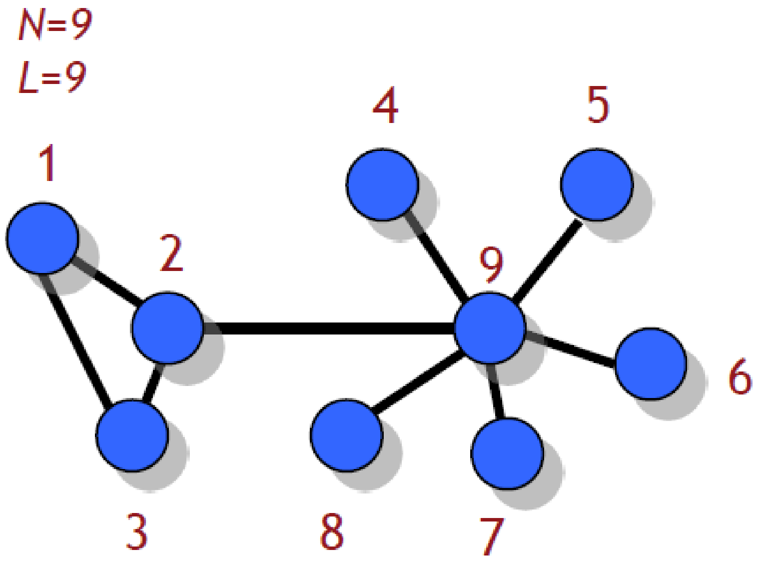
Lecture Two: Fundamentals of Networks

A ***Network*** is a set of elements with connections between them



A network (graph) G = *(V,E)* consists of a set of $V = \{v_1, v_2, \dots, v_N\}$ nodes and a set of $E = \{e_1, e, \dots, e_M\}$ links, where *V* denotes a finite nonempty set of vertices, and *E* denotes a set of edges between pairs of vertices.

A graph is the mathematical abstraction of a network. We will use both terms, graph and network, as synonyms in our course though it is not rigorous.

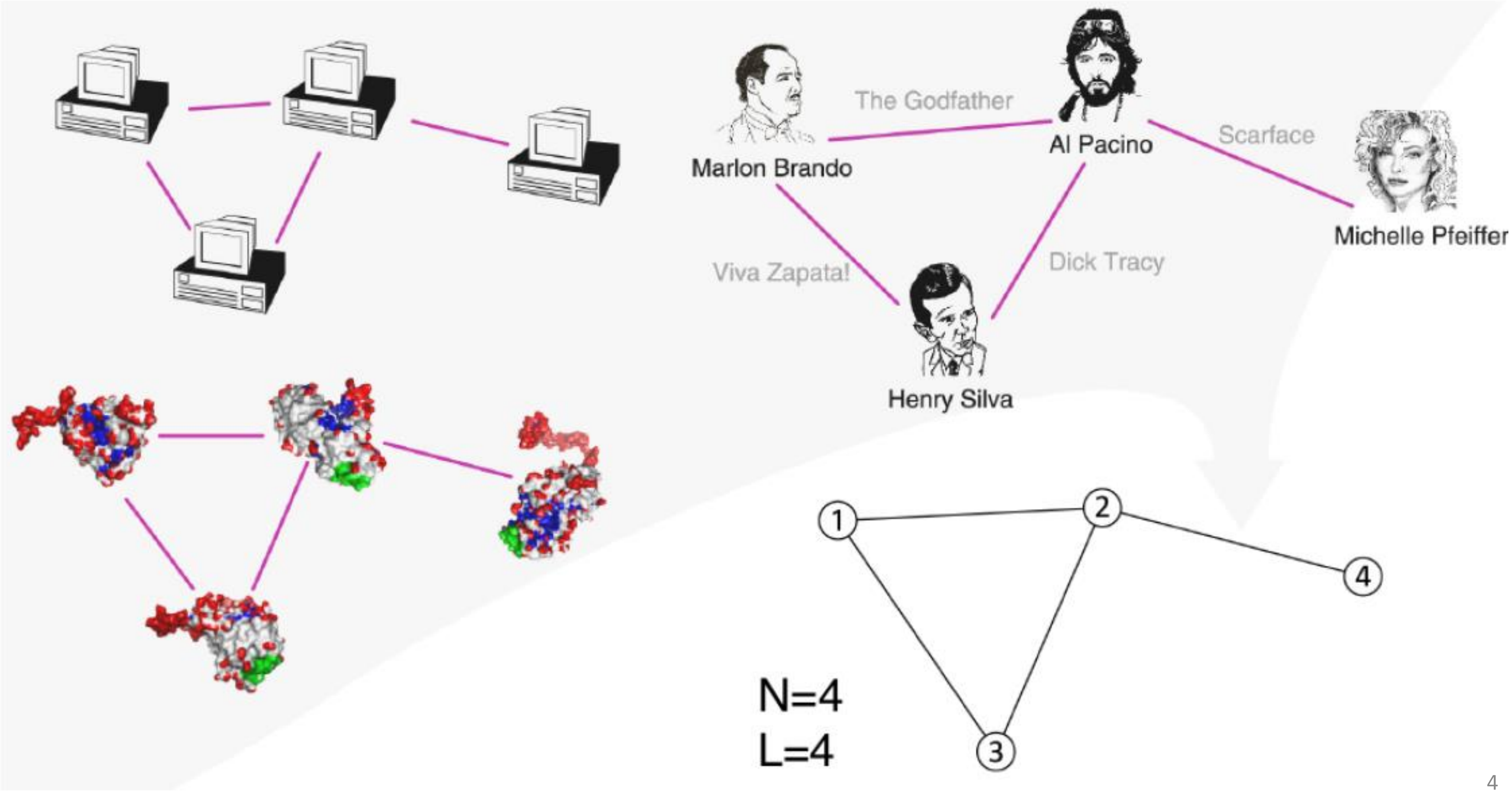
From this viewpoint, each element is represented by a site (physics), node (computer science), actor(sociology) or vertex(graph theory) and the interaction between two elements corresponds to a bond (physics), link(computer science), tie(sociology) or edge (graph theory).

Examples of Networks:

Network (Graph)	Node (Vertex)	Link (Edge)
Internet	Computer or router	Cable or wireless data connection
World Wide Web	Web page	Hyperlink
Citation network	Article, patent, or legal case	Citation
Power grid	Generating station or substation	Transmission line
Friendship network	Person	Friendship
Metabolic network	Metabolite	Metabolic reaction
Neural network	Neuron	Synapse
Food web	Species	Predation

Table 1: ***Nodes and links in networks.*** Some examples of nodes and links in particular networks.

Nodes and links



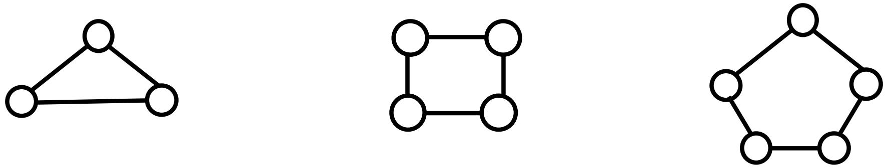
Some Basics of Networks

- ***Density*** is the number of lines in a simple network, expressed as a proportion of the maximum possible number of lines.
- A ***complete*** network is a network with maximum density.
- ***Degree*** of a node is the number of lines attached to it.
- Two nodes are ***adjacent*** (distance =1) when connected by a link.
- The ***in-degree*** of a node is the number of arcs it receives.
- The ***out-degree*** is the number of arcs it sends.

Some Basics of Graphs

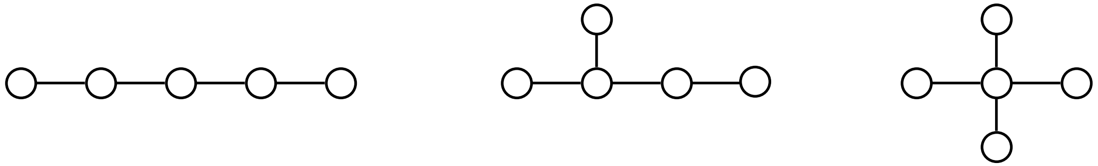
Two vertices in a graph can be connected in different ways.

Cycle



- a cycle is a *closed* path that visits a set of vertices only *once* (apart from the end-vertices that coincide)

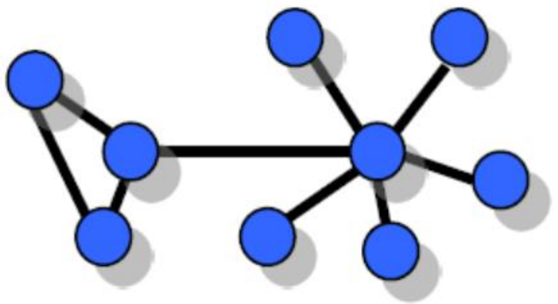
Tree



- a set of vertices connected to each other without cycles is a tree

- a set of *disconnected* trees is a forest

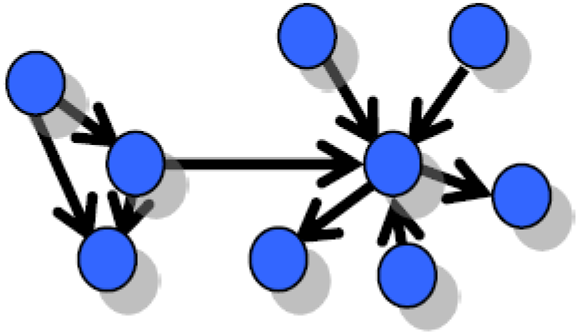
Some Basics of Graphs



An undirected network

An undirected link indicates a symmetric interaction. The graphical representation of undirected links is a line. If node *j* is linked to node *i* then also node *i* is linked to node *j*.

An ***undirected network*** is a network in which every link is undirected.



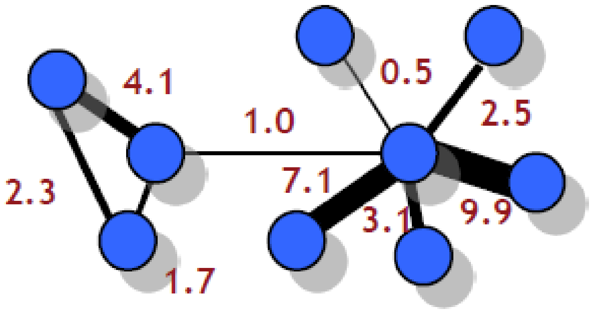
A directed network

A directed link indicates an interaction between nodes that is not symmetrical. The graphical representation of directed links is an arrow. If node *j* points to node i the arrow starts from node j and points to node *i*.

A ***directed network*** is a network where all the links are directed. A ***basis*** of a directed network (digraph) is a minimal set of vertices such that every other vertex can be reached from some vertex in this set by a directed path

Some Basics of Graphs

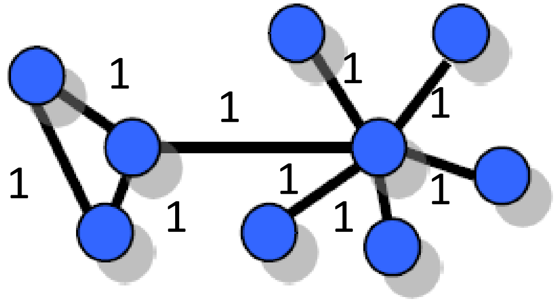
A weighted network



A *weighted* link between node *i* and node *j* is a link to which we assign an integer or real number indicating the intensity of the interaction. When the weight is integer the weighted link is also called multiple link.

A weighted network is a network where all the links are weighted. It can be either directed or undirected.

An unweighted network



An *unweighted* network is a network in which all the links are unweighted. It can be either directed or undirected.

*** One can also have networks that have weights on their nodes, or more exotic variables on either the links or nodes, or enumerative variables like color.

Some Basics of Graphs

More definitions:

- A signed link is a link associated with a sign (either positive or negative). Signed networks can be also weighted and/or directed. A signed network is a network of which all the links are signed.

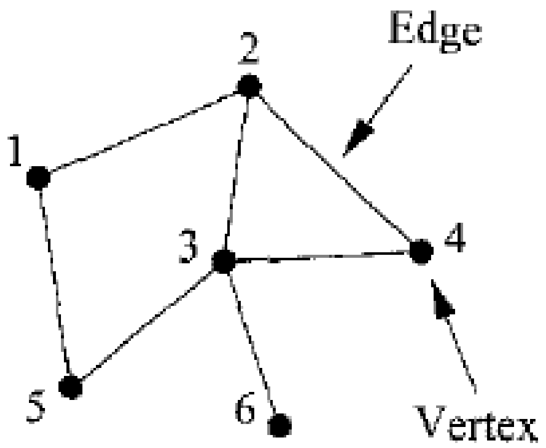
- An unsigned network is a network in which all the links have the same sign.

- Tadpoles are links that connect a node with itself. Tadpoles can be directed or undirected.

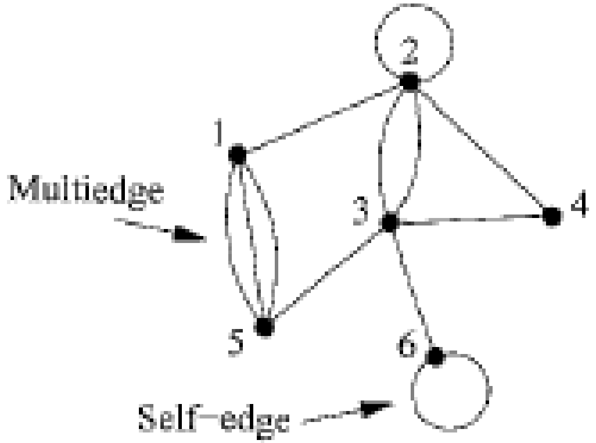
- A simple network is an undirected, unweighted, and unsigned network without tadpoles.

*** One can also have networks that have weights on their nodes, or more exotic variables on either the links or nodes, or enumerative variables like color.

Some Basics of Graphs



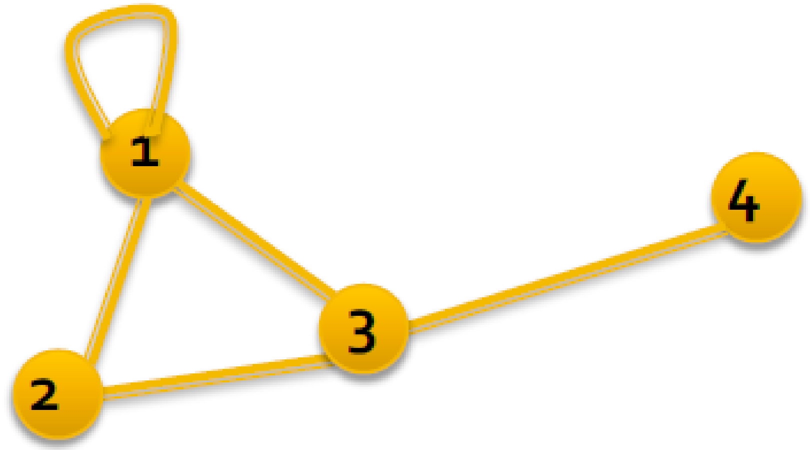
A simple graph with no self-loop edge and multi-edges.



A graph with self-loop edges and multi-edges.

Adjacency matrix

If we denote an edge between vertices i and j by (i, j) then the complete network can be specified by giving the number of vertices N and a list of all the edges.



$N = 4$

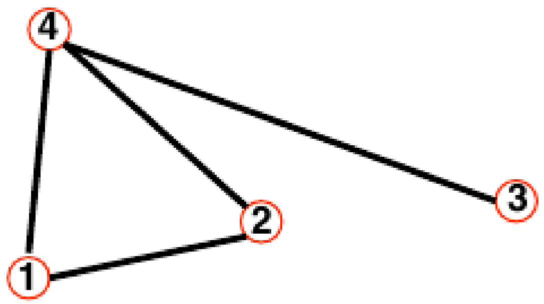
Edge list:

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

Edge lists are sometimes used to store the structure of networks on computers, but for mathematical developments like those in this chapter they are rather **cumbersome.**

Adjacency matrix

A better representation of a network for present purposes is the *adjacency matrix*. The adjacency matrix A of a simple graph is the matrix with elements A_{ij} such that



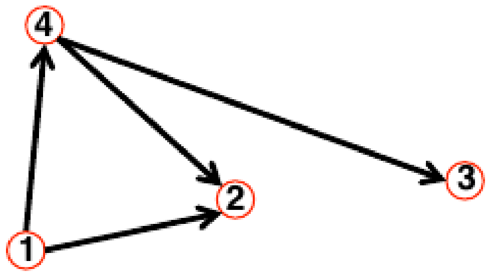
$A_{ij} = 1$, if there is a link between nodes i and j

$A_{ij} = 0$, if nodes i and j are not connected to each other

$$A_{ij} = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

Adjacency matrix

For a directed graph:



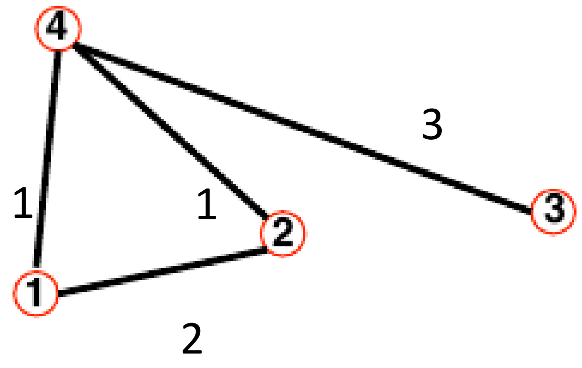
$$A_{ij} = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

$A_{ij} = 1$, if there is a link pointing from node *i* to node *j*

$A_{ij} = 0$, if there is no link pointing from node *i* to node *j*

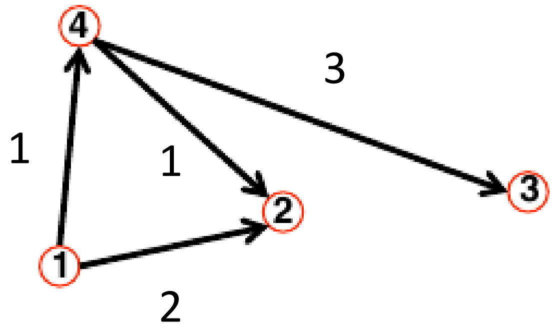
Adjacency matrix

An undirected but weighted graph



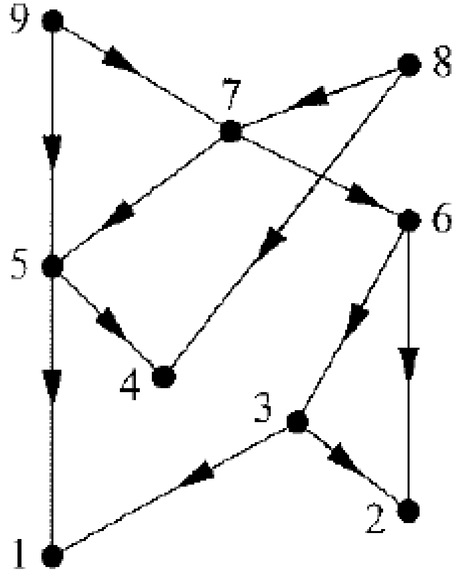
$$\begin{bmatrix} 0 & 2 & 0 & 1 \\ 2 & 0 & 0 & 1 \\ 0 & 0 & 0 & 3 \\ 1 & 1 & 3 & 0 \end{bmatrix}$$

A directed and weighted graph



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

Directed acyclic graph (DAG)



$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

A directed acyclic graph (DAG), is a finite directed graph with no directed cycles. An example is a citation network of scientific papers. In a DAG, there must be at least one vertex somewhere on the network that has ingoing edges only and no outgoing ones.

A simple algorithm for determining whether a network is acyclic is:

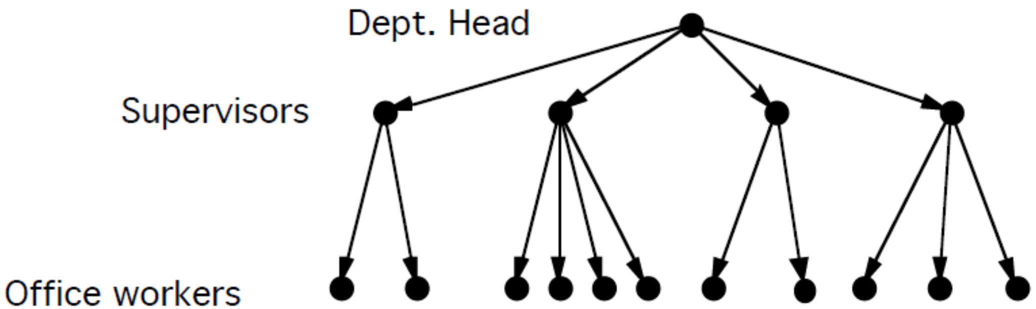
- Find a vertex with no outgoing edges.
- If no such vertex exists, the network is cyclic. Otherwise, if such a vertex does exist, remove it and all its ingoing edges from the network.
- If all vertices have been removed, the network is acyclic. Otherwise go back to step 1.

**The adjacency matrix of an acyclic network has all eigenvalues zero and a network is acyclic if its adjacency matrix has all eigenvalues zero.

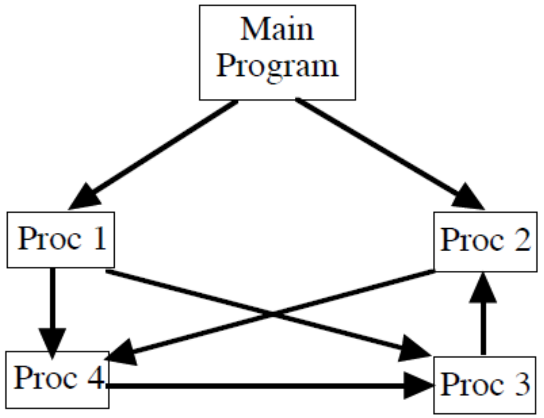
Adjacency matrix for the corresponding network shown above. The matrix is **upper triangular.

Directed acyclic graph (DAG)

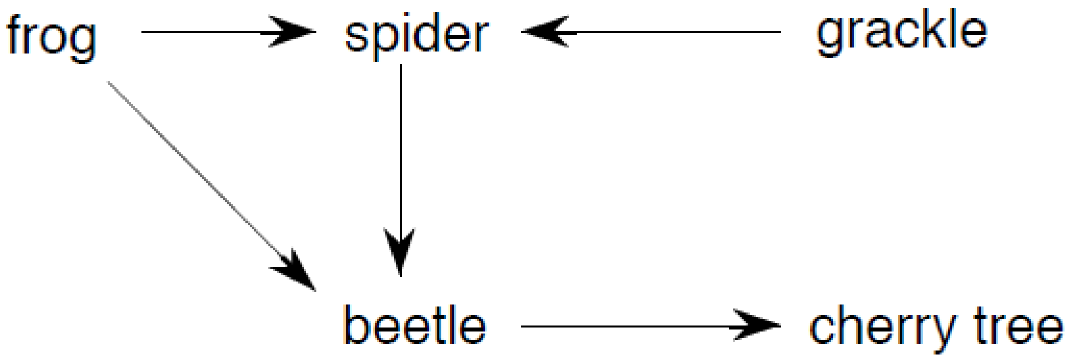
Some Examples:



A corporate hierarchy



Call graph of a computer program



A small food web

Hypergraphs

A hypergraph is a generalization of a graph in which an edge can join any number of vertices. a hypergraph ***H*** is a pair ***H = (X,E)*** where ***X*** is a set of elements called nodes or vertices, and ***E*** is a set of non-empty subsets of ***X*** called hyperedges or edges.

A hypergraph and corresponding bipartite graph.
These two networks show the same information -- the membership of five vertices in four different groups, (a) The hypergraph representation in which the groups are represented as hyperedges, denoted by the loops circling sets of vertices. (b) The bipartite representation in which we introduce four new vertices (open circles) representing the four groups, with edges connecting each vertex to the groups to which it belongs.

Hypergraphs

Network	Vertex	Group
Film actors	Actor	Cast of a film
Coauthorship	Author	Authors of an article
Boards of directors	Director	Board of a company
Social events	People	Participants at social event
Recommender system	People	Those who like a book, film, etc.
Keyword index	Keywords	Pages where words appear
Rail connections	Stations	Train routes
Metabolic reactions	Metabolites	Participants in a reaction

Table 2: ***Hypergraphs and bipartite graphs.*** Examples of networks that can be represented as hypergraphs or equivalently as bipartite graphs.

Bipartite Networks

A bipartite network $\boldsymbol{G_B = (V;U;E)}$ is a network formed by two non-overlapping sets of nodes $\boldsymbol{U}$ and $\boldsymbol{V}$ and by a set of links $\boldsymbol{E}$, such that every link joins a node in $\boldsymbol{V}$ with a node in $\boldsymbol{U}$.

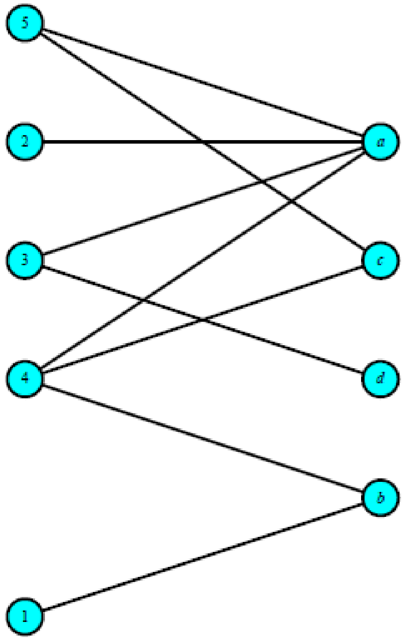
Bipartite network	**Nodes $i \in V$**	**Groups $a \in U$**
Bipartite Actors network	Actors	Films
Bipartite Collaboration networks	Scientists	Papers
Bipartite Board of Directors	Directors	Board of a company
Bipartite Metabolic network	Metabolites	Chemical reaction

Table 3: Examples of bipartite complex networks.

Bipartite Networks

A bipartite network $\boldsymbol{G_B} = \boldsymbol{(V;U;E)}$ is described by an ***incidence matrix***. The incidence matrix of a bipartite network $\boldsymbol{G_B} = \boldsymbol{(V;U;E)}$ is an $N_V \times N_U$ matrix of elements B_{ij} defined as follows:

$$\mathrm{B}_{ij} = \begin{cases} 1 & \text{if vertex } i \text{ belongs to group } j, \\ 0 & \text{otherwise} \end{cases}$$

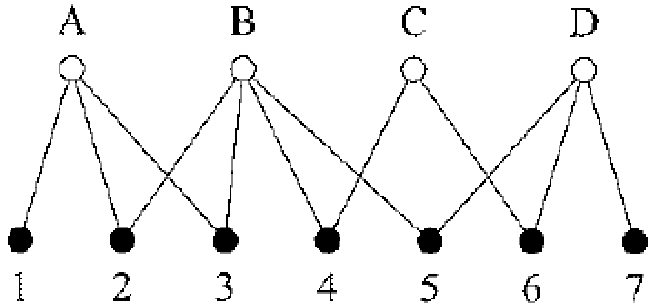


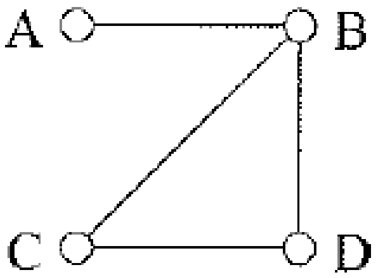
$$\mathbf{B} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

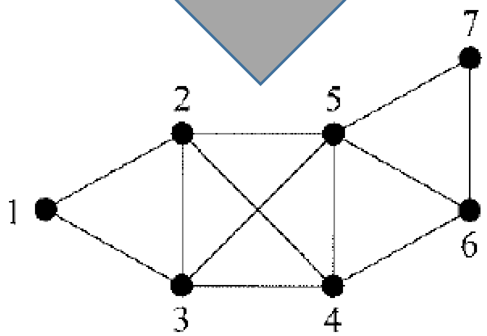
Example of a bipartite undirected and unweighted network of $N_V = 5$ nodes and $N_U = 4$ groups

Bipartite Networks

We can also use the bipartite network to infer direct connections between vertices of just one type, creating a *one-mode projection* from the two-mode bipartite form.

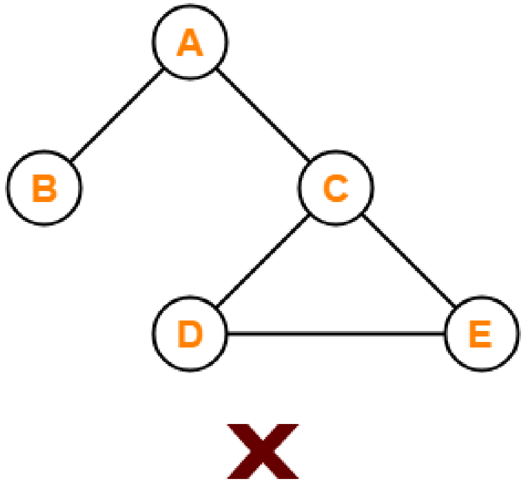




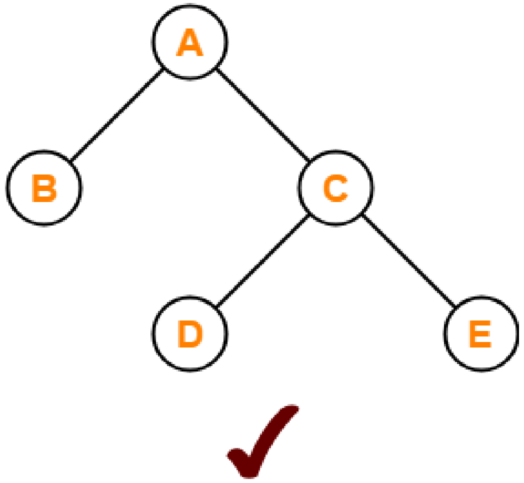


The two one-mode projections of a bipartite network.
The bipartite network with four vertices of one type (open circles labeled A to D) and seven of another (filled circles, 1 to 7). On the right and at the bottom we show the one-mode projections of the network onto the two sets of vertices.

Trees and Planar Graphs



This graph is not a Tree



This graph is a Tree

Trees

A *tree* is a connected, undirected graph that contains no closed loops. “Connected” here means that every vertex in the graph is reachable from every other via some path through the graph.

****Examples: minimum spanning trees (MST), Bethe Lattices*

- A *forest* is a graph in which all its parts are trees. Since there are no closed loops, there is exactly one path between any pair of vertices.

- A tree of n vertices always has exactly n - 1 edges. Likewise, any connected graph with n vertices and n - 1 edges is a tree.

Trees and Planar Graphs

Planar Graphs

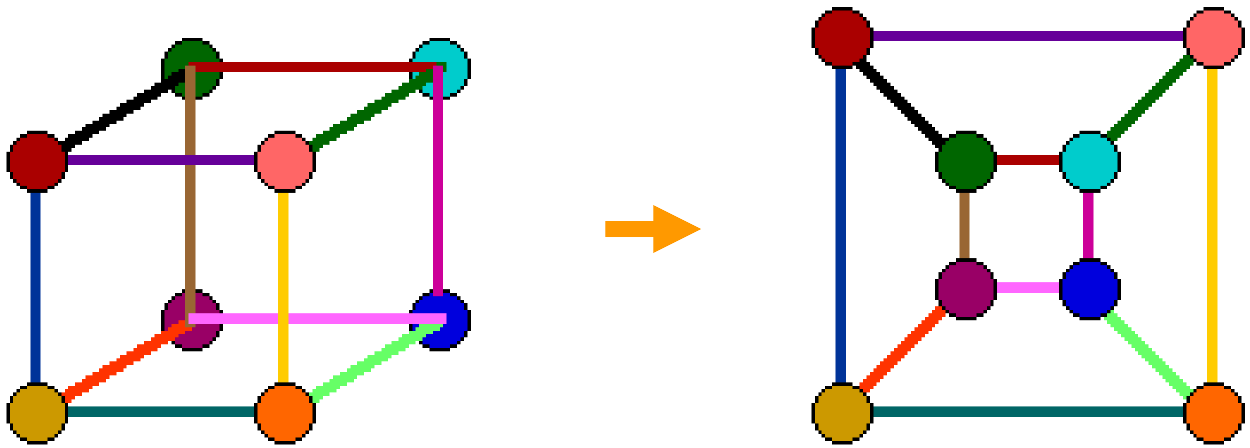
A *planar graph* is a graph that can be drawn on a plane without having any edges crossing each other. Notice that it is in most cases possible to find a way to draw a planar network so that some edges do cross. (See the example below)

****Examples: River networks, road networks*

Two drawings of a planar graph. (a) A small planar graph with four vertices and six edges. The graph is planar, since it has no edges that cross. (b) The same graph redrawn with two of its edges crossing. Even though the edges cross, the graph is still planar → a graph is planar if it *can* be drawn without crossing edges.

Trees and Planar Graphs

Planar Graphs: *An example of a 3-D graph mapping into a 2-D planar graph*



Trees and Planar Graphs

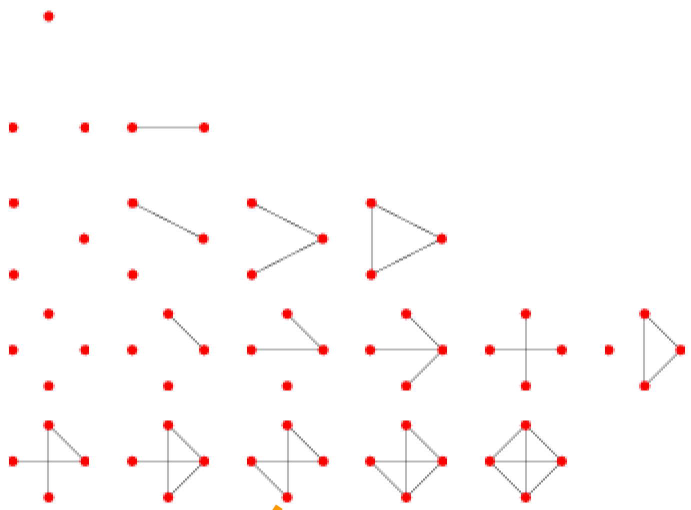
Number of planar graphs of different sizes:

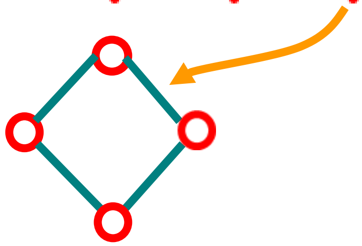
1: 1

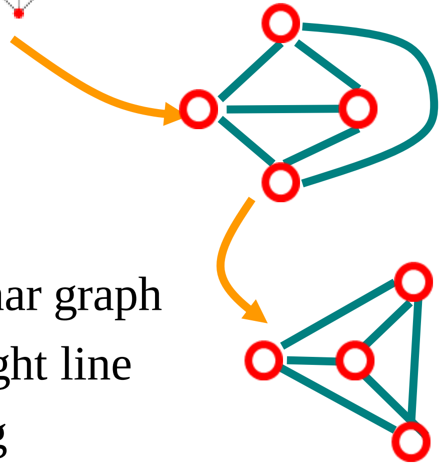
2: 2

3: 4

4: 11



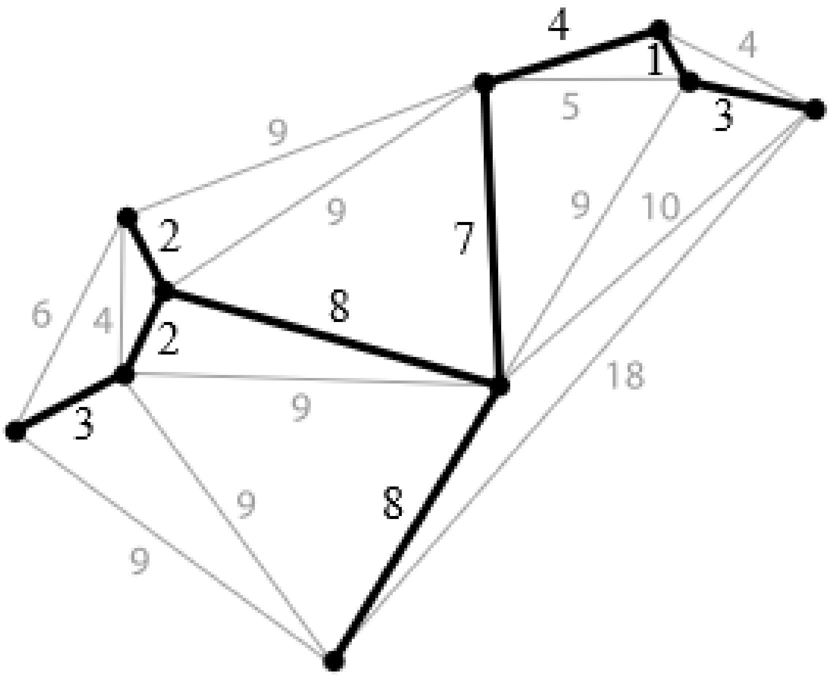




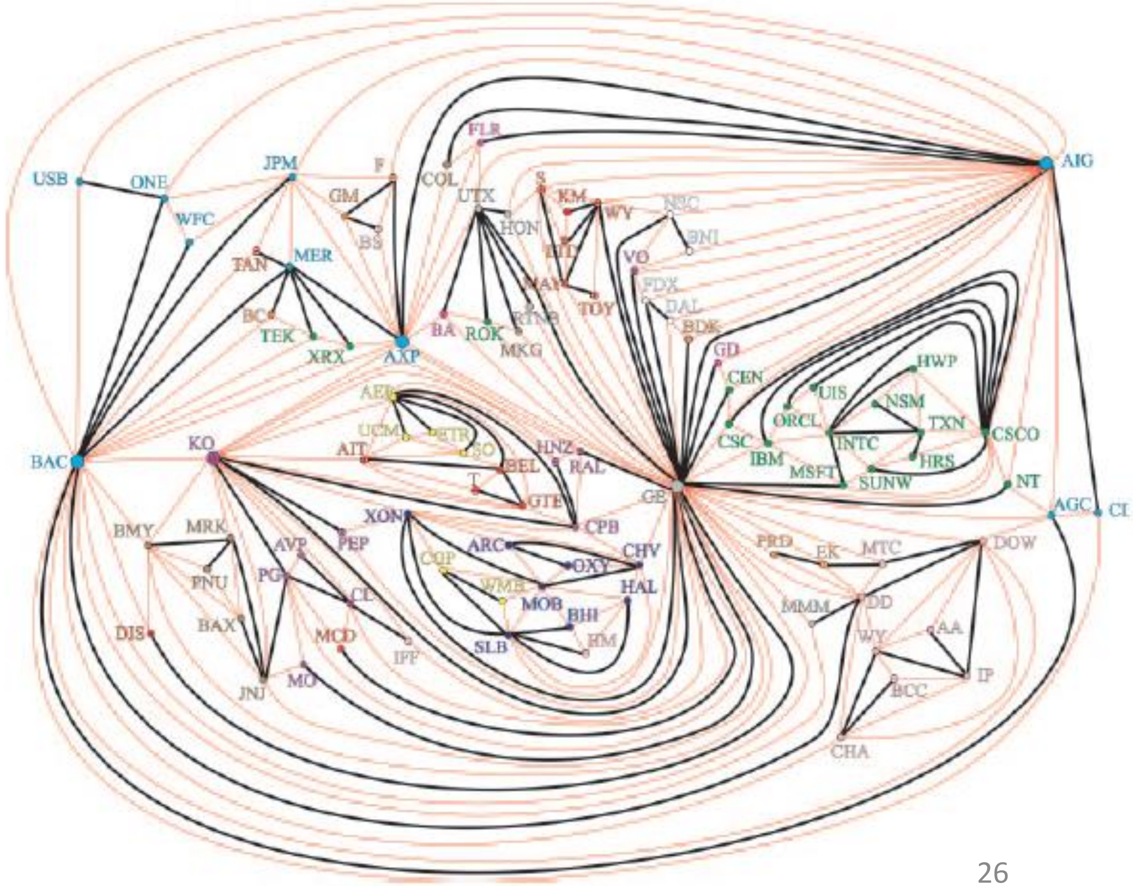
Every planar graph has a straight line embedding

Trees and Planar Graphs

A ***minimum spanning tree (MST)*** is a subset of the edges of a connected, edge-weighted (un)directed graph that connects all the vertices together, without any cycles and with the minimum possible total edge weight.

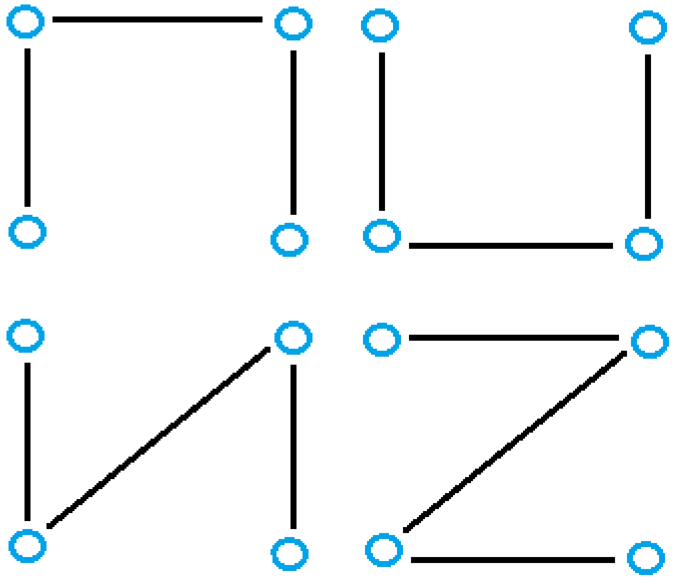


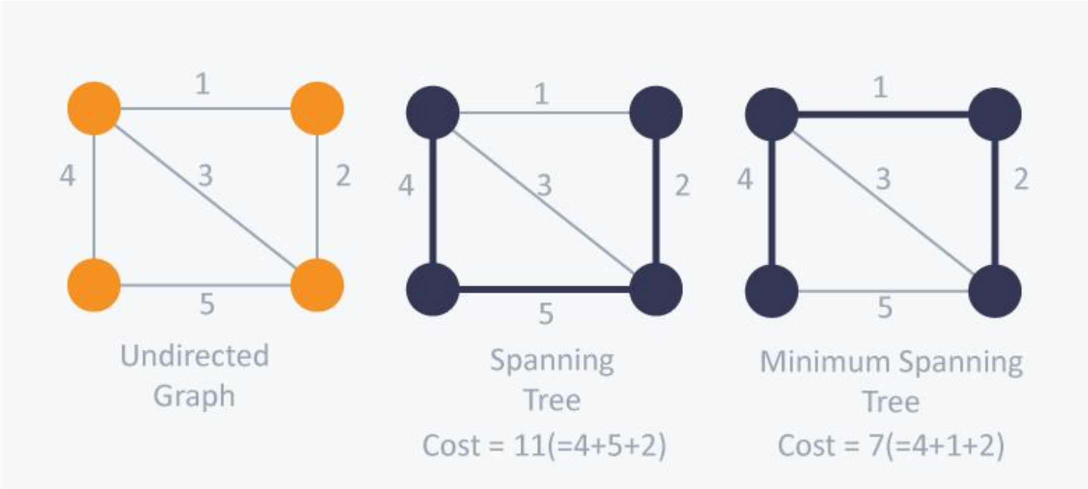
Planar Maximally Filtered Graph (PMFG)



Minimum Spanning Tree (MST)

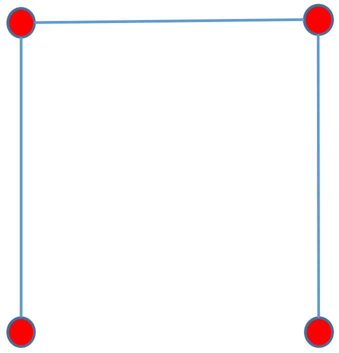
If there are n vertices in the graph, then each spanning tree has $n-1$ edges. If each edge has a distinct weight then there will be only one, unique minimum spanning tree. Otherwise, there may be several minimum spanning trees of the same weight; in particular, if all the edge weights of a given graph are the same, then every spanning tree of that graph is minimum.





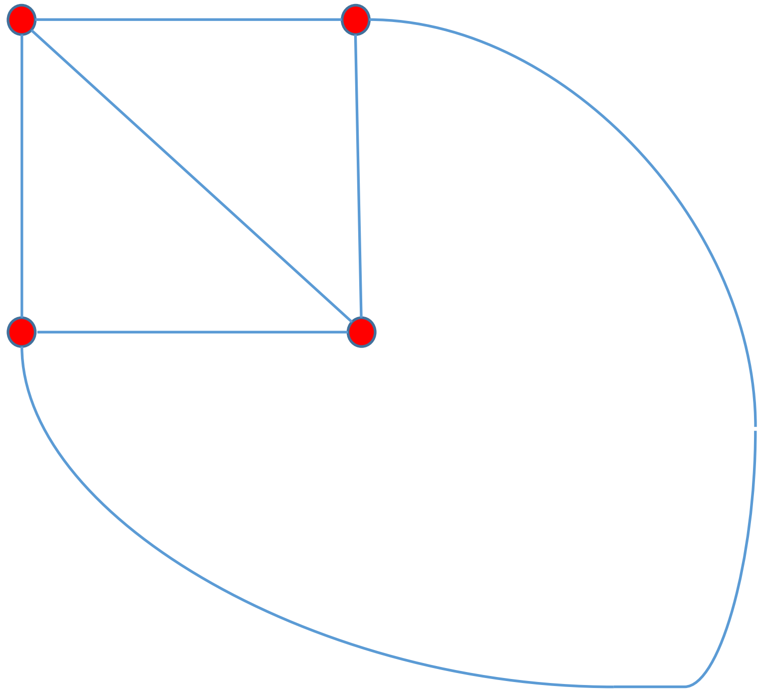
Trees and Planar Graphs

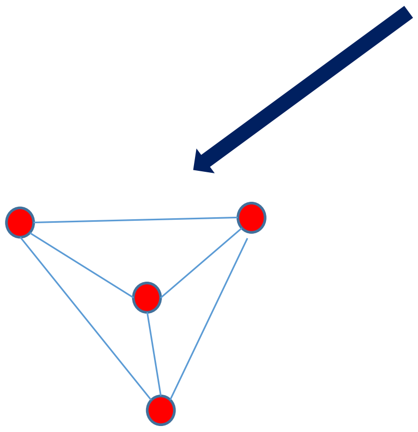
Minimum Spanning Tree (MST)



n = 4; *L* = *n* – 1 = 3

Planar Maximally Filtered Graph (PMFG)

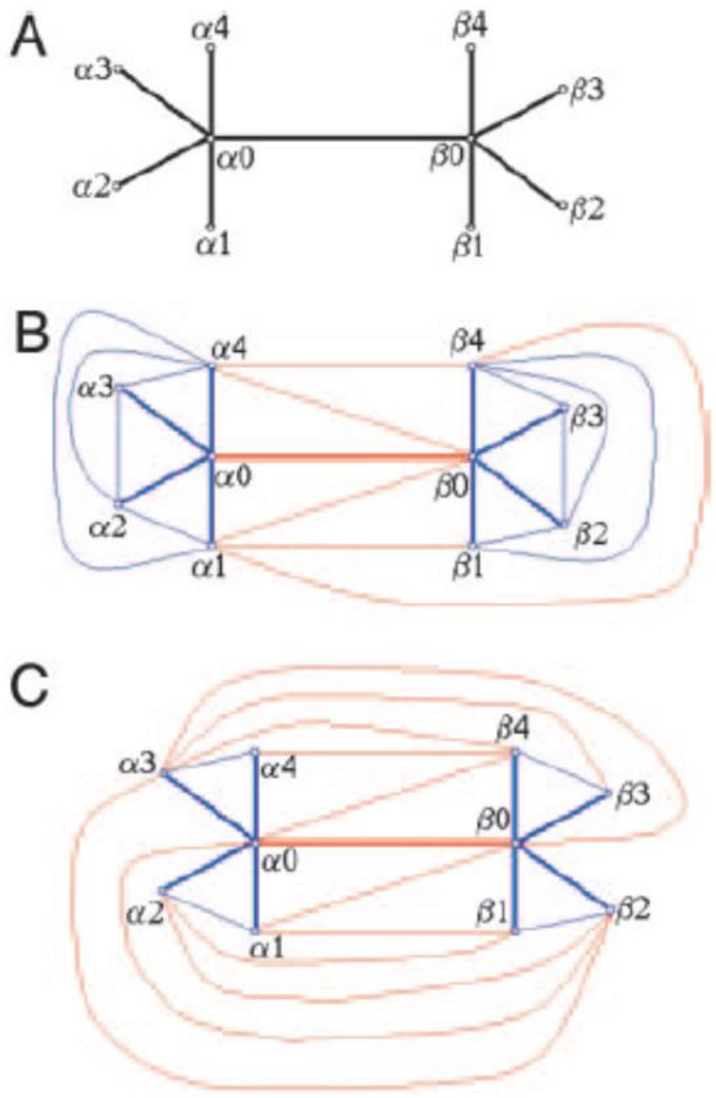




**Every planar graph has a straight line embedding

n = 4; *L* = 3(*n* – 2) = 6

Trees and Planar Graphs



An ***illustrative example*** of two graphs that share the same MST but have distinct PMFGs. (A) MST of a simple system of 10 vertices. (B and C) PMFG of two systems with the same MST (the one drawn in A). The thicker lines are identifying links belonging to both the MST and the PMFG, whereas the thinner lines belong to the PMFG only.

PMFG has 3(n - 2) edges.

Node degree

The *degree* of a node in a network is the number of links connected to it. We will denote the degree of node i by k_i. For an undirected network of *N* nodes the degree can be written in terms of the adjacency matrix as

$$k_i = \sum_{j=1}^{N} A_{ij}$$

Every link in an undirected network has two ends and if there are *L* links in total then there are 2*L* ends of links. The number of ends of links is also equal to the sum of the degrees of all the nodes, therefore

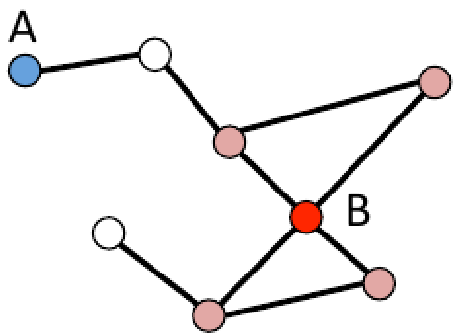
$$2L = \sum_{i=1}^{N} k_i$$

or

$$L = \frac{1}{2}\sum_{i=1}^{N} k_i = \frac{1}{2}\sum_{ij} A_{ij}$$

Node degree

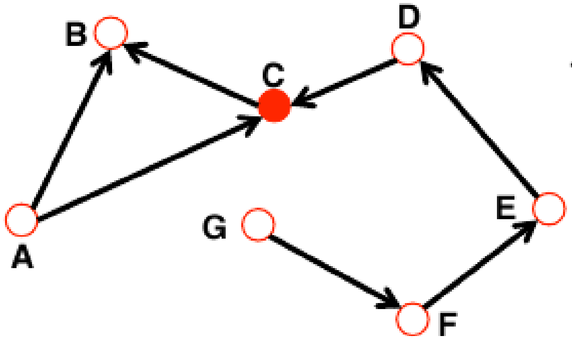
Undirected



Node degree: the number of links connected to the node.

$$k_A = 1 \qquad k_B = 4$$

Directed



In *directed networks* we can define an in-degree and out-degree.

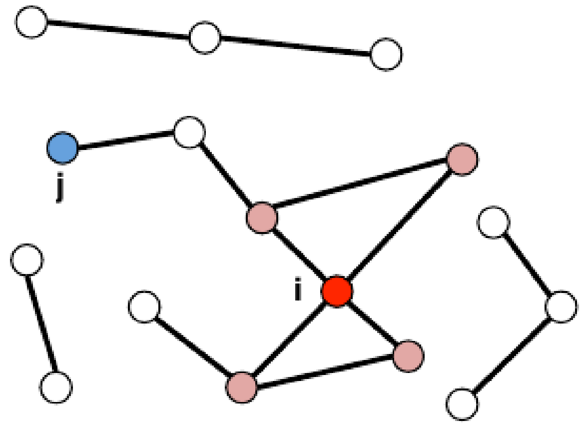
The (total) degree is the sum of in- and out-degree.

$$k_C^{in} = 2 \qquad k_C^{out} = 1 \qquad k_C = 3$$

Source: a node with $k^{in} = 0$; Sink: a node with $k^{out} = 0$.

Node degree

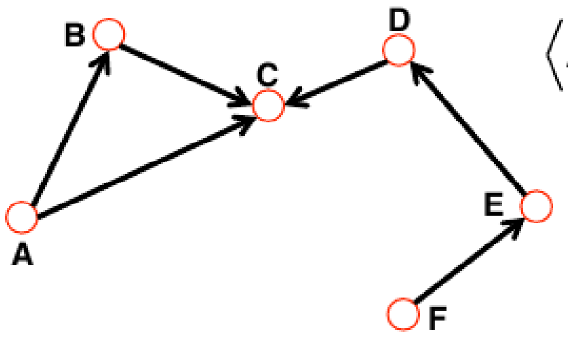
Undirected



$$\langle k \rangle \equiv \frac{1}{N}\sum_{i=1}^{N} k_i \qquad \langle k \rangle \equiv \frac{2L}{N}$$

N – the number of nodes in the graph

Directed



$$\langle k^{in} \rangle \equiv \frac{1}{N}\sum_{i=1}^{N} k_i^{in}, \quad \langle k^{out} \rangle \equiv \frac{1}{N}\sum_{i=1}^{N} k_i^{out}, \quad \langle k^{in} \rangle = \langle k^{out} \rangle$$

$$\langle k \rangle \equiv \frac{L}{N}$$

Node degree

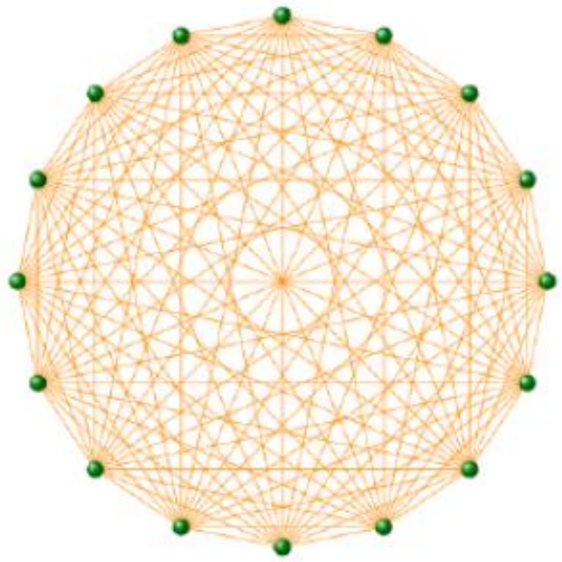
NETWORK	NODES	LINKS	DIRECTED UNDIRECTED	N	L	⟨k⟩
Internet	Routers	Internet connections	Undirected	192,244	609,066	6.33
WWW	Webpages	Links	Directed	325,729	1,497,134	4.60
Power Grid	Power plants, transformers	Cables	Undirected	4,941	6,594	2.67
Mobile Phone Calls	Subscribers	Calls	Directed	36,595	91,826	2.51
Email	Email addresses	Emails	Directed	57,194	103,731	1.81
Science Collaboration	Scientists	Co-authorship	Undirected	23,133	93,439	8.08
Actor Network	Actors	Co-acting	Undirected	702,388	29,397,908	83.71
Citation Network	Paper	Citations	Directed	449,673	4,689,479	10.43
E. Coli Metabolism	Metabolites	Chemical reactions	Directed	1,039	5,802	5.58
Protein Interactions	Proteins	Binding interactions	Undirected	2,018	2,930	2.90

Node degree

The maximum number of links of a network of *N* nodes can have is

$$L_{\max} = \binom{N}{2} = \frac{N(N-1)}{2}$$

A graph with $L = L_{max}$ is called a ***complete graph*** and its average degree is $N - 1$.



The Density (or Connectance) ρ of a network is therefore defined as

$$\rho = \frac{\langle k \rangle}{N-1}$$

A network for which the density ρ tends to a constant as $N \rightarrow \infty$ is said to be *dense*. In such a network the fraction of non-zero elements in the adjacency matrix remains constant as the network becomes large. A network in which $\rho \rightarrow 0$ as $N \rightarrow \infty$ is said to be *sparse*.

Node degree

Most networks observed in real systems are indeed sparse:

$$L \ll L_{max} \quad \text{or} \quad \langle k \rangle \ll N - 1$$

WWW (ND Sample):	N=325,729;	L=1.4 10^6	$L_{max}=10^{12}$	<k>=4.51
Protein (*S. Cerevisiae*):	N= 1,870;	L=4,470	$L_{max}=10^7$	<k>=2.39
Coauthorship (Math):	N= 70,975;	L=2 10^5	$L_{max}=3\ 10^{10}$	<k>=3.9
Movie Actors:	N=212,250;	L=6 10^6	$L_{max}=1.8\ 10^{13}$	<k>=28.78

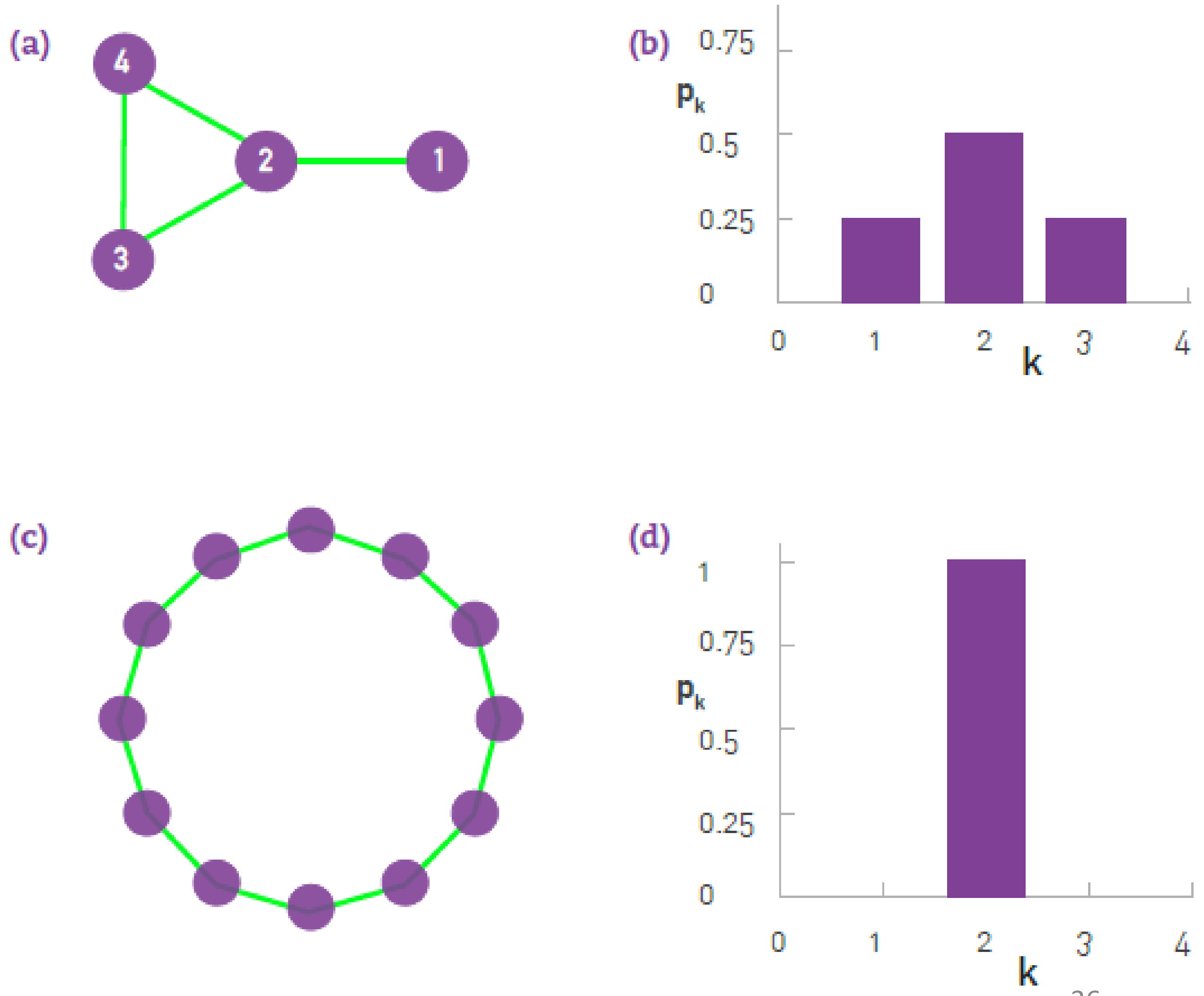
(Source: Albert, Barabasi, RMP2002)

Degree distribution

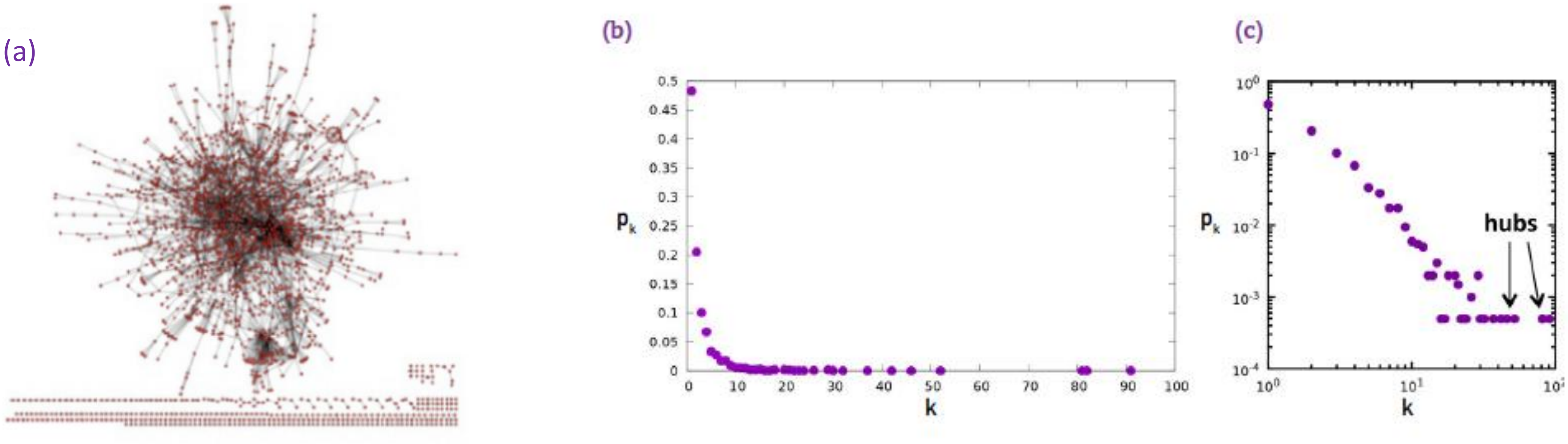
P(k): probability that a randomly chosen node has degree *k*

N_k = # nodes with degree *k*

$P(k) = \frac{N_k}{N}$ → plot



Degree distribution



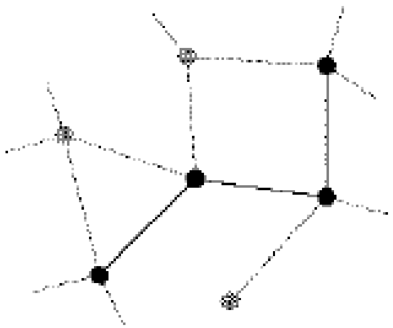
Degree Distribution of a Real Network

(a) A layout of the protein interaction network of yeast. Each node corresponds to a yeast protein and links correspond to experimentally detected binding interactions. Note that the proteins shown on the bottom have self-loops, hence for them k=2.

(b) The degree distribution of the protein interaction network shown in (a). The observed degrees vary between k=0 (isolated nodes) and k=92, which is the degree of the most connected node, called a **hub**. There are also wide differences in the number of nodes with different degrees: Almost half of the nodes have degree one (i.e. p_1 = 0.48), while we have only one copy of the biggest node (i.e. p_{92} = 1/N=0.0005).

(c) The degree distribution is often shown on a log-log plot, in which we either plot log p_k in function of ln k, or, as we do in (c), or we use logarithmic axes.

Paths



A path of length three in a network.

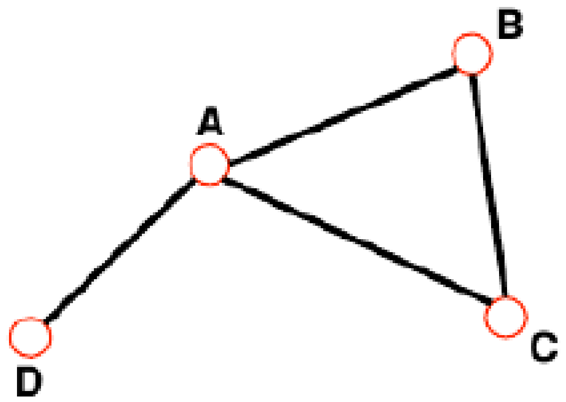
A *path* in a network is any sequence of nodes such that every consecutive pair of nodes in the sequence is connected by a link in the network. Paths can be defined for both directed and undirected networks.

A path can intersect itself, visiting again a node it has visited before, or even running along a link or set of links more than once. Paths that do not intersect themselves are called *self-avoiding paths*.

The *length* of a path in a network is the number of links traversed along the path. Links can be traversed more than once, and are counted separately each time they are traversed.

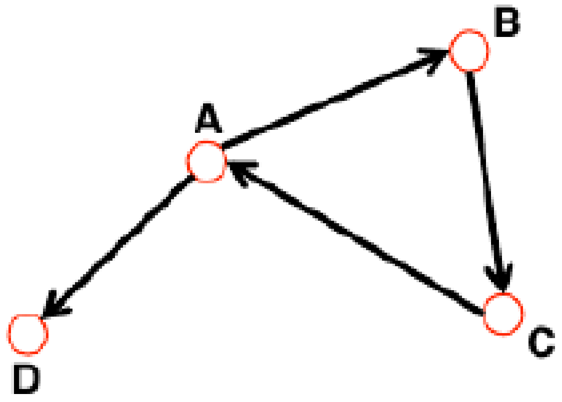
One uses the adjacency matrix to calculate the path length. If the *ij* element of the product $[A^n]$ is 1, there is a path of length *n* from *j* to *i*.

Paths



The distance (shortest path, geodesic path) between two nodes is defined as the number of edges along the shortest path that connects the two.

*If two nodes are disconnected, the distance is infinite



In directed graphs, each path has to follow the arrows of the edges. Therefore, the distance from node A to B is in general different from the distance from node B to A.

Paths

Let N_{ij} denote the number of paths between any two nodes *i* and *j:*

Length n=1: If there is a link between *i* and *j*, then $A_{ij} = 1$ and $A_{ij} = 0$ otherwise.

Length n=2: If there is a path of length two between *i* and *j*, then $A_{ik}A_{kj} = 1$ and $A_{ik}A_{kj} = 0$ otherwise.

The number of paths of length 2 is

$$N_{ij}^{(2)} = \sum_{k=1}^{N} A_{ik}A_{kj} = [A^2]_{ij}$$

Length n: In general, if there is a path of length *n* between *i* and *j*, then $A_{ik} \dots A_{lj} = 1$ and $A_{ik} \dots A_{lj} = 0$ otherwise.

The number of paths of length *n* between *i* and *j* is*

$$N_{ij}^{(n)} = [A^n]_{ij}$$

*holds for both directed and undirected networks.

Cycles or loops: closed paths

The total number (L_n) of cycles (or loops) of length *n* anywhere in a network over all possible starting points *i* is given by

$$L_n = Tr(A^n)$$

Paths

Connectivity:

Undirected → Connected: there is a path between every pair of vertices.

Directed
- Strongly connected: there is a directed path between every pair of vertices.
- Weakly connected: connected after replacing all directed edges with undirected edges.

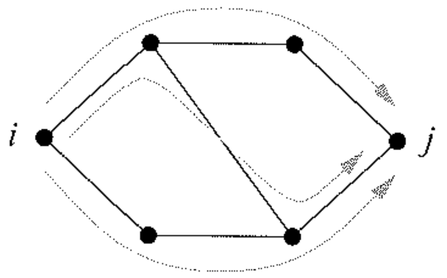
Diameter: $\boldsymbol{d_{max}}$ the maximum distance between any pair of nodes in the graph.

Average path length/distance, <d>, for a connected graph:

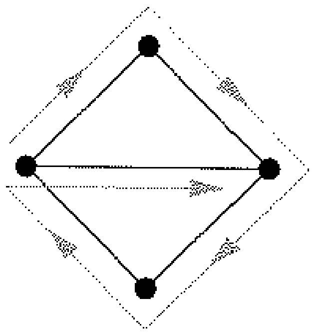
$$\langle d \rangle \equiv \frac{1}{2L_{\max}} \sum_{i,j \neq i} d_{ij}$$

where d_{ij} is the distance from node *i* to node j

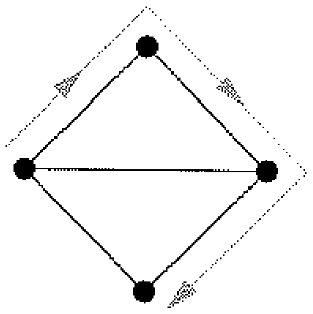
Paths



Geodesic paths between nodes i and j of length three.



An Eulerian path in a network



An Hamiltonian path in a network

More about Paths

- A *geodesic path*, also called a *shortest path*, is a path between two vertices such that no shorter path exists. Geodesic paths are necessarily self-avoiding but are not necessarily unique.

- An *Eulerian path* is a path that traverses each edge in a network exactly once.

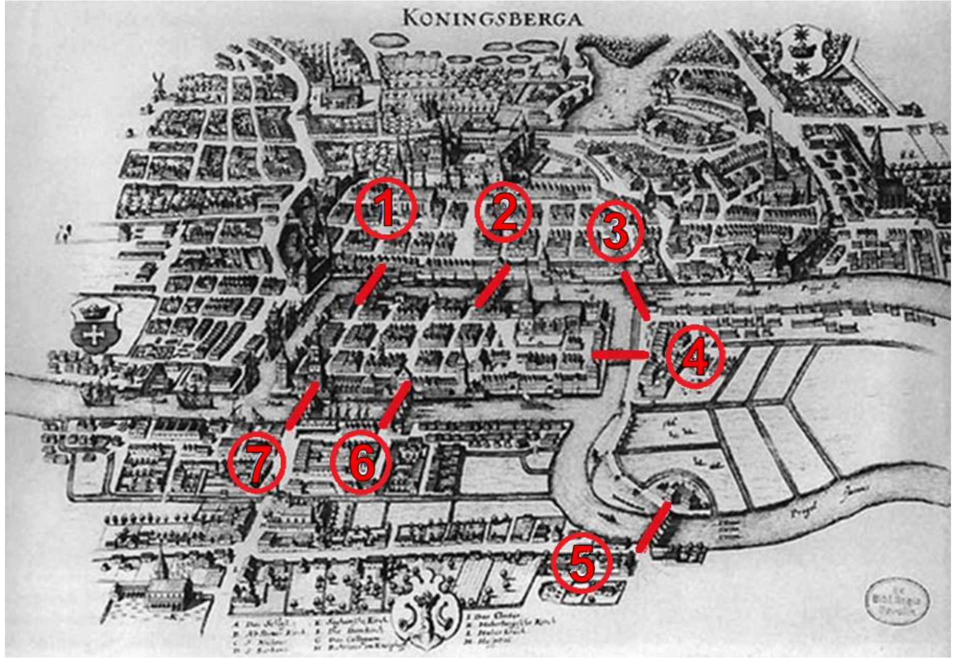
- A *Hamiltonian path* is a path that visits each vertex exactly once.

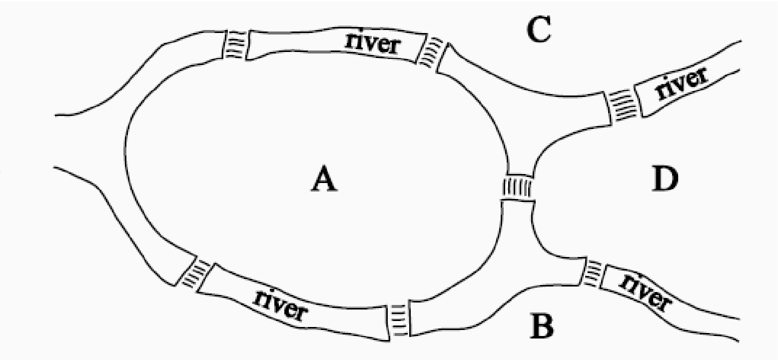
- A *Hamiltonian path* is by definition self-avoiding, but an *Eulerian path* need not be. **A *Hamiltonian path* problem was also the first problem solved by using a DNA-based computer. *(Adleman, L. M., "Molecular computation of solutions to combinatorial problems", Science 266, 1021-1024 (1994).)*

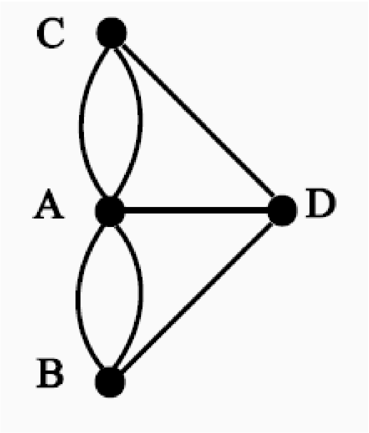
Paths

The Seven Bridges of Königsburg Revisit

Can one walk across all the seven bridges, once and only once, and then return to the starting point ?







*The bridge problem becomes a problem of finding an **Eulerian path** on the rightmost network*

Euler's Solution (1736):
(1) If a graph has more than two nodes of odd degree. There is no path.
(2) If a graph is connected and has no odd degree nodes, it has at least one path.

***A network can have an Eulerian path only if there are exactly two or zero vertices of odd degree.

Paths

Independent Paths, Connectivity and Cut Sets

- There are two species of independent path: *edge-independent* and *vertex independent*
- Two paths connecting a given pair of vertices are *edge-independent* if they share no edges.
- Two paths are *vertex-independent* (or *node-independent)* if they share no vertices other than the starting and ending vertices. (***If two paths are vertex-independent, then they are also edge-independent, but the reverse is not true*)
- Independent paths are also sometimes called *disjoint paths*.
- The number of independent paths between a pair of vertices is called the *connectivity* of the vertices. The vertices A and B in the following figure have *edge connectivity* 2 but *vertex connectivity* 1.

Edge independent paths. (a) There are two edge-independent paths from A to B in this figure, as denoted by the arrows, but there is only one vertex-independent path, because all paths must pass through the center vertex C. (b) The edge-independent paths are not unique; there are two different ways of choosing two independent paths from A to B in this case.

Cut Sets and Maximum Flow

- A ***vertex cut set*** is a set of vertices whose removal will disconnect a specified pair of vertices.
- An ***edge cut set*** is a set of edges whose removal will disconnect a specified pair of vertices.
- A ***minimum cut set*** is the smallest cut set that will disconnect a specified pair of vertices
  - Need not to be unique

- ***Menger's theorem:*** If there is no cut set of size less than *n* between a pair of vertices, then there are at least *n* independent paths between the same vertices
  - Implies that the size of minimum cut set is equal to maximum number of independent paths
    - for both edge and vertex independence

- ***Maximum Flow*** between a pair of vertices is the number of edge independent paths times the edge capacity

Cocitation and Bibliographic coupling

- Cocitation of two vertices i and j is the number of vertices that have outgoing edges to both, e.g., two papers *i* and *j* are being cited by another paper *k*

$$C_{ij} = \sum_{k=1}^{n} A_{ik}A_{jk} = \sum_{k=1}^{n} A_{ik}{A_{kj}}^{T}$$

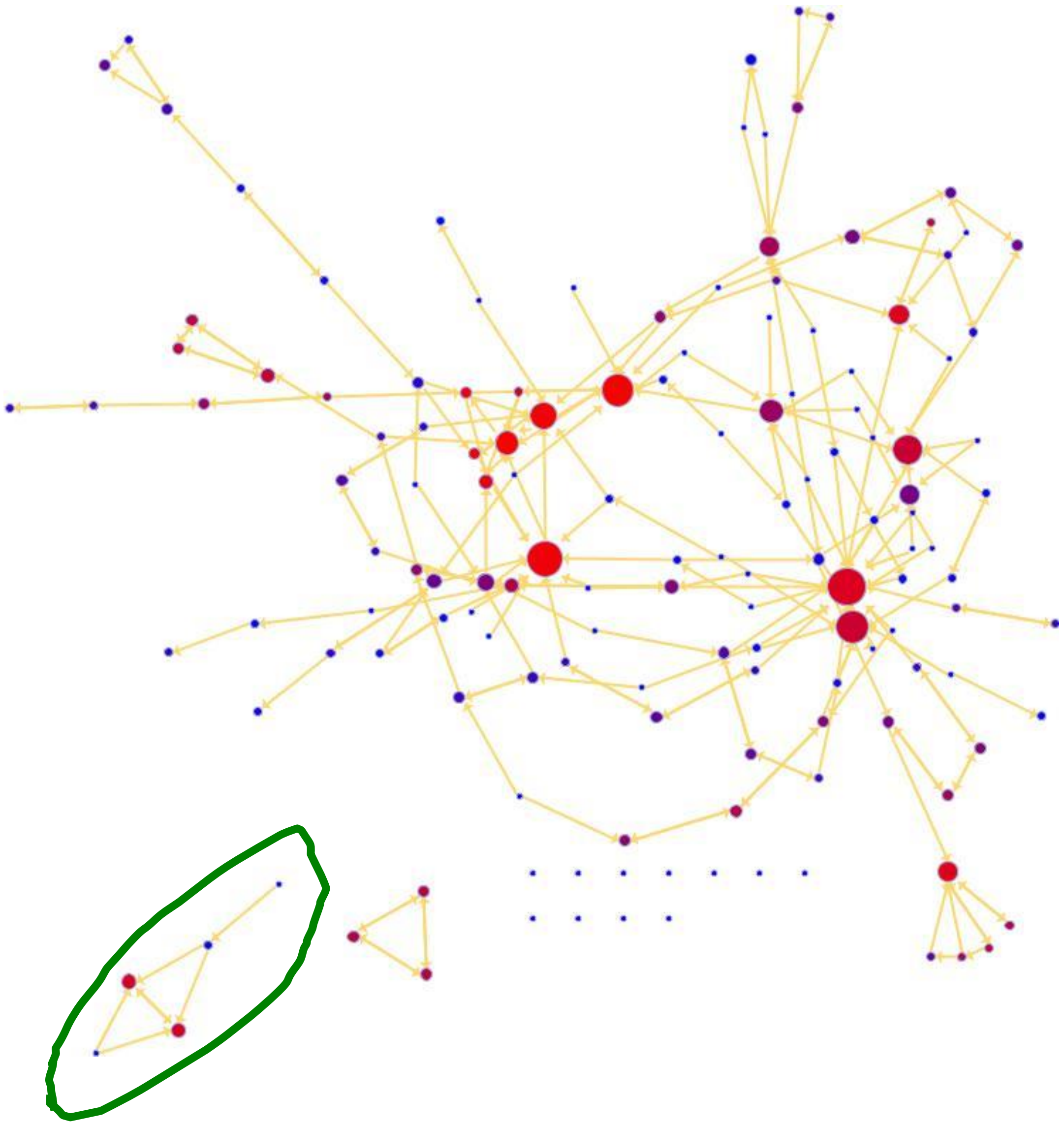
$$C = AA^{T}$$

- Bibliographic coupling is the number of vertices to which both point, e.g., papers *i* and *j* cite another paper *k*

$$B_{ij} = \sum_{k=1}^{n} A_{ki}A_{kj} = \sum_{k=1}^{n} {A_{ik}}^{T}A_{kj}$$

$$B = A^{T}A$$

Characterizing networks: Is everything connected?



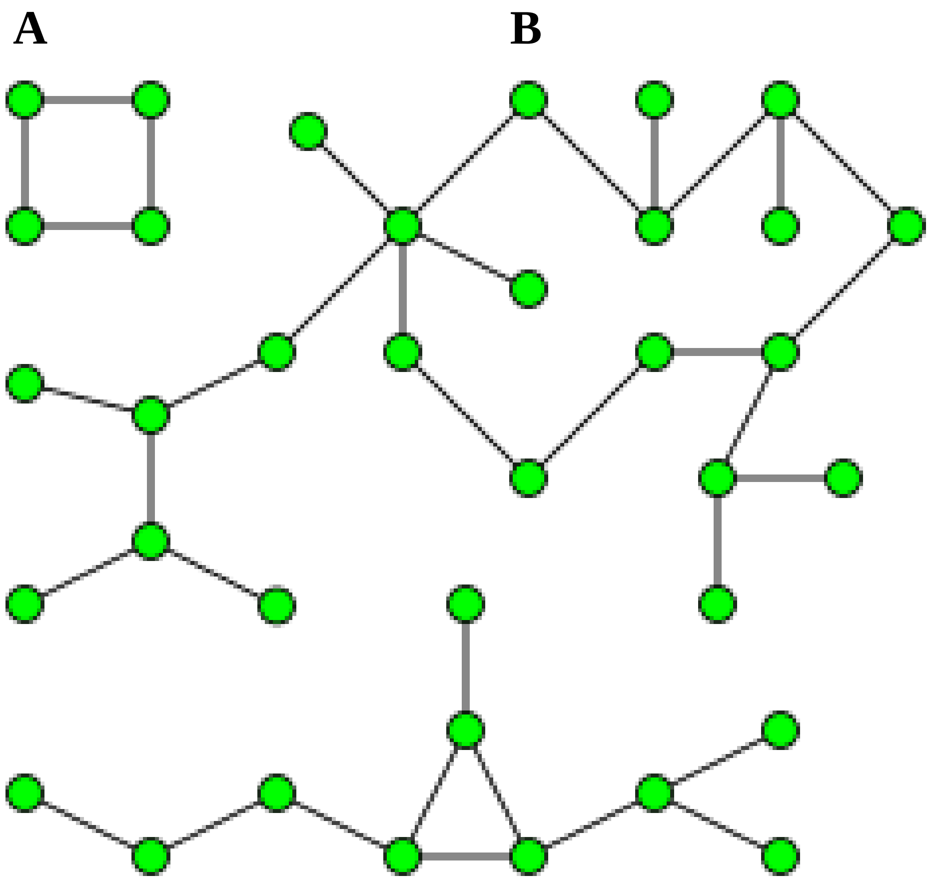
Components

It is possible for there to be no path at all between a given pair of vertices in a graph. In the graph, there is no path from vertex *A* to vertex *B*.

A graph of this kind is said to be ***disconnected***. Conversely, if there is a path from every vertex in a graph to every other the graph is ***connected.***

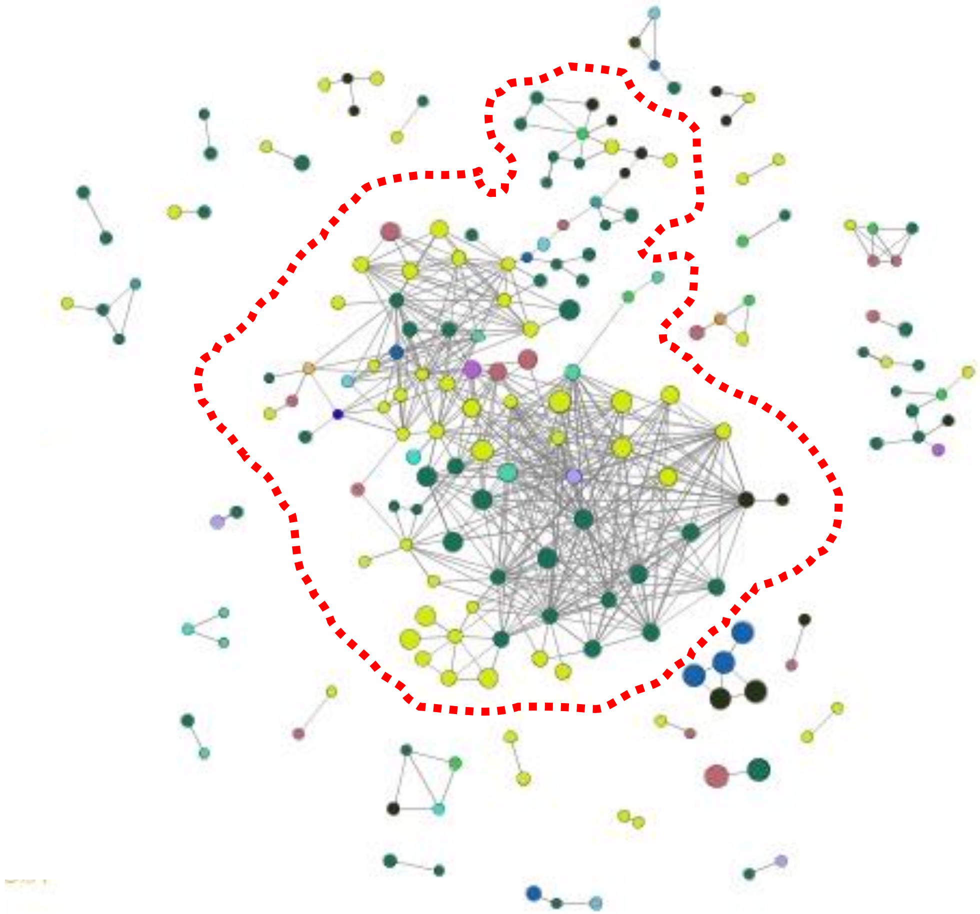
The ***subgraphs*** in a graph are called ***components.***

A ***connected component*** is a maximal connected subgraph of a graph G. Each vertex belongs to exactly one connected component, as does each edge.



Components

If the largest component encompasses a significant fraction of the graph, it is called the **giant component**

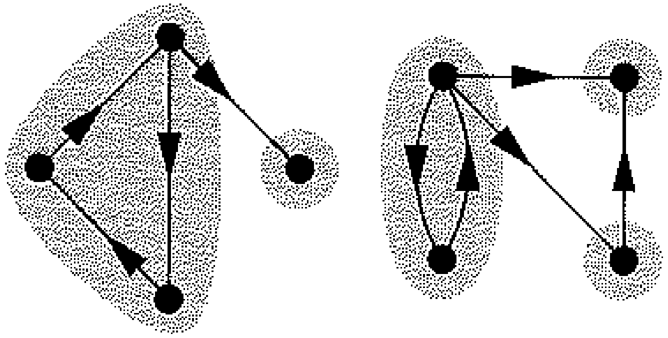


Components

The adjacency matrix **A** of a graph with subgraphs will look like:

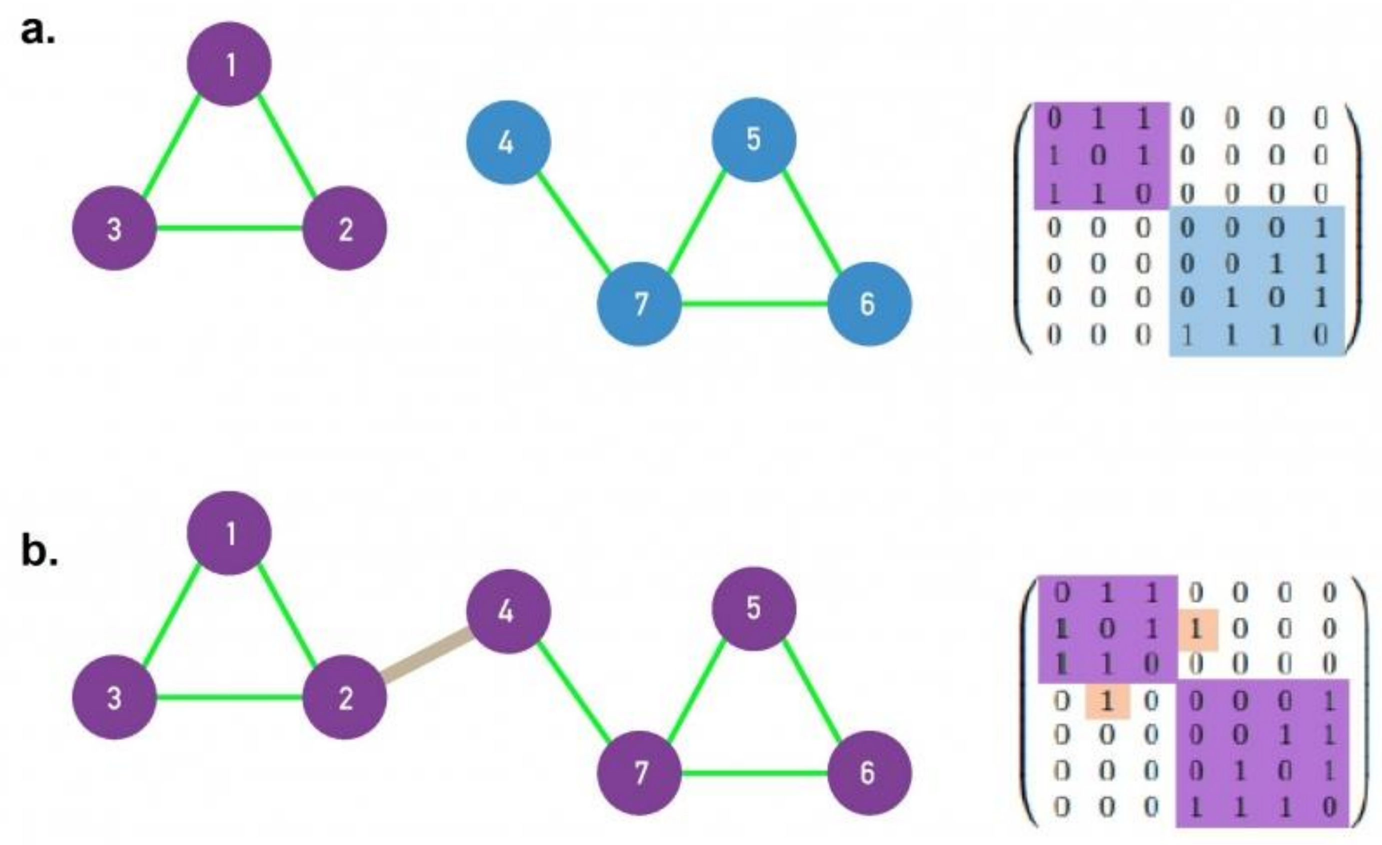
$$\mathbf{A} = \begin{bmatrix} \square & 0 & \cdots \\ 0 & \square & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

Components in Directed Networks



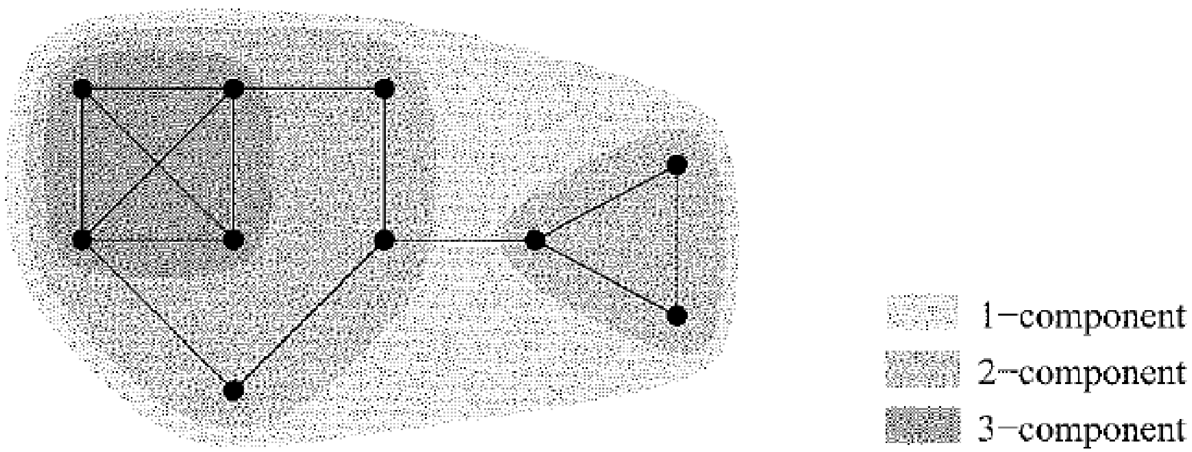
This network has two weakly connected components of four vertices each, and five strongly connected components (shaded).

Components



Components

K-components



The k-components in a small network. The shaded regions denote the *k*-components in this small network, which has a single *1*-component, two *2*-components, one *3*-component, and no *k*-components for any higher value of *k*. Note that the *k*-components are nested within one another, the *2*-components falling inside the *1*-component and the *3*-component falling inside one of the *2*-components.

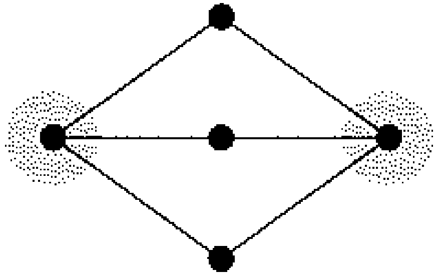
A *k*-component (sometimes also called a *k*-connected component) is a maximal subset of vertices such that each is reachable from each of the others by at least *k* vertex-independent paths.

For the common special cases $k = 2$ and $k = 3$, *k*-components are also called bi-components and tri-components respectively.

A *1*-component by this definition is just an ordinary component -- there is at least one path between every pair of vertices – and *k*-components for $k \geqslant 2$ are nested within each other. A *2*-component or bi-component, for example, is necessarily a subset of a *1*-component, since any pair of vertices that are connected by at least two paths are also connected by at least one path.

Components

K-components



The two highlighted vertices in this network form a tri-component, even though they are not directly connected to each other. The other three vertices are not in the tri-component.

The idea of a *k*-component is a natural one in network analysis, being connected with the idea of *network robustness*. For instance, in a data network such as the Internet. A pair of vertices connected by three paths cannot be disconnected by the failure of any two routers. A *k*-component with high *k*, so that it would be difficult for points on the backbone to lose connection with one another.

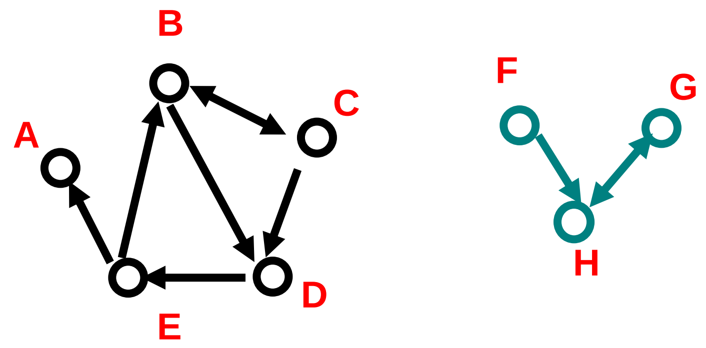
Components in directed networks

- **Weakly connected components:**
  - every node can be reached from every other node by following links in either direction

Weakly connected components

A B C D E

G H F



- **Strongly connected components**
  - Each node within the component can be reached from every other node in the component by following directed links

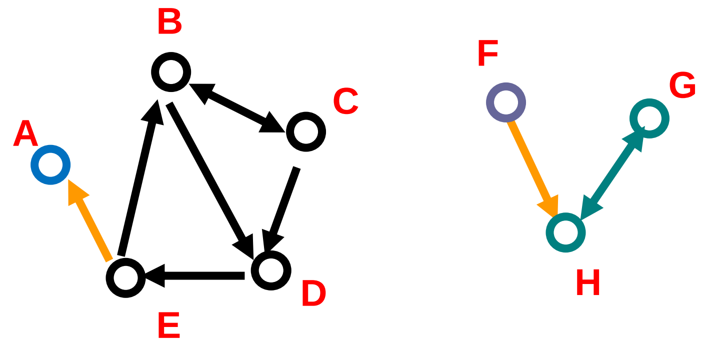
Strongly connected components

B C D E

A

G H

F



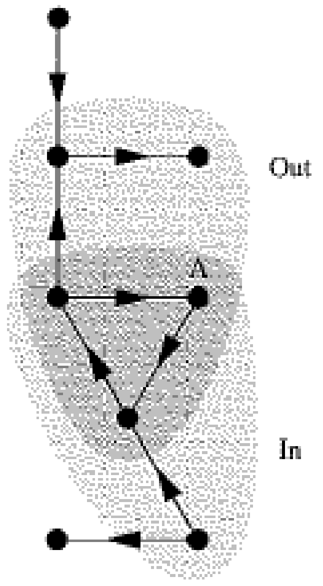
Components in directed networks

- Every strongly connected component of more than one vertex has at least one cycle

- **Out-component**: set of all vertices that are reachable via directed paths starting at a specific vertex **v**

  - Out-components of all members of a strongly connected component are identical

- **In-component**: set of all vertices from which there is a direct path to a vertex v
  - In-components of all members of a strongly connected component are identical



The in- and out-components of a vertex A in a small directed network.

Graph Laplacian

The graph Laplacian **L** is defined as

$$\mathbf{L} = \mathbf{D} - \mathbf{A}$$

A is the adjacency matrix and **D** is the diagonal matrix with the vertex degrees along its diagonal as

$$\mathbf{D} = \begin{pmatrix} k_1 & 0 & 0 & \cdots \\ 0 & k_2 & 0 & \cdots \\ 0 & 0 & k_3 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

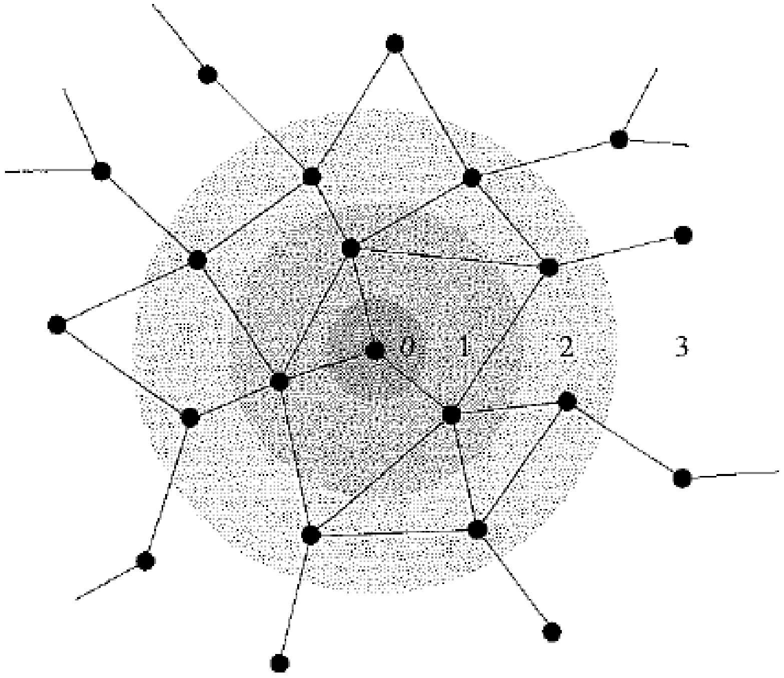
The Laplacian is a symmetric matrix, and its eigenvalues are real and non-negative. At least one of its eigenvalues is zero (**L.1** = 0). Furthermore, the number of zero eigenvalues is exactly equal to the number of components of the graph.

Corollary: The second eigenvalue of the graph Laplacian is non-zero if and only if the network is connected, i.e., consists of one single component.

The graph Laplacian appears in a variety of different places, including diffusion processes, random walks on networks, resistor networks, graph partitioning, and network connectivity.

Breadth-First Search: Shortest Path and Component

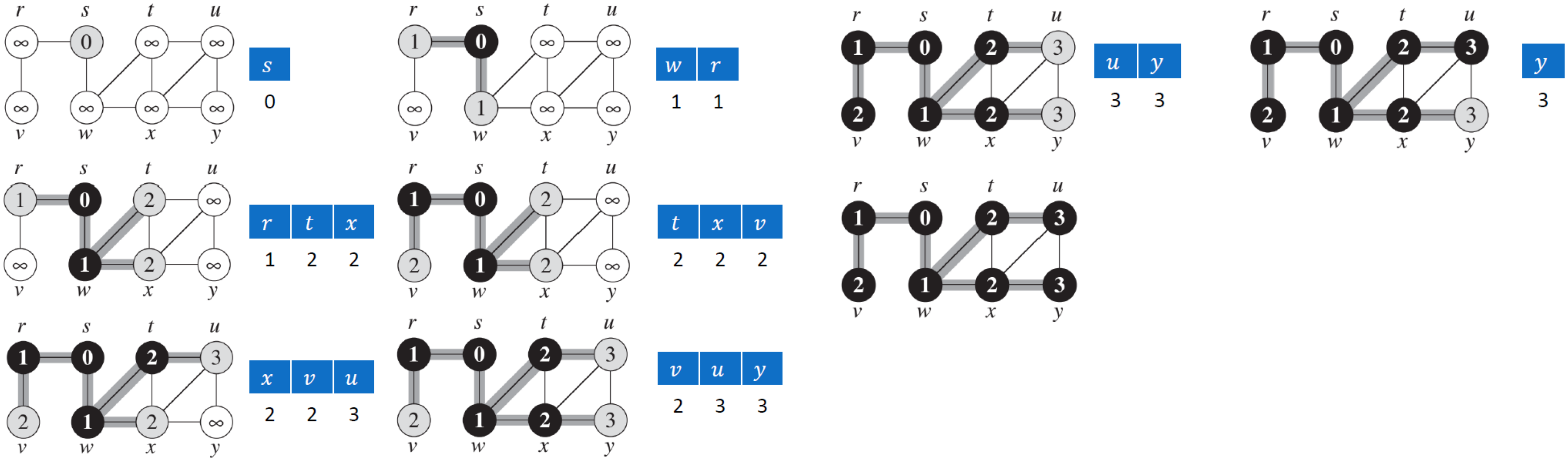
Breadth-first search algorithm finds the shortest (geodesic) distance from a single source vertex *s* to every other vertex in the same component of the network as *s*. It also finds the component to which vertex *s* belongs, and is the algorithm of choice for finding components in networks.



Breadth-first search. A breadth-first search starts at a given vertex, which by definition has distance 0 from itself, and grows outward in layers or waves. The vertices in the first wave, which are the immediate neighbors of the starting vertex, have distance *1*, the neighbors of those neighbors have distance 2, and so forth.

Breadth-First Search: Shortest Path and Component

An illustration:



When the links are weighted, one can use the Dijkstra's algorithm.

Dijkstra, E. W. “*A note on two problems in connexion with graphs*”. Numerische Mathematik 1 (1959) 269 – 271