Complex Networks:

Lecture 3a: Introduction to Graph Theory

EE 6605

Instructor: G Ron Chen

What is a Graph?

- A ***graph*** is a diagrammatical representation of some physical structure

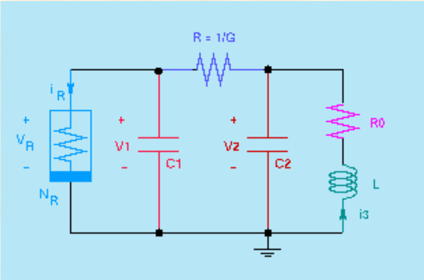
  such as:

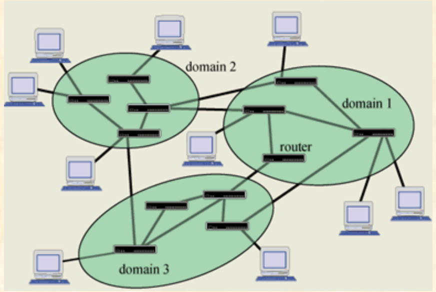
  a circuit

  a computer network

  a human relationship network

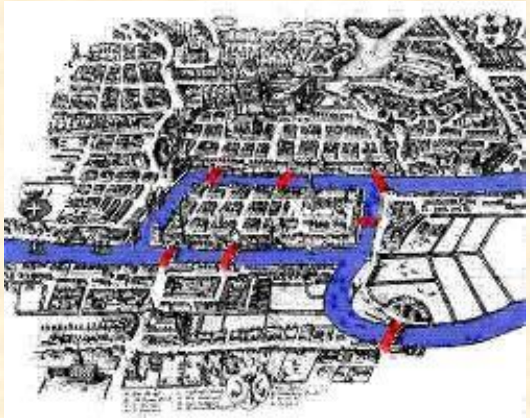
  … and so on.

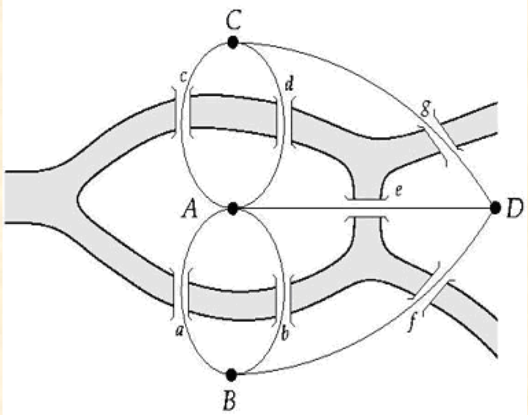




Beginning of Graph Theory

Euler (1707-1783) proved that the Königsburg seven-bridge problem has no solution





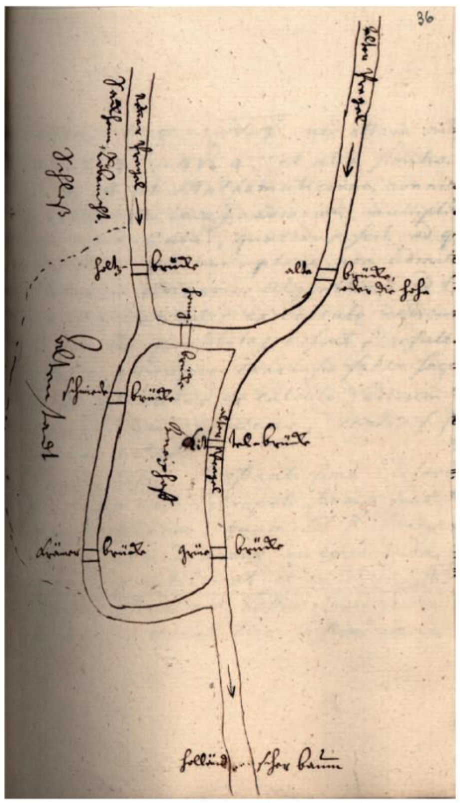
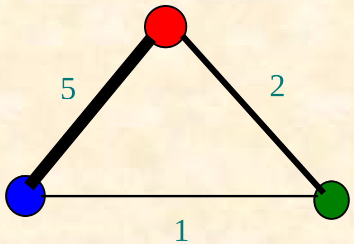


Figure 1: Ehler's drawing of Königsberg, 1736

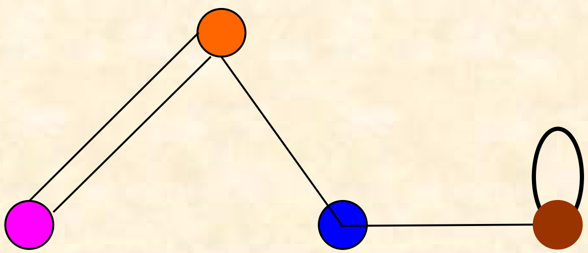
Notation

- Let G be a non-empty graph with at least one **node** (or, vertex).
- In a non-isolated case, G has at least one **edge** (or, link); thus, it has at least two nodes.
- Let $N(G)$ and $M(E)$ denote the set of its nodes and the set of its edges, respectively.
- In general, $N(G)$ and $M(E)$ are finite sets.
- Such a non-empty pair $(N(G), M(E))$ is referred to as a **simple graph**

Examples of General (non-simple) Graphs



weighted graphs
sometimes considered



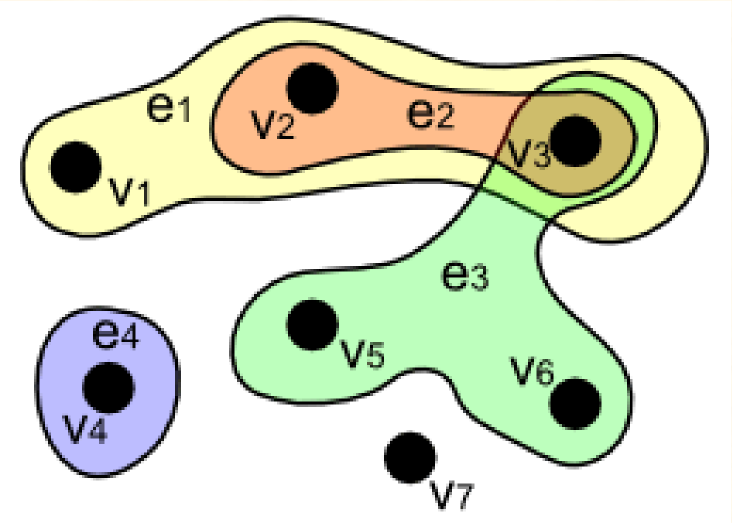
graph with self-loop
or multiple edges
are not allowed

Convention:  not allowed

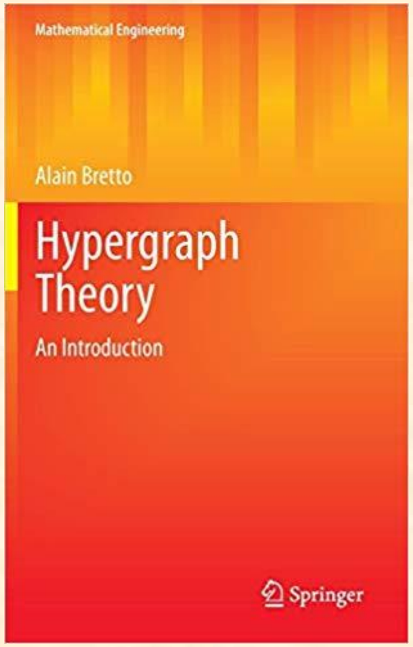
When a node is removed, all its edges will also be removed

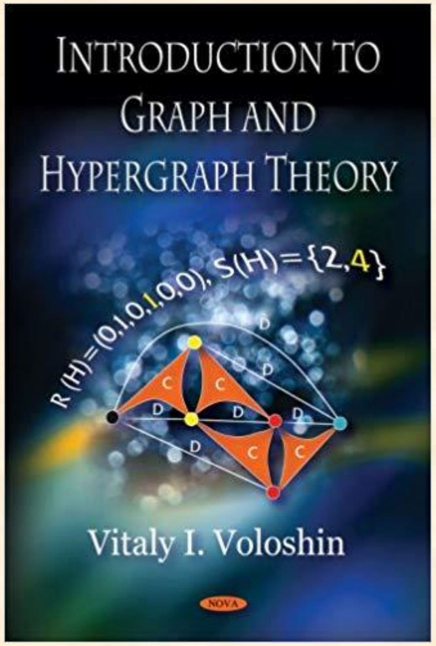
Hyper-Graph

- Concept: A generalization of a graph, where an edges can connect any number of nodes.
- Formally, a hypergraph H is a pair $H = (X,E)$ where X is a set of nodes {v} and E is a set of subsets {e} of X (called hyper-edges).

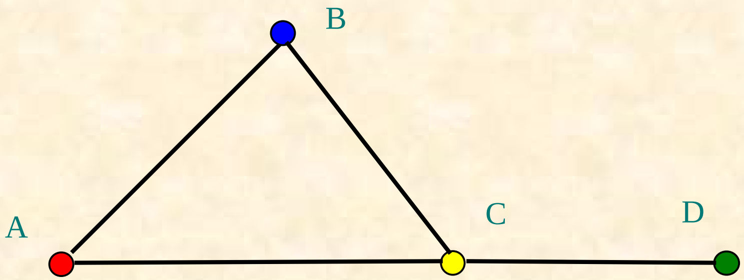


Hyper-graphs are not simple graphs





Examples of Simple Graphs



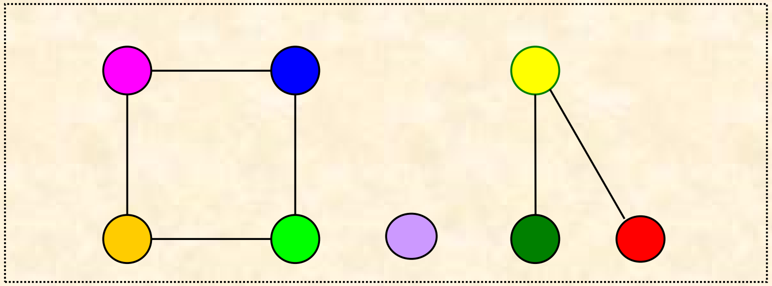
$$N(G) = \{A, B, C, D\}$$

$$M(E) = \{AB, AC, BC, CD\}$$

Subgraph: ABC, AB, D etc. (needed not be connected)

Circuit (Cycle): ABC

Component: A self-connected subgraph, but un-connected with other parts of the same graph



← A simple graph with 3 components

- **Theorem:** *If a simple graph G with N nodes has K components, then the number of edges, M, of G satisfies*

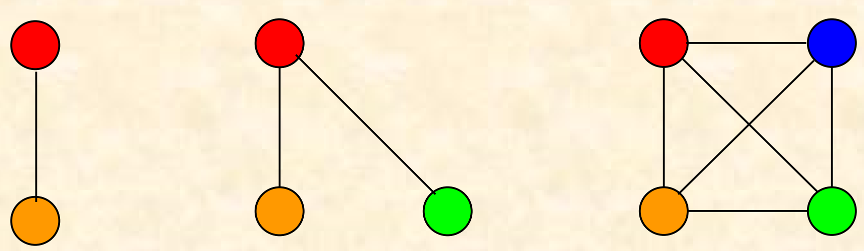
$$N-K \le M \le \frac{1}{2}(N-K)\ (N-K+1)$$

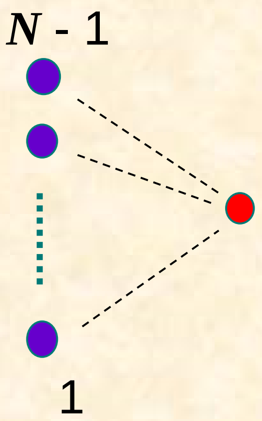
In particular, for a connected graph ($K=1$), it reduces to

$$N-1 \le M \le \frac{1}{2}N(N-1)$$

- ***Proof.*** The general case is proved in the lecture notes, while the case of $K=1$ is obvious: A connected graph with N nodes has at least $N-1$ edges and at most $N(N-1)/2$ edges.

Examples:





A graph of N nodes has $N(N-1)/2$ edges

Corollary: *If a simple graph of N nodes satisfies* $M > \frac{1}{2}(N-1)(N-2)$ *then it must be connected.*

Proof. If not connected, then $K \geq 2$ in $N - K \leq M \leq \frac{1}{2}(N-K)(N-K+1)$

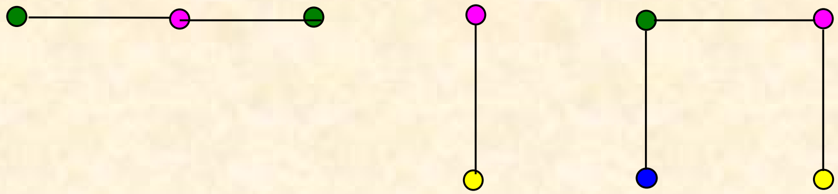
In case of $K = 2$: $N - 2 \leq M \leq \frac{1}{2}(N-1)(N-2)$

But, now it is assumed $M > \frac{1}{2}(N-1)(N-2)$

This is a contradiction.

The simplest connected graph is a chain, which has $M = N - 1$ edges

Examples:



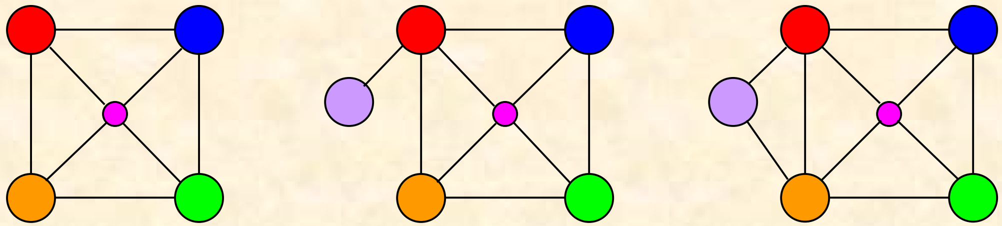
Some Basic Results

- **Theorem** (Handshaking Lemma) *The total node degree of a graph is always an even number.*

- *Proof*. Since every edge joins two nodes, so the total node degree is twice of the number of edges.

- **Corollary:** *In any graph, the number of nodes of odd degrees must be even.*

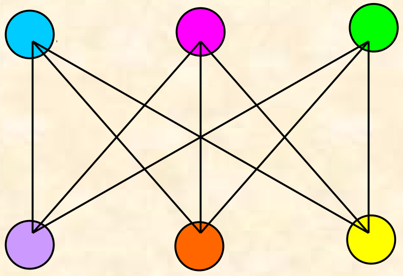
Examples:

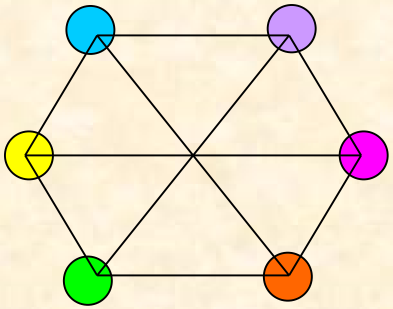


Isomorphism

- Two graphs G_1 and G_2 are said to be **isomorphic**, if there is a one-one correspondence between the nodes of G_1 and those of G_2, with the property that the number of edges joining any two nodes of G_1 is equal to the number of edges joining the two corresponding nodes of G_2.

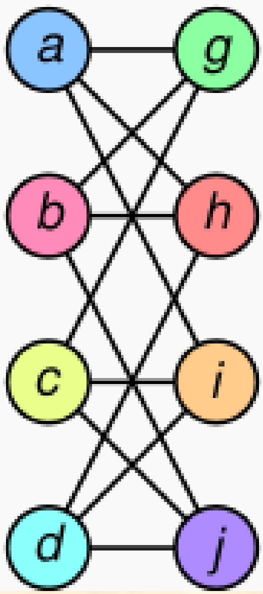
- Example:



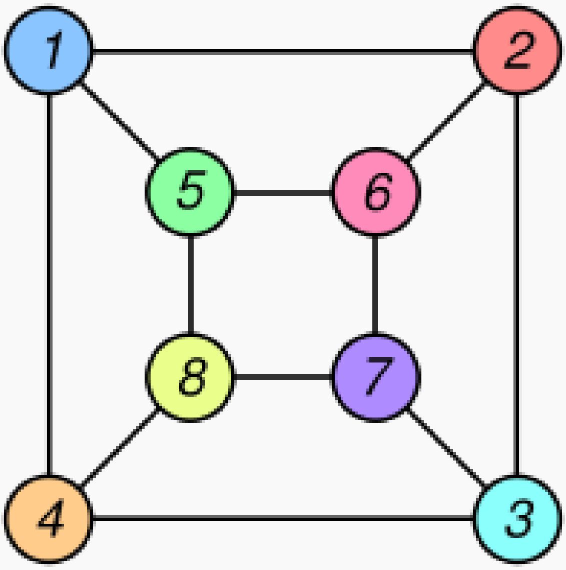


Graph Isomorphism is a Mapping

Graph G



Graph H



An isomorphism between G and H

$f(a) = 1$

$f(b) = 6$

$f(c) = 8$

$f(d) = 3$

$f(g) = 5$

$f(h) = 2$

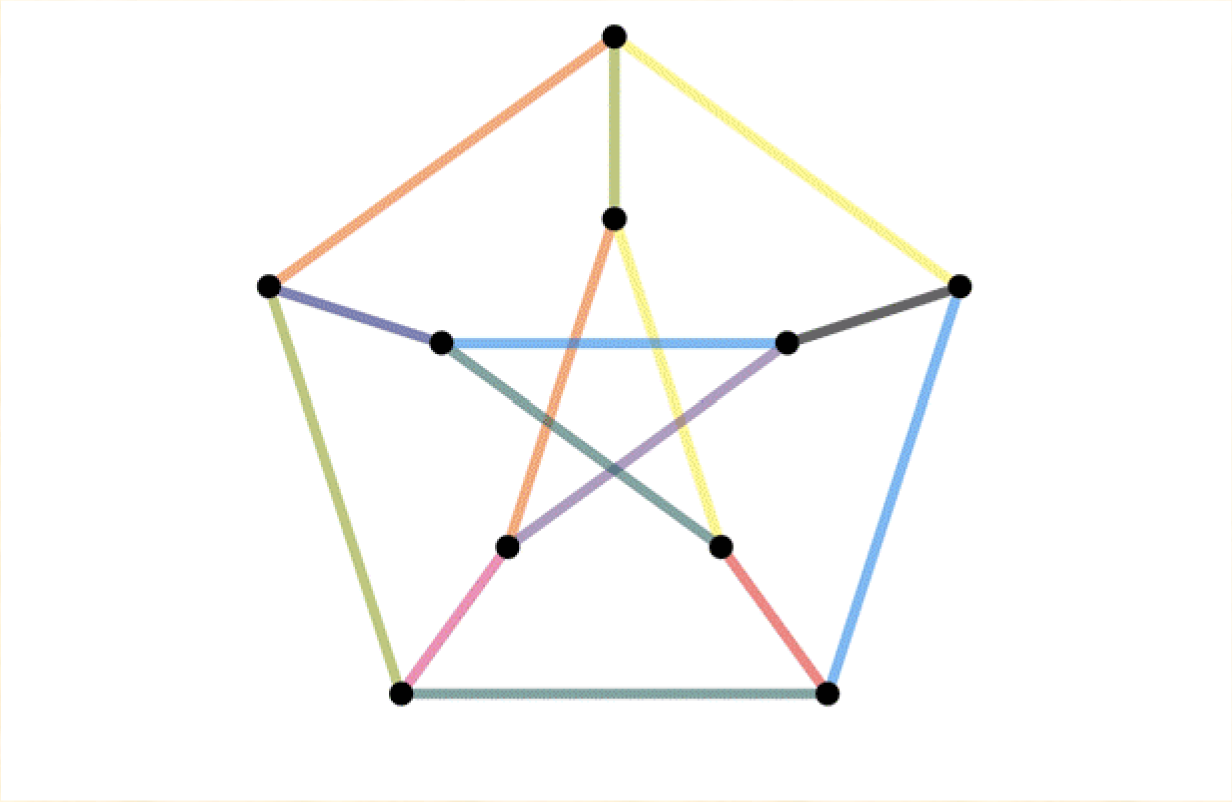
$f(i) = 4$

$f(j) = 7$

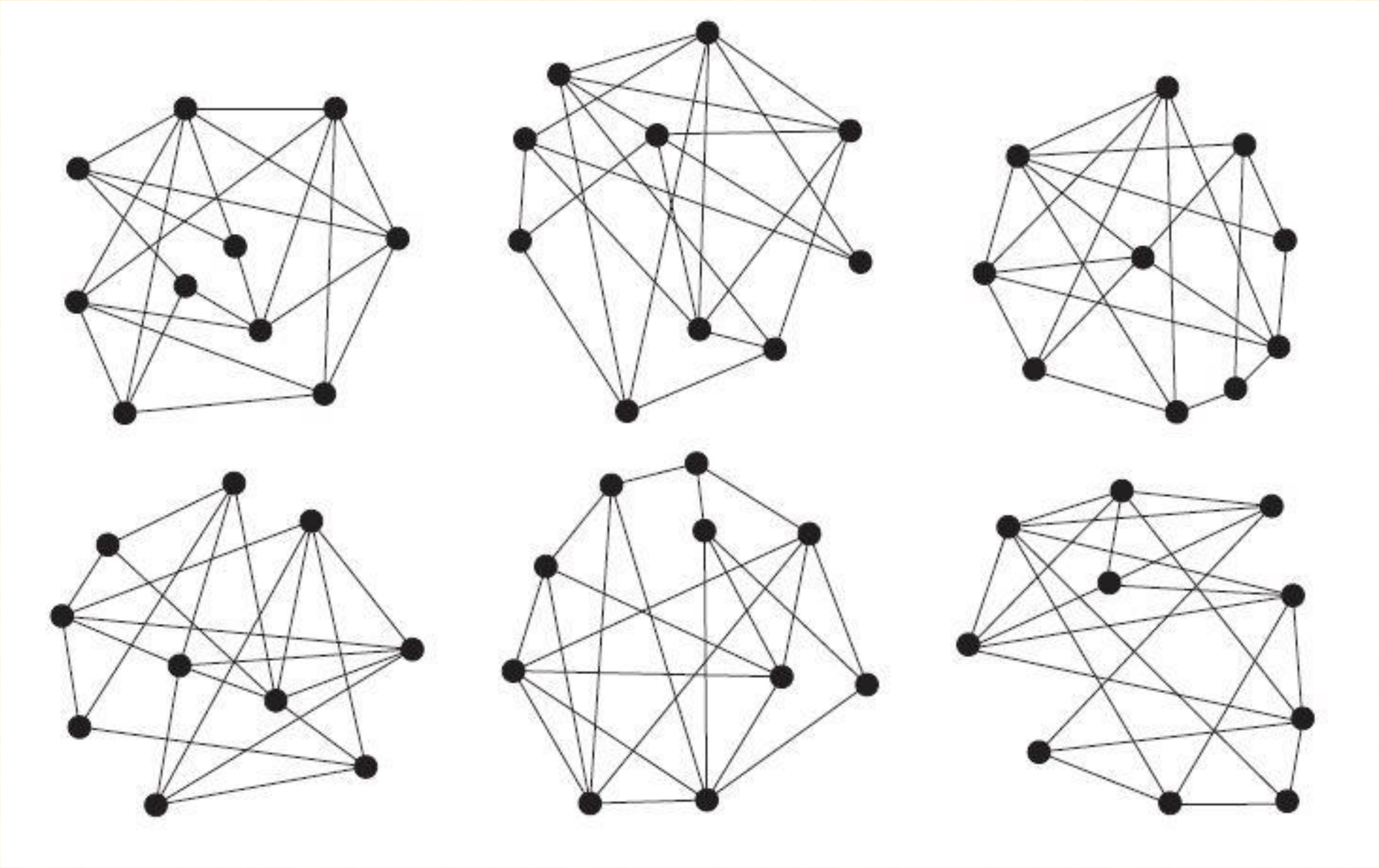
bipartite graph

This mapping is one-to-one and onto, hence invertible.

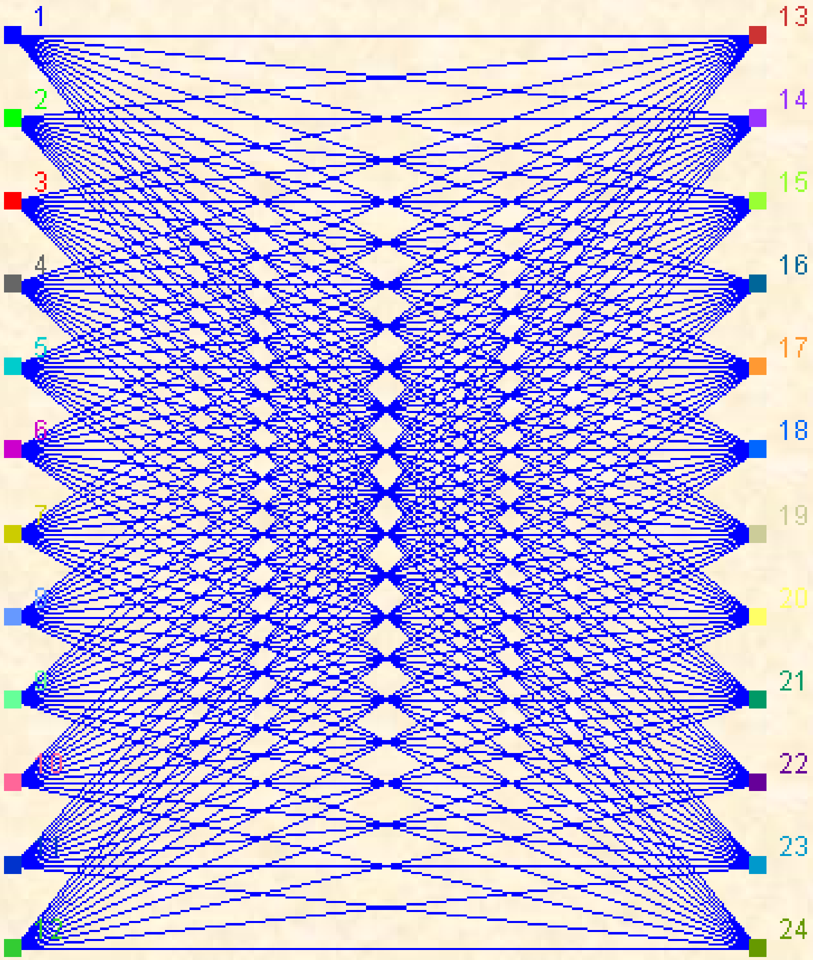
Example



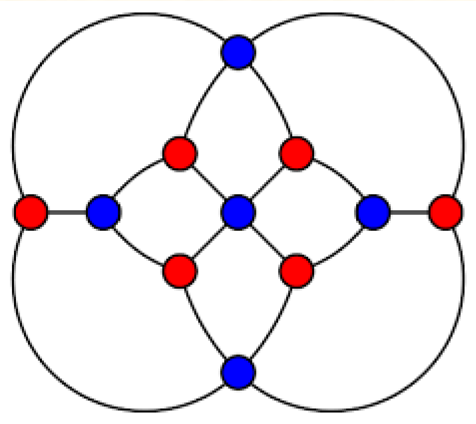
Example

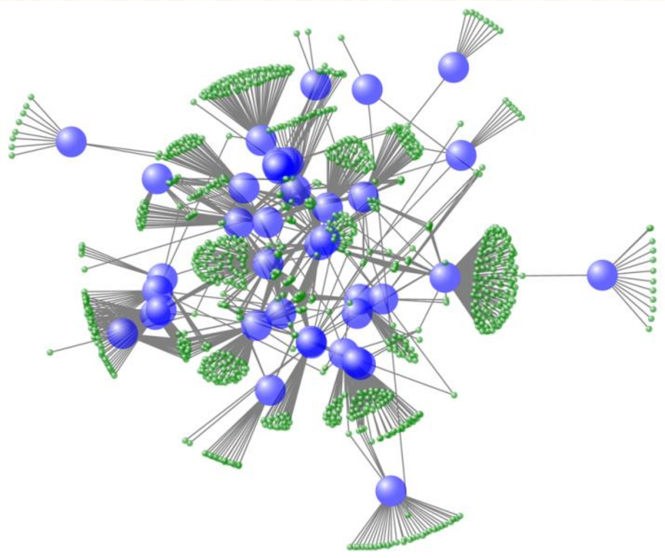


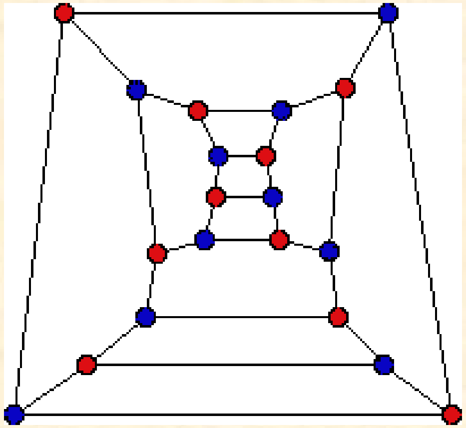
Bipartite Graphs

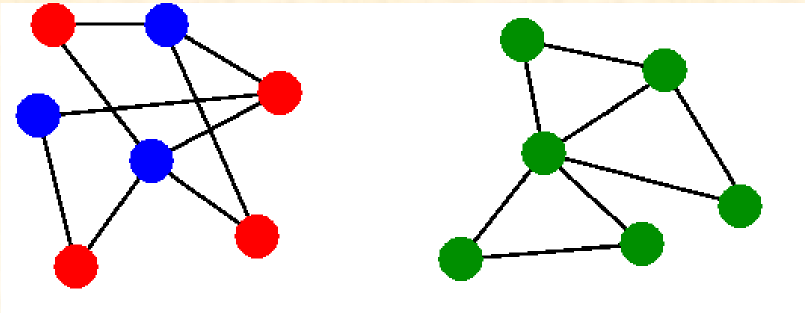


Bipartite Graphs







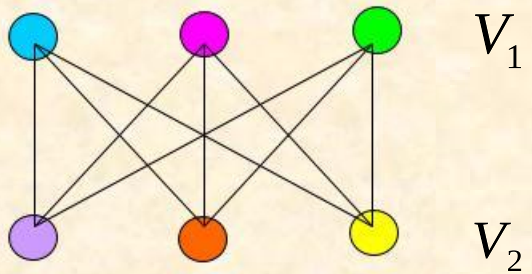


A) A Bipartite Graph B) A non–Bipartite Graph

Circuits in Bipartite Graphs

- **Theorem:** *A graph is bipartite if and only if every circuit (cycle) has an even number of edges in the path.*
- ***Proof.*** Let $v_1 \to v_2 \to \cdots \to v_n \to v_1$ be a circuit in the bipartite graph $G = G(V_1, V_2)$. Assume, without loss of generality, that $v_1 \in V_1$. Then, since G is bipartite, one must have $v_2 \in V_2, v_3 \in V_1$ etc. Finally, one must have $v_n \in V_2$ in order to form a circuit, yielding an even number of paths.

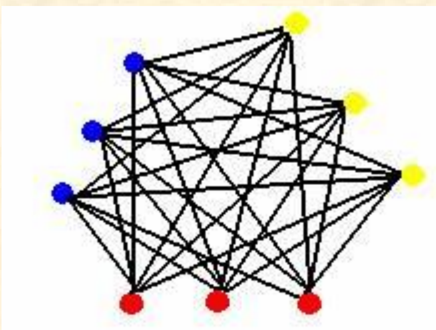
Another direction is obvious, since to form a circuit any path has to return to the same side, which has even number of edges.

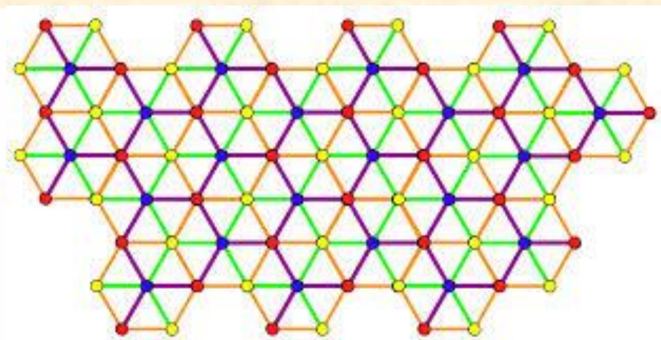


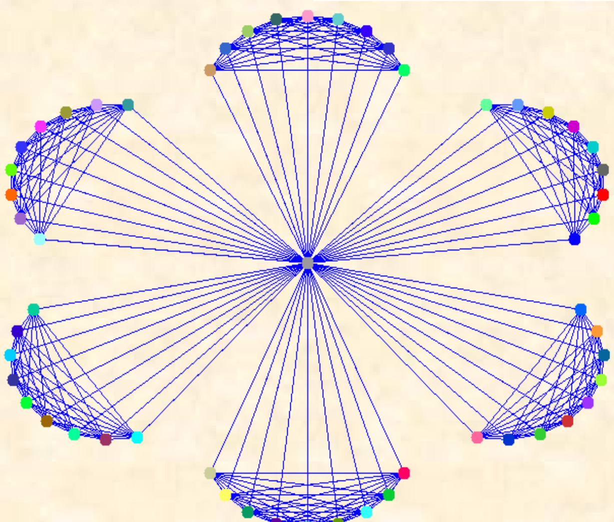
Circuit = Loop = Circle = Closed Path

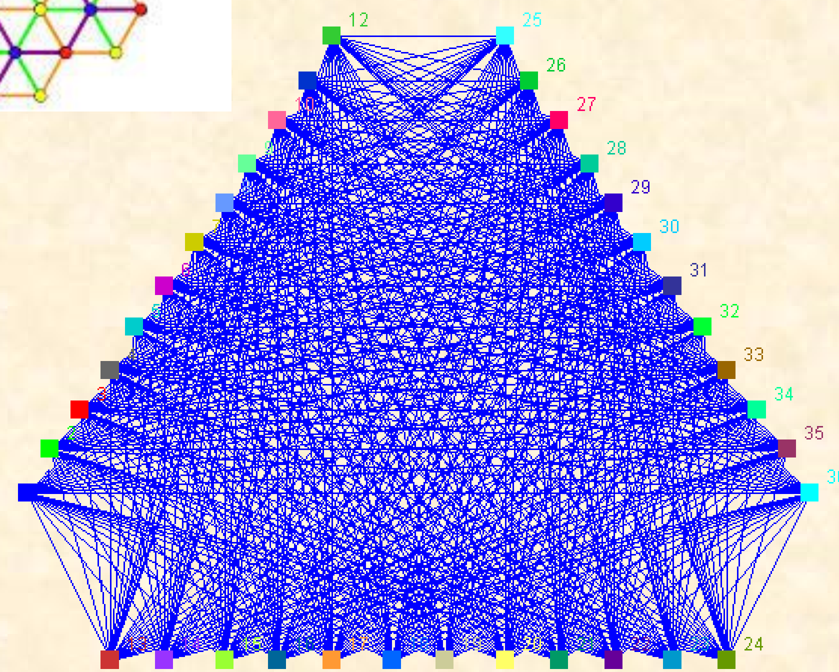
Tripartite Graphs

Tripartite Graph: All nodes are partitioned into three sets in such a way that no two vertices contained in any one of the three parts are adjacent





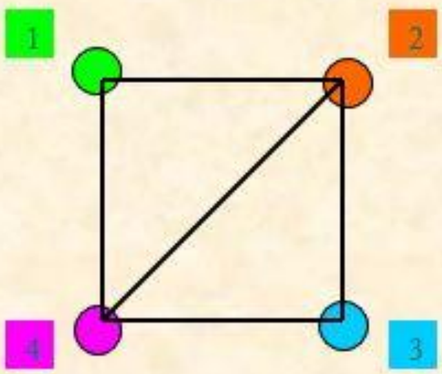




Adjacency Matrix

- For a graph G with nodes $N(G) = \{1, 2, \ldots, n\}$, its adjacency matrix A is defined to be the $n \times n$ constant matrix those ijth entry is 1 if node i connects node j; or 0 otherwise.

- Example:



$$
\begin{array}{c c}
 & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\
A = \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}
\end{array}
$$

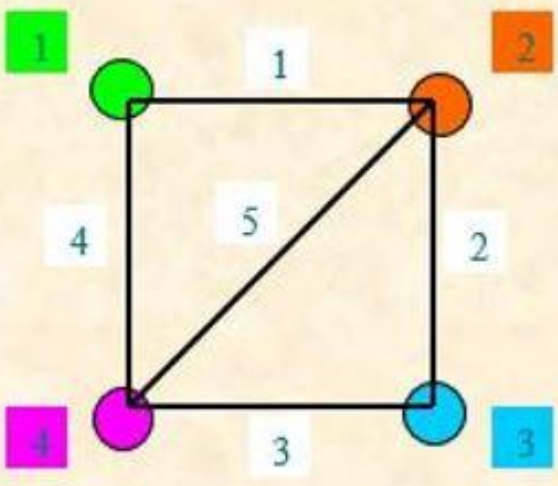
(always square and symmetrical)

Computer/algorithm uses the adjacency matrix to store a network

Incidence Matrix

- For a graph G with edges $M(E)=\{1,2,\ldots,m\}$, its incidence matrix M is defined to be the $n\times m$ constant matrix whose ijth entry is 1 if node i connects edge j ; or 0 otherwise.

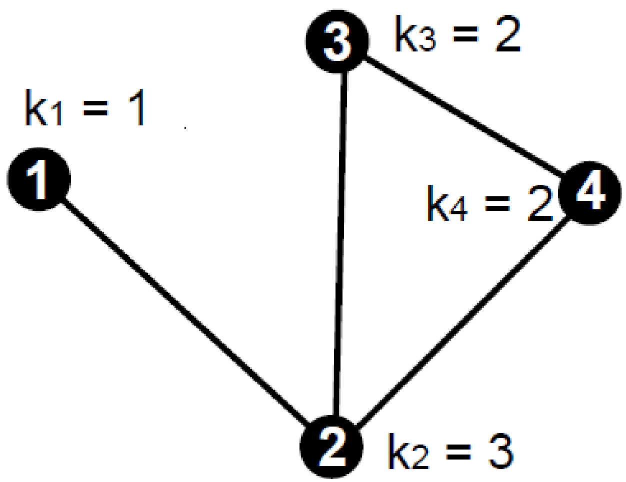
- Example:



$$M=\begin{array}{c} \\ 1\\ 2\\ 3\\ 4 \end{array}\!\!\begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 1 & 0\\ 1 & 1 & 0 & 0 & 1\\ 0 & 1 & 1 & 0 & 0\\ 0 & 0 & 1 & 1 & 1 \end{bmatrix} \end{array}$$

(usually non-square)

Second-Order Adjacency Matrix



$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

$$A^2 = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 3 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix}$$

degrees

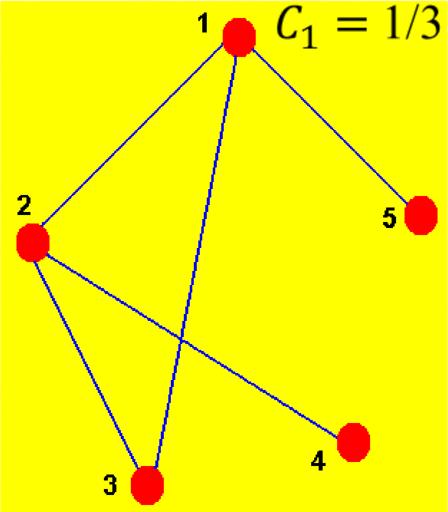
Clustering Coefficient

$$C_i = \frac{1}{k_i(k_i - 1)} \times \sum_{j,k=1\ (j\neq k)}^{N} a_{ij} a_{jk} a_{ki}$$

$A = [a_{ij}]$ -- adjacency matrix

$$C_1 = \frac{1}{3(3-1)} \times \big((a_{12}a_{23}a_{31}) + (a_{13}a_{34}a_{41}) + (a_{14}a_{45}a_{51}) + (a_{12}a_{24}a_{41}) + (a_{12}a_{25}a_{51}) + (a_{13}a_{35}a_{51})\big) \times 2$$

$$= \frac{1}{6}(1+0+0+0+0) \times 2 = 1/3$$



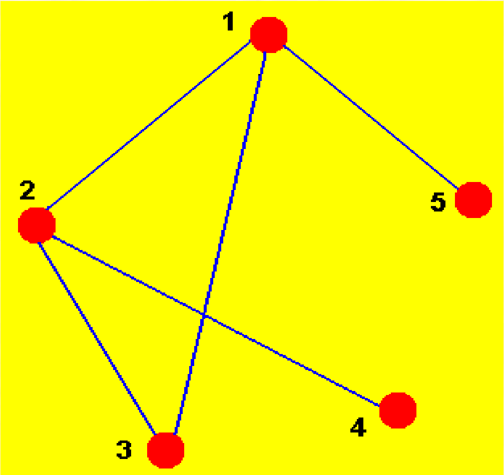
Average Clustering Coefficient of graph:

$$C = \frac{tr(A^3)}{\sum_{i,j=1(i\neq j)}^{N} [(A^2)_{i,j}]}$$

Density of Graph

$$Den = \frac{\overset{\text{total number of edges}}{L}}{N(N-1)/2} = \frac{2\sum_{i>j=1}^{N} a_{ij}}{N(N-1)}$$

- Fully connected graph: $\sum_{i>j=1}^{N} a_{ij} = N(N-1)/2$, so $Den = 1$
- Isolated nodes without edges: $\sum_{i>j=1}^{N} a_{ij} = 0$, so $Den = 0$

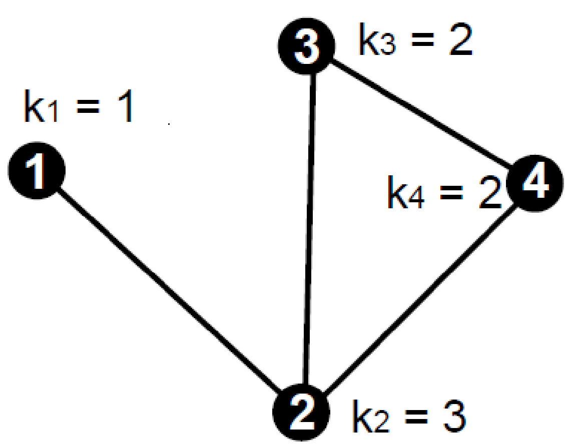


$$\begin{aligned} &Den \\ &= \frac{2}{5 \times 4}(a_{21} + a_{31} + a_{41} + a_{51} + a_{32} + a_{42} \\ &+ a_{52} + a_{43} + a_{53} + a_{54}) \\ &= \frac{1}{10}(1 + 1 + 0 + 1 + 1 + 1 + 0 + 0 + 0 + 0) \\ &= \frac{5}{10} = \frac{1}{2} \end{aligned}$$

Number of Paths of Length n

Path Length formula:

The number of paths of length n between node i and node j $= (A^n)_{i,j}$



$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

number of paths of length 1

$$A^2 = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 3 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix}$$

number of paths of length 2

Laplacian Matrix

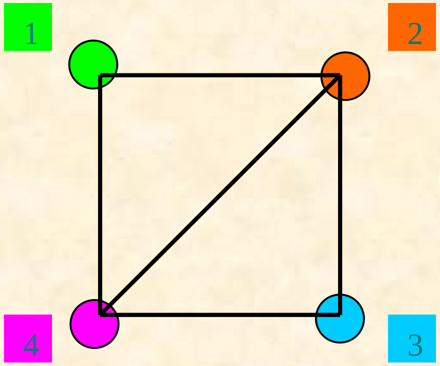
Definition: Laplacian matrix (admittance matrix, Kirchhoff matrix), denoted $L = \lfloor L_{ij} \rfloor$, is defined as

$$L_{ij} = \begin{cases} k_i & if \quad i = j \\ -1 & if \quad i \neq j, \quad v_i \quad adjacent\ with \quad v_j \\ 0 & otherwise \end{cases}$$

where k_i is the degree of node v_i

It has zero row-sum (and, by symmetry, zero column-sum)

Example:

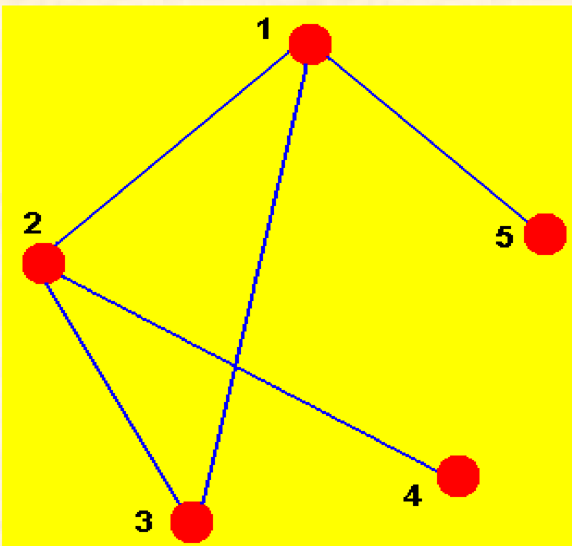


Degree matrix

$$\begin{array}{cccc} & \boldsymbol{D} & A & \boldsymbol{L} \\ L = & \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 3 \end{bmatrix} - & \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} = & \begin{bmatrix} 2 & -1 & 0 & -1 \\ -1 & 3 & -1 & -1 \\ 0 & -1 & 2 & -1 \\ -1 & -1 & -1 & 3 \end{bmatrix} \end{array}$$

L = D - A

Relationships



$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$L = \begin{bmatrix} 3 & -1 & -1 & 0 & -1 \\ -1 & 3 & -1 & -1 & 0 \\ -1 & -1 & 2 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Eigenvalues of Laplacian Matrix

For a connected undirected and unweighted graph G of size N, its Laplacian matrix L is symmetrical and semi-positive definite, with zero row-sums (hence, zero column-sums). Let the eigenvalues of L be $\{\lambda_1, \lambda_2, \ldots, \lambda_N\}$, which are real and nonnegative. Then

$$0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_N$$

If G has $m < N$ components (in the above, $m = 1$), then

$$0 = \lambda_1 = \lambda_2 = \cdots = \lambda_m < \lambda_{m+1} \leq \cdots \leq \lambda_N$$

In particular, if $m = N$ then

$$0 = \lambda_1 = \lambda_2 = \cdots = \lambda_N$$

This corresponds to a set of N isolated nodes for which $L = 0$.

$\lambda_2 > 0$ is called the spectral gap of the graph, or the algebraic connectivity of the network

Verification

To verify the algebraic connectivity $\lambda_2 > 0$ for a connected network, let x be a nonzero eigenvector associated with the eigenvalue $\lambda_1 = 0$ of its Laplacian matrix L.

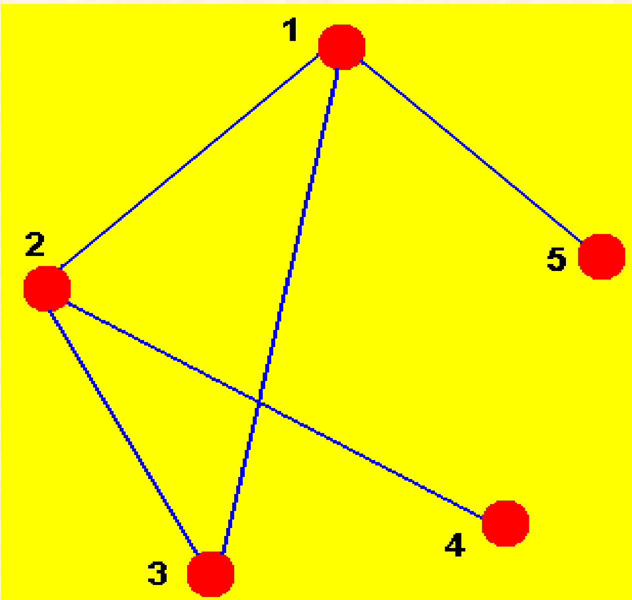
Then, $Lx = 0$, so that $x^T L x = \sum_{(u,v)\in E} (x_u - x_v)^2 = 0$.

Consequently, $x_u = x_v$ for every pair of nodes (u,v) in the network since the network is connected.

This implies that $x = a\,[1,1,\ldots,1]^T$ for some constant $a \neq 0$, namely, the eigenvalue $\lambda_1 = 0$ has multiplicity 1; therefore $\lambda_2 \neq 0$.

By the semi-positiveness of the Laplacian matrix L, one has $\lambda_2 > 0$.

Examples



$$L = \begin{bmatrix} 3 & -1 & -1 & 0 & -1 \\ -1 & 3 & -1 & -1 & 0 \\ -1 & -1 & 2 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

symmetrical with zero row-sum

eigenvalues of L: $\lambda_1 = 0 < \lambda_{2,3} = \frac{5}{2} \pm \frac{\sqrt{5}}{2}, \lambda_{4,5} = \frac{5}{2} \pm \frac{\sqrt{13}}{2}$

$$\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

eigenvalues L: $\lambda_1 = 0 \quad \lambda_2 = 2$

Regular Graphs

- A graph in which all nodes have the same degree is called a regular graph; if every node has degree r then the graph is called a regular graph of degree r (or, r-regular)

- **Theorem:** *A regular graph of degree r with N nodes has $r\,N\,/\,2$ edges.*

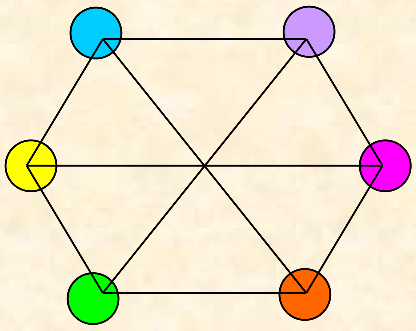
- *Proof.* Since every node connects with r edges, there are $r\,N$ connecting edges. However, each edge has been doubly counted, so it should be divided by two.

- Example:

  A regular graph of degree 3

  (3-regular), which has 6 nodes

  and $3 \times 6 / 2 = 9$ edges



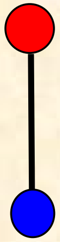
Eigenvalues of Regular Graphs

Let G be a connected r-regular graph with N nodes, with eigenvalues $\{\mu_1, \mu_2, ..., \mu_{N-1}, \mu_N\}$ of its adjacency matrix A

(i) $-r \le \mu_1 \le \cdots \le \mu_{N-1} < \mu_N = r$

(ii) G is bipartite if and only if the above eigenvalues are symmetrical about 0; and if and only if $\mu_1 = -r$

$$-r = \mu_1 < \mu_2 \le \cdots \le \mu_{N-1} < \mu_N = r$$



$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ $\quad r = 1 \quad$ $-1 = \mu_1 < \mu_2 = 1$

Line Graphs

Given a graph G, its line graph $L(G)$ is a graph such that

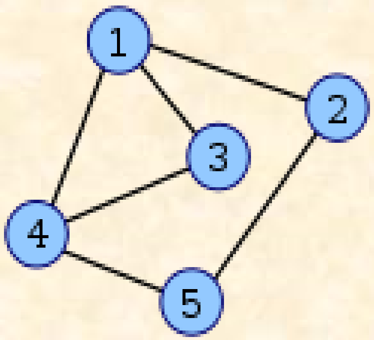
i) each node of $L(G)$ represents an edge of G

ii) two nodes of $L(G)$ are adjacent if and only if their corresponding edges share a common node in G

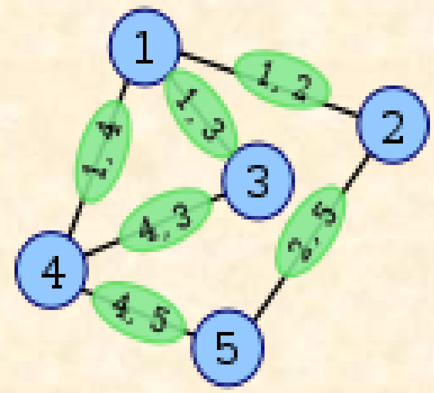
That is, it is the intersection graph of the edges of G, representing each edge by the set of its two end-nodes

There are linear-time algorithms for recognizing line graphs and reconstructing their original graphs

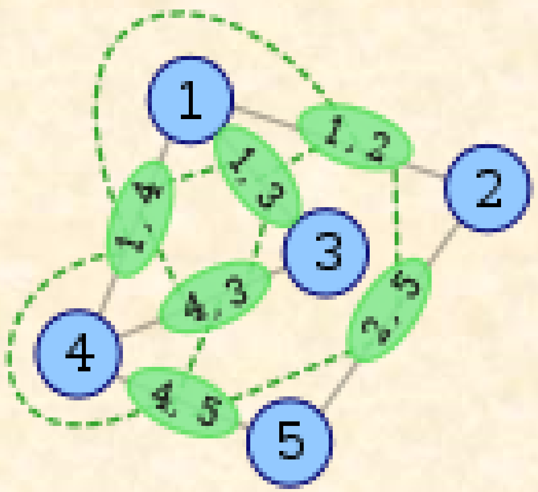
Line Graphs



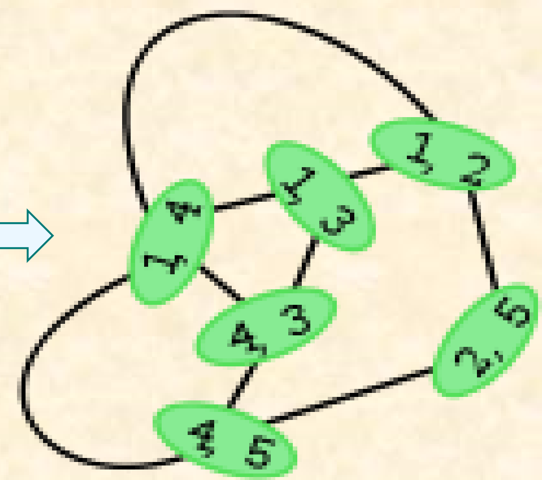
Graph G



Nodes in $L(G)$



Added edges in $L(G)$

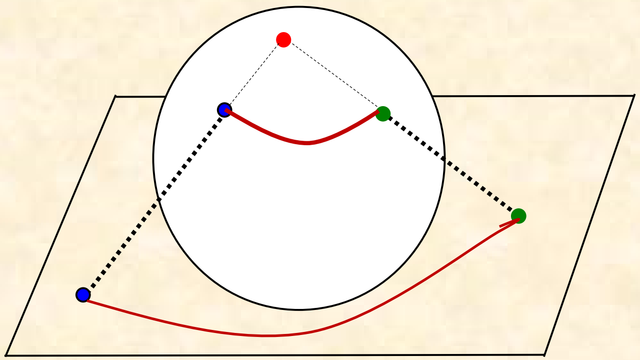


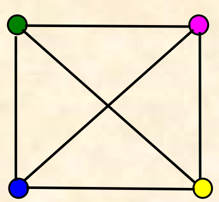
Line graph $L(G)$

Planar Graphs

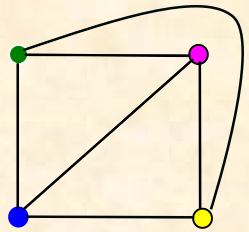
- **Plane graph** is one that can be drawn on the 2D plane without crossing edges
- **Planar graph** is one that is isomorphic to a plane graph
- Every planar graph can be embedded on a 2D plane (within the Euclidean 3D space)

- **Theorem:** *A graph is planar if and only if it can be embedded on the surface of a sphere.*



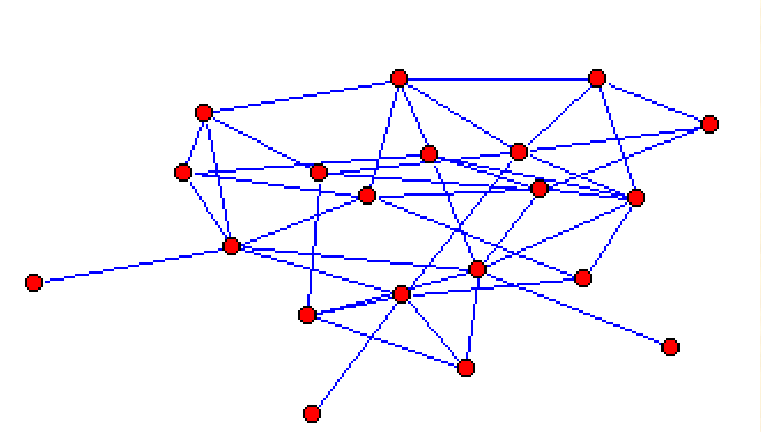


Planar



Plane

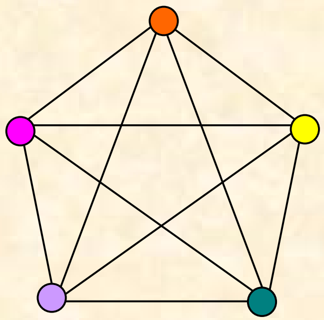
They can be embedding on to a 3-D sphere



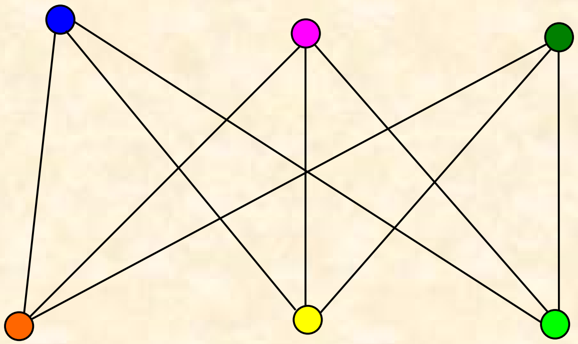
Impossible to do so

Non-Planar Graphs

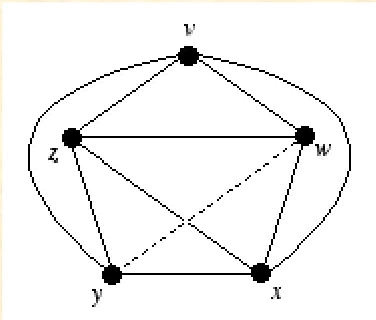
- Two special yet important non-planar graphs:



Graph K_5



Graph $K_{3,3}$



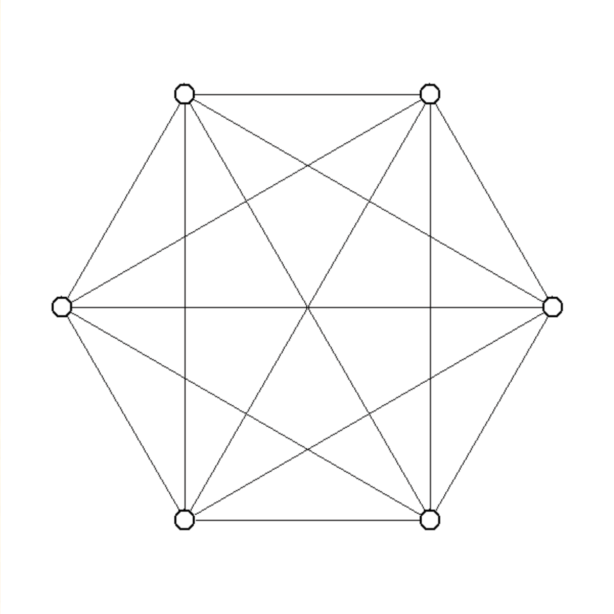
K. Kuratowski
(1896 -1980)

Ramsey Problem

6 people meet in a party

If two persons know each other, then connect them with a red edge

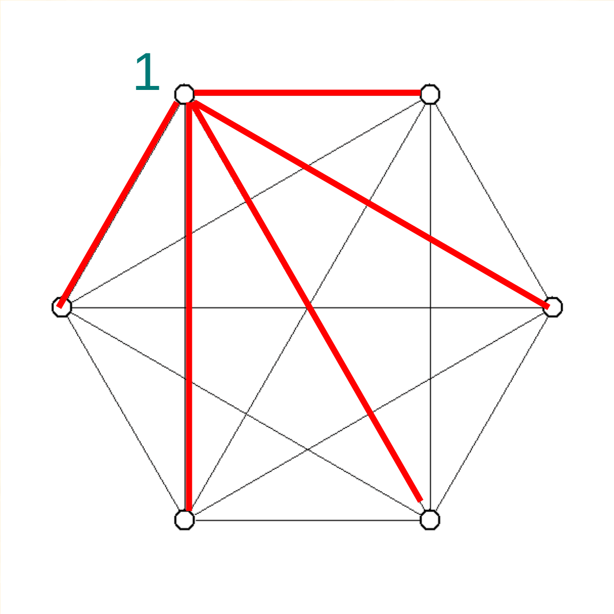
If two persons don't know each other, then connect them with a blue edge

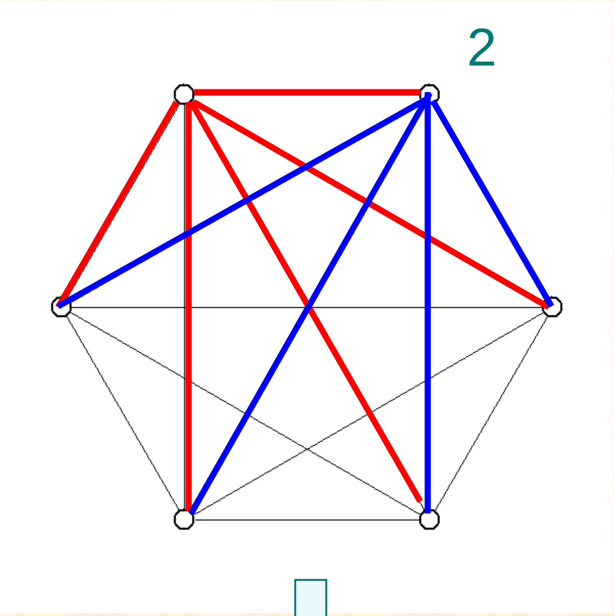


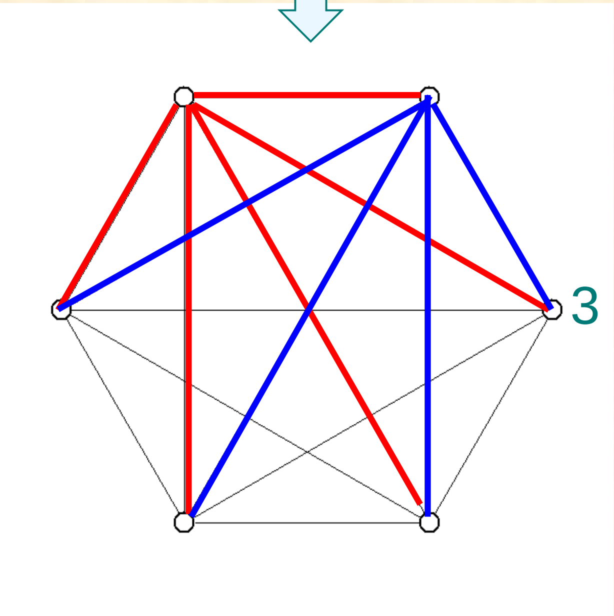
⇓

How many red triangles ?
How many blue triangles?

Try one case







?

Ramsey Theorem:

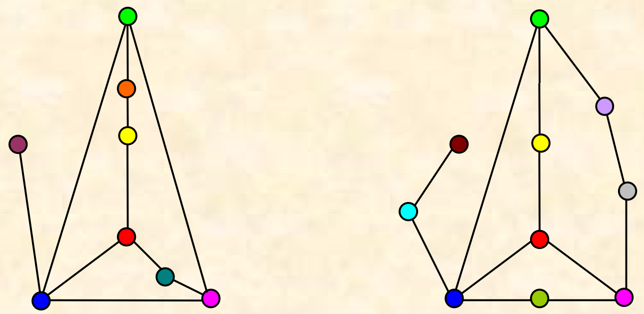
There exists at least one red triangle or one blue triangle

→

3 know each other OR
3 don't know each other

Homeomorphism

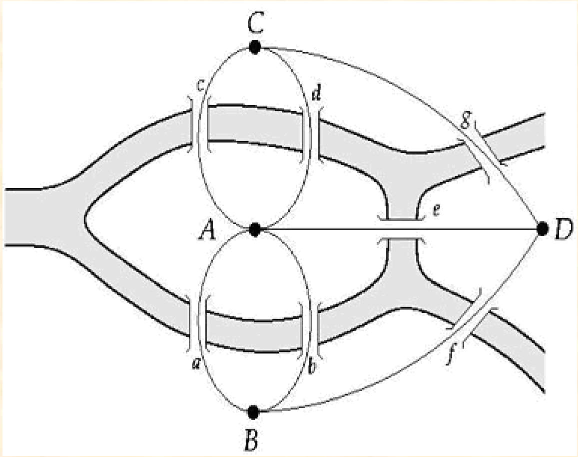
- Two graphs are said to be **homeomorphic**, if they can both be obtained from the same graph by inserting new nodes of degree-2 into edges.
(namely, identical to within nodes of degree-2)



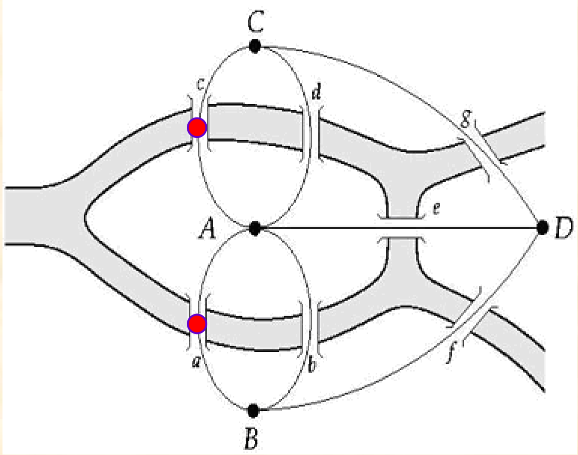
- **Theorem** (Kuratowski Theorem) *A graph is planar if and only if it contains no subgraphs homeomorphic to* K_5 *or* $K_{3,3}$

An Application Example

Recall: The Königsburg seven-bridge problem



Homeomorphic



With multiple edges

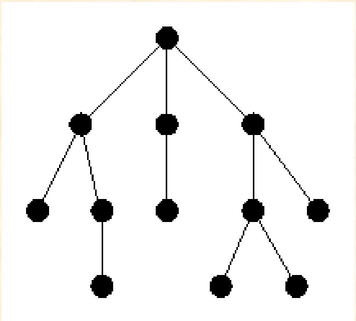
A simple graph

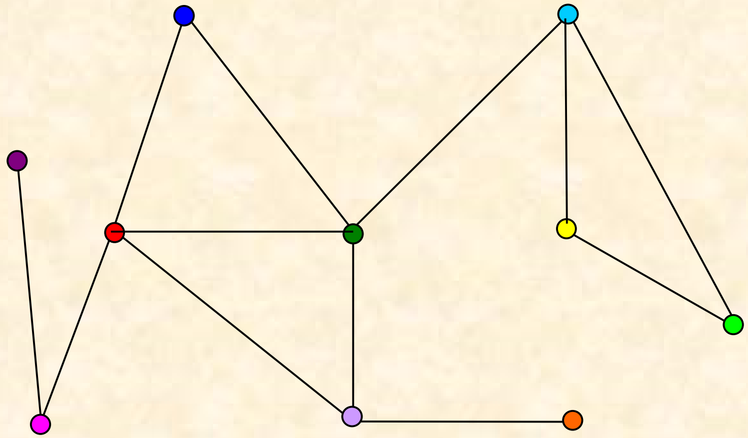
BREAK

10 minutes

More Concepts

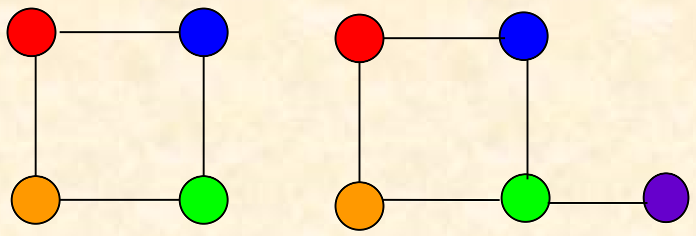
- **Walk**: A finite sequence of edges, one after another, in the form of $v_1v_2, v_2v_3, \ldots, v_{n-1}v_n$ where $N(G) = \{v_1, v_2, \ldots, v_n\}$ are nodes
- A walk is denoted by $v_1 \to v_2 \to \cdots \to v_n$ and the number of edges in a walk is called its **length**
- **Trail**: A walk in which all <u>edges</u> are distinct
- **Path**: A trail in which all <u>nodes</u> are distinct, except perhaps $v_1 = v_n$ which is called a **closed path**, often called a **circuit** (or, a **cycle**)
- **Tree**: A graph with no circuits





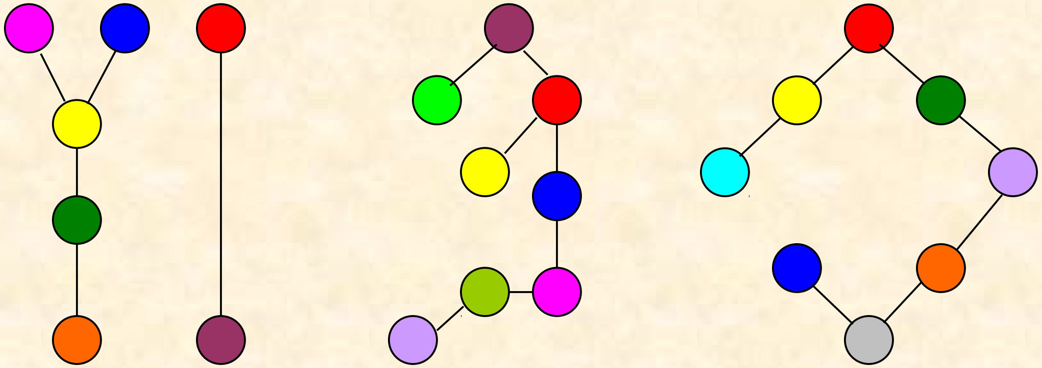
- **Lemma:** *If every node in a graph has degree $k \geq 2$ then this graph contains a circuit (cycle).*

- *Proof.*
  Consider a simple graph. Starting from any node v_0, construct a walk $v_0 \to v_1 \to v_2 \to \cdots$ in such a way that v_1 is any adjacent node of v_0 and, for $i = 1,2,...,$ node v_{i+1} is any (except v_{i-1}) adjacent node of v_i. Since every node has degree $k \geq 2$, such a node v_{i+1} exists. Because the graph has finitely many nodes, the walk eventually connects to a node that has been chosen before. This walk yields a circuit in the graph.

The converse may not be true: 

Trees

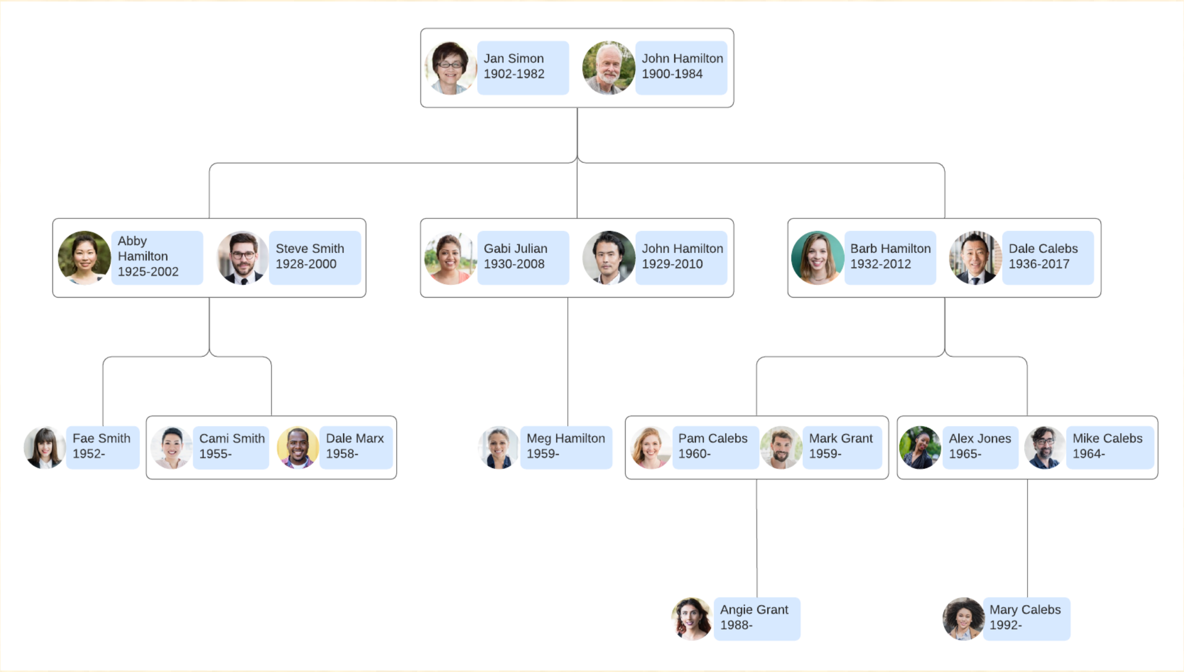
- **Tree**: A connected graph without circuits
- **Forest**: A family of unconnected trees



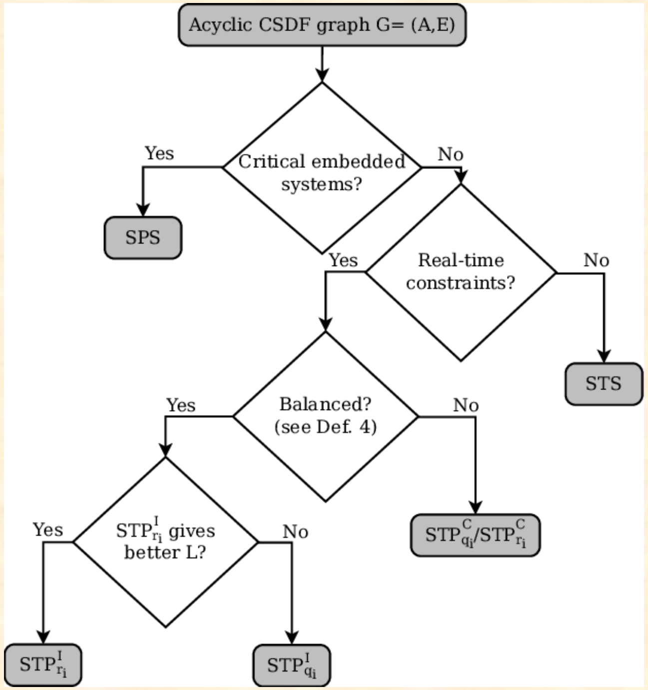
- A tree with N nodes has $N - 1$ edges
- Sum of node degrees in a tree

  = 2 x (number of edges) = $2\,(N - 1)$

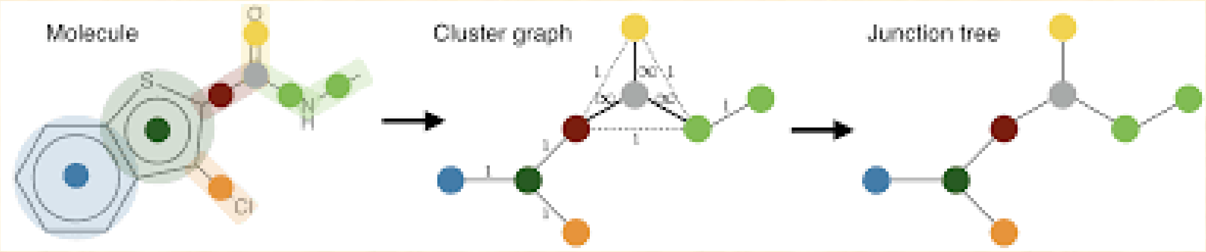
Family Tree



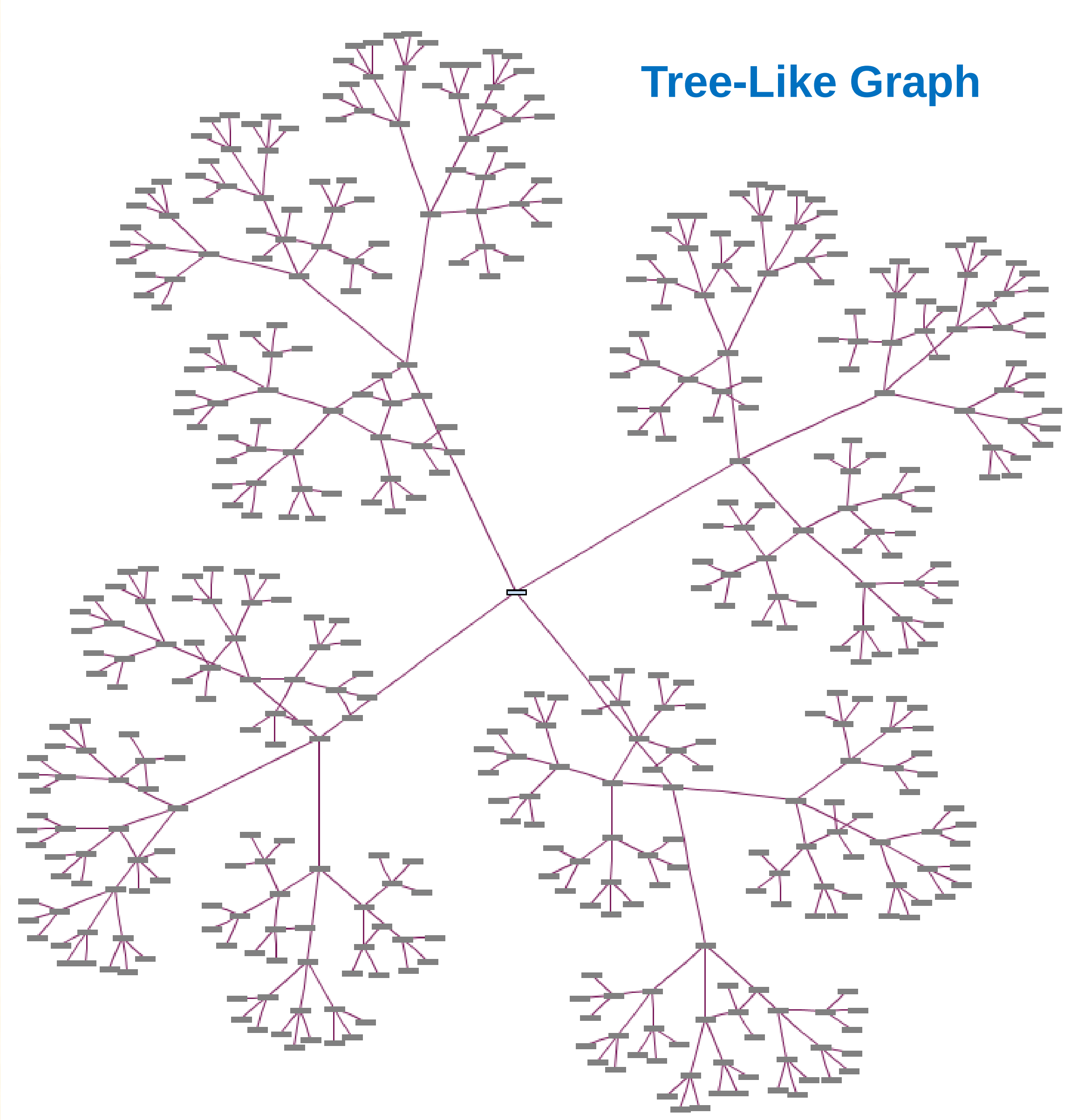
Decision Tree

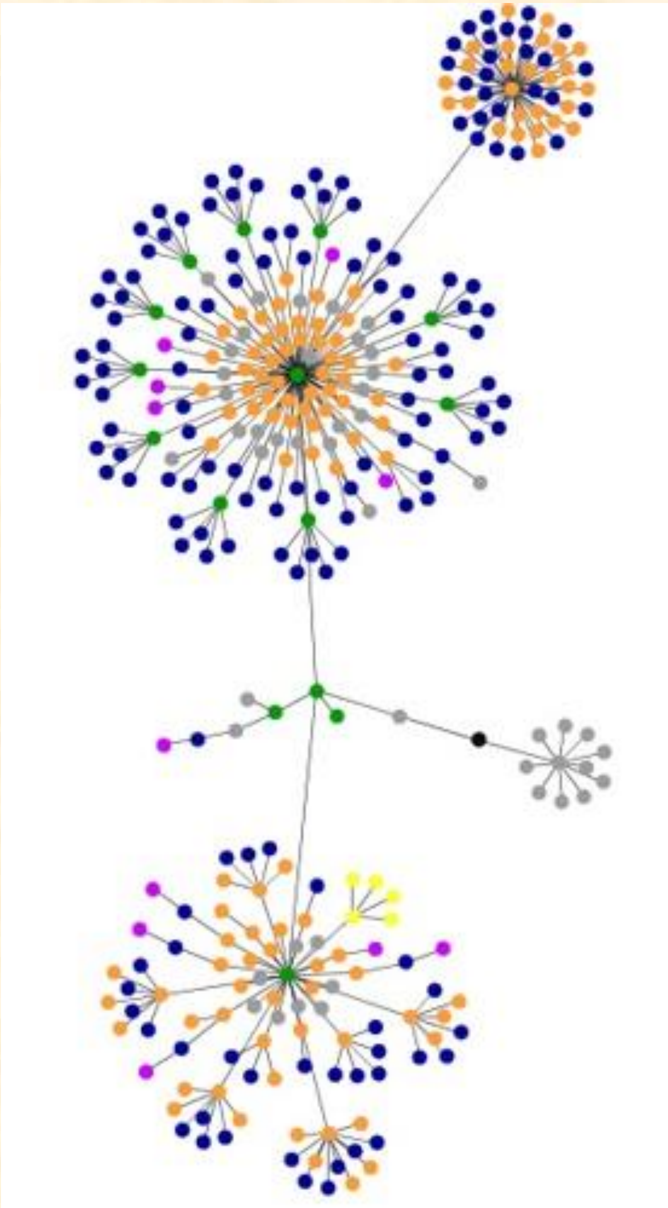


Chemistry



Tree-Like Graph





Some Basic Results

Theorem: *Let* T *be a graph with* N *nodes. Then, the following statements are equivalent:*

- T *is a tree*
- T *has* $N - 1$ *edges but contains no circuits*
- T *has* $N - 1$ *edges and is connected*
- T *is connected, but the removal of any edge will disconnect the graph*
- *every pair of nodes of* T *are connected by exactly one path*
- T *contains no circuits, but the addition of any new edge creates exactly one circuit*

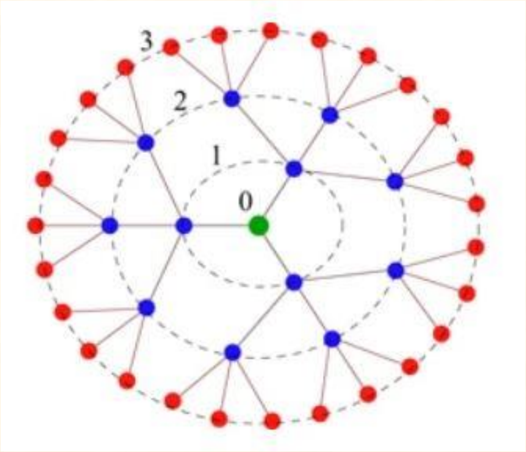
Fractal Tree Graphs

Number of nodes: N

Number of breaches: d

Number of generations: g

Number of peripheral nodes: m



$$N = \frac{d^{g+1}-1}{d-1}$$

$$m = d^g$$

$N = 40$
$d = 3$
$g = 3$
$m = 27$

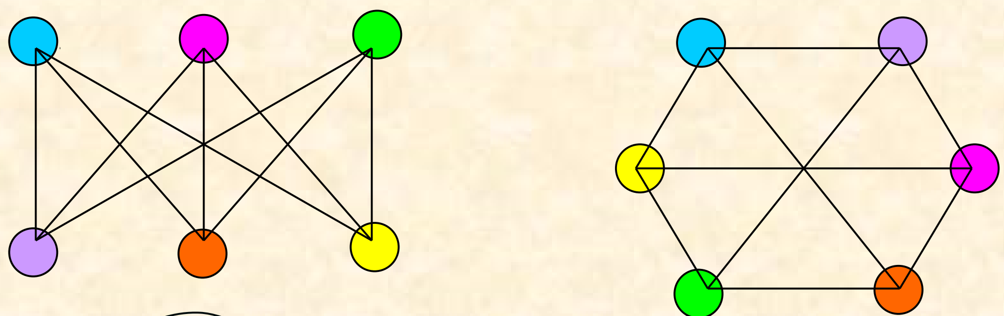
$$40 = \frac{3^{3+1}-1}{3-1}$$

$$27 = 3^3$$

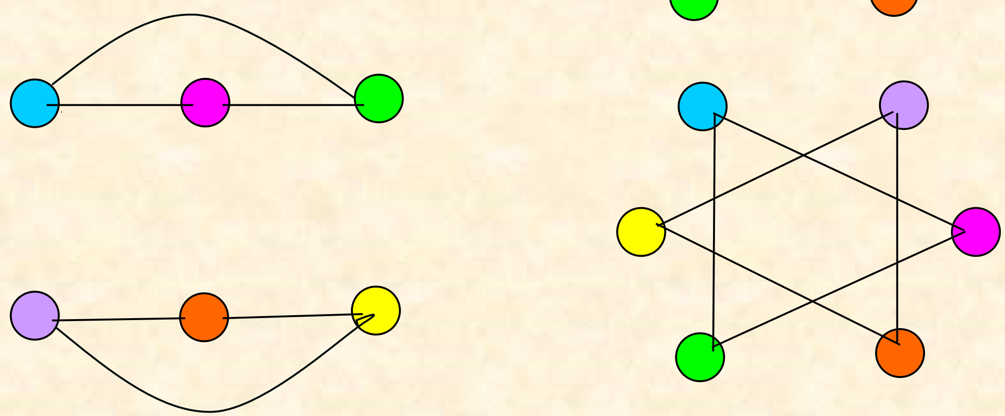
Complementary Graph

For a given graph G, its *complementary graph* G^c is the graph containing all the nodes of G and all the edges that are not in G

G



G^c

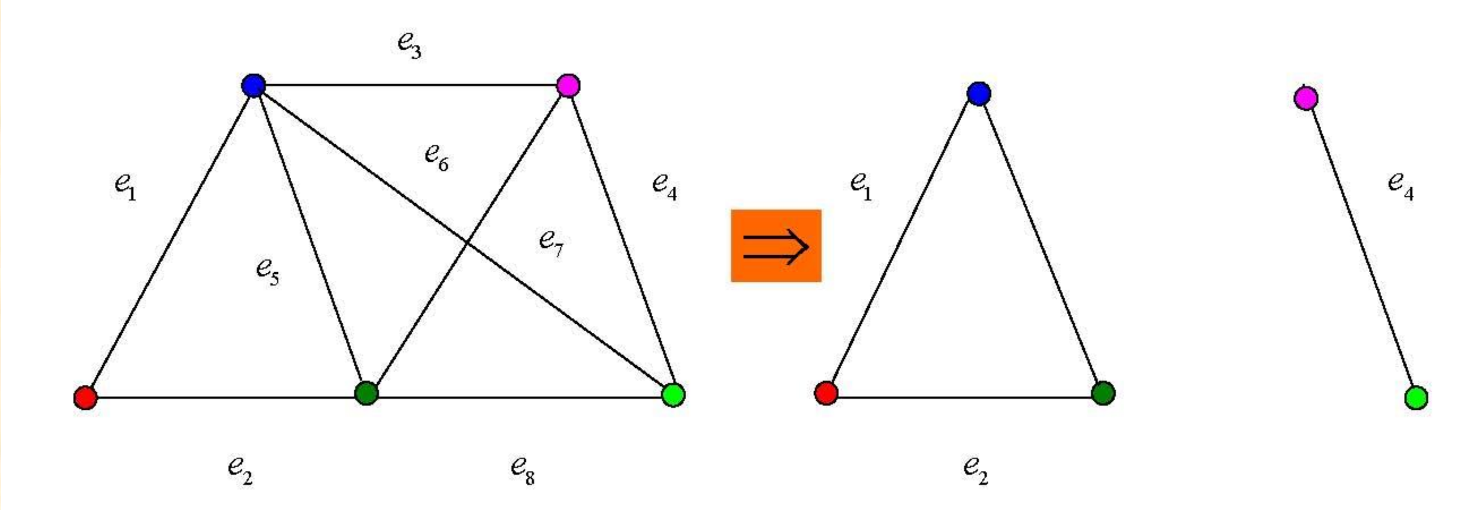


Graph Connectivity

- **Q:** How many edges or nodes must be removed in order to disconnect an originally connected graph?

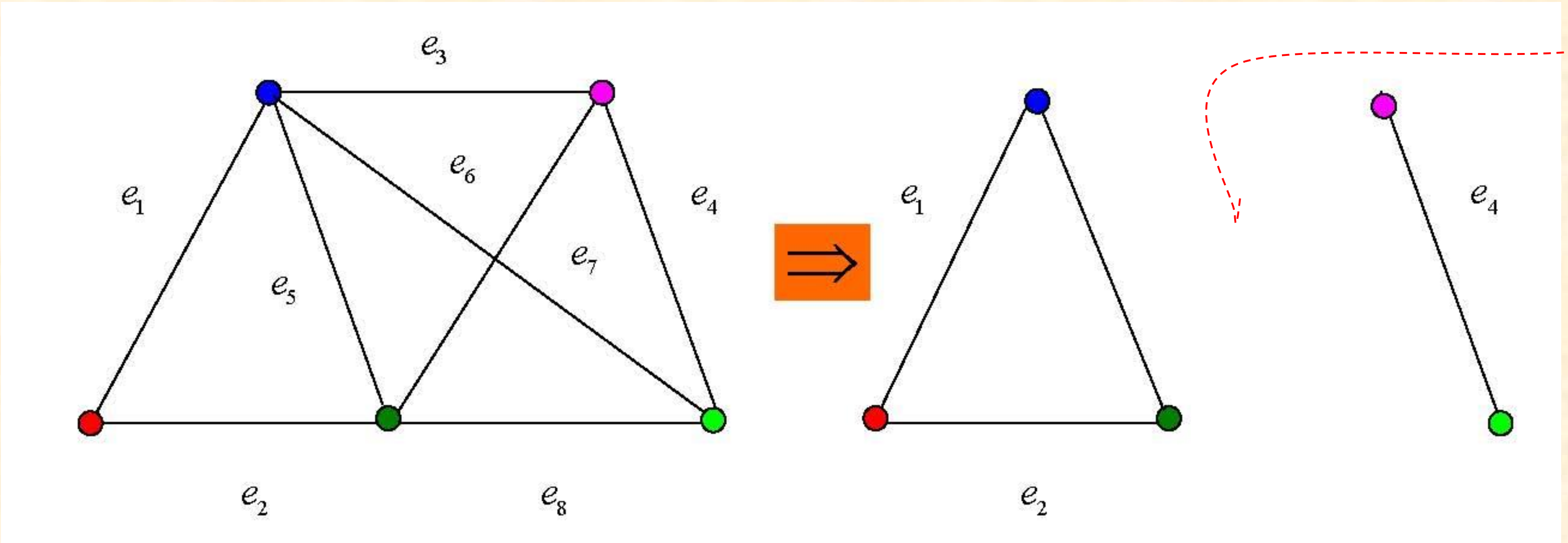
- Note: If a node is removed, then all edges joining it will also be removed; but the converse is not so.

Example:



Disconnecting Sets and Cut-Sets

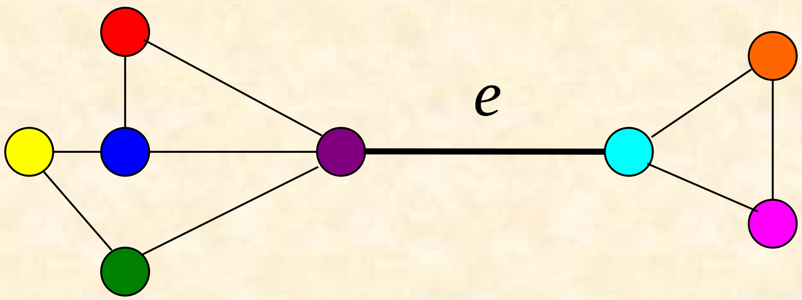
- **Disconnecting set**: A set of edges, denoted as $E_0(G)$, such that after it is being removed, the graph G will become unconnected
- **Cut-Set**: The smallest disconnecting set, i.e., no proper subset of which is a disconnecting set
- **Example**: $E_0^1(G)=\{e_1,e_2\}$ $E_0^2(G)=\{e_1,e_2,e_5\}$ $E_0^3(G)=\{e_3,e_6,e_7,e_8\}$ are disconnecting sets, in which both $E_0^1(G)$ and $E_0^3(G)$ are cut-sets



Bridges

Bridge: A cut-set with only one edge

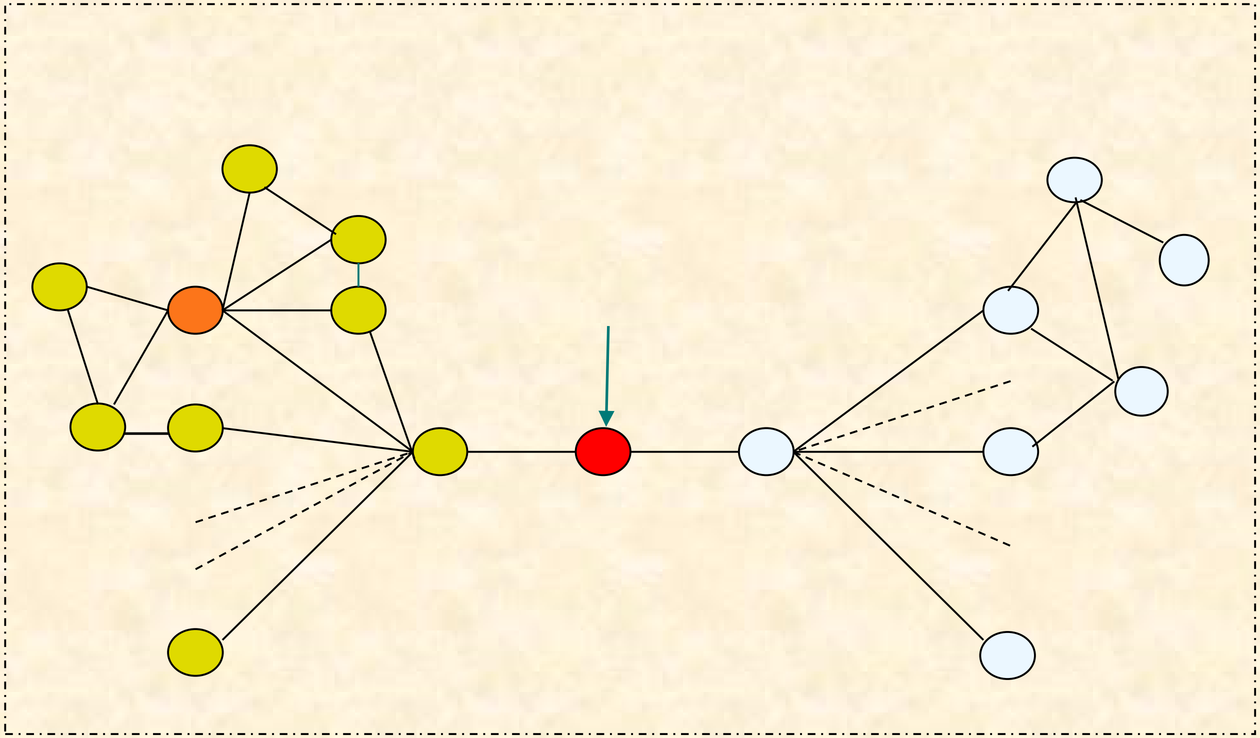
Example: cut-set $\{e\}$ below is a bridge



Bridge is also called Edge Connectivity

Importance of bridges

In a network, a node of low degree may be more important than a node of high degree. For example, on a bridge:



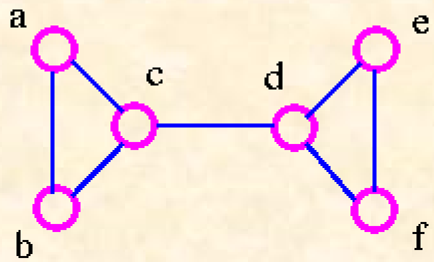
Node Connectivity

Node Connectivity, $NC(G)$, of a connected graph G is the minimum number of nodes whose removal disconnects G

When $NC(G) \geq k$, the graph is said to be k-connected

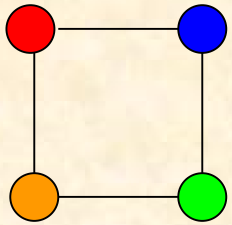
$NC(G) = 1$

1-connected

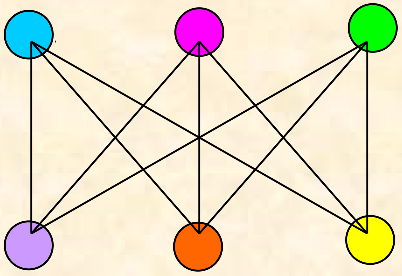


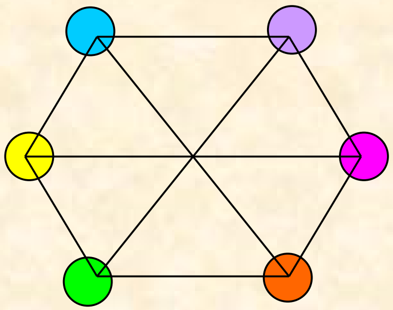
$NC(G) = 2$

2-connected



$NC(G) = ?$



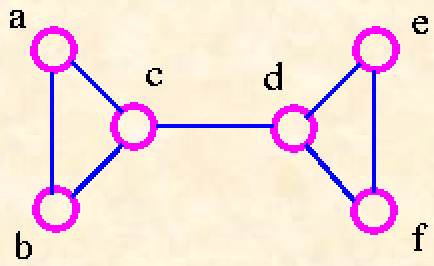


Edge Connectivity

Edge Connectivity, $EC(G)$, of a connected graph G is the minimum number of edges whose removal disconnects G

Let $MD(G)$ be the minimum node degree of a graph

Theorem: $NC(G) \leq EC(G) \leq MD(G)$



$NC(G) = 1$
$EC(G) = 1$
$MD(G) = 2$

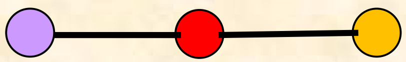
Closeness

A node is considered to be more important if it is “closer” to all other nodes. For node v_i in a network of N nodes:

Closeness: $$C(v_i) = \left[\sum_{j=1}^{N} d(v_i, v_j) \right]^{-1}$$

Normalization:

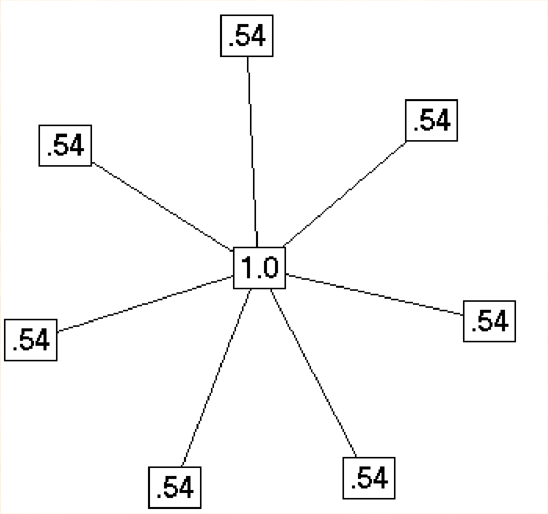
$$C_n(v_i) = C(v_i) \times (N-1)$$



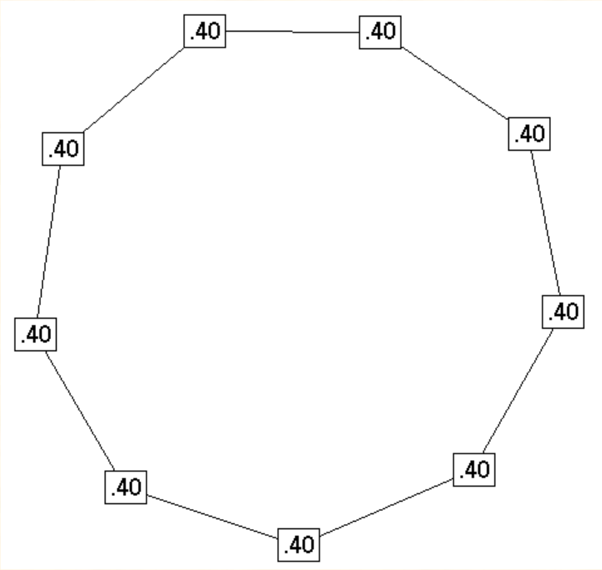
C(yellow) = 1 / (1 + 2) = 1 / 3

C(red) = 1 / (1 + 1) = 1 / 2 (> 1 / 3)

Closeness



Distance								Closeness	normalized
0	1	1	1	1	1	1	1	.143	1.00
1	0	2	2	2	2	2	2	.077	.538
1	2	0	2	2	2	2	2	.077	.538
1	2	2	0	2	2	2	2	.077	.538
1	2	2	2	0	2	2	2	.077	.538
1	2	2	2	2	0	2	2	.077	.538
1	2	2	2	2	2	0	2	.077	.538
1	2	2	2	2	2	2	0	.077	.538

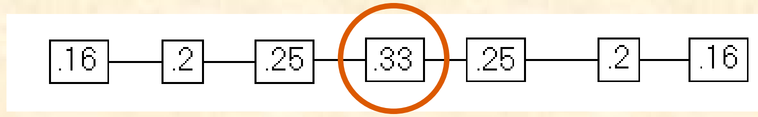


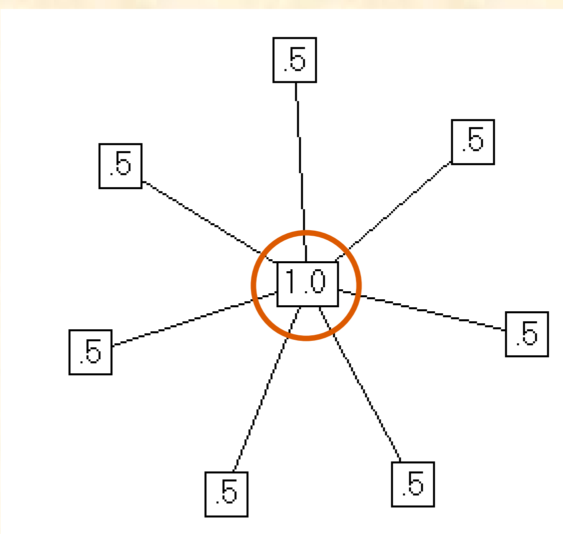
Distance									Closeness	normalized
0	1	2	3	4	4	3	2	1	.050	.400
1	0	1	2	3	4	4	3	2	.050	.400
2	1	0	1	2	3	4	4	3	.050	.400
3	2	1	0	1	2	3	4	4	.050	.400
4	3	2	1	0	1	2	3	4	.050	.400
4	4	3	2	1	0	1	2	3	.050	.400
3	4	4	3	2	1	0	1	2	.050	.400
2	3	4	4	3	2	1	0	1	.050	.400
1	2	3	4	4	3	2	1	0	.050	.400

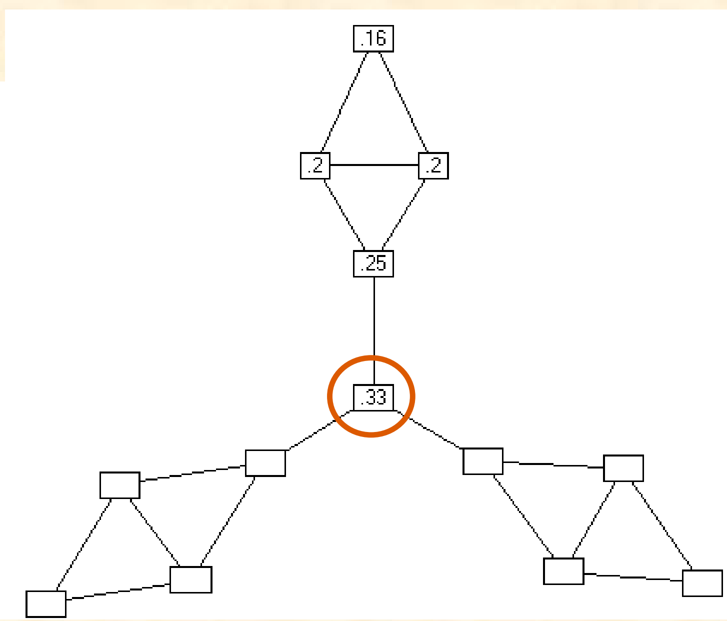
Graph-Theoretic Centre

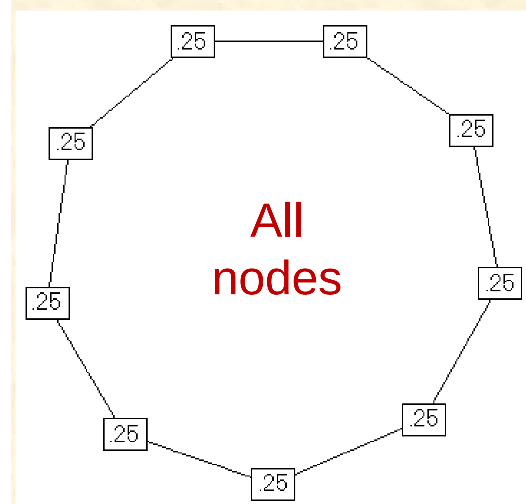
Also called Barry Centre or Jordan Centre

The set of all nodes, A, satisfying that the longest distance $d(A,B)$ to other nodes B, is minimal (or, $1 / d(A,B)$ is maximal)



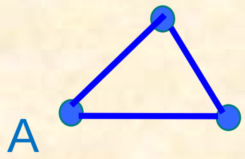




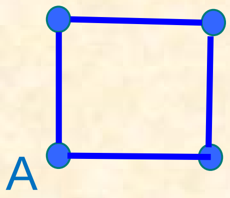


Girth of a Node

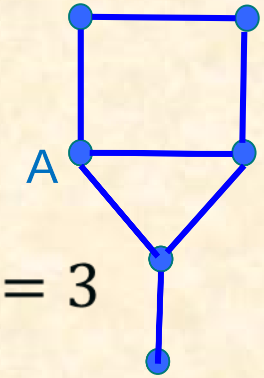
Girth of Node A = length of the shortest cycle in the graph, which passes Node A



$g = 3$

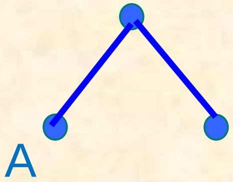


$g = 4$

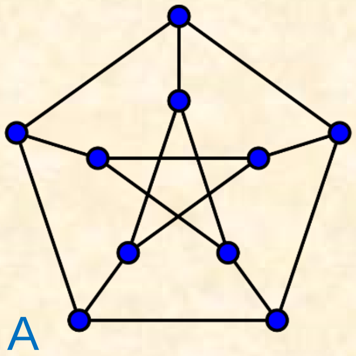


$g = 3$

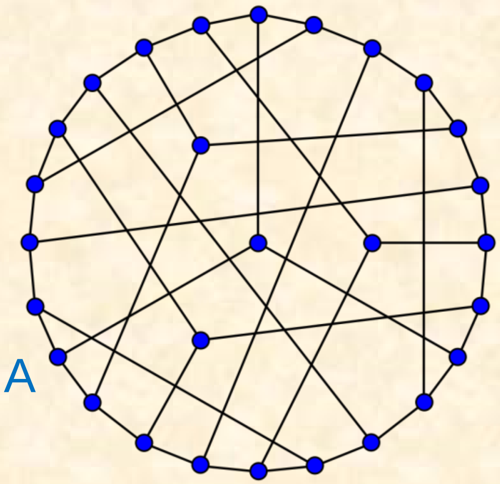
$g := 0$



$g := \infty$



$g = 5$



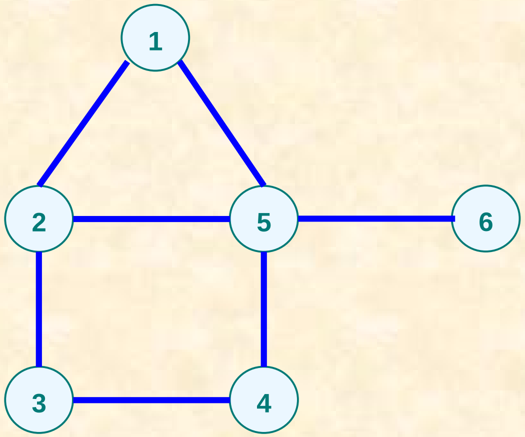
$g = ?$

Cyclic Coefficient

Definition:

S_{ijk} = girth that passes nodes $i,\ j,\ k$

Example: Node 5 to all its neighbors



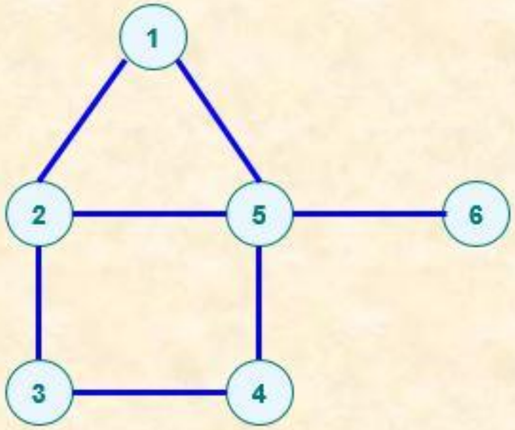
$S_{512} = 3$ $S_{513} = 5$

$S_{514} = 5$ $S_{523} = 4$

$S_{524} = 4$ $S_{534} = 4$

$S_{516} = S_{526} = S_{536} = S_{546} = \infty$

$S_{56} = \infty, S_6 = 0$



$S_{512} = 3 \qquad S_{513} = 5$

$S_{514} = 5 \qquad S_{523} = 4$

$S_{524} = 4 \qquad S_{534} = 4$

$S_{516} = S_{526} = S_{536} = S_{546} = \infty$

Definition: Cyclic Coefficient of Node i

$$\theta_i = \frac{1}{k_i(k_i-1)/2} \sum_{k>j=1}^{N} \frac{a_{ij}a_{jk}}{S_{ijk}}$$

where $A = [a_{ij}]$ is adjacency matrix

The graph:

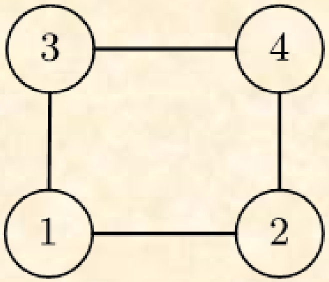
$$\theta = \frac{1}{N}\sum_{i=1}^{N} \theta_i$$

Example: Node 5

$$\theta_5 = \frac{2}{4(4-1)}\left(\frac{a_{51}a_{12}}{S_{512}} + \frac{a_{52}a_{23}}{S_{523}} + 0 + \cdots + 0\right) = \frac{1}{6}\left(\frac{1}{3}+\frac{1}{4}\right) = \frac{7}{72}$$

Graphical Meaning:

Cyclic coefficient is the percentage of two ordered adjacent edges over all possibilities in an orderly motion (order: $k > j$)

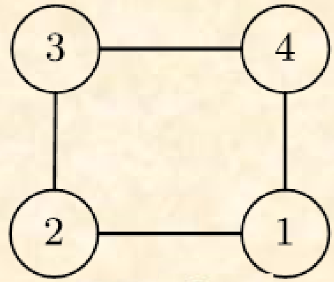


$$\theta_3 = \frac{a_{31}a_{12}}{S_{312}} = \frac{1}{4}$$

$$\theta_4 = 0$$

$$\theta_2 = \frac{a_{21}a_{13}}{S_{213}} = \frac{1}{4}$$

$$\theta_1 = \frac{a_{12}a_{24}}{S_{124}} + \frac{a_{13}a_{34}}{S_{134}} = \frac{1}{2}$$



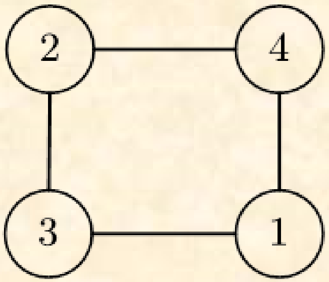
S_{312}

$$\theta_3 = 0$$

$$\theta_4 = \frac{a_{41}a_{12}}{S_{412}} = \frac{1}{4}$$

$$\theta_1 = \frac{a_{12}a_{23}}{S_{123}} = \frac{1}{4}$$

$$\theta_2 = \frac{a_{21}a_{14}}{S_{214}} + \frac{a_{23}a_{34}}{S_{234}} = \frac{1}{2}$$



$$\theta_2 = 0$$

$$\theta_4 = \frac{a_{41}a_{13}}{S_{413}} + \frac{a_{42}a_{23}}{S_{423}} = \frac{1}{2}$$

$$\theta_1 = 0$$

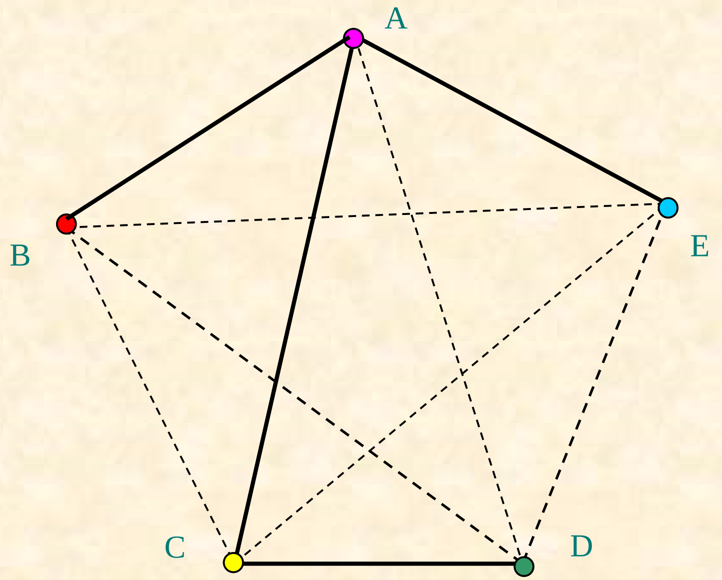
$$\theta_3 = \frac{a_{31}a_{14}}{S_{314}} + \frac{a_{32}a_{24}}{S_{324}} = \frac{1}{2}$$

Node 3: Path 312 has 1 ordered adjacent edge ⇨ 1 out of 4 = 1/4

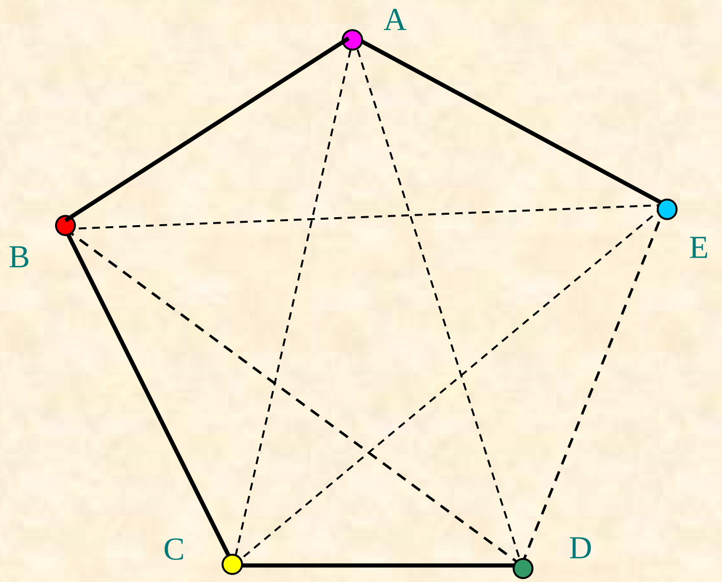
Node 1: Path 124 has 1 and 134 has 1 → 1+1=2 ⇨ 2 out of 4 = 1/2

More about Trees

- **Spanning tree** of a graph *G* is a subgraph that is a tree and it connects all nodes of *G*
- Spanning tree usually is not unique



For example →
Two spanning trees of a graph

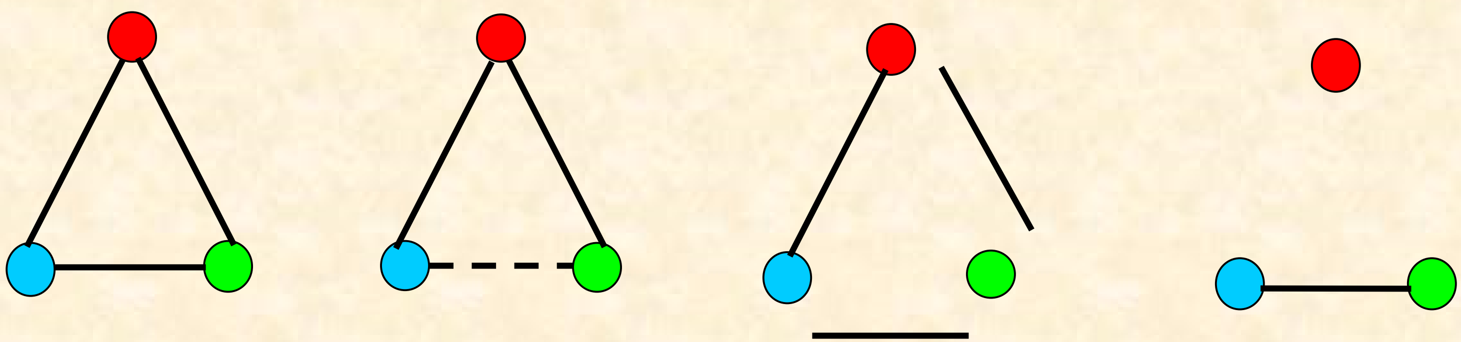


Some Results

Theorem: *Let* T *be any spanning tree of graph* G*. Then*

1) *every cut-set of* G *has an edge in common with* T
2) *every circuit of* G *has an edge in common with the complementary graph of* T

Spanning tree



Cut-set

(2 edges)

Complementary graph

(lower part)

Eigenvalues of Spanning Trees

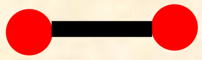
Let L be the Laplacian matrix of a connected graph G of size N, with eigenvalues

$$0 = \lambda_1 < \lambda_2 \le \cdots \le \lambda_N$$

Let $L(i, j)$ be the matrix obtained from L by deleting its i row and j column. Then, the number of spanning trees, n , in G, satisfies

$$n = \left| \det[L(i, j)] \right| = \frac{\lambda_2 \lambda_3 \cdots \lambda_N}{N}$$

Example:



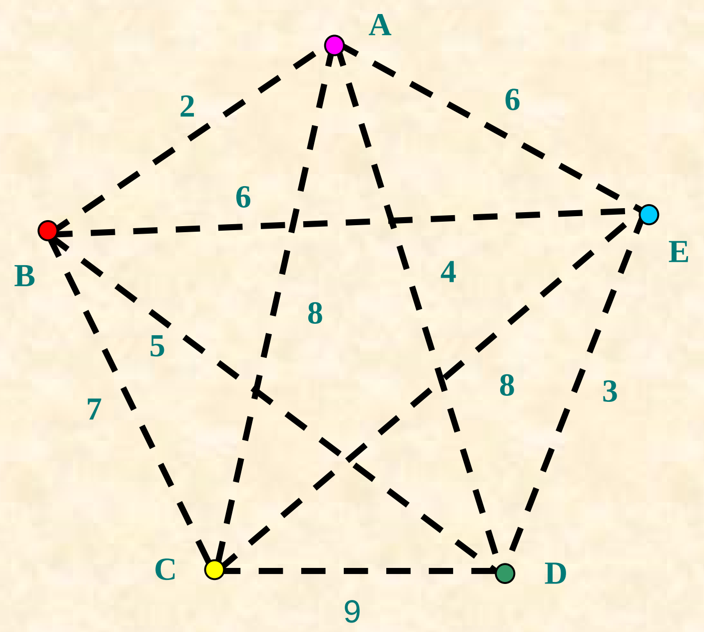
$$L = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \qquad \lambda_1 = 0 \quad \lambda_2 = 2 \qquad n = 1$$

Minimum Connector Problem

- **Minimum connector problem**: Suppose that one wants to build a highway network connecting N given cities, in such a way that a car can travel from any city to any other city, but the total mileage of the highways is minimum.

- Clearly, the graph formed by taking the N cities as nodes and the connecting highways as edges must be a tree.

- The problem is to find an efficient algorithm to decide which tree connecting these cities reaches the minimum total mileage.



Greedy Algorithm

- **Theorem** (Kruskal **Greedy Algorithm**) *Let G be a connected graph with N nodes. Then, the following constructive scheme yields a solution to the minimum connector problem:*
- *let e_1 be an edge of G with the smallest weight;*
- *choose $e_2,\ldots,e_{N-1}$ one by one, by choosing an edge e_i (not previously chosen) with a smallest weight, subject to the condition that it forms no circuit with all the previous edges $\{e_1,\ldots,e_{i-1}\}$;*
- *repeat this procedure until no more edge can be chosen*
- *the resulting graph is a spanning tree, i.e., the subgraph of G with edges $\{e_1,\ldots,e_{N-1}\}$*

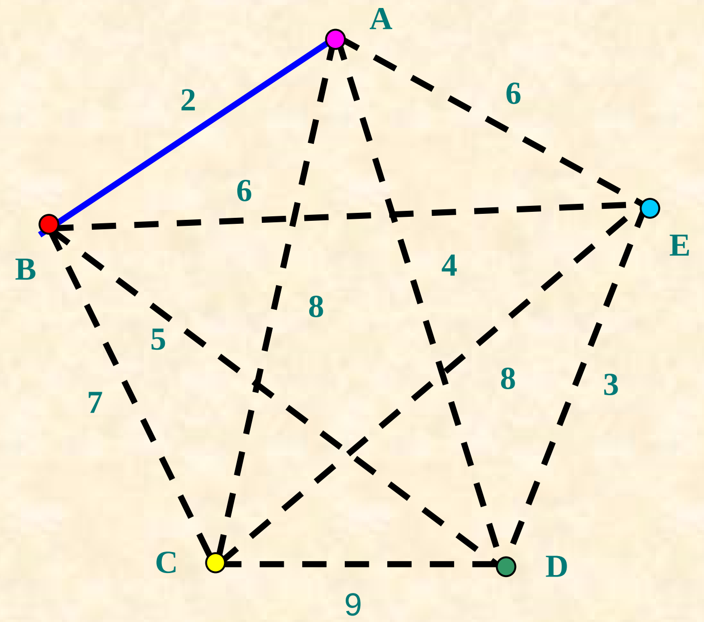
It is a **minimum spanning tree** (usually not unique)

Method 1

Staring from the shortest edge,

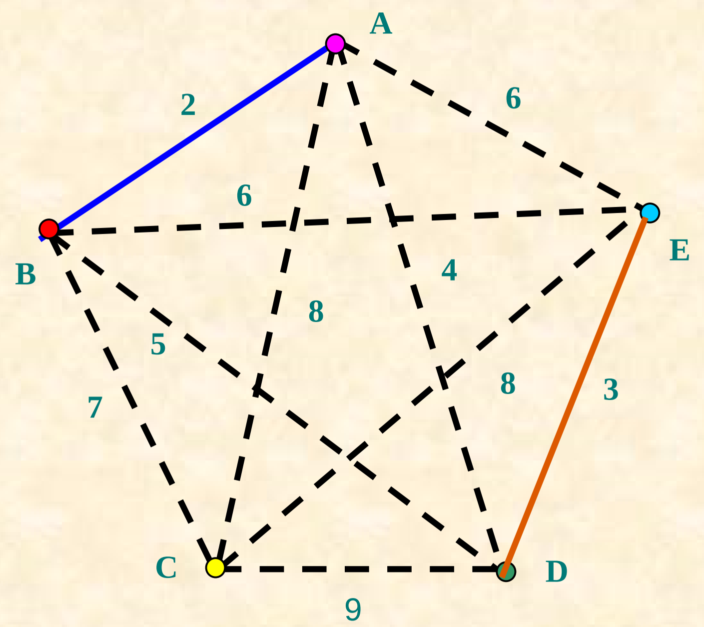
apply the greedy algorithm →

Keep the edge



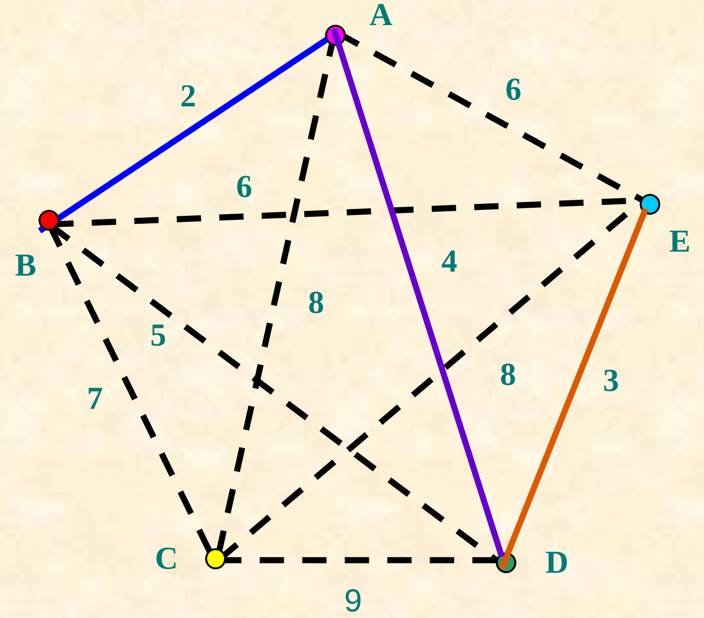
Method 1

Continue the greedy algorithm →



Method 1

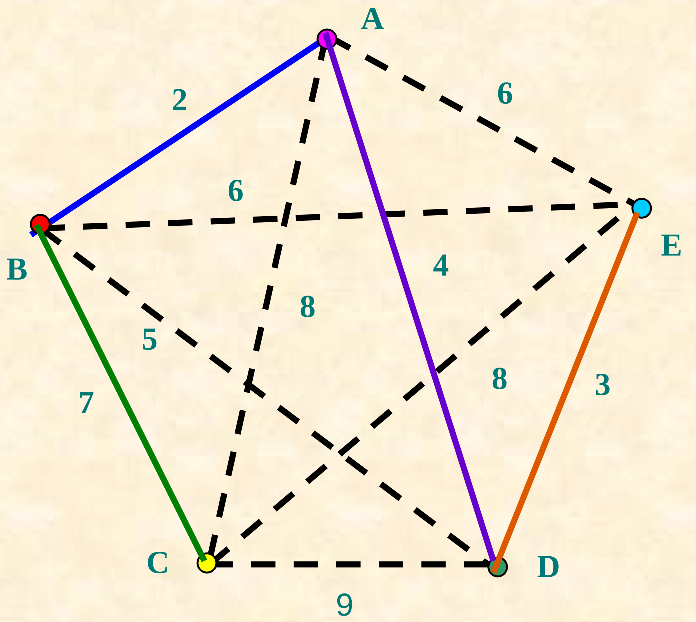
Continue the greedy algorithm →



Method 1

Continue the greedy algorithm →

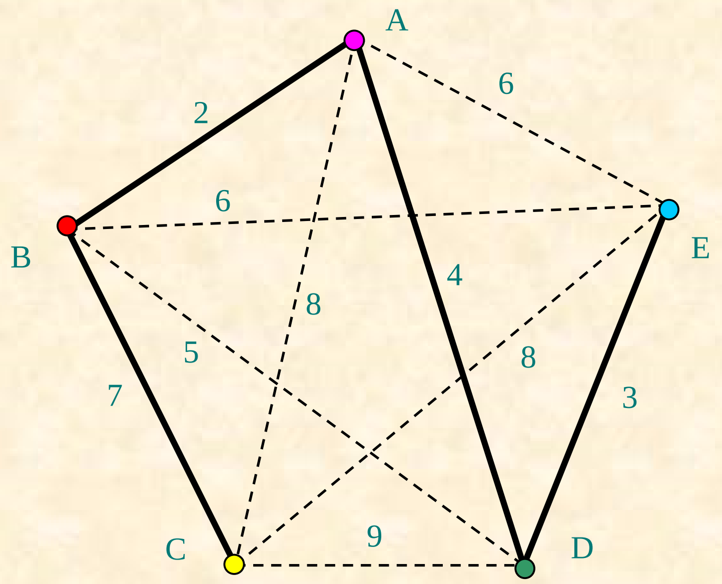
Because BD = 5 or
AE = 6 or BE= 6
will form a circuit



Result

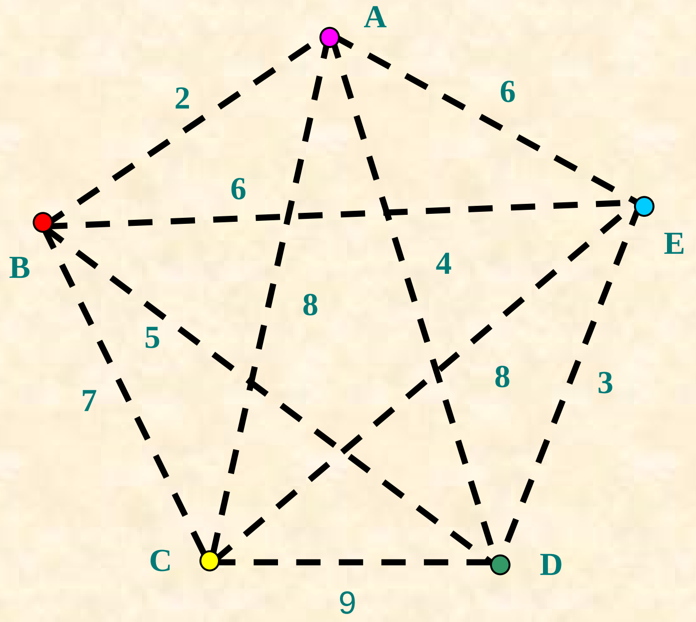
The greedy algorithm finally yields:

No more edge
can be added



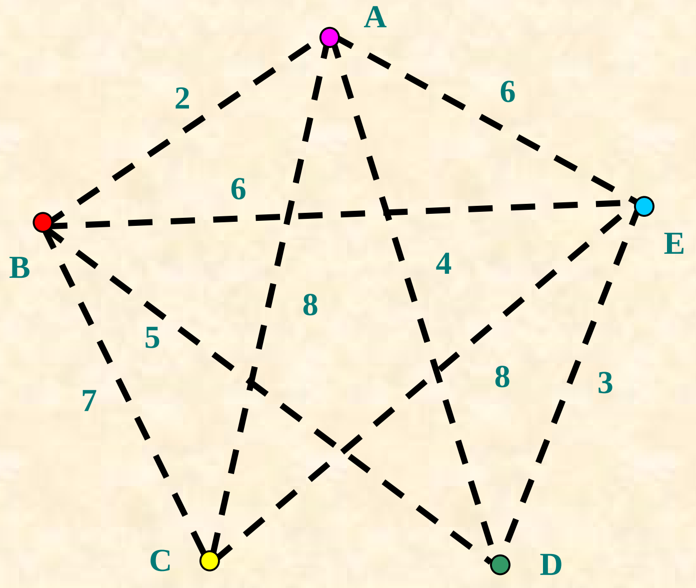
Method 2

Staring from the longest edge,
apply the greedy algorithm →
Remove the edge to break loop



Method 2

Staring from the longest edge,
apply the greedy algorithm →
Remove the edge to break loop

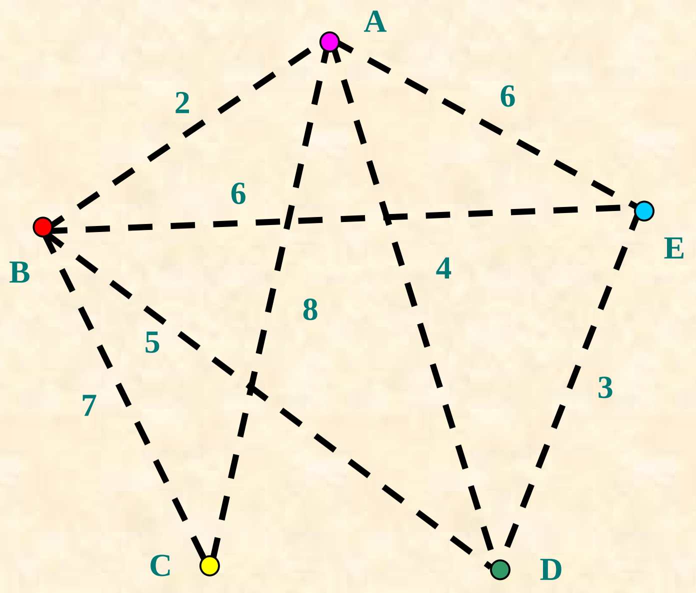


Method 2

Staring from the longest edge,

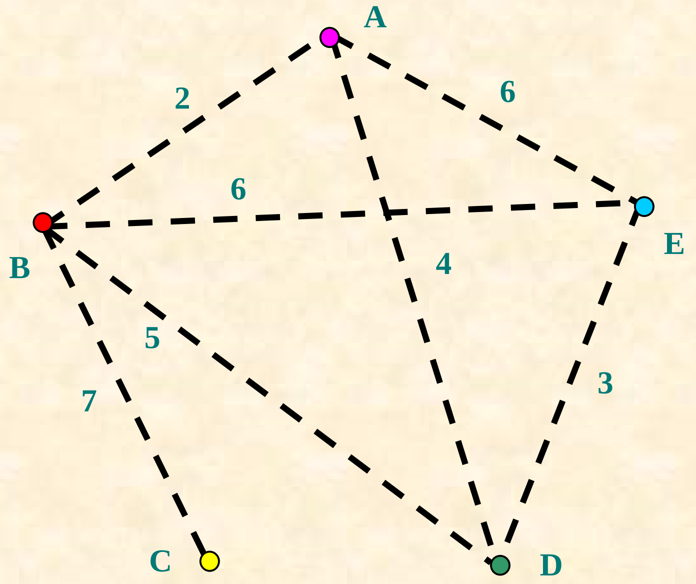
apply the greedy algorithm →

Remove the edge to break loop



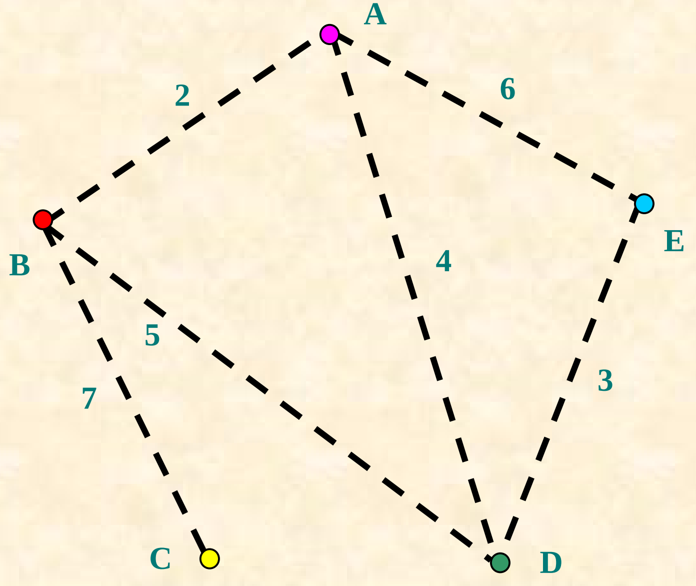
Method 2

Staring from the longest edge,
apply the greedy algorithm →
Remove the edge to break loop



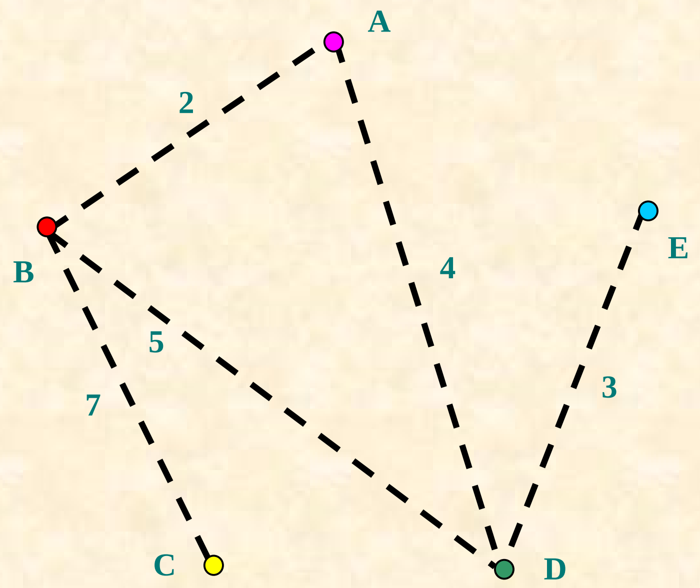
Method 2

Staring from the longest edge,
apply the greedy algorithm →
Remove the edge to break loop



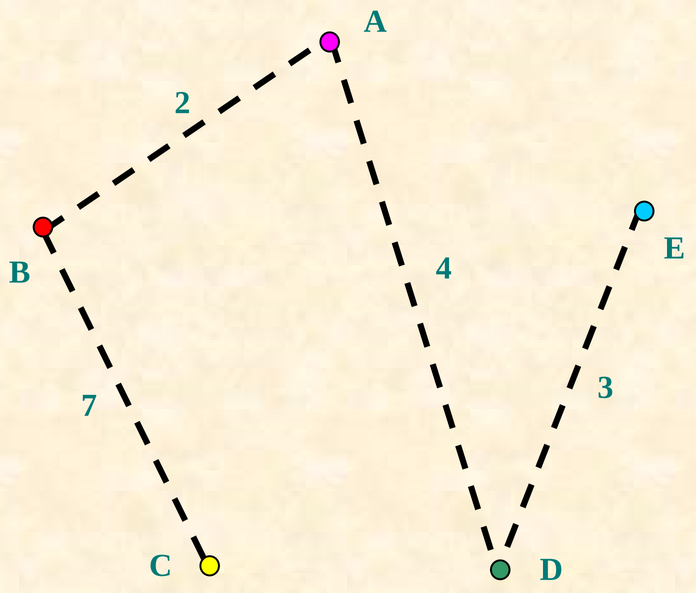
Method 2

Staring from the longest edge,
apply the greedy algorithm →
Remove the edge to break loop



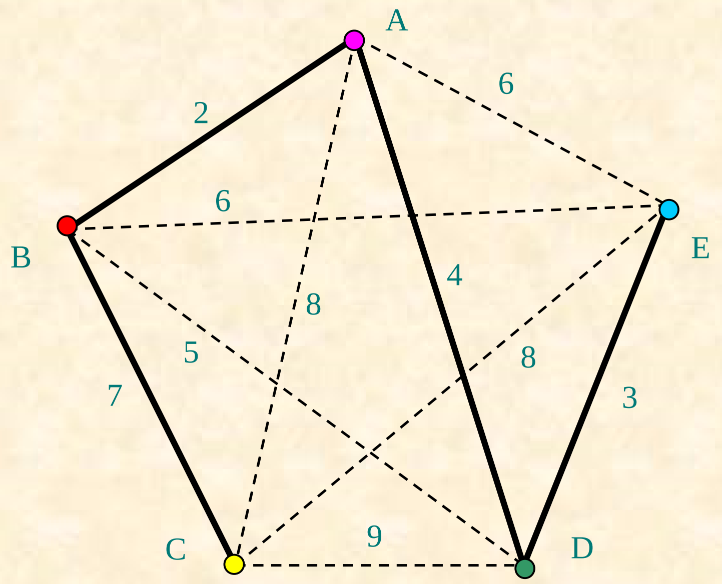
Method 2

Staring from the longest edge,
apply the greedy algorithm →
Remove the edge to break loop



Result

The greedy algorithm finally yields:



No more edge can be removed

End