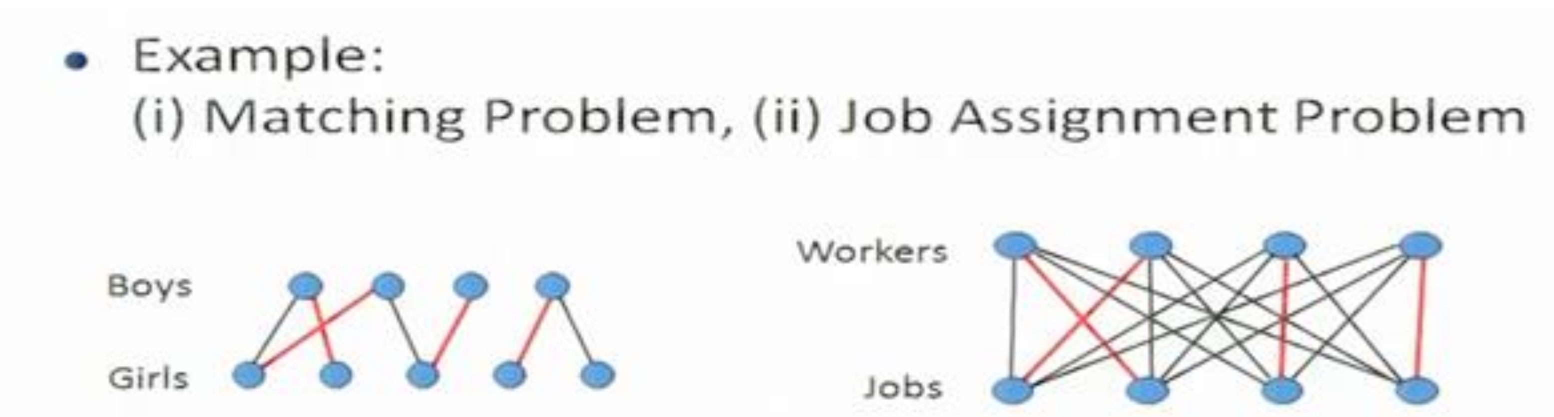





a
c
e
b
d
{a, c, e}
independent set

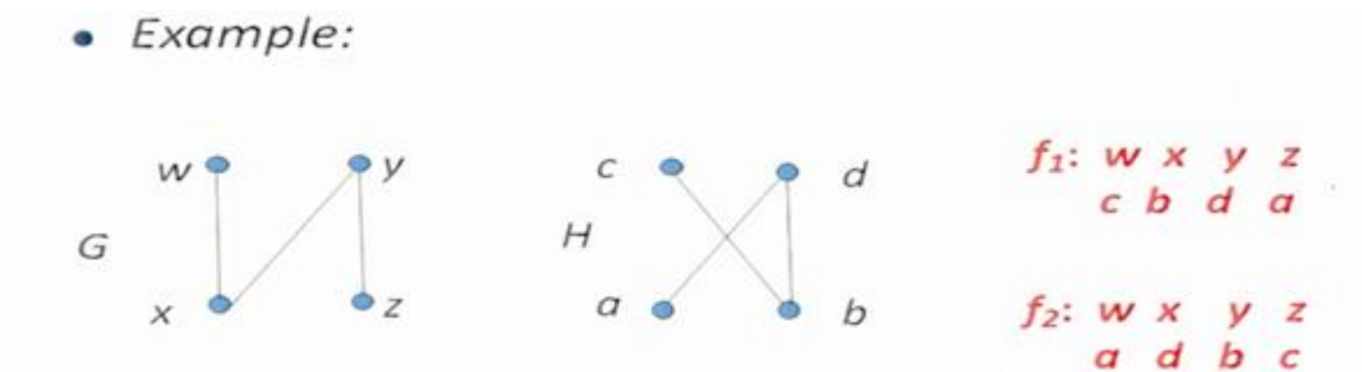
Bipartite graph

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



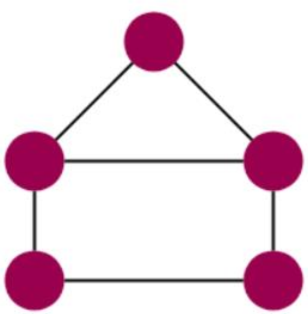
Isomorphism

- An *isomorphism* from G to H is a bijection $f: V(G) \rightarrow V(H)$ such that $uv \in E(G)$ if and only if $f(u)f(v) \in E(H)$.
  - We say that *G is isomorphic to H*, written as G≡H, if there is an isomorphism from G to H.

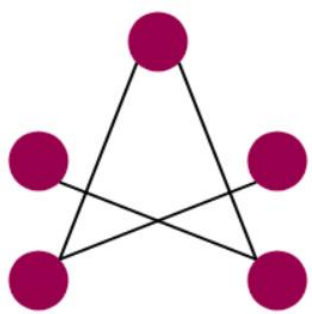


Complement

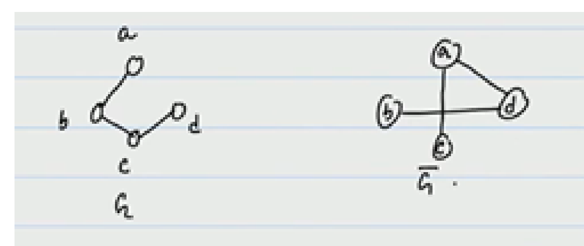
- The *complement* G′ of a simple graph G is the simple graph with vertex set V(G) and edge set defined by:
  - $uv \in E(G')$ if and only if $uv \notin E(G)$



Graph G



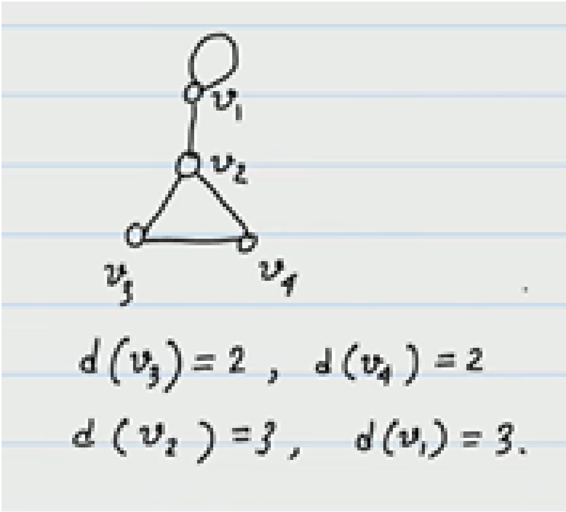
Complement Graph $\overline{G}$



Adjacency, Incidence and Degree

Assume that **e_i is an edge**, whose end points are **{v_j, v_k}**

- Vertices v_j and v_k are called **adjacent to each other**.
- The edge **e_i** is said to be **incident upon vertex v_j and v_k**
- **Degree** of a vertex v_k is the **number of edges incident** upon v_k. It's denoted as d(v_k)



Adjacency, Incidence and Degree

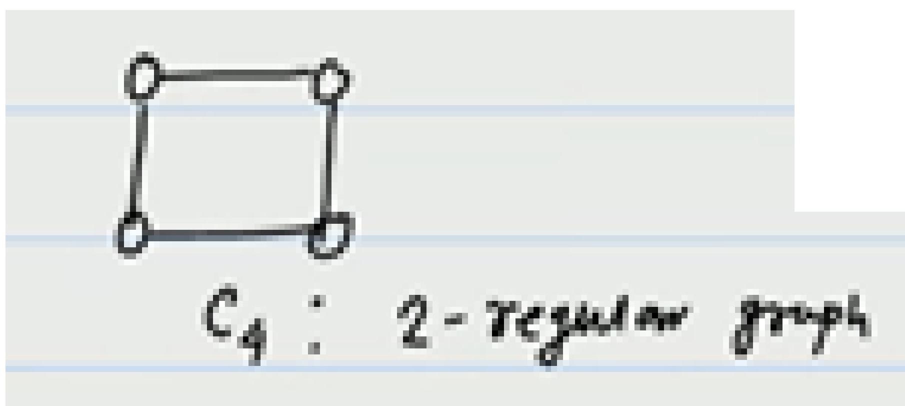
- The **degree** of vertex v in a graph G, written $d_G(v)$, or $d(v)$, is the number of edges incident to v, except that each loop at v counts twice
- The **maximal degree** is $\Delta(G)$
- The **minimum degree** is $\delta(G)$

$d(B) = 3, \; d(C) = 2$

$\Delta(G) = 3, \delta(G) = 2$

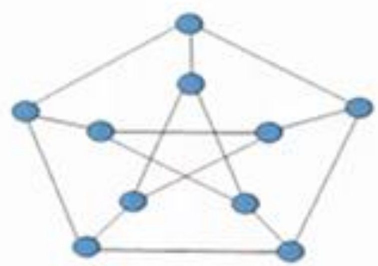
K-regular graph

- A graph G is called k-regular if d(v)=k for all vertices v € V
- A complete graph with n vertices are (n-1) regular



K-regular graph

- G is ***regular*** if $\Delta(G) = \delta(G)$ ✓
- G is ***k-regular*** if the common degree is k.
- The ***neighborhood*** of v, written $N_G(v)$ or $N(v)$ is the set of vertices adjacent to v.

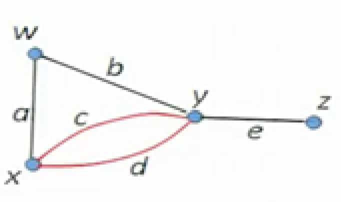


3-regular

Graph representation

Adjacency matrix

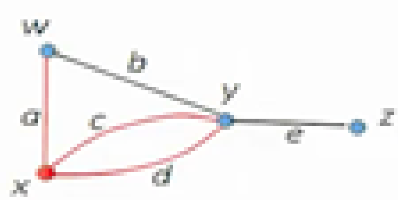
- Let G(V, E), |V|=n, |E|=m
- The **adjacency matrix** of G is written as A(G) is the **n × n matrix**, in which each entry a{i,j} is the number of edges with end points (v_i , v_j)



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

Graph representation

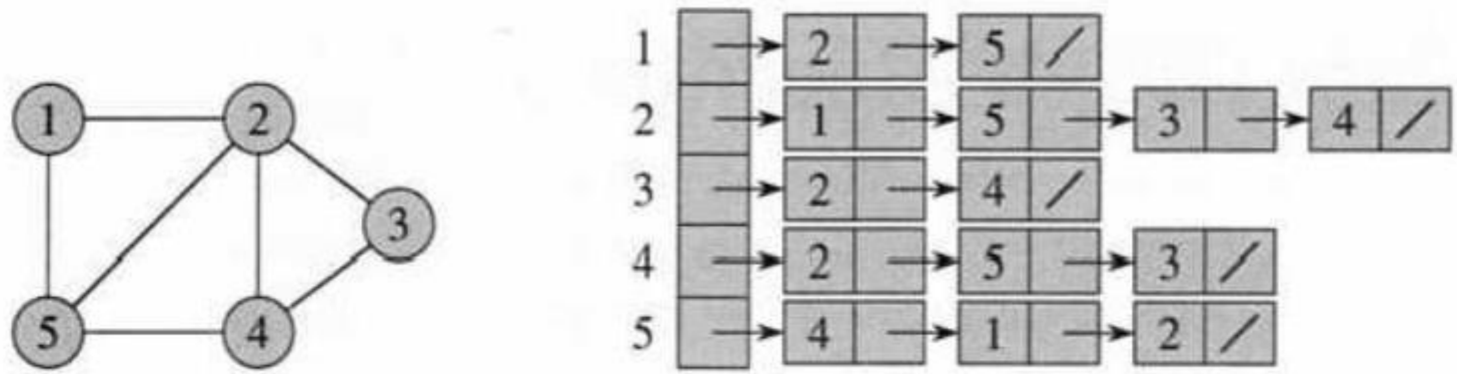
Incidence matrix

- Let G(V, E), |V|=n, |E|=m
- **Incidence matrix** M(G) is the **n × m matrix** in which entry m_{i, j}=1 if node v_i is the end point of edge e_j, otherwise 0
- 

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

Graph representation

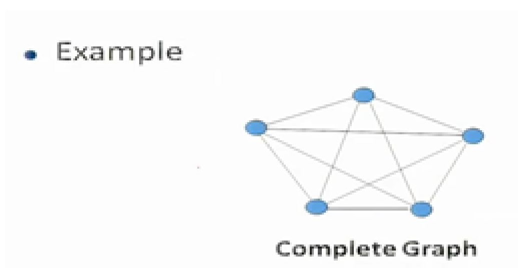
Adjacency List

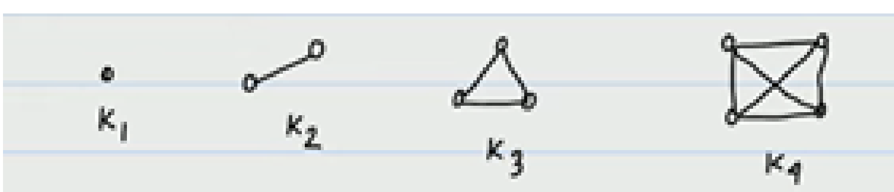


(a) (b)

complete graph

- A *complete graph* or a *clique* is a simple graph in which every pair of vertices is an edge.
  - We use the notation K_n to denote a clique of *n* vertices
  - The complement K_n' of K_n has no edges





Complete bipartite graph

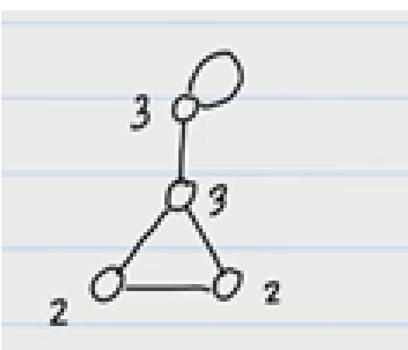
- Complete bipartite graph is a simple bipartite graph, where **two vertices are adjacent** if and only if they are in **different partite set**.

Complete Bipartite Graph

Handshaking Lemma

Theorem: If a graph G with e edges has vertices v_1, v_2,…. , v_n , then show that

$$\sum_{i=1}^{n} d(v_i) = 2e$$



Number of edges e=5

Sum of the degrees: 3+3+2+2=10=2×e

Handshaking Lemma

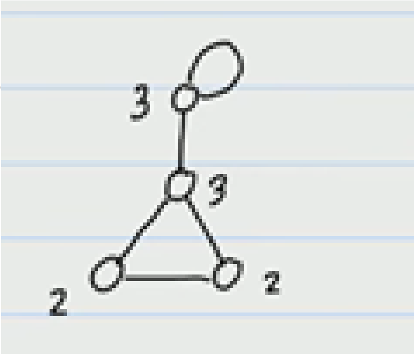
- Every edge in a graph is incident on **exactly two vertices**.
- When we sum the degrees of all vertices, **each edge is counted twice**—once at each of its endpoints.
- Since there are e edges, the total contribution to the degree sum is

$$2 \times e = 2e.$$

Therefore,

$$\deg(v_1) + \deg(v_2) + \cdots + \deg(v_n) = 2e.$$

$$\boxed{\therefore \sum_{i=1}^{n} \deg(v_i) = 2e.}$$



Corollary 1

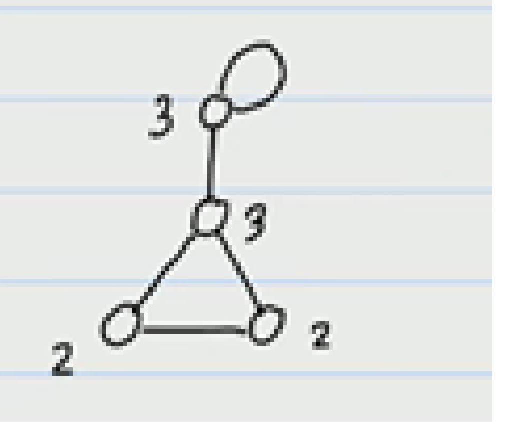
Corollary

In a graph G, the average degree is $2e/n$

Hence, $\delta(G) <= 2e/n <= \Delta(G)$

Corollary 2

In any graph G, the number of vertices with odd degree is even



Corollary 2

In any graph G, the number of vertices with odd degree is even

- Since 2|E| is even, the sum of the degrees of all vertices is even.
- The sum of all even degrees is always even.
- Therefore, the sum of the odd degrees must also be even.
- A sum of odd integers is even only if there are an even number of odd integers.
- Hence, the number of vertices with odd degree is even.

Suppose $v_1, v_2, \cdots v_t$ are odd vertices

$v_{t+1}, v_{t+2}, \cdots v_n$ are even vertices

$$d(v_1) + d(v_2) + \cdots + d(v_t) + d(v_{t+1}) + \cdots + d(v_n) = 2e$$

$$d(v_1) + \cdots + d(v_t) = 2e - \Big(\underbrace{d(v_{t+1}) + \cdots + d(v_n)}\Big)$$

Corollary

- **Corollary** 1.3.4. In a graph G, the average vertex degree is $\frac{2e(G)}{n(G)}$,and hence $\delta(G) \leq \frac{2e(G)}{n(G)} \leq \Delta(G)$.

- **Corollary** 1.3.5. Every graph has an even number of vertices of odd degree. No graph of odd order is regular with odd degree.

- **Corollary** 1.3.6. A k-regular graph with n vertices has $nk/2$ edges.

- The maximum number of edges in a simple graph with (n) vertices is

$$\boxed{\binom{n}{2} = \frac{n(n-1)}{2}}.$$