Lecture 12
Graph Theory and Applications

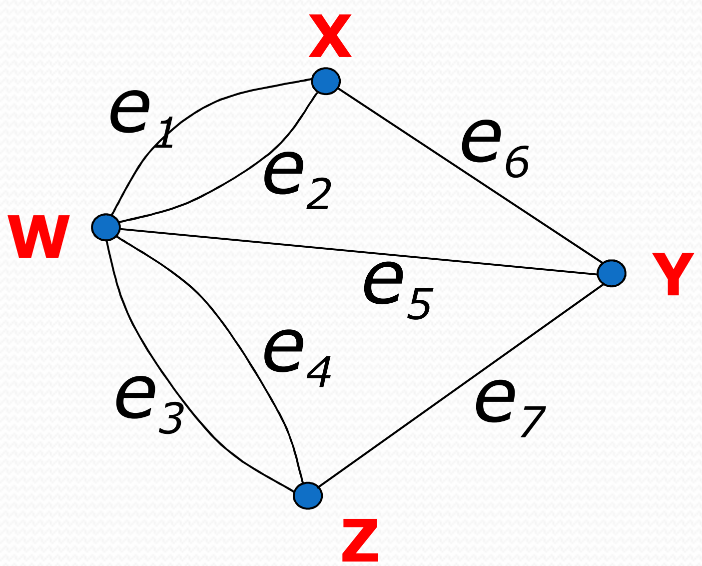
- Introduction to Graph Theory
- Historical Problems
- Graph Theory and Networks
- Graph and its basic components
- Application to Circuit Analysis

Introduction to graph theory

- **Graph theory** – study of graphs and their applications
- **Graph** – mathematical object consisting of a set of:
  - V = **nodes** (vertices, points).
  - E = **edges** (links, arcs) between pairs of nodes.
  - Denoted by $G = (V, E)$.
  - Captures pairwise relationship between objects.
  - **Graph size** parameters: $n = |V|$, $m = |E|$.

What Is a Graph?

- A graph G is a triple consisting of:
  - A vertex set $V(G)$
  - An edge set $E(G)$
  - A relation between an edge and a pair of vertices



Examples of Applications

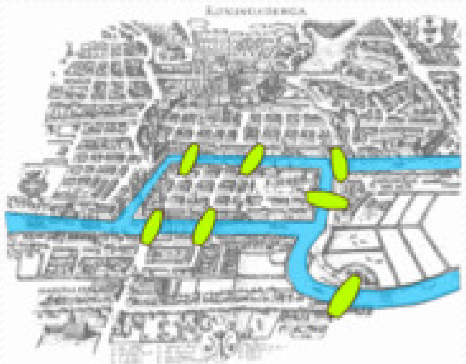
- Graphs can be used to model many types of relations and processes in physical, biological, social and information systems.
- In computer science, can be used to represent networks of communication, data organization, computational devices, the flow of computation, etc.
- Chemistry, e.g., model of molecule (atom & bond)
- Physics, e.g., interactions of system components.
- Sociology, e.g., social network (friendship, acquaintance, work collaboration, etc.)
- Biology, e.g., animal migration, spread of disease.

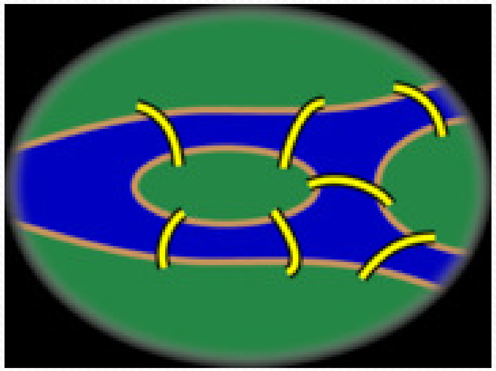
Graph Theory - History

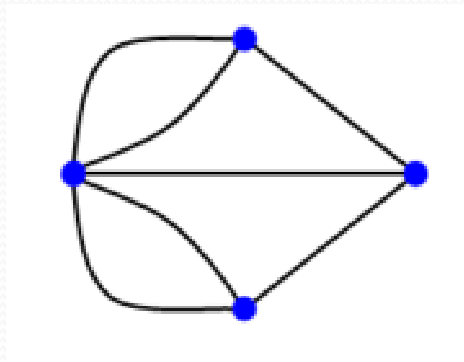
Leonhard Euler's paper on “7 *Bridges of Königsberg*” , published in 1736.

Here, vertices = islands;

edges = bridges







The 7 Bridges of Königsburg

- Königsburg (now called Kalingrad) is a city on the Baltic Sea wedged between Poland and Lithuania.
- A river runs through the city which contains a small island.
- There are 7 bridges which connect the various land masses of the city.

The Problem

- The people of Königsburg made a sport during the 18th century of trying to cross each and every one of the 7 bridges exactly once.
- This was to be done in such a way that one would always end up where one began.

Euler and Graph Theory

- Euler’s solution to the Königsburg bridge problem was more than a trivial matter.
- He didn’t just solve the problem as stated; he made a major contribution to graph theory. Indeed, he essentially invented the subject.
- His contribution has many practical applications.

Some Vocabulary

- A graph is a set of vertices connected by edges.
- The valence (degree) of a vertex is the number of edges that meet there.
- An Euler Circuit is a path within a graph that covers each and every edge exactly once and returns to its starting point.

Euler's Theorem

- ***A connected graph has an Euler circuit if and only if every vertex has an even valence.***
- The Königsburg bridge problem translated into a graph in which all valences were odd. Thus there was no way to walk on each bridge precisely once.

Euler's Theorem

Why is it true?

- Any vertex with odd valence must be either a starting point or an ending point.
- All points that are neither starting nor ending points must be left as often as they are entered.

Why is it important?

- There are many, many examples of circuits that one wishes to traverse such that every edge is covered and no edge is repeated.
- Routes for letter carriers, meter readers, and the like, share these characteristics.

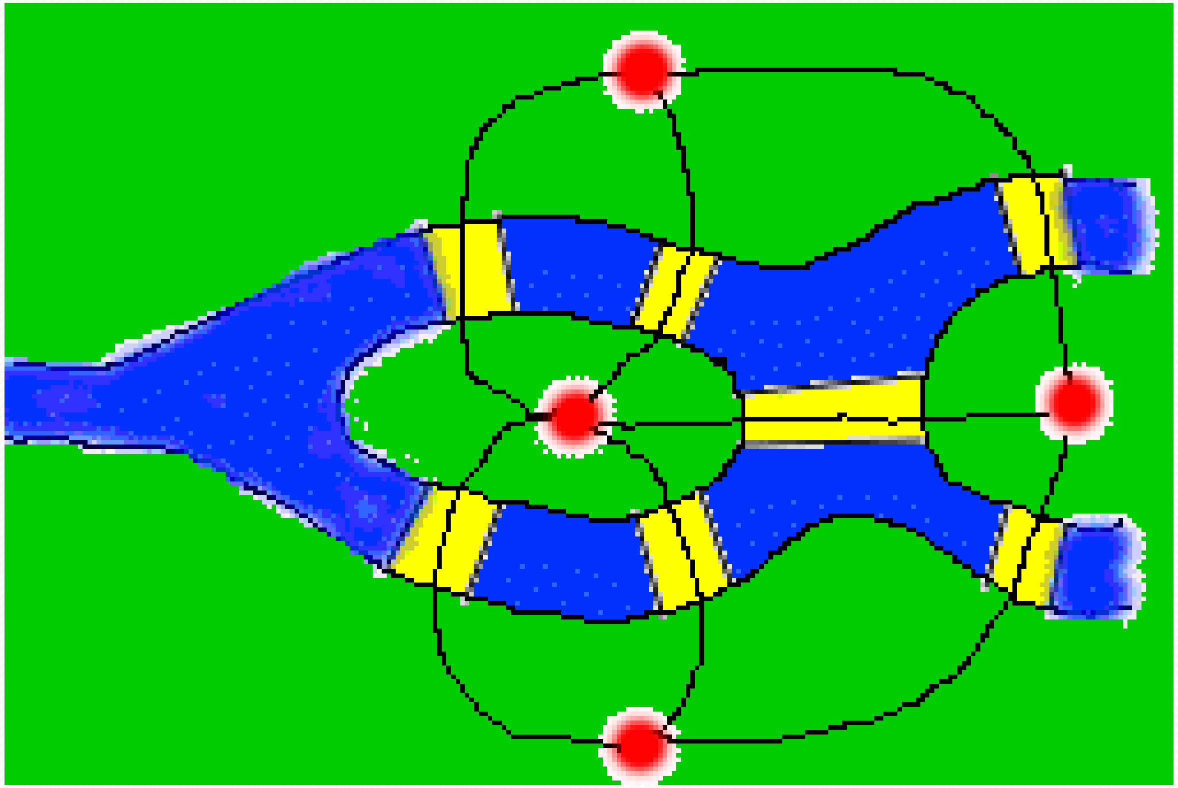
Not all graphs have even valence on all vertices --- What then?

- One cannot expect that every street layout or route will translate into a graph with all vertices of even valence.
- In these cases, one can try to minimize the number of edges that are repeated.
- There is an algorithm to do this. It is called Eulerizing the graph.

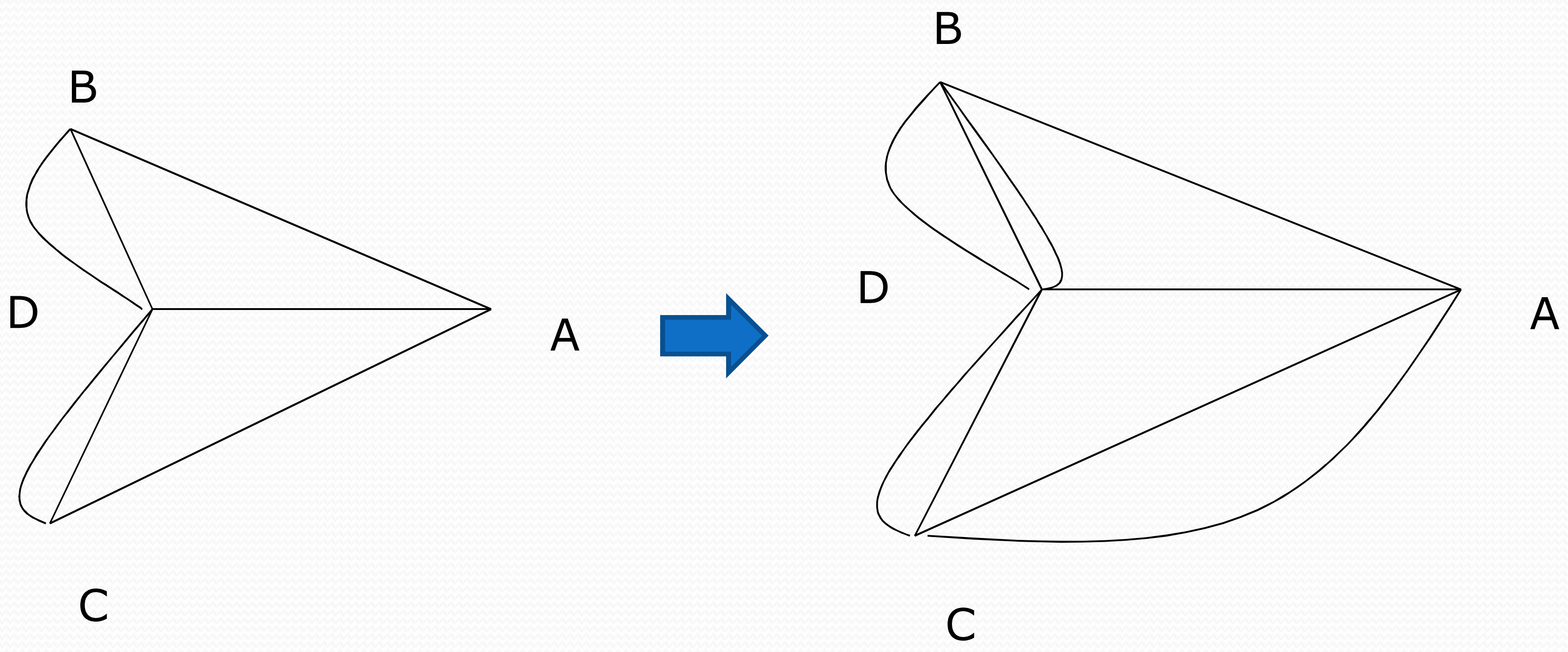
Eulerizing a Graph

- Select pairs of vertices in the graph that have odd valence.
- Do this in such a way that the vertices are as close together (have the fewest edges between them) as possible.
- Neighboring vertices would be the best choice, if possible.
- For each edge on the path that connects a pair of odd-valenced vertices, generate a "phantom edge" duplicating that edge.
- Do this for each pair of odd-valenced vertices.
- In general, there will be more than one Eulerization of a graph. The fewer duplicated edges, the better.

Recall the City of Königsburg



Let us Eulerize Königsburg I



Eulerizing Königsburg II

- Here, we have selected pairs of odd-valenced vertices, BD and AC.
- We have added a “phantom” edge between these pairs of vertices. These phantom edges are edges that are traversed twice.
- Now, with the addition of just two edges, the graph has all even-valenced vertices.

A Troublesome Question

- How do we know that we can always do this?
- In particular, how do we know that the odd-valenced vertices will occur in pairs?

Theorem: The number of odd-valenced vertices is even.

<u>Proof</u> Suppose that there are N edges, thus, there are 2N "ends" of edges. The sum of all the valences must be 2N. Therefore, it is not possible to have an odd number of odd-valenced vertices. Hence, the odd-valence vertices occur in pairs.

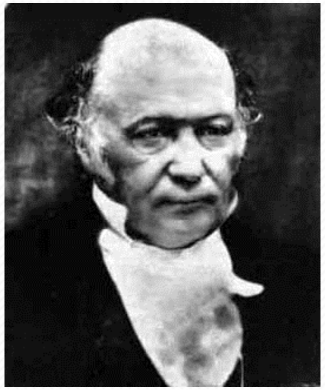
Euler Circuits: In Summation

- A very simple and elegant idea has led to a wide variety of real-world applications.
- Nearly any process which involves routing (and there are many) can be made more efficient by these methods.
- Many millions of dollars can be saved in the process!!

Graph Theory - History

Cycles in Polyhedra

Thomas P. Kirkman

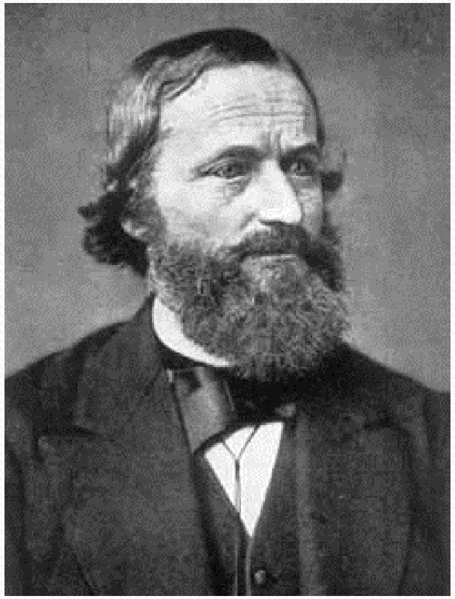


William R. Hamilton

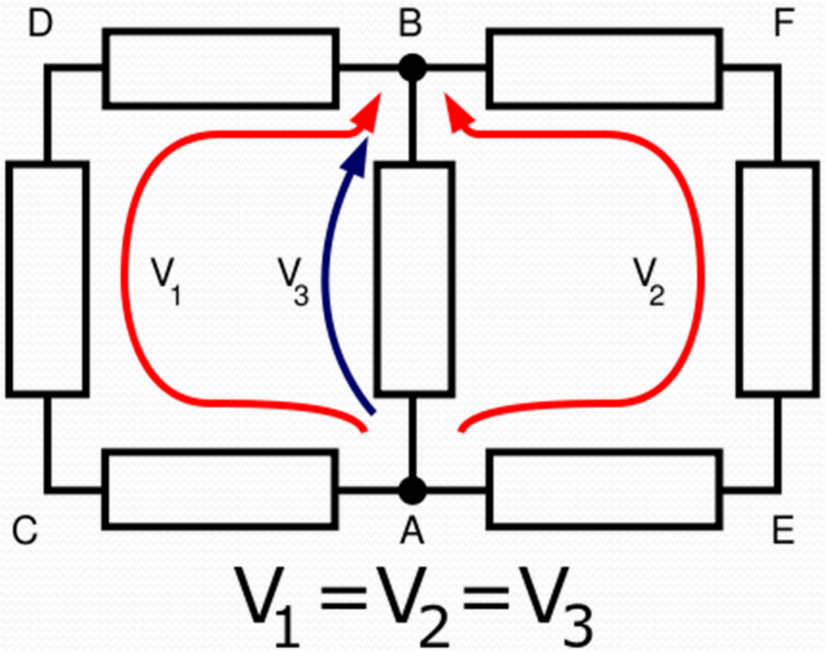
Hamiltonian cycles in Platonic graphs

Graph Theory - History

Trees in Electric Circuits

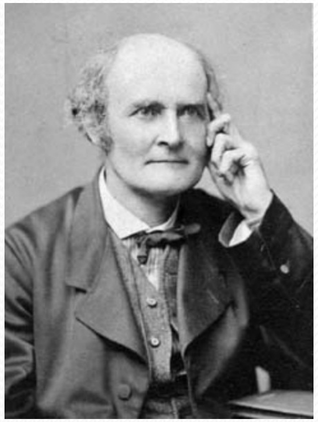


Gustav Kirchhoff

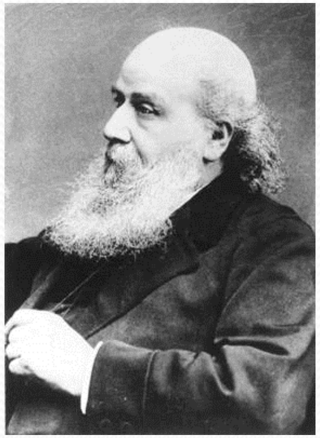


Graph Theory - History

Enumeration of Chemical Isomers –n.b. topological distance a.k.a chemical distance

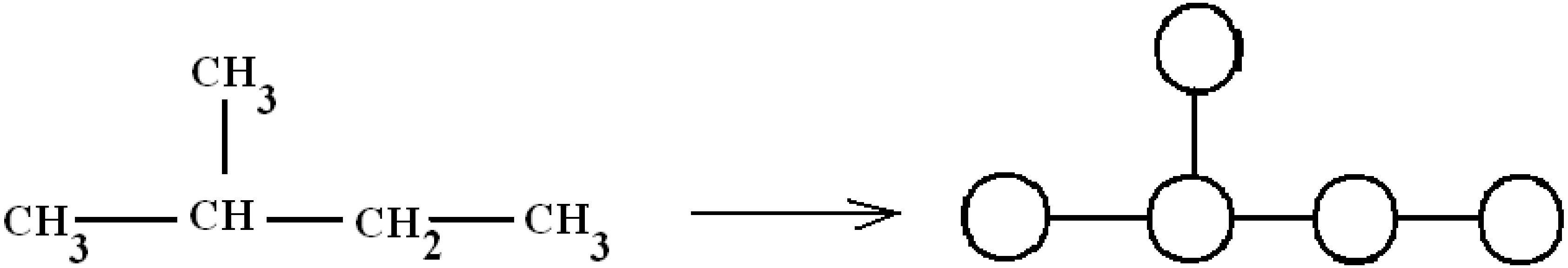


Arthur Cayley



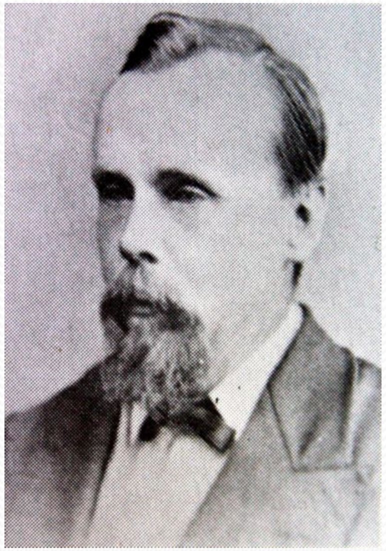
James J. Sylvester

George Polya



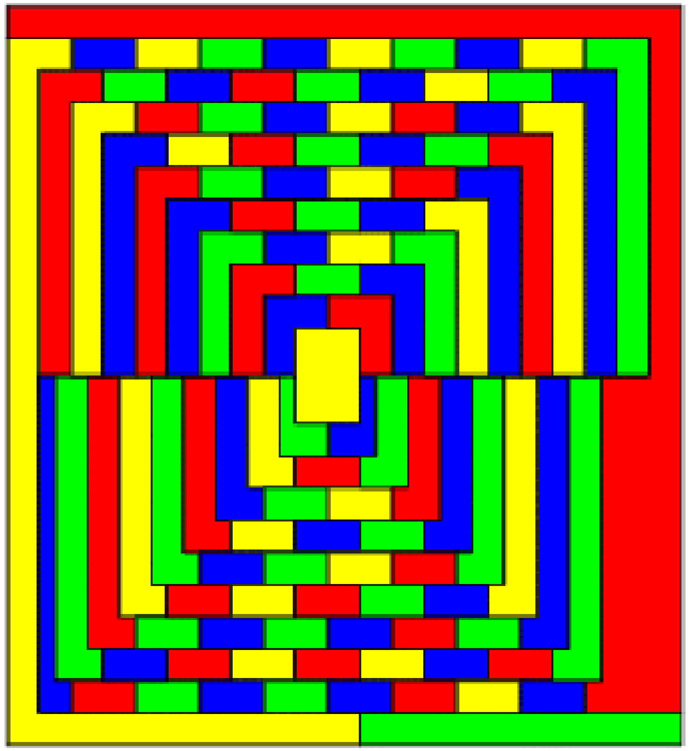
Graph Theory - History

Four Colors of Maps

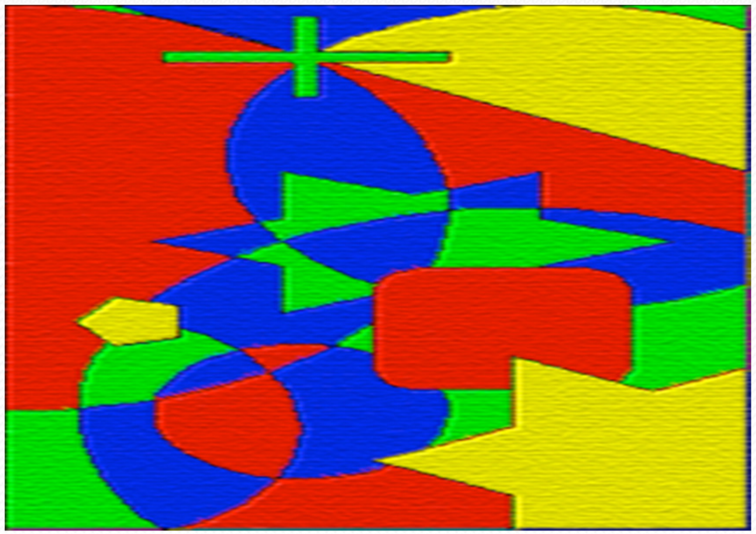


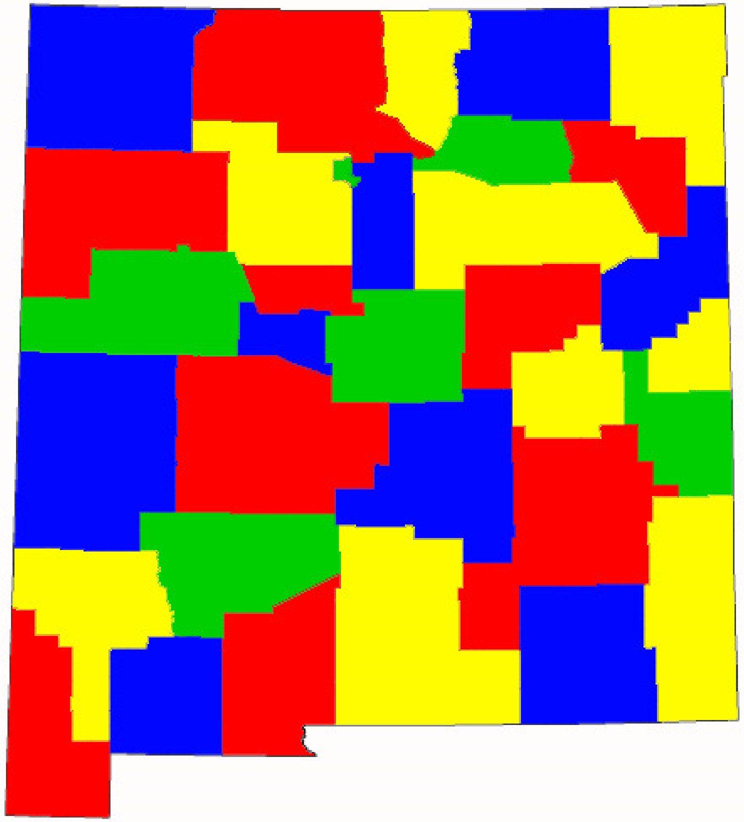
Francis Guthrie

Auguste DeMorgan

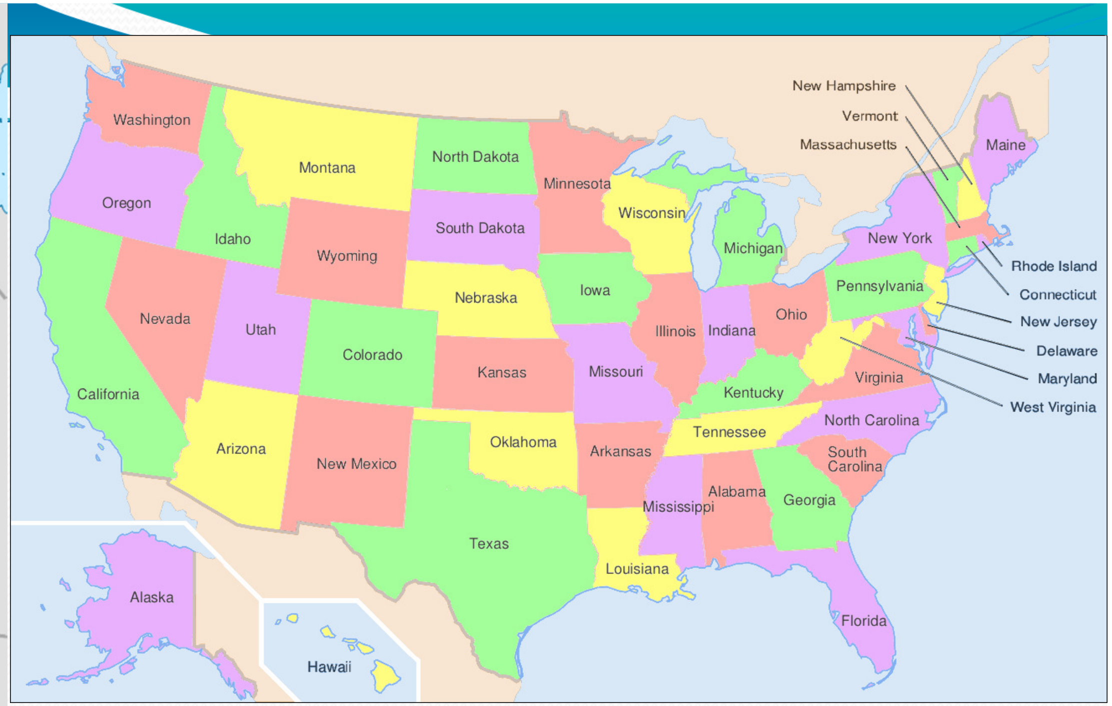


- The theorem asserts that four colors are enough to color any geographical map in such a way that no neighboring two countries are of the same color.






Chiang Rai
Phayao
Chiang Mai
Nan
Mae Hong Son
Lampang
Phrae
Lamphun
Uttaradit
Bueng Kan
Nong Khai
Udon Thani
Nakhon Phanom
Loei
Nong-Bua-Lamphu
Sakon Nakhon
Sukhothai
Phitsanulok
Tak
Khon Kaen
Kalasin
Mukdahan
Kamphaeng-Phet
Phichit
Phetchabun
Maha-Sarakham
Yasothon
Chaiyaphum
Roi Et
Amnat-Charoen
Nakhon Sawan
Uthaithani
Ubon-Ratchathani
Chainat
Lopburi
Nakhon-Ratchasima
Singburi
Buriram
Surin
Sisaket
Suphanburi
Ang-Thong
Saraburi
Kanchanaburi
Ayutthaya
Nakhon-Nayok
Prachin-Buri
Nakhon-Pathom
Pathum-Thani
Sa Kaew
Ratchaburi
Chachoengsao
Chonburi
Phetchaburi
Rayong
Chanthaburi
Trat
Samut Prakan
Bangkok
Nonthaburi
Samut Sakhon
Samut Songkhram
Prachuap Khiri Khan
Chumphon
Ranong
Surat Thani
Phang Nga
Nakhon Si Thammarat
Krabi
Phuket
Phatthalung
Trang
Songkhla
Satun
Pattani
Yala
Narathiwat


Washington
Montana
North Dakota
Minnesota
New Hampshire
Vermont
Massachusetts
Maine
Oregon
Idaho
South Dakota
Wisconsin
Michigan
New York
Wyoming
Rhode Island
Connecticut
Nevada
Nebraska
Iowa
Pennsylvania
New Jersey
Utah
Illinois
Indiana
Ohio
Delaware
Maryland
Colorado
Kansas
Missouri
Virginia
West Virginia
California
Kentucky
North Carolina
Tennessee
Oklahoma
Arkansas
South Carolina
Arizona
New Mexico
Alabama
Georgia
Mississippi
Texas
Louisiana
Alaska
Florida
Hawaii


N Africa
TUNISIA
MOROCCO
ALGERIA
LIBYA
EGYPT
WESTERN SAHARA
CAPE VERDE
MAURITANIA
MALI
NIGER
SENEGAL
GAMBIA
CHAD
SUDAN
ERITREA
GUINEA BISSAU
DJIBOUTI
BURKINA
CAMEROON
SOMALIA
GUINEA
NIGERIA
ETHIOPIA
SIERRA LEONE
BENIN
CENTRAL AFRICAN REPUBLIC
SOUTH SUDAN
UGANDA
LIBERIA
TOGO
GHANA
Indian Ocean
CÔTE D'IVOIRE
KENYA
GABON
SAO TOME & PRINCIPE
DEMOCRATIC REPUBLIC OF THE CONGO
Atlantic Ocean
CONGO REPUBLIC
TANZANIA
COMOROS
SEYCHELLES
SAINT HELENA ISLAND
MAYOTTE ISLAND
ANGOLA
ZAMBIA
MALAWI
JUAN DE NOVA ISLAND
ZIMBABWE
MADAGASCAR
NAMIBIA
MOZAMBIQUE
MAURITIUS
BASSIS DE INDIA
BOTSWANA
RÉUNION
1. EQUATORIAL GUINEA
2. RWANDA
3. BURUNDI
SWAZILAND
EUROPA ISLAND
SOUTH AFRICA
BOUVET ISLAND
LESOTHO
© maps.com

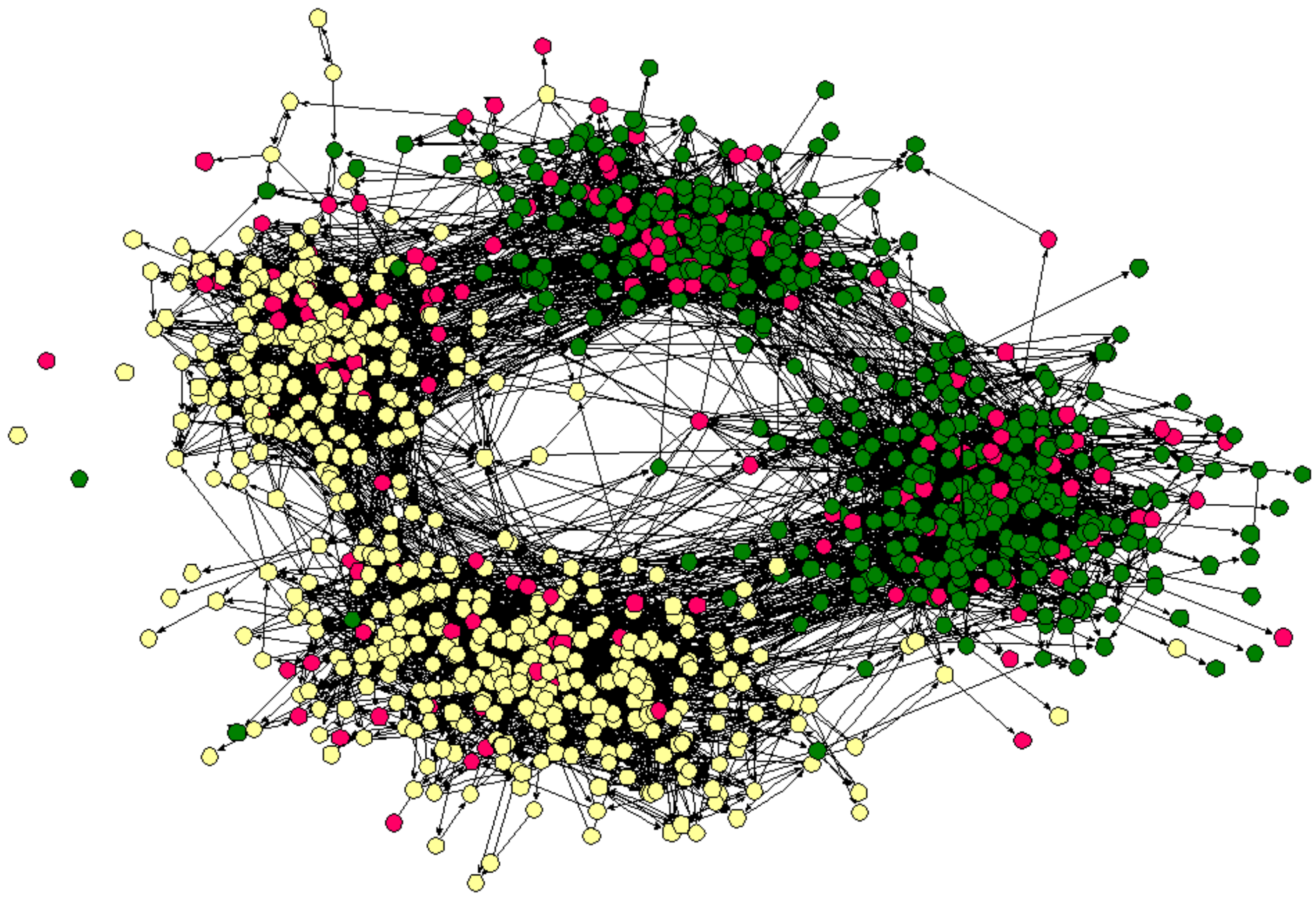
Graph Representation

- Representing a as a graph can provide a different point of view
- Representing a problem as a graph can make a problem much simpler
  - More accurately, it can provide the appropriate tools for solving the problem

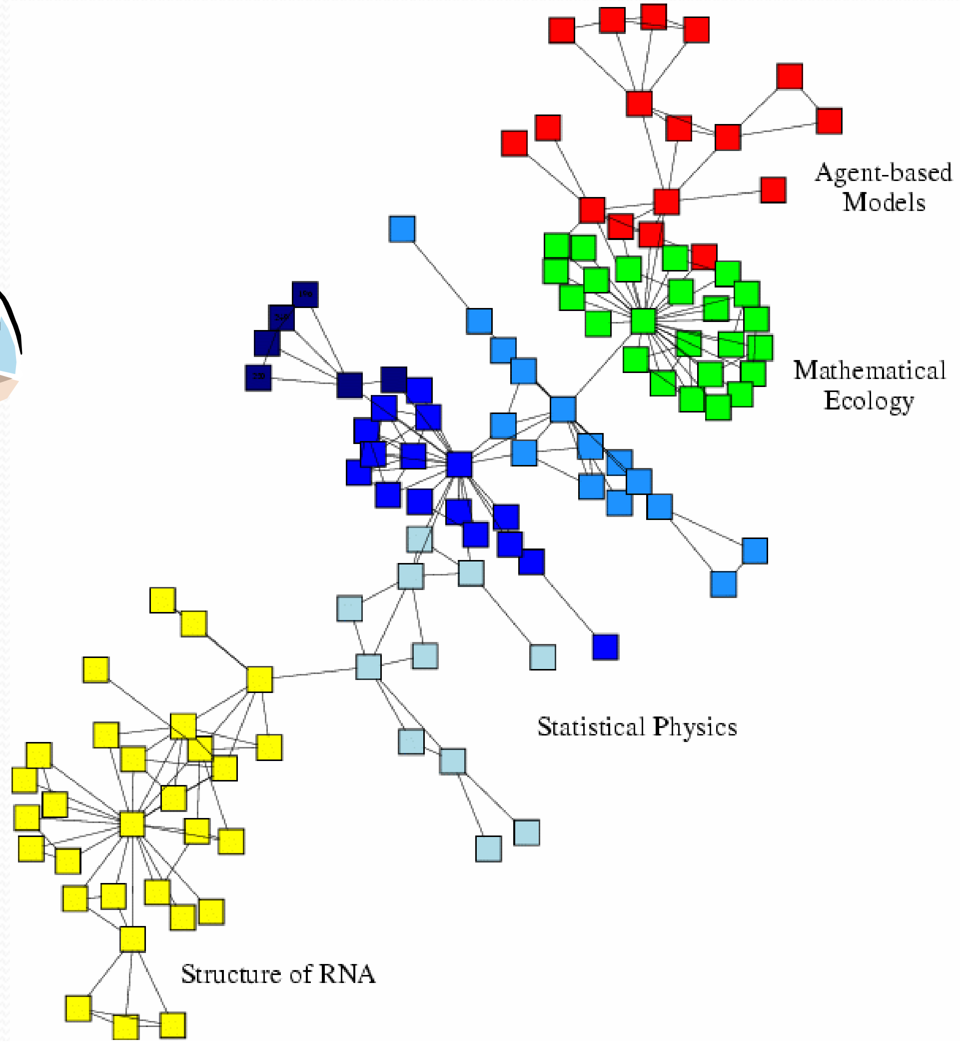
What makes a problem graph-like?

- There are two components to a graph
  - Nodes and edges
- In graph-like problems, these components have natural correspondences to problem elements
  - Entities are nodes and interactions between entities are edges
- Most complex systems are graph-like

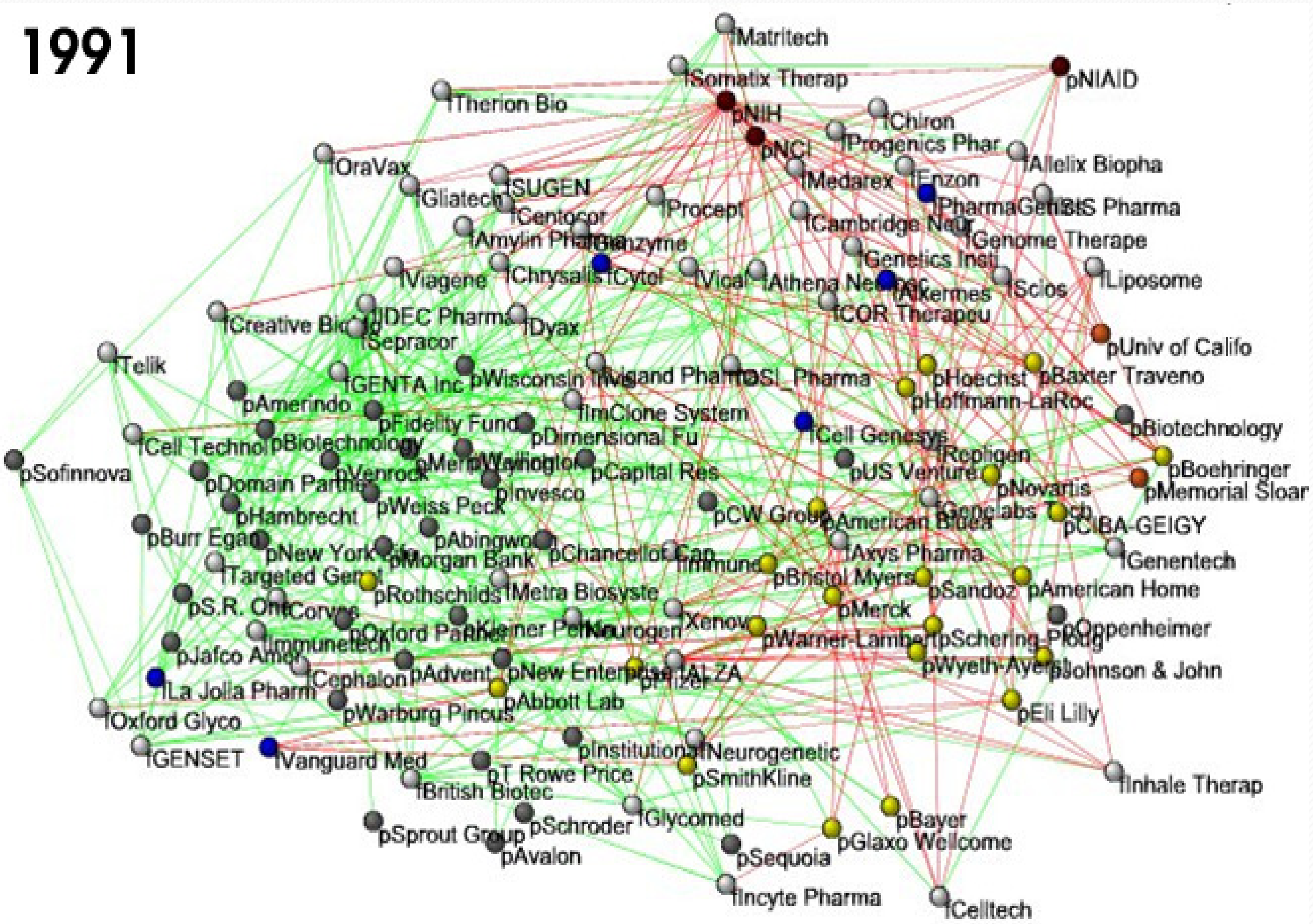
Friendship Network



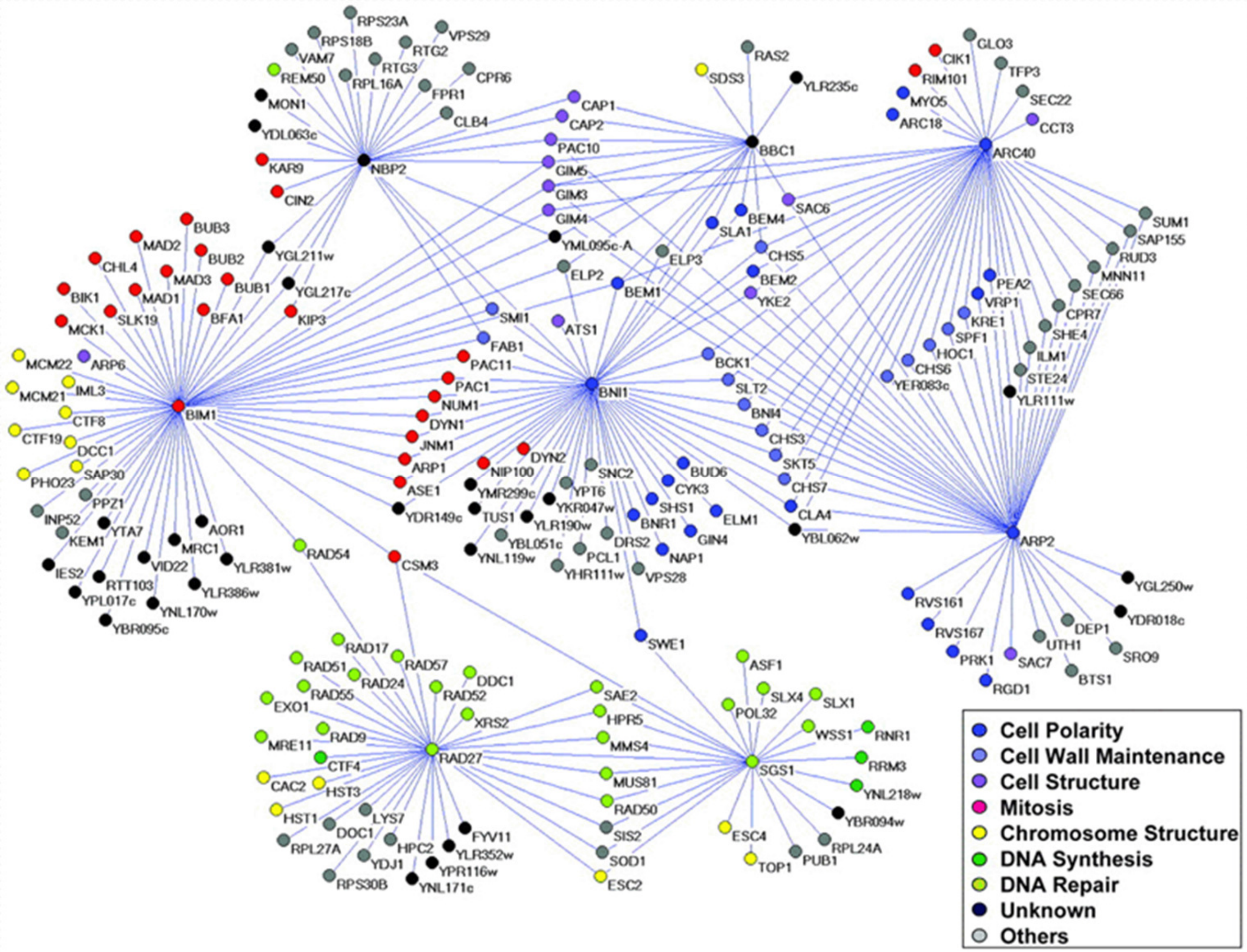
Scientific collaboration network



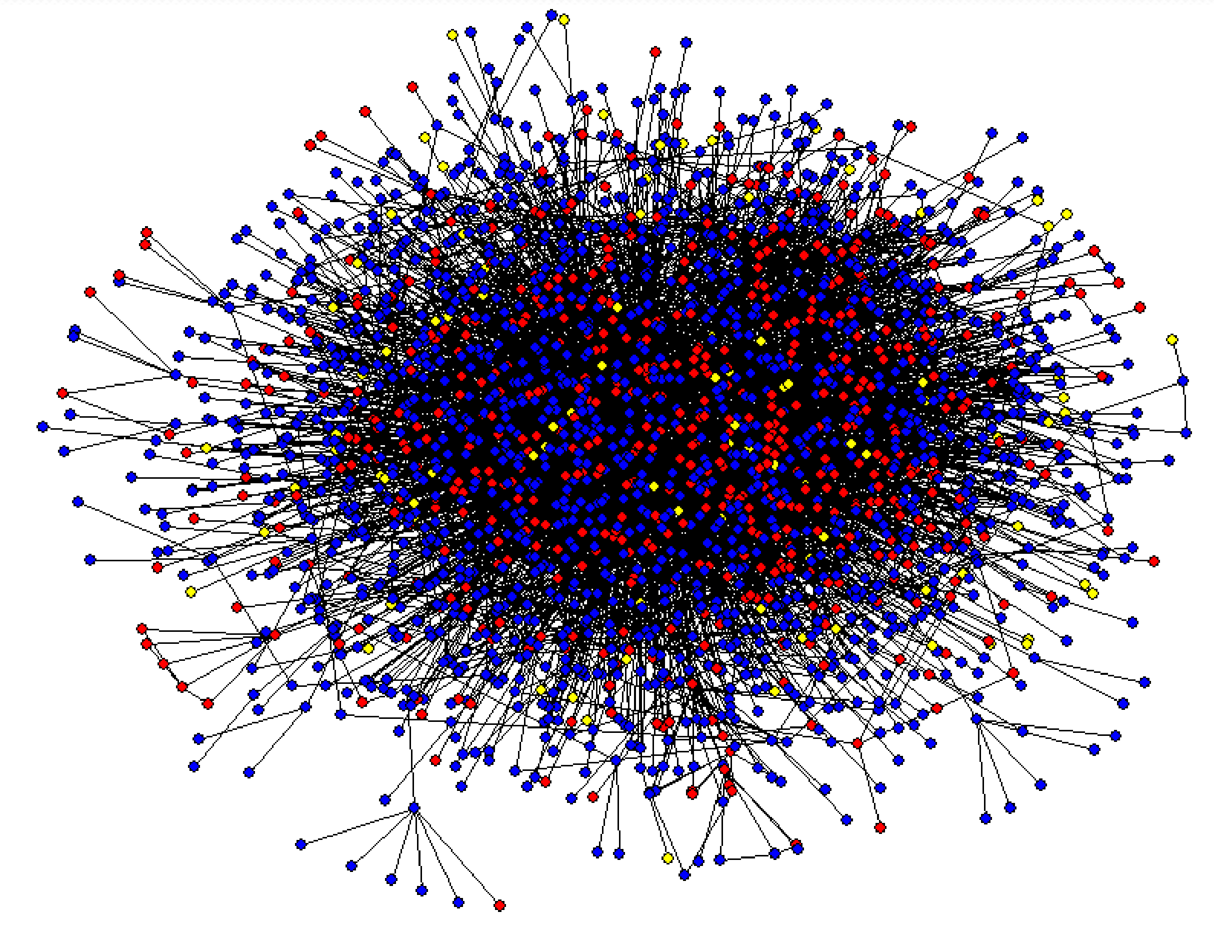
Business ties in US biotech-industry



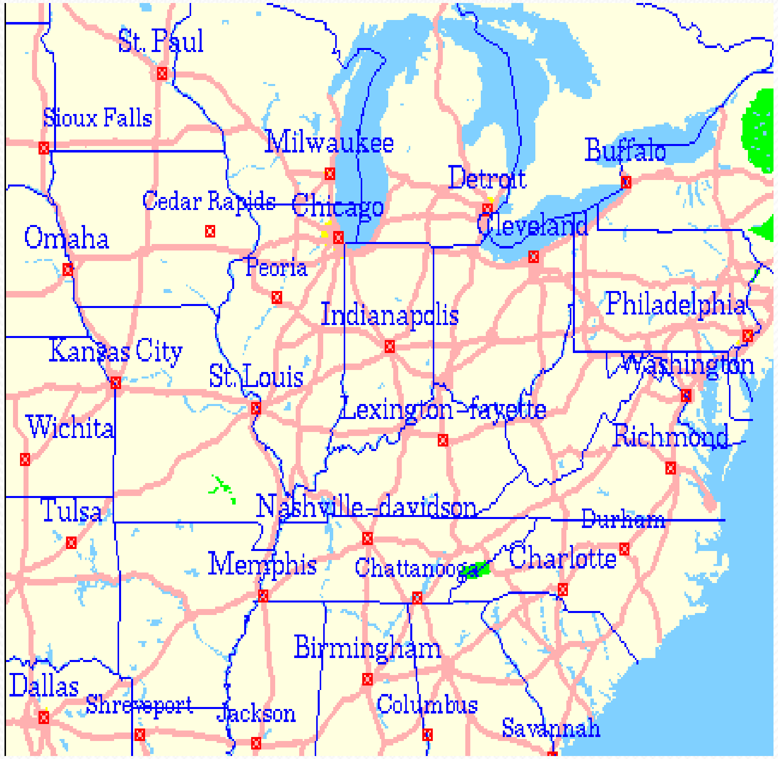
Genetic interaction network

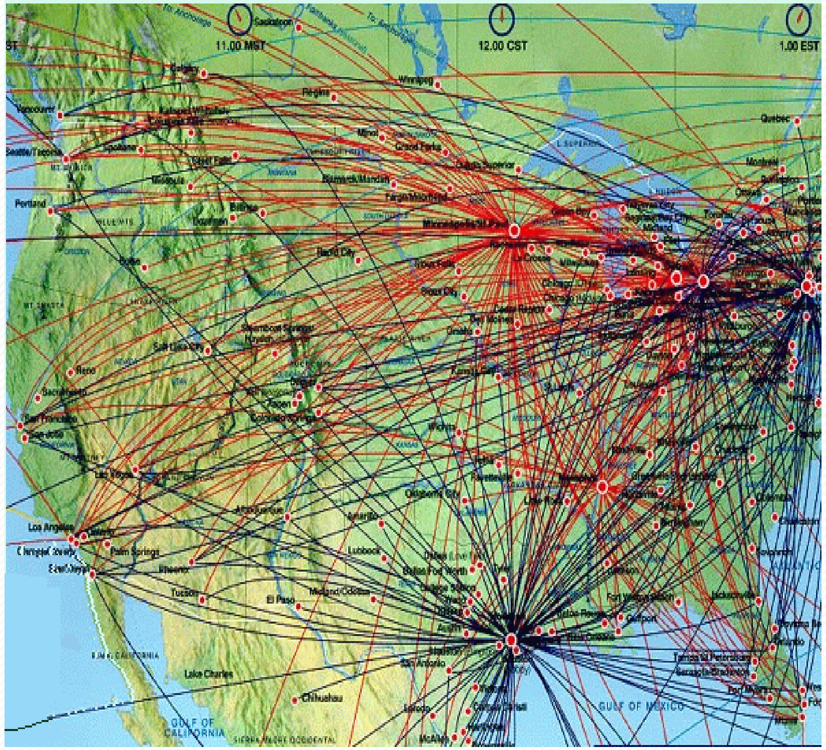


Protein-Protein Interaction Networks

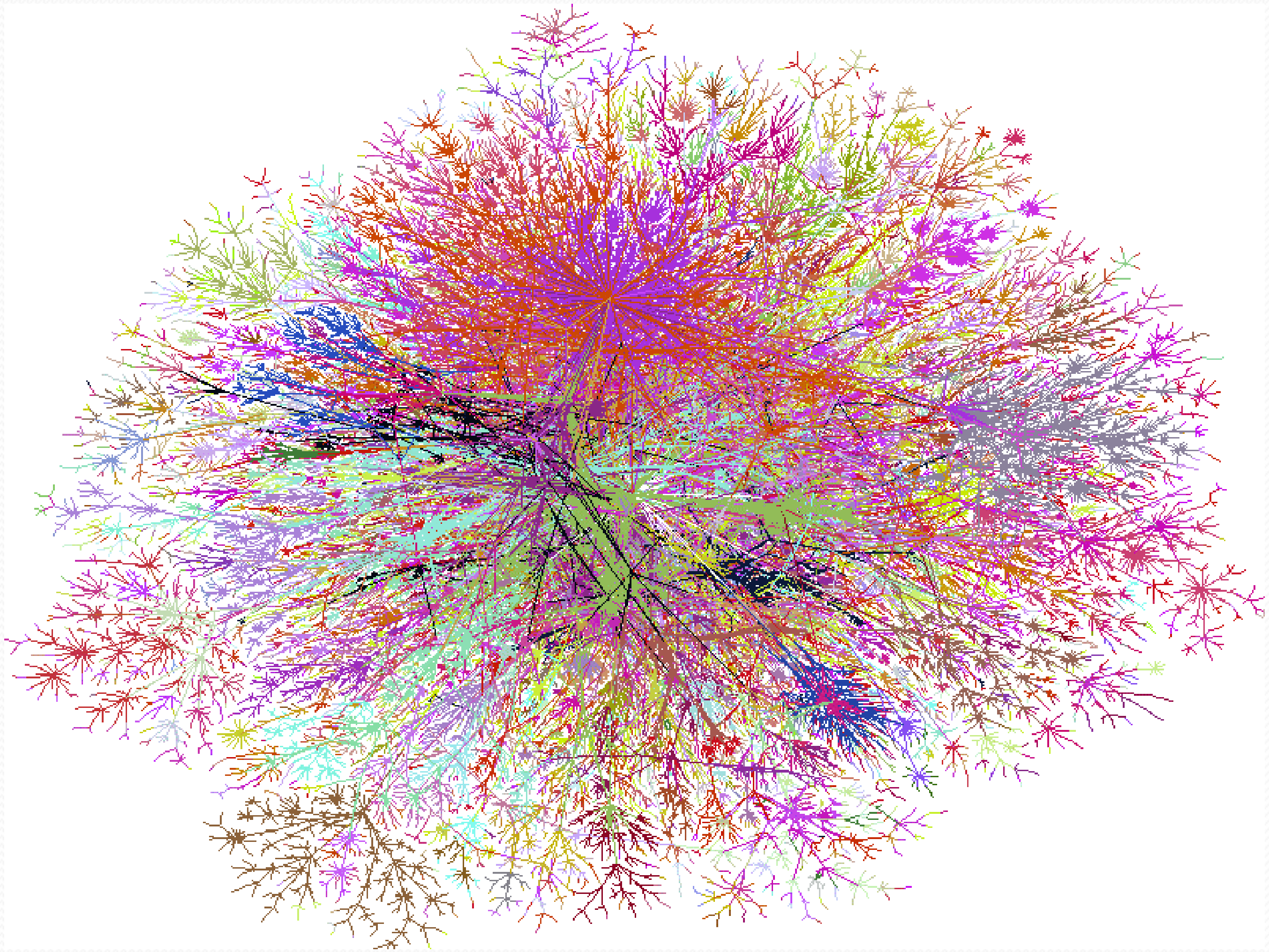


Transportation Networks

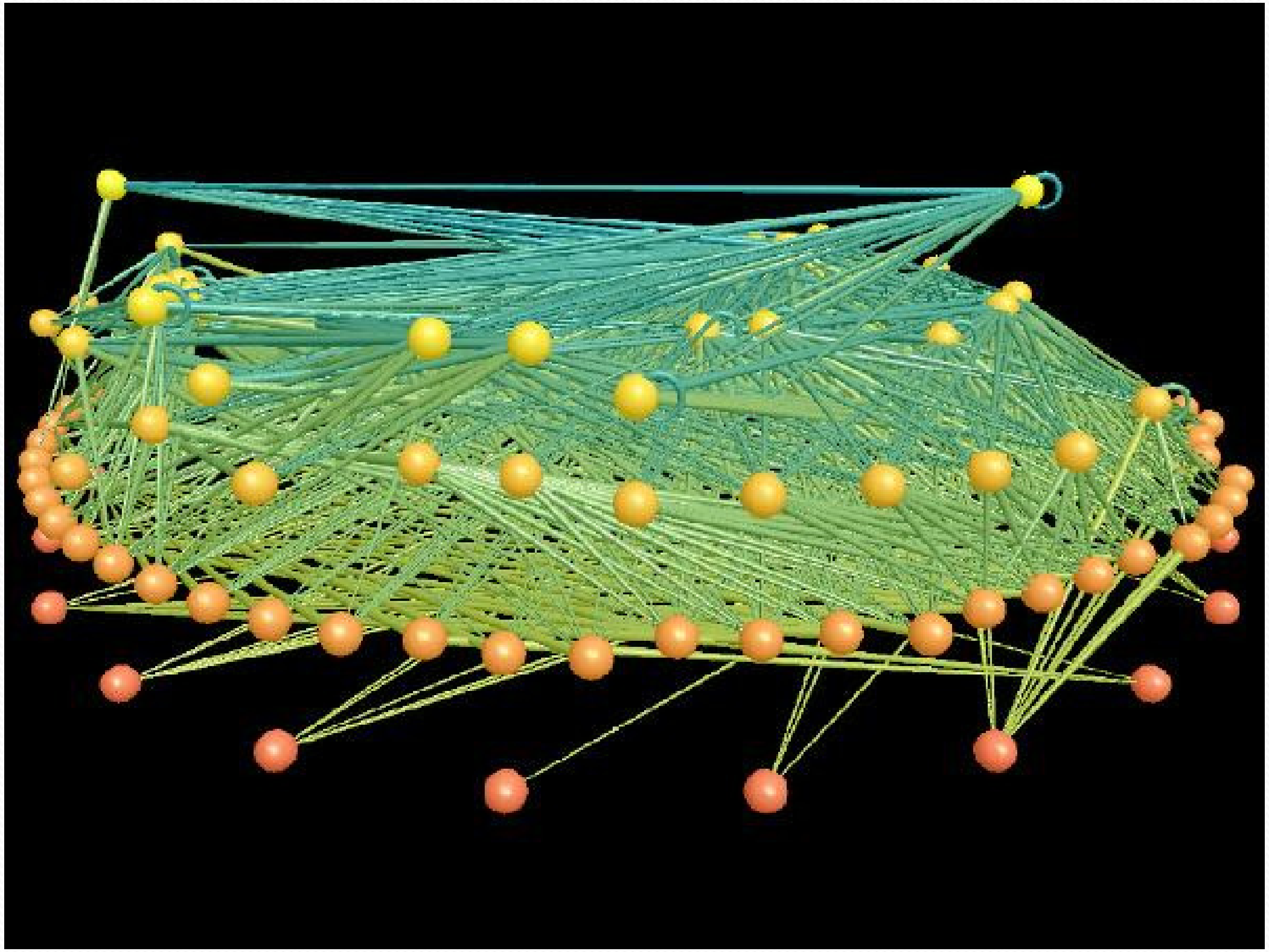




Internet



Ecological Networks



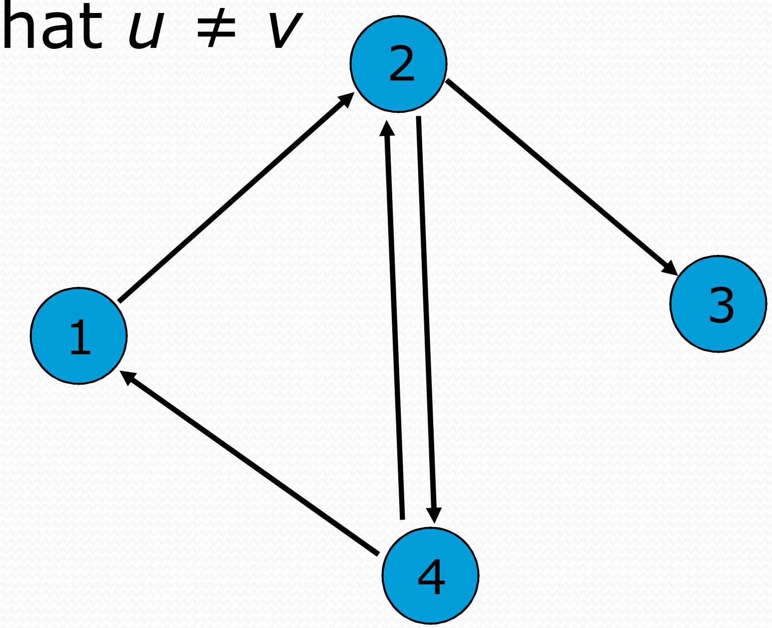
Graphs $\leftrightarrow$ Networks

Graph (Network)	Vertexes (Nodes)	Edges (Arcs)	Flow
Communications	Telephones exchanges, computers, satellites	Cables, fiber optics, microwave relays	Voice, video, packets
Circuits	Gates, registers, processors	Wires	Current
Mechanical	Joints	Rods, beams, springs	Heat, energy
Hydraulic	Reservoirs, pumping stations, lakes	Pipelines	Fluid, oil
Financial	Stocks, currency	Transactions	Money
Transportation	Airports, rail yards, street intersections	Highways, railbeds, airway routes	Freight, vehicles, passengers

Graph Theory : Terminology

Directed Graph (Digraph)

An edge $e \in E$ of a directed graph is represented as an ordered pair (u,v), where $u, v \in V$. Here u is the initial vertex and v is the terminal vertex. Also assume here that $u \neq v$

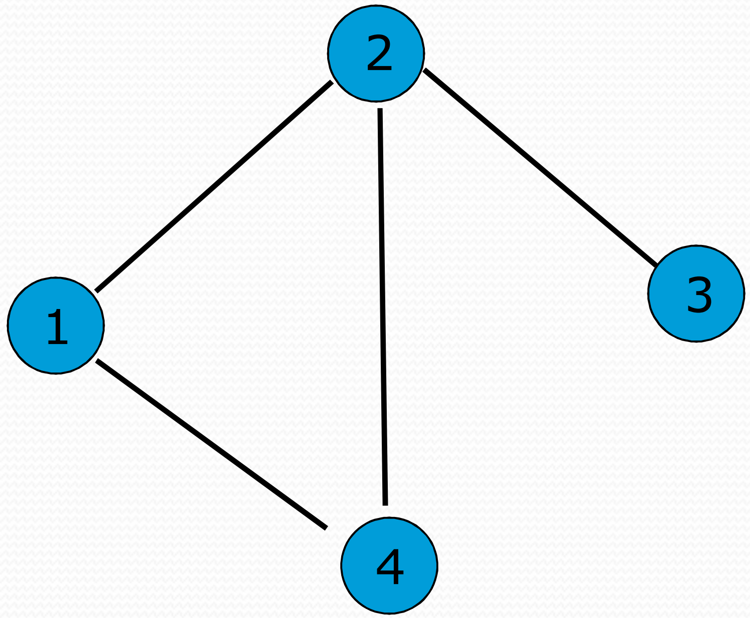


$V = \{ 1, 2, 3, 4\}$, $| V | = 4$

$E = \{(1,2), (2,3), (2,4), (4,1), (4,2)\}$, $| E | = 5$

Undirected Graph

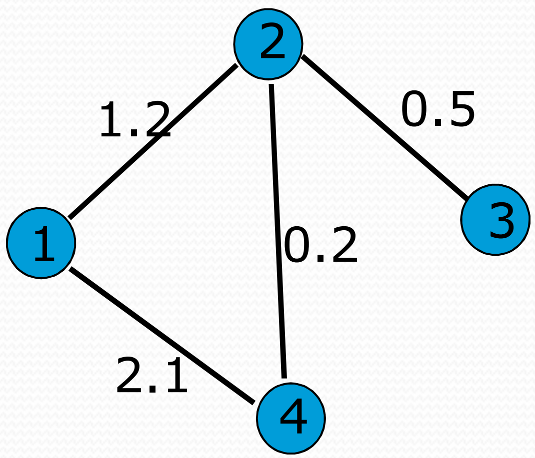
An edge $e \in E$ of an undirected graph is represented as an unordered pair *(u,v)=(v,u)*, where $u, v \in V$. Also assume that $u \neq v$

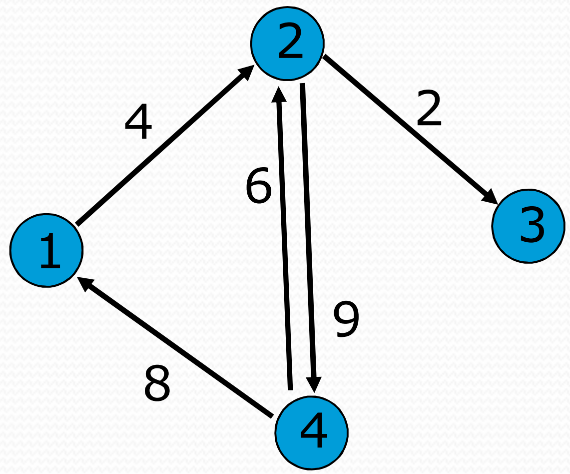


$V = \{ 1, 2, 3, 4\}, | V | = 4$
$E = \{(1,2), (2,3), (2,4), (4,1)\}, | E | = 4$

Weighted Graph

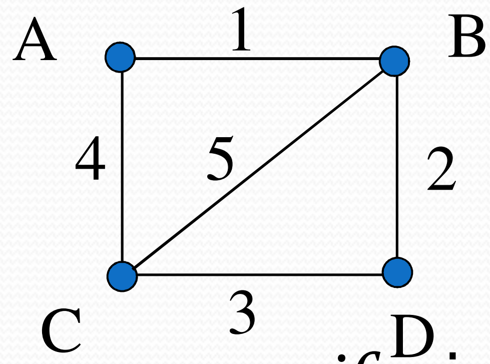
A *weighted graph* is a graph for which each edge has an associated *weight*, usually given by a *weight function* $w: E \rightarrow R$





Adjacent, neighbor, incident

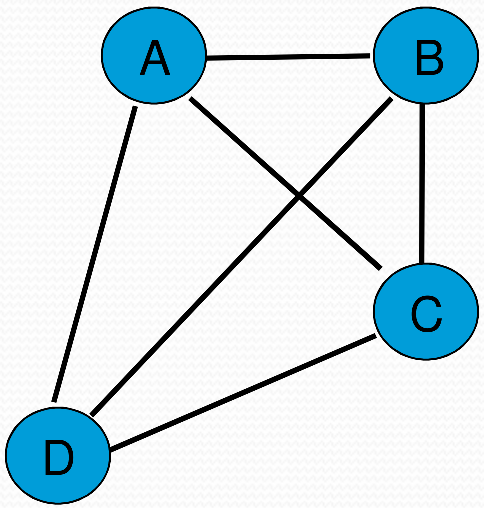
- Two vertices are *adjacent* and are *neighbors* if they are the endpoints of an edge
- Example:
  - *A* and *B* are adjacent
  - *A* and *D* are not adjacent



- The edge e_i is said to be ***incident upon*** v_j, v_k *if* e_i is an edge whose endpoints are (v_j, v_k), e.g., edge *1* is ***incident upon*** *A*,*B*.
- ***Degree*** of a vertex v_k is the number of edges incident upon v_k. It is denoted as $d(v_k)$. e.g., $d(\mathrm{A}) = 2$

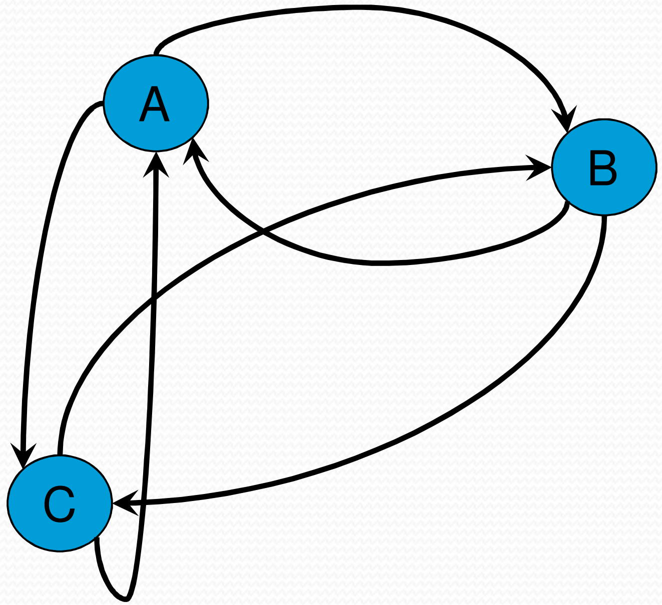
Complete Graphs

A *complete graph* is an undirected/directed graph in which every pair of vertices is *adjacent*. If (*u*, *v*) is an edge in a graph *G*, we say that vertex *v* is *adjacent* to vertex *u*.



4 nodes and (4*3)/2 edges

V nodes and *V**(*V*-1)/2 edges

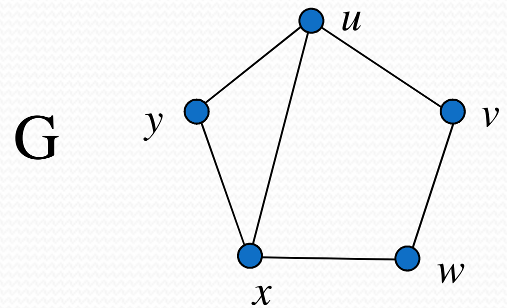


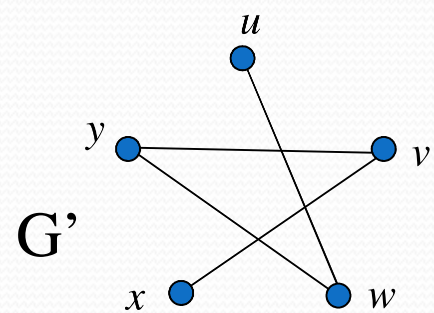
3 nodes and 3*2 edges

V nodes and *V**(*V*-1) edges

Complement

- ***Complement of G***: The complement G' of a simple graph G :
  - A simple graph
  - $V(G') = V(G)$
  - $E(G') = \{ uv \mid uv \notin E(G) \}$

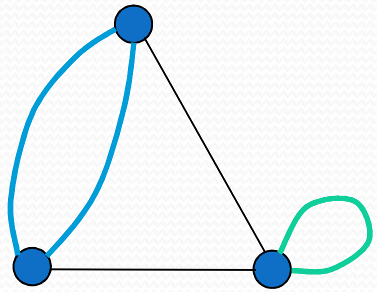




Loop, Multiple edges, Simple Graph

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

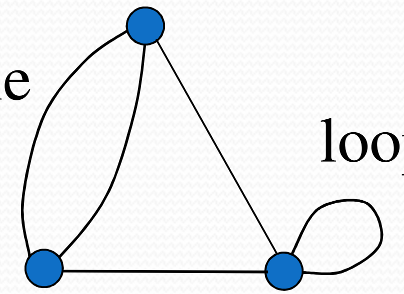
Multiple edges



loop

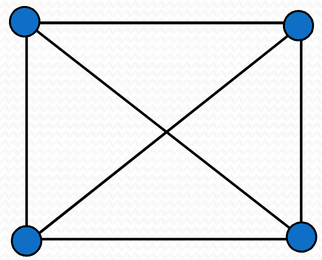
- *Simple graph* : A graph has no loops or multiple edges

Multiple edges



loop

It is not simple.

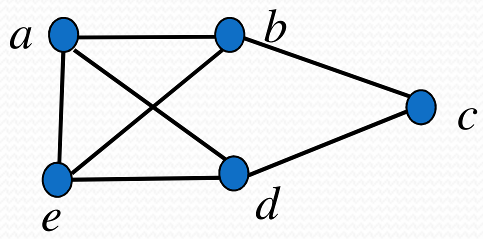


It is a simple graph.

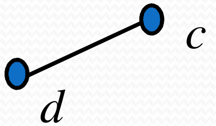
Subgraphs

- A ***subgraph*** of a graph G is a graph H such that:
  - $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$ and
  - The assignment of endpoints to edges in H is the same as in G.
- Example:: H_1, H_2, and H_3 are subgraphs of G

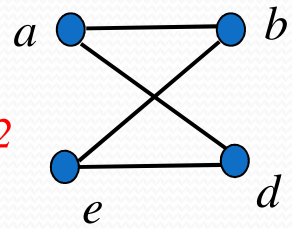
G



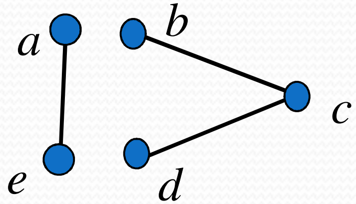
H_1



H_2

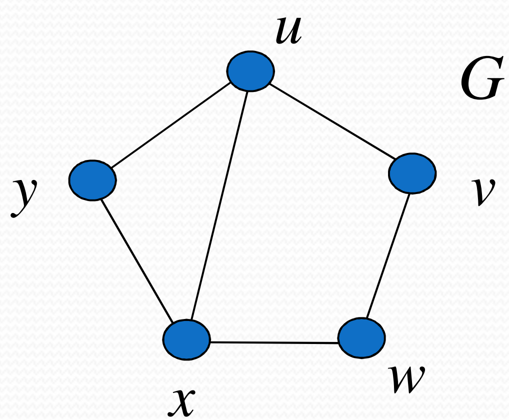


H_3



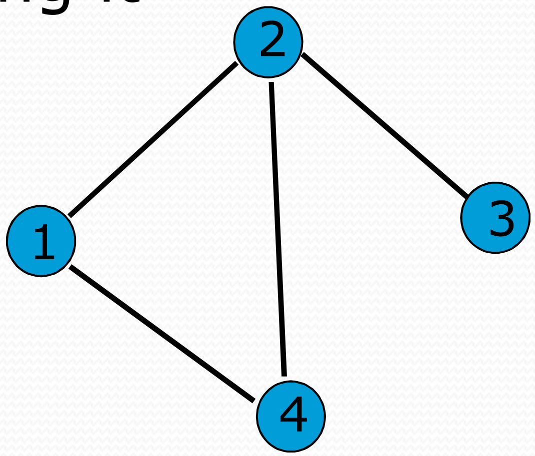
Clique and Independent set

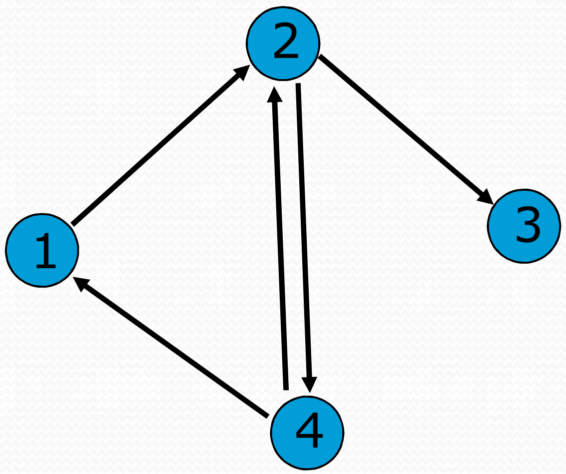
- A ***Clique*** in a graph: a set of pairwise adjacent vertices (a complete subgraph)
- An ***independent set*** in a graph: a set of pairwise nonadjacent vertices
- Example:
  - $\{x, y, u\}$ is a clique in G
  - $\{u, w\}$ is an independent set



Degree of a Vertex

Degree of a vertex in an undirected graph is the number of edges incident on it. In a directed graph, the *out degree* of a vertex is the number of edges leaving it and the *in degree* is the number of edges entering it

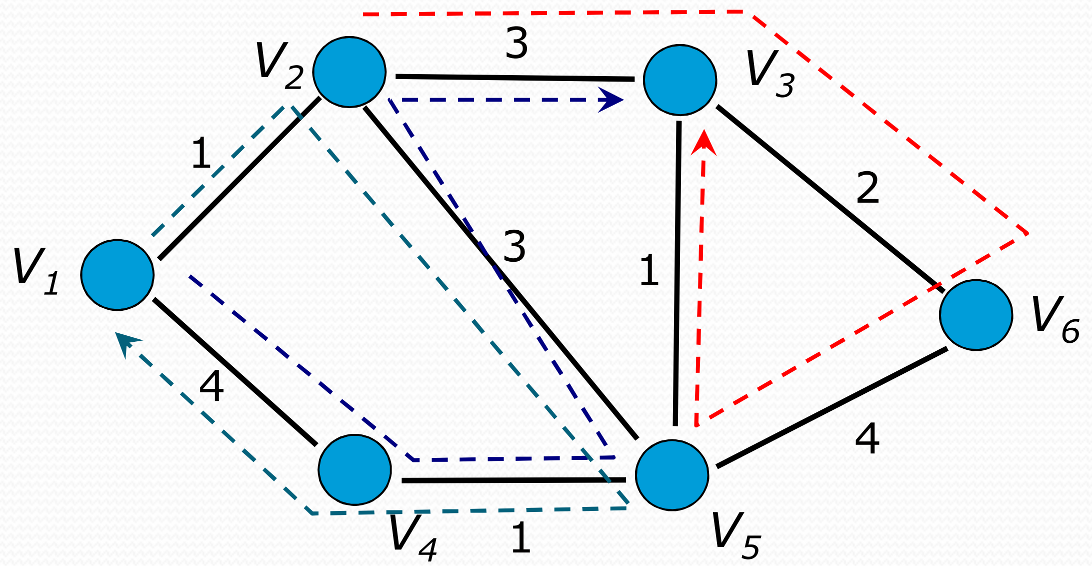




The *degree* of vertex 2 is 3

The *in degree* of vertex 2 is 2 and the *in degree* of vertex 4 is 1

Walks and Paths



A *walk* is an alternating sequence of vertices and edges, *e.g.* (V_2, e_3, V_3, e_2, V_6, e_4, V_5, e_1, V_3)

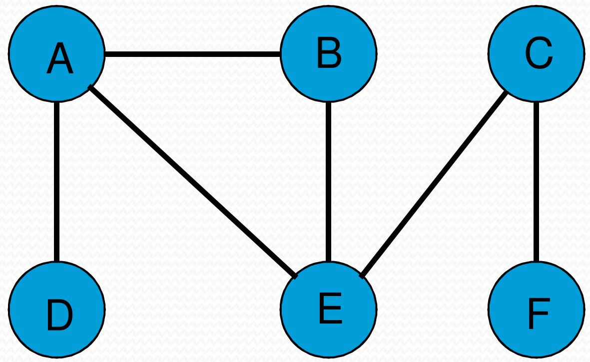
A *simple path* is a walk with no repeated nodes, e.g. (V_1, V_4,V_5, V_2,V_3)

A *cycle* is a closed path (v_1, v_2,..., v_L) where v_1=v_L with no other nodes repeated and L>3, e.g. (V_1, V_2,V_5, V_4,V_1)

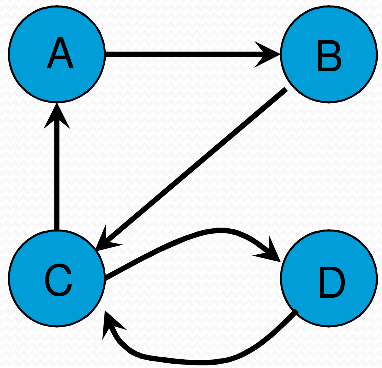
A graph is called *cyclic* if it contains a cycle; otherwise it is called *acyclic*

Connected Graphs

An undirected graph is *connected* if you can get from any node to any other by following a sequence of edges OR any two nodes are connected by a path



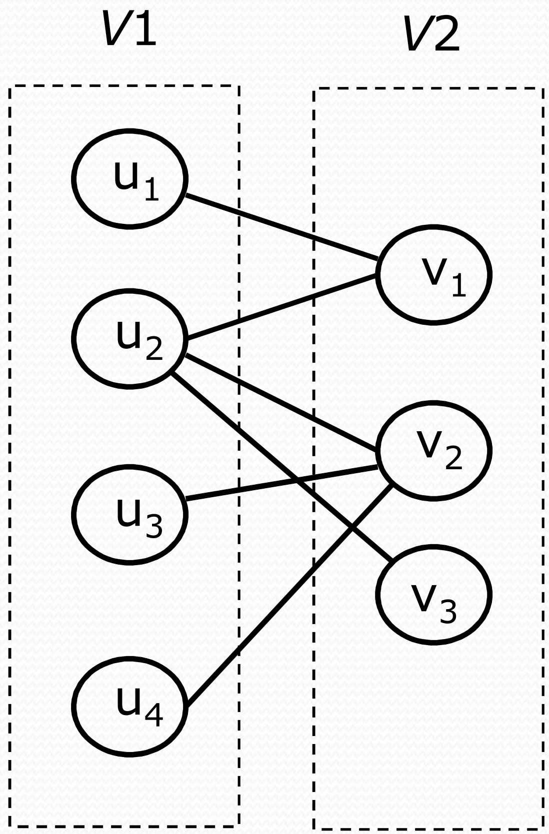
A directed graph is *strongly connected* if there is a directed path from any node to any other node



- A graph is *sparse* if $|E| \approx |V|$
- A graph is *dense* if $|E| \approx |V|^2$

Bipartite Graph

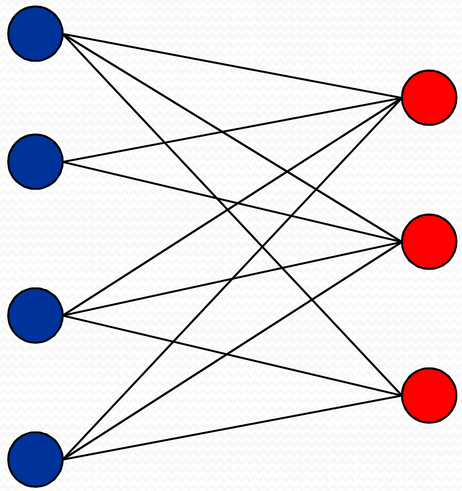
A *bipartite graph* is an undirected graph $G = (V,E)$ in which V can be partitioned into 2 sets $V1$ and $V2$ such that $(u,v) \in E$ implies either
$u \in V1$ and $v \in V2$
OR
$v \in V1$ and $u \in V2$.



An example of bipartite graph application to telecommunication problems can be found in, C.A. Pomalaza-Ráez, "A Note on Efficient SS/TDMA Assignment Algorithms," *IEEE Transactions on Communications*, September 1988, pp. 1078-1082.

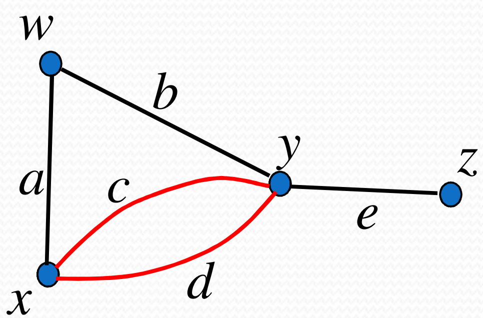
Applications of Bipartite Graph

- Stable marriage: men = red, women = blue.
- Scheduling: machines = red, jobs = blue.
- Metabolic networks: metabolites = blue, enzymes = red.



Adjacency matrix

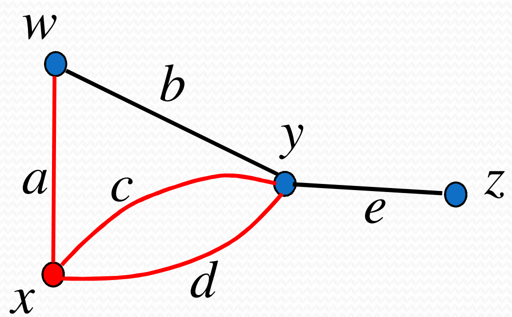
- Let $G = (V, E)$, $|V| = n$ and $|E|=m$
- The *adjacency matrix* of G written $A(G)$, is the n-by-n matrix in which entry $a_{i,j}$ is the number of edges in G with endpoints $\{v_i, v_j\}$.



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

Incidence Matrix (undirected)

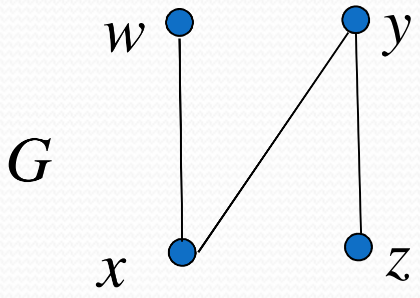
- Let $G = (V, E)$, $|V| = n$ and $|E|=m$
- The *incidence matrix* $M(G)$ is the n-by-m matrix in which entry $m_{i,j}$ is 1 if v_i is an endpoint of e_i and otherwise is 0. Note that for digraphs, the entry is 1 for "outward" connection, and -1 for "inward".

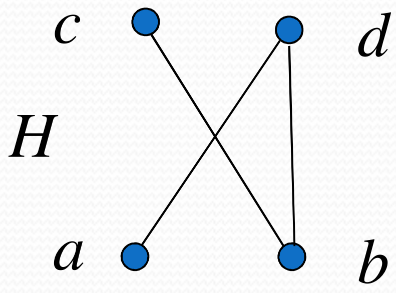


$$\begin{array}{c} \\ w \\ x \\ y \\ z \end{array}\begin{array}{c} \begin{matrix} a & b & c & d & e \end{matrix} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

Isomorphism

- An *isomorphism* from a simple graph G to a simple graph 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 "*G is isomorphic to H*", written $G \cong H$





f_1: w x y z
c b d a

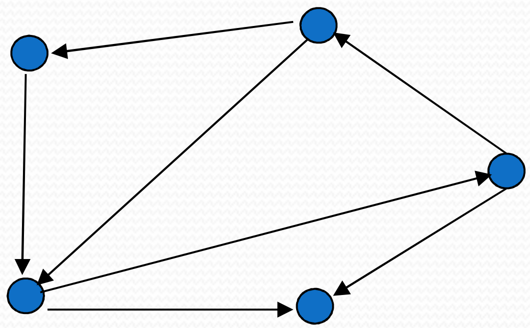
f_2: w x y z
a d b c

Directed Graph and Its edges

- A ***directed graph*** or ***digraph*** $\mathbf{G}$ is a triple:
  - A ***vertex set*** $\mathbf{V(G)}$,
  - An ***edge set*** $\mathbf{E(G)}$, and
  - A function assigning each edge an ordered pair of vertices.
    - The first vertex of the ordered pair is the ***tail*** of the edge
    - The second is the ***head***
    - Together, they are the ***endpoints***.
- An edge is said to be ***from*** its tail ***to*** its head.
  - The terms "head" and "tail" come from the arrows used to draw digraphs.

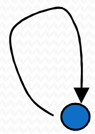
Directed Graph and its edges

- As with graphs, we
  - assign each vertex a point in the plane and
  - each edge a curve joining its endpoints.
- When drawing a digraph, we give the curve a direction from the tail to the head.
- When a digraph models a relation, each ordered pair is the (head, tail) pair for at most one edge.
  - In this setting as with simple graphs, we ignore the technicality of a function assigning endpoints to edges and simply treat an edge as an ordered pair of vertices

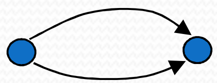


Loop and multiple edges in directed graph

- In a graph, a ***loop*** is an edge whose endpoints are equal.
- ***Multiple edges*** are edges having the same ordered pair of endpoints.
- A digraph is ***simple*** if each ordered pair is the head and tail of the most one edge; one loop may be present at each vertex.
- In the simple digraph, we write uv for an edge with tail u and head v.
  - If there is an edge form u to v, then v is a ***successor*** of u, and u is a ***predecessor*** of v.
  - We write $u \rightarrow v$ for "there is an edge from u to v".



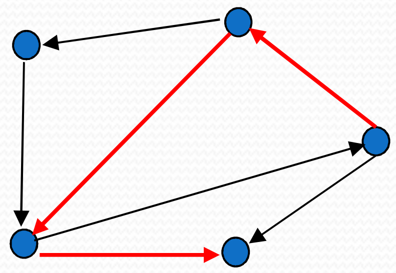
Loop



Multiple edges

Path and Cycle in Digraph

- A digraph is a *path* if it is a simple digraph whose vertices can be linearly ordered so that there is an edge with tail u and head v if and only if v immediately follows u in the vertex ordering.
- A *cycle* is defined similarly using an ordering of the vertices on the cycle.

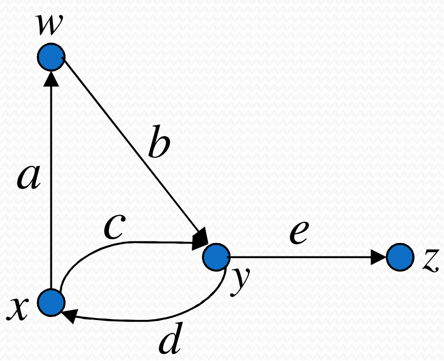


Adjacency Matrix and Incidence Matrix of a Digraph

- In the ***adjacency matrix*** $A(G)$ of a digraph G, the entry in position i, j is the number of edges from v_i to v_j.
- In the ***incidence matrix*** $M(G)$ of a loopless digraph G, we set $m_{i,j}$=+1 if v_i is the tail of e_j and $m_{i,j}$= -1 if v_i is the head of e_j.

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

$A(G)$



G

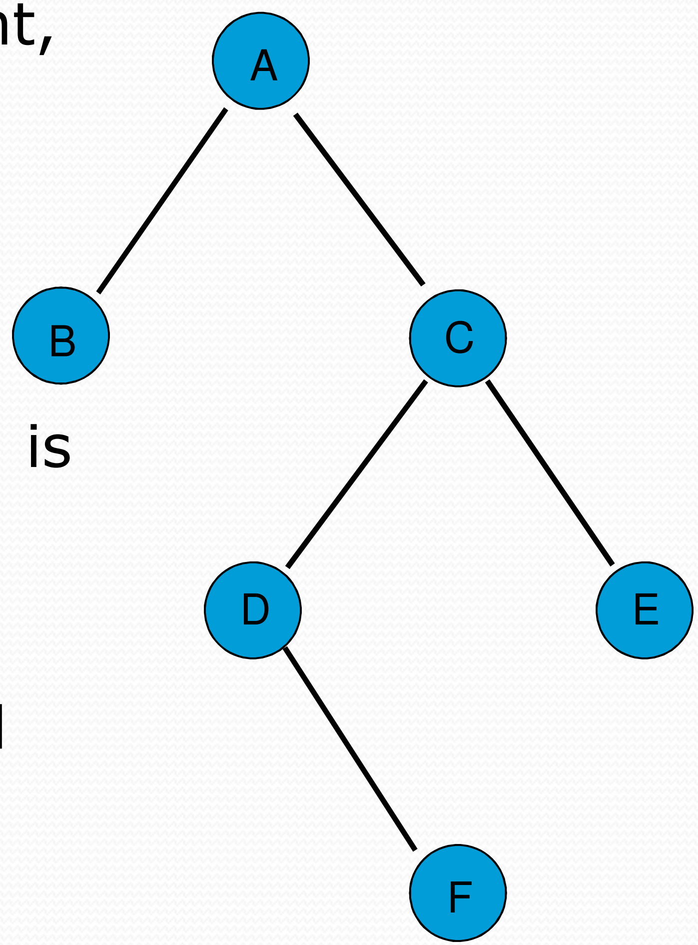
$$\begin{array}{c} \\ w \\ x \\ y \\ z \end{array} \begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{bmatrix} -1 & +1 & 0 & 0 & 0 \\ +1 & 0 & +1 & -1 & 0 \\ 0 & -1 & -1 & +1 & +1 \\ 0 & 0 & 0 & 0 & -1 \end{bmatrix} \end{array}$$

$M(G)$

Trees

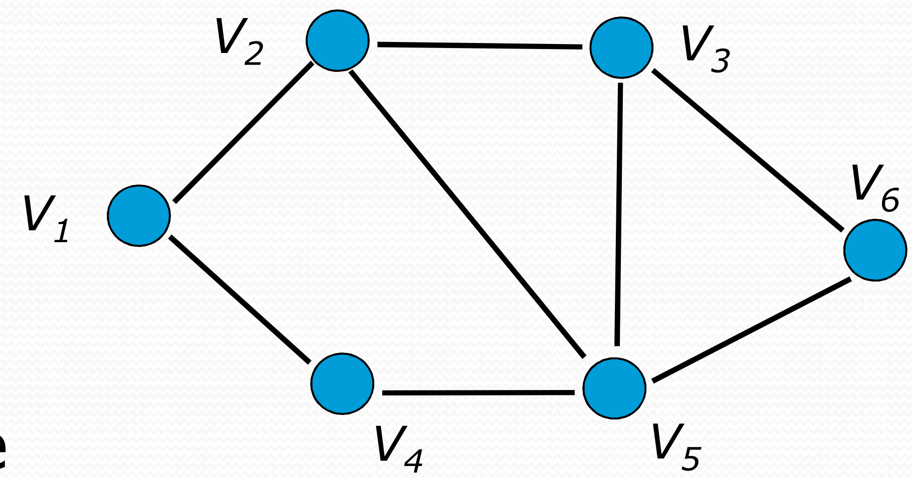
Let $G = (V, E)$ be an undirected graph.
The following statements are equivalent,

1. G is a tree
2. Any two vertices in G are connected by unique simple path
3. G is connected, but if any edge is removed from E, the resulting graph is disconnected
4. G is connected, and $|E| = |V| - 1$
5. G is acyclic, and $|E| = |V| - 1$
6. G is acyclic, but if any edge is added to E, the resulting graph contains a cycle



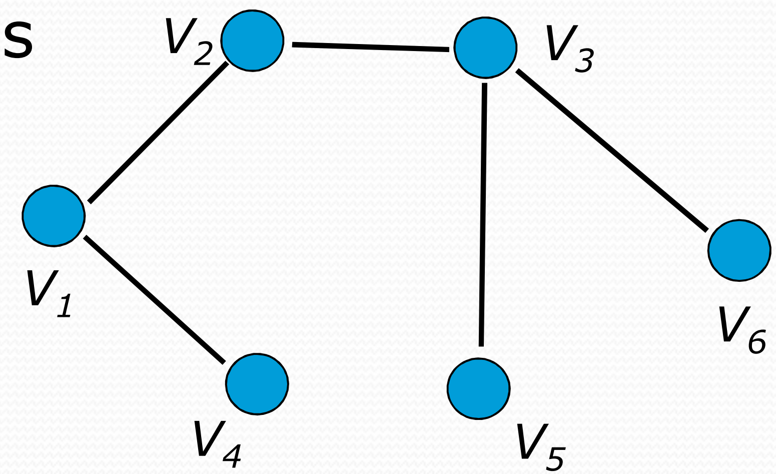
Spanning Tree

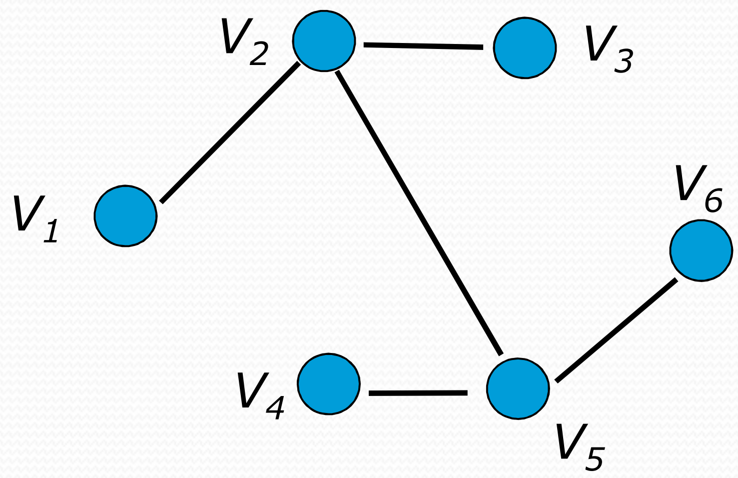
A tree (T) is said to span $G = (V,E)$ if $T = (V,E')$ and $E' \subseteq E$



For the graph shown on the right two possible spanning trees are shown below

For a given graph there are usually several possible spanning trees





Planarity

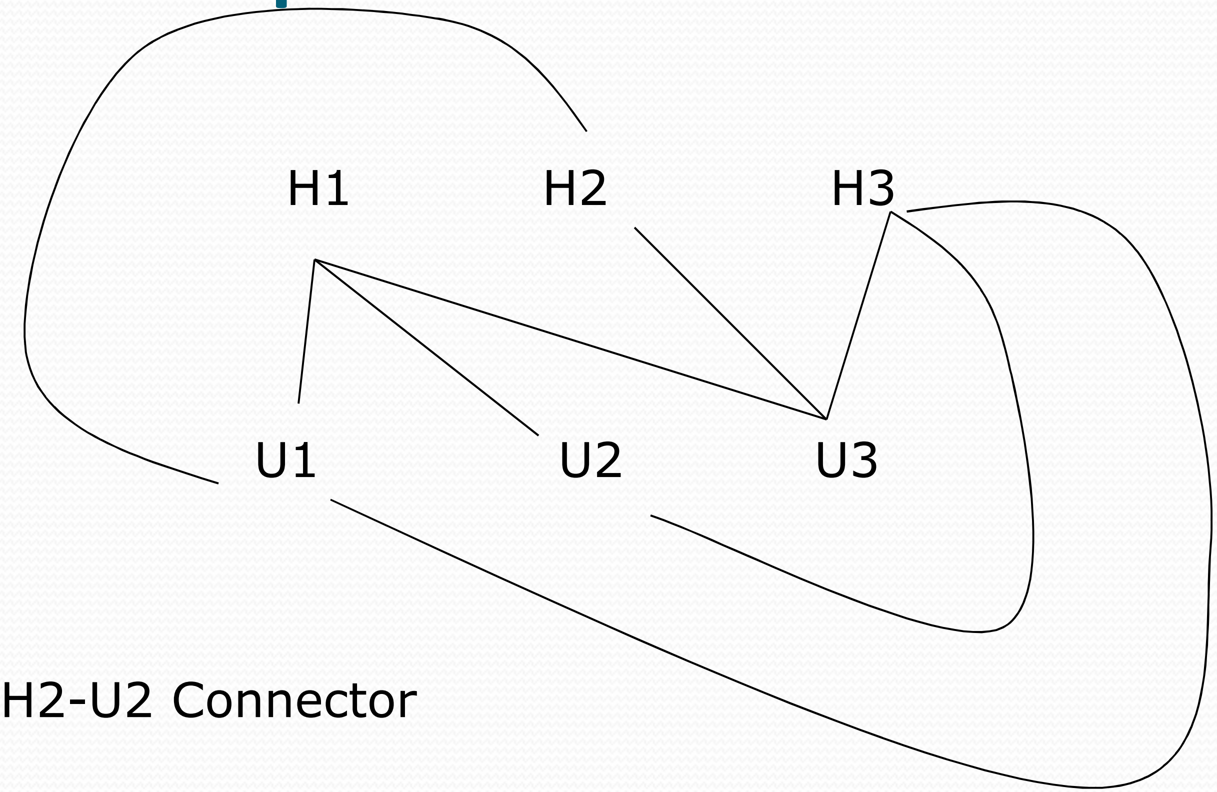
- Another problem in graph theory also has a simple solution that has major consequences.
- The question of planarity refers to whether a graph can be drawn in the plane without any edges crossing any other ones.
- Example : Connect 3 houses to 3 utilities

H1 H2 H3

U1 U2 U3

Draw edges from each U to each H without crossing edges.

An Attempted Solution

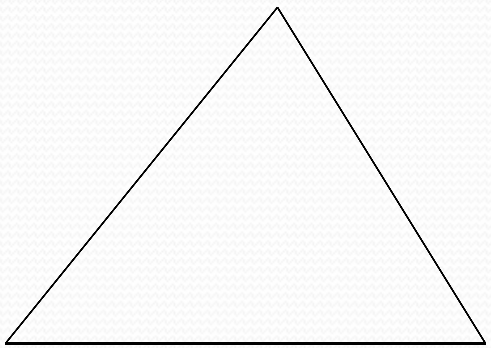


No H2-U2 Connector

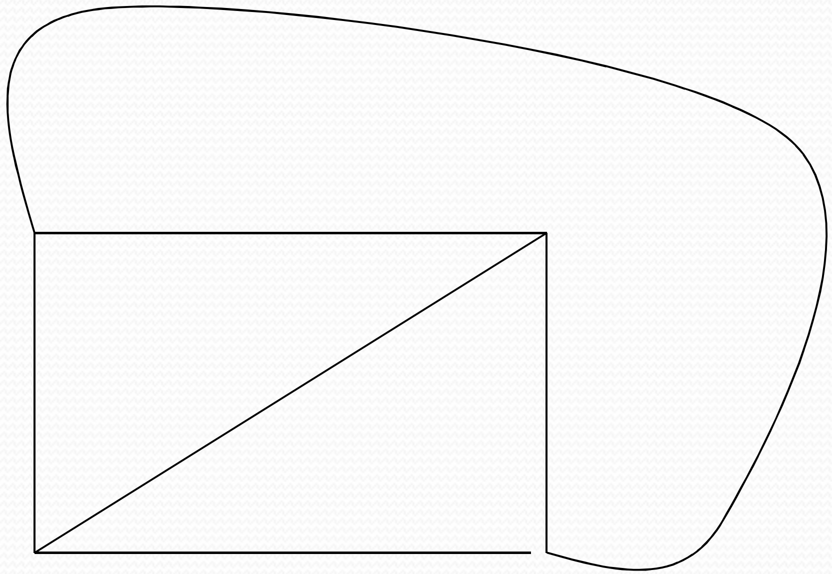
The graph connecting all vertices of a set of three to all vertices to another set of three is called $K_{3,3}$
This graph is not planar. That is to say, it is not possible to draw it in the plane with no edges crossing others.

K_n

- K_n is called the complete graph on n vertices. It is the graph one gets by starting with n vertices and drawing an edge between each pair.
- K_n is planar or not depending upon n.



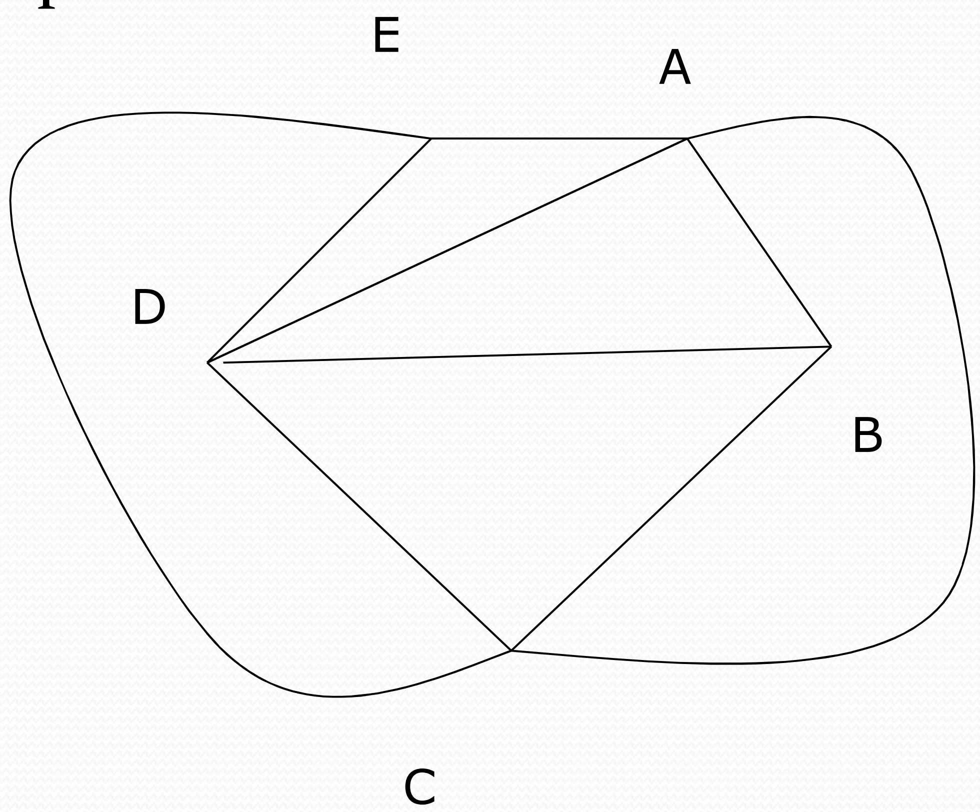
n=3



n=4

K_n Is Not Planar for n > 4

- As shown below, K_5 is not planar.
- If n is bigger than or equal to 5 then K_n couldn't possibly be planar.



Planar Graphs --- A Theorem

- All non-planar graphs (those that cannot be drawn in the plane without crossing edges) contain either a copy of K_5 or $K_{3,3}$ as a sub-graph.
- Conversely, if neither K_5 nor $K_{3,3}$ is to be found embedded anywhere inside a graph, that graph will be planar.

Why's it important?

- Any physical interpretation of a graph that wants to avoid crossings of edges needs to take this into account.
- The most obvious examples are printed circuit boards and micro-chips