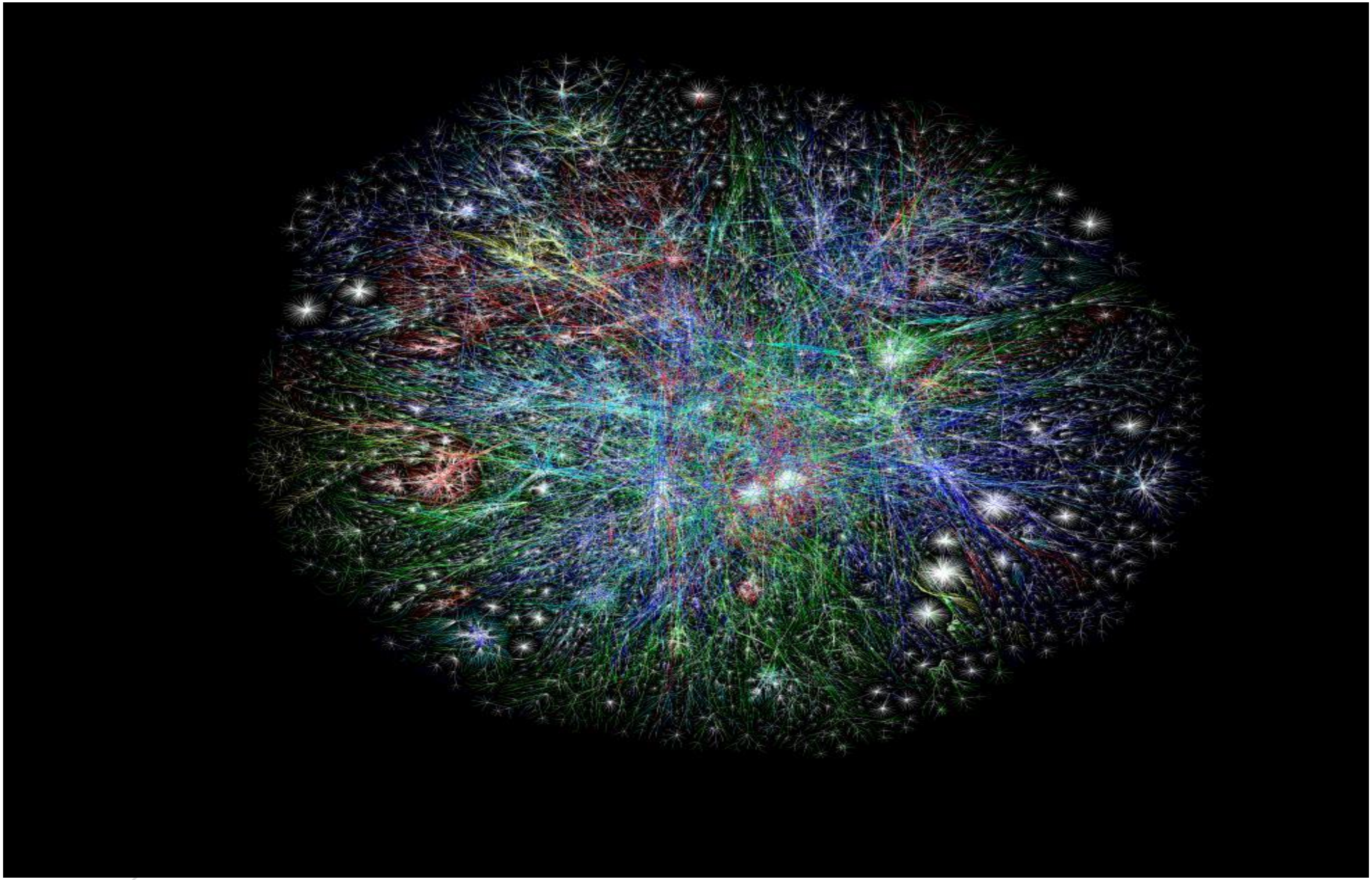
CS544 Introduction to Network Science

Introduction

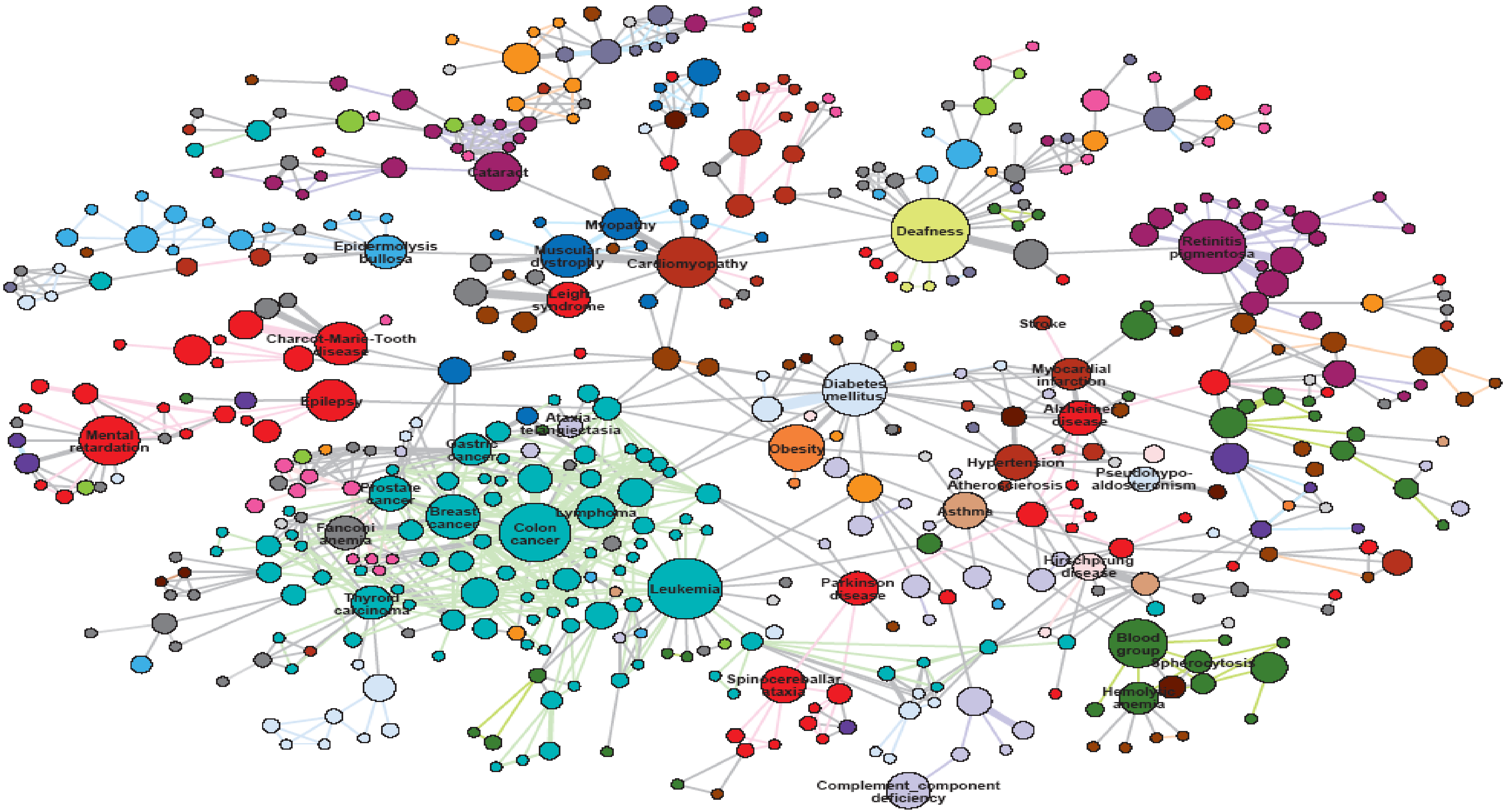
Most slides adapted from Konstantinos Pelechrinis course on Network Science & Analysis, University of Pittsburgh

The Internet

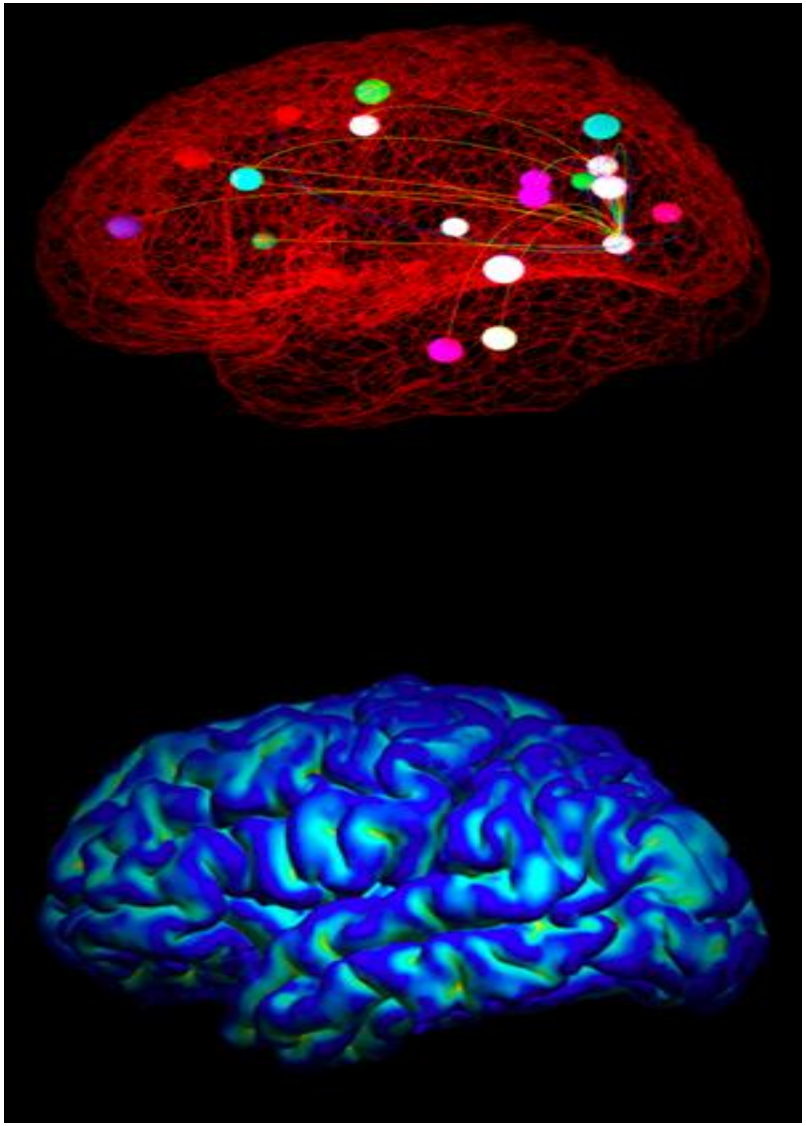


Social Networks

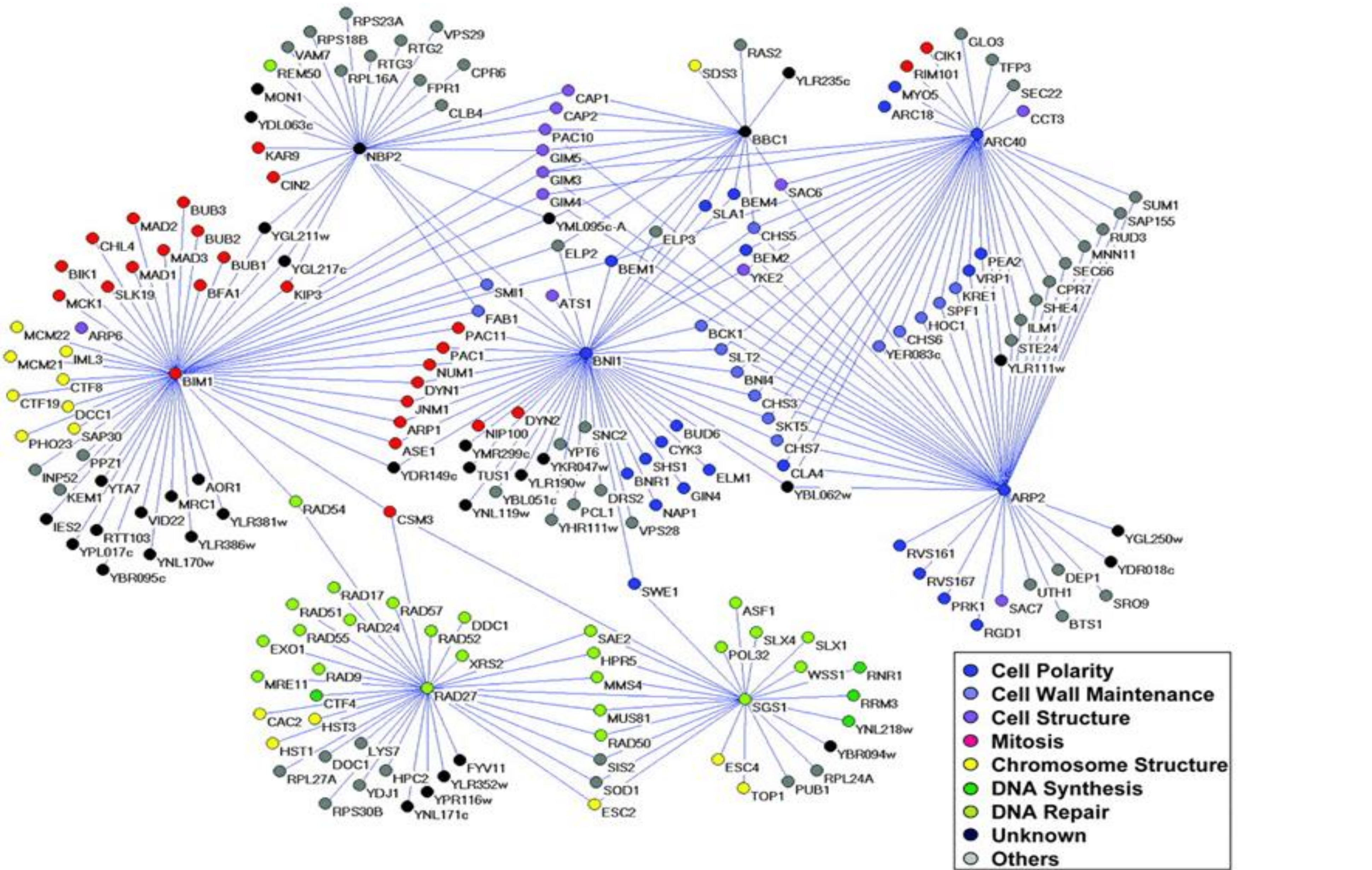
Human Disease Network

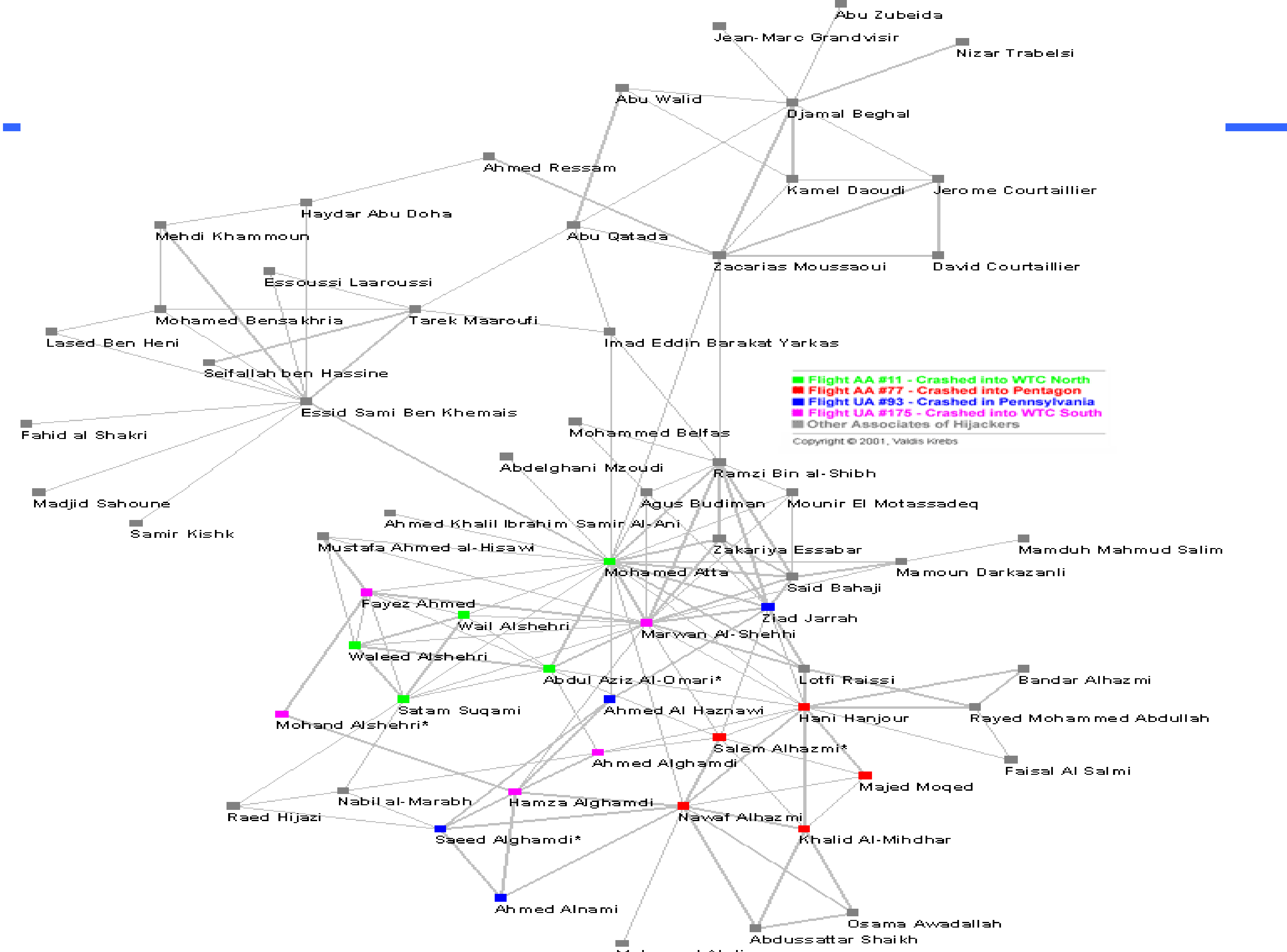


Neuro Networks

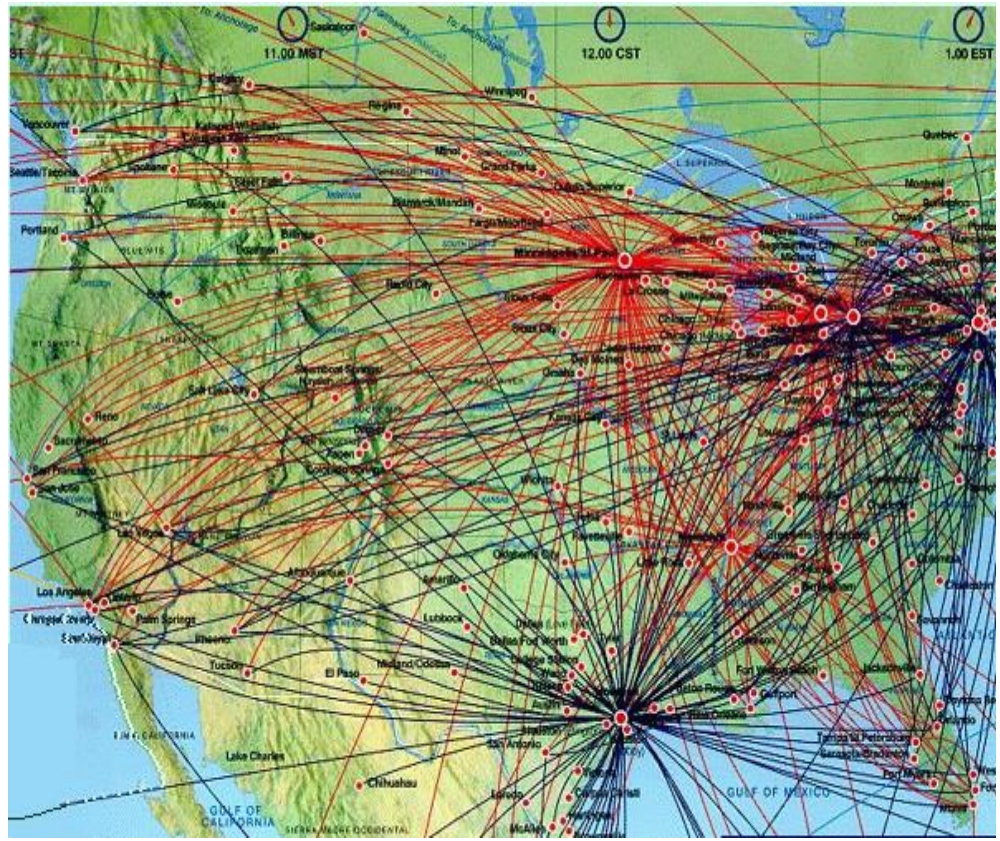


Genetic interaction network




Abu Zubeida
Jean-Marc Grandvisir
Nizar Trabelsi
Abu Walid
Djamal Beghal
Ahmed Ressam
Kamel Daoudi
Jerome Courtaillier
Haydar Abu Doha
Mehdi Khammoun
Abu Qatada
Zacarias Moussaoui
David Courtaillier
Essoussi Laaroussi
Mohamed Bensakhria
Tarek Maaroufi
Lased Ben Heni
Imad Eddin Barakat Yarkas
Seifallah ben Hassine
Flight AA #11 - Crashed into WTC North
Flight AA #77 - Crashed into Pentagon
Flight UA #93 - Crashed in Pennsylvania
Flight UA #175 - Crashed into WTC South
Other Associates of Hijackers
Copyright © 2001, Valdis Krebs
Essid Sami Ben Khemais
Fahid al Shakri
Mohammed Belfas
Abdelghani Mzoudi
Ramzi Bin al-Shibh
Madjid Sahoune
Agus Budiman
Mounir El Motassadeq
Samir Kishk
Ahmed Khalil Ibrahim Samir Al-Ani
Mustafa Ahmed al-Hisawi
Zakariya Essabar
Mamduh Mahmud Salim
Mohamed Atta
Mamoun Darkazanli
Said Bahaji
Fayez Ahmed
Ziad Jarrah
Wail Alshehri
Marwan Al-Shehhi
Waleed Alshehri
Abdul Aziz Al-Omari*
Lotfi Raissi
Bandar Alhazmi
Satam Suqami
Ahmed Al Haznawi
Hani Hanjour
Rayed Mohammed Abdullah
Mohand Alshehri*
Salem Alhazmi*
Ahmed Alghamdi
Faisal Al Salmi
Majed Moqed
Nabil al-Marabh
Hamza Alghamdi
Raed Hijazi
Nawaf Alhazmi
Saeed Alghamdi*
Khalid Al-Mihdhar
Ahmed Alnami
Osama Awadallah
Abdussattar Shaikh

Road and Airlines Network



Other Data as Graphs

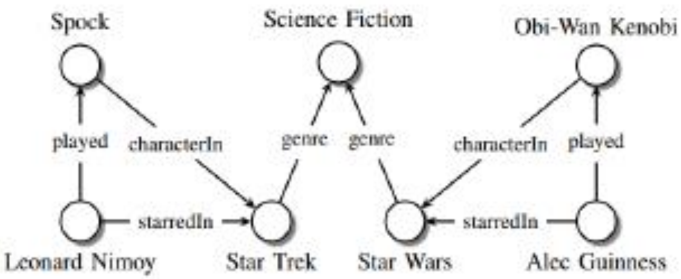


Image credit: Maximilian Nickel et al

Knowledge Graphs

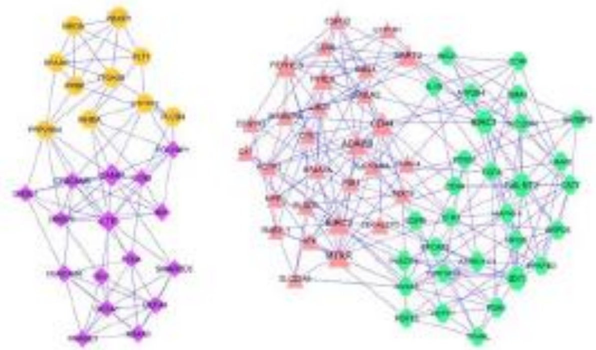


Image credit: ese.wustl.edu

Regulatory Networks

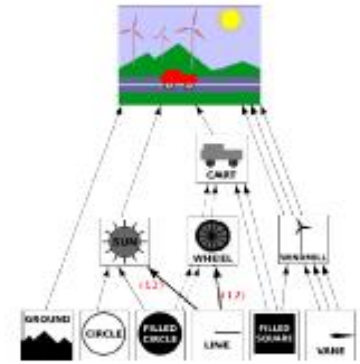


Image credit: math.hws.edu

Scene Graphs

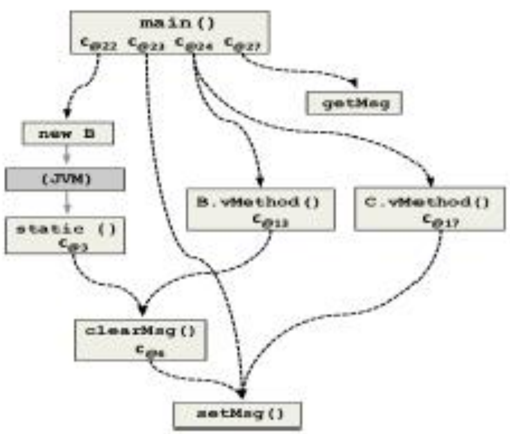


Image credit: ResearchGate

Code Graphs

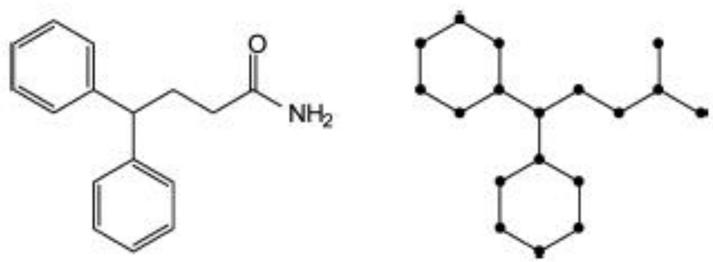


Image credit: MDPI

Molecules

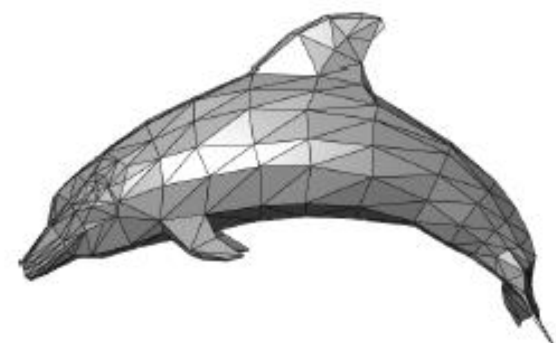


Image credit: Wikipedia

3D Shapes

Source: CS224W Stanford

Why Networks?

- Nothing happens in isolation: “everything is connected, caused by, and interacting with a huge number of other pieces of a complex universal puzzle” (AL Barabasi, “Linked”)
- The power of the network is in the links
- However, most people don’t see the links till they are exposed to them

Why Network Science?

- **Study of the structural evolution of large networks**
- **It studies networks holistically**
- **Modeling phenomena around us using networks can be done in multiple ways and at different levels/depths**
  - Macroscopic World
    - ✓ Contains the things that we can see with our eyes
      - In language world, grammar rules and vocabularies
  - Microscopic World
    - ✓ Contains the building blocks of matters like atoms, molecules etc.
      - In language world, utterances (syllables, phonemes, graphemes)
  - Mesoscopic World
    - ✓ An intermediate between the macroscopic and microscopic level
      - In language world, linguistic entities like letters, words and phrases

Why Network Science?

- Networks have shifted from simple and small to complex and extremely large (data explosion)
- Network modeling transitioned from static graphs to dynamic graphs
- Network nodes (objects) to be studied transitioned from one type to a variety of data types (multimodal data)

Books

- **Main text book**
  - Will Hamilton, “Graph Representation Learning”
  - M.J.E. Newman, “Networks: An Introduction”, Oxford University Press (2010)

- **Other possible references**
  - D. Easley and J. Kleinberg, “Networks, Crowds and Markets: Reasoning About a Highly Interconnected World”, Cambridge University Press (2010)
  - T.G. Lewis, “Network Science: Theory and Applications”, Wiley (2009)
  - D.J. Watts, “Six Degrees: The Science of a Connected Age”, W.W. Norton & Company (2003)
  - Research papers (pointers will be provided)

Assignments and codes

- **Codes will be in Python**
- **Libraries**
  - NetworkX
  - PyTorch Geometric (PyG)
  - DeepSNAP
  - GraphGym
  - SNAP.py

Mathematics of Networks

The representation of networks

- **The network consists of *entities* connected with each other**

- **The structure of these connections are represented through <u>graphs</u>**

- **A graph is represented by two sets**
  - A vertex set V of the entities participating in the network. In the rest of the slides typically, n will be the number of vertices
    - ✓ Also called node or actor set
  - An edge set E of the connections between vertices. In the rest of the slides typically, m will be the number of edges
    - ✓ Also called link or tie set

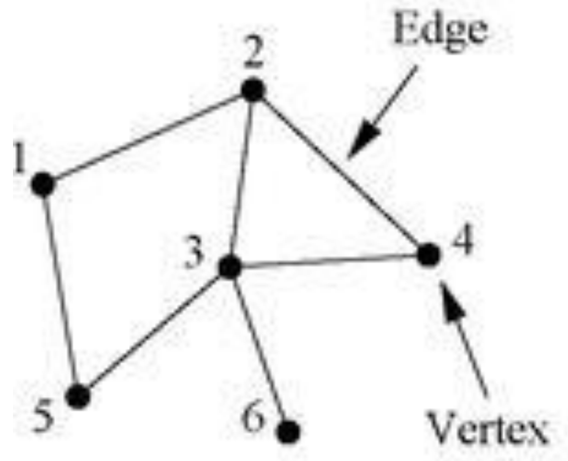
Graph terminology

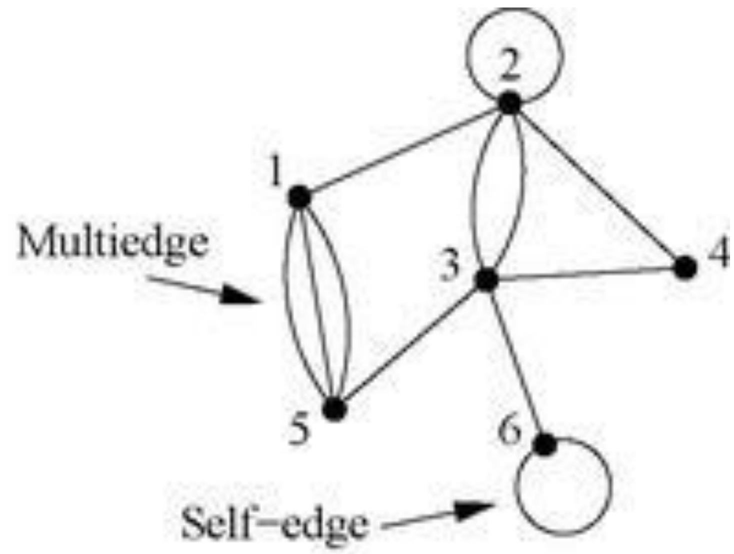
- **A graph whose vertices are connected by at most one link is called <u>simple network</u> or <u>simple graph</u>**
  - Most of the graphs we will examine will be simple

- **When two nodes connect with more than one edge, we refer to all those edges collectively as <u>multiedge</u>**
  - The corresponding graph is called <u>multigraph</u>

- **Depending on the type of connection a node might be connected to itself**
  - <u>Self-edges</u> or <u>self-loops</u>

Example





The adjacency matrix

- **If we label the nodes with IDs 1, 2, … n we can denote each edge as a pair (i,j)**
  - This is an edge list specification
    - ✓ Good for storing and processing networks in computers, but not for mathematical development
- **The adjacency matrix A of a simple graph is a matrix with elements A_{ij} such that:**

$$A_{ij} = \begin{cases} 1, & \text{if there is an edge between vertices i and j} \\ 0, & \textit{otherwise} \end{cases}$$

Example

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

Observations

- For the (typical) case of no self-loops, the diagonal of the matrix is all 0
- For undirected networks, since the existence of edge (i,j) assumes the existence of the edge (j,i), matrix A is symmetric
- If node i has a self edge then we set A_{ii}=2 (why?)
- Multiedges are represented by setting the corresponding entry equal to the number of distinct edges

Weighted networks

- **Some relations are not simple on/off (1/0) relations**
- **In a weighted network links can have weights**
  - The corresponding adjacency matrix entry is equal to the weight
  - Weights can represent the frequency of contacts between the actors, the capacity of a channel connecting two routers etc.
- **When weights are integer it might be convenient to think of the weight as multiedges**

Directed networks

- **In some phenomena the *direction* of the underlying relation between two nodes matters → relations are not reciprocal**
  - E.g., twitter connections, world wide web links, paper citations etc.

- **These relations are captured through <u>directed networks/graphs</u>**

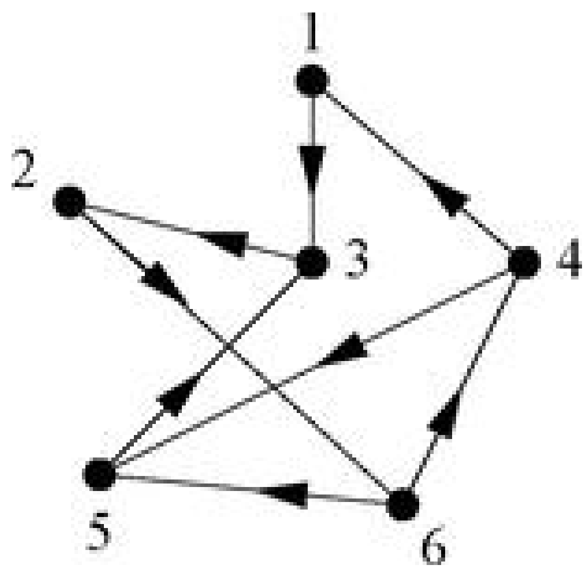
- **The adjacency matrix of a directed graph (or a digraph) is given by:**

A is in general not symmetric

$$A_{ij} = \begin{cases} 1, & \text{if there is an edge from j to i} \\ 0, & \textit{otherwise} \end{cases}$$

Self edges are represented by setting $A_{ii}=1$

Example

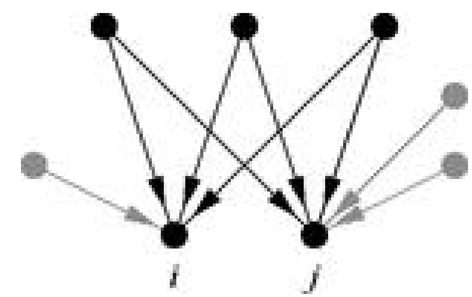


$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Co-citation coupling

- **The co-citation of two edges i and j in a directed network is the number of nodes that have outgoing edges pointing to both nodes i and j**
    - So in order for node k to contribute to the co-citation of i and j, the following needs to be true: $A_{ik}A_{jk}=1$
    - Hence the co-citation of nodes i and j is:

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



- **The nxn adjacency matrix of the corresponding co-citation network is:**

$$C = AA^{T}$$

Co-citation coupling

- **The non-diagonal elements can be larger than 1**
  - Either we consider it as a weighted undirected graph or we map every non-diagonal entry greater than 1 to 1

- **The diagonal elements of C can be non zero**
  - Since A_{ik}=0 or A_{ik}=1 (assuming a simple graph – no multiedges)

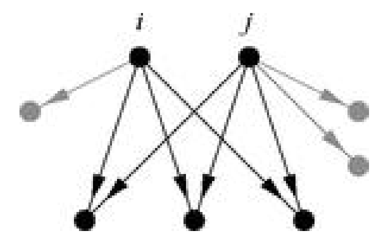
$$C_{ii} = \sum_{k=1}^{n} A_{ik}^2 = \sum_{k=1}^{n} A_{ik}$$

  - C_{ii} provides the number of nodes that point to node i
    - ✓ In constructing the co-citation network we set by definition all the diagonal elements to zero!

Bibliographic coupling

- **The bibliographic coupling of two edges i and j in a directed network is the number of nodes that they both point to**
  - So in order for node k to contribute to the bibliographic coupling of i and j, the following needs to be true: $A_{ki}A_{kj}=1$
  - Hence the bibliographic coupling of nodes i and j is:

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



- **The nxn adjacency matrix of the corresponding bibliographic coupling network is:**

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

Bibliographic coupling

- **The diagonal elements of B can be non zero**
  - Since A_{ki}=0 or A_{ki}=1 (assuming a simple graph – no multiedges)

$$B_{ii} = \sum_{k=1}^{n} A_{ki}^2 = \sum_{k=1}^{n} A_{ki}$$

  - B_{ii} provides the number of nodes that node i points to
    - ✓ Again when considering the bibliographic coupling network, we set by definition all the diagonal elements to zero!

- **The off diagonal elements of B can again be greater than 0 and hence, we can define a weighted undirected network**
  - We can also map any positive off diagonal element to 1

Notes on coupling

- **While both coupling methods are similar they can give completely different structures**
  - Co-citation is heavily based on incoming edges, while bibliographic coupling on outgoing edges

- **They can be used for vertex similarity**
  - E.g., the highest the co-citation of two papers in a paper citation network, the more possible is that these papers deal with similar topics

- **While these coupling methods help us convert an directed network to undirected, we lose some structural information during this transformation**

Problem 1

- **Consider certain papers and the references that they cite**
  - 1 → {2,4,5}
  - 2 → {3,4,5}
  - 3 → {4}
  - 4 → {}
- **Which of these papers do you expect to be likely of similar topic?**

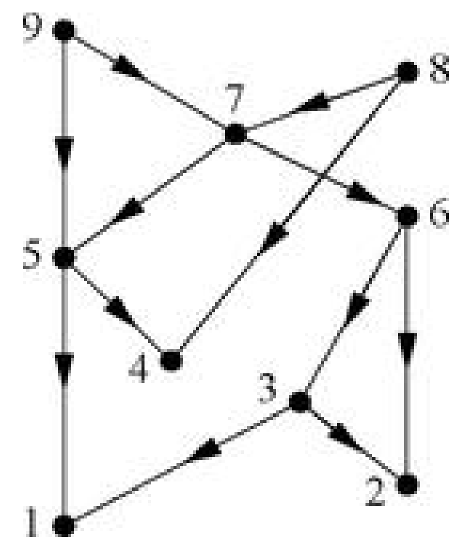
Problem 2

- **Consider a product website that maintains a purchase profile of the users over a time period T. A snap shot of the same for 4 users are as follows:**
  - A → {1,2,3}
  - B → {1,3}
  - C → {1,2, 3, 4}
  - D →{ 3,4}

- **If A purchases an unknown product (say 5), can you say which other user will most likely purchase the same product?**

Acyclic directed networks

- **A <u>cycle</u> in a directed network is a closed loop of edges with the arrows on each of the edges pointing the same way around the loop**
  - Networks without cycles are called *acyclic*, while those with cycles are called *cyclic*
    - ✓ Self edges also count as cycles
- **An acyclic network can be drawn with all edges pointing downward (not necessarily unique drawing)**



Determining if a network is acyclic

- **A simple algorithm for determining whether a network is acyclic is:**
  - Find a vertex with no outgoing edges (step 1)
  - If no such vertex exists, the network is cyclic. Otherwise, if such a vertex does exist, remove it and all its incoming edges from the network (step 2)
  - If all vertices have been removed, the network is acyclic. Otherwise go back to step 1

Adjacency matrix of an acyclic network

- **If we construct an ordering of the vertices of an acyclic graph as per the previous algorithm, there can be an edge from vertex i to vertex j only if j > i**
  - Adjacency matrix is triangular (only the elements above the diagonal have non zero entries)
    - ✓ More precisely, since the acyclic graph does not have self loops, the diagonal elements are zero → strictly triangular matrix

- <u>**All of the eigenvalues of an adjacency matrix are zero if and only if the network is acyclic**</u>

Bipartite networks

- **In bipartite networks (aka *two-mode networks*) there are two kinds of vertices**
  - One type represents that original vertices (e.g., people) and the other represents the groups – or affiliations or foci – that the first type of vertices belong to (e.g., company)

- **The edges of a bipartite network run only between nodes of different type**

- **A bipartite network is defined by the <u>incidence matrix B</u>. If there are *n* number of actors and *g* number of groups:**

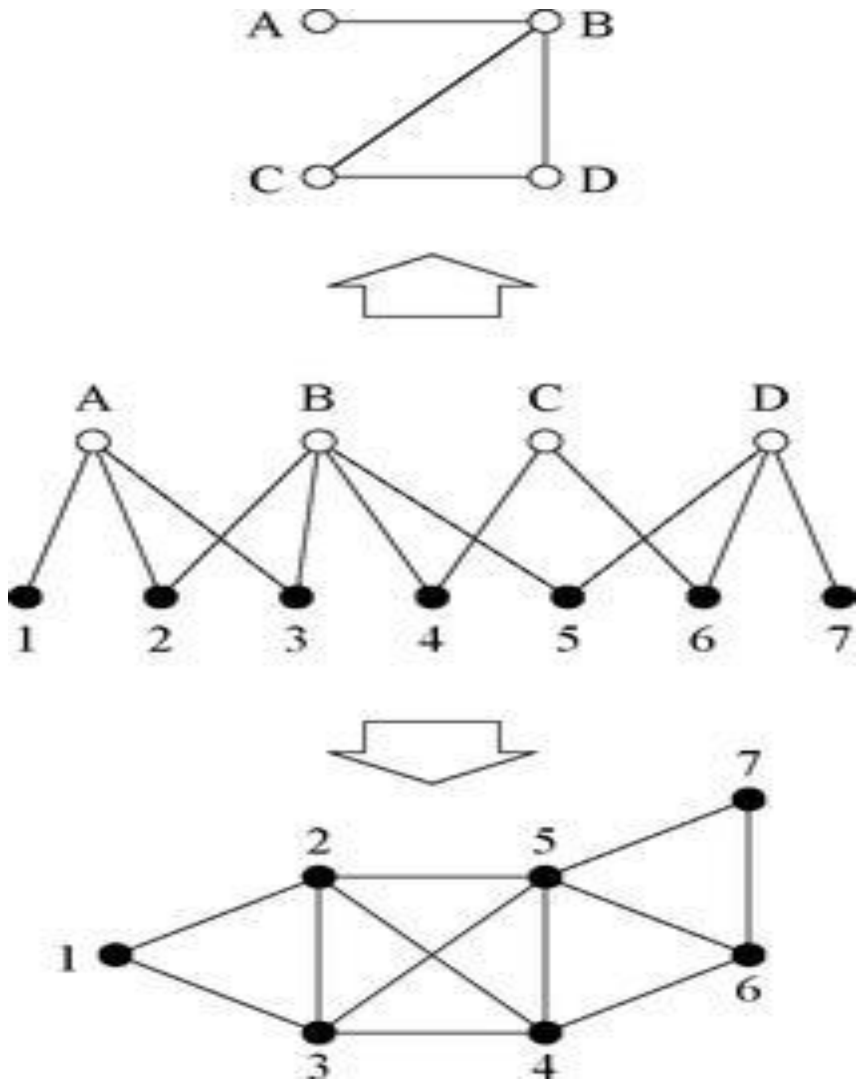
$$B_{ij} = \begin{cases} 1, & \text{if vertex j belongs to group i} \\ 0, & \textit{otherwise} \end{cases}$$

One-mode projections

- **We can use the bipartite network to infer connections between nodes of the same type**
  - Projection on the actors → n-vertex network, where nodes are the actors of the original bipartite graph and an edge between two actors exists if they participate to at least one common group
  - Projection on the groups → g-vertex network, where nodes are the groups of the original bipartite graph and an edge between two groups exists if they include at least one common actor

- **One mode projections discard a lot of the information present in the structure of the original bipartite graph**

Projections



Mathematics of actors one-mode projection

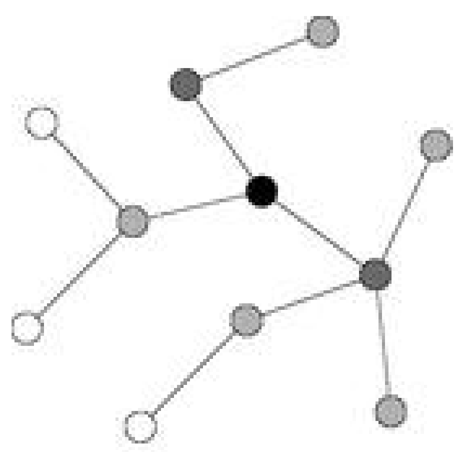
- **Two actors i and j belong to group k iff $B_{ki}B_{kj}=1$**
  - The total number of groups P_{ij} that i and j belong is given by:

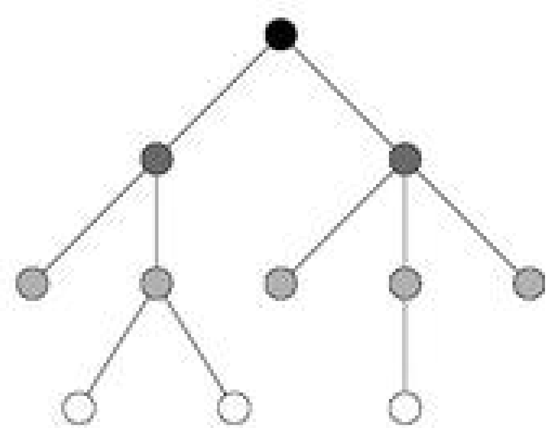
$$P_{ij} = \sum_{k=1}^{g} B_{ki}B_{kj} = \sum_{k=1}^{g} B_{ik}^{T}B_{kj}$$

- **The nxn matrix $P=B^TB$ is similar to the adjacency matrix for the weighted one-mode projection onto the n vertices (actors)**
  - Why only *similar* and not the same?

Trees

- **Connected, undirected network with no closed loops**
  - They are usually drawn in a *rooted* manner
    - ✓ A *root node* at the top and a branching structure going down
    - ✓ Vertices at the bottom of the tree, connected only to one other vertex are called *leaves*





Trees

- **Since there are no closed loops there is exactly one path from every vertex to any other**
  - This property is important because it makes some calculations particularly simple

- **A tree with n vertices has exactly n-1 edges**
  - Reverse is also true!

Degree

- **The <u>degree</u> k_i of a vertex i in a graph is the number of edges connected to it**
  - For undirected graphs we have:

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

  - And the number of edges of a graph is given by:

$$m = \frac{1}{2}\sum_{i=1}^{n} k_i = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij}$$

- **Mean degree c of a vertex in an undirected graph is:**

$$c = \frac{1}{n}\sum_{i=1}^{n} k_i = \frac{2m}{n}$$

Connectance

- **The maximum number of possible edges in a simple graph is**

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

- **<u>Connectance</u> or <u>density ρ</u> of a graph is the fraction of these edges that are actually present:**

$$\rho = \frac{m}{\binom{n}{2}} = \frac{2m}{n(n-1)} = \frac{c}{n-1}$$

  - A network for which the density ρ tends to a (positive) constant as n→∞ is called <u>dense</u>
  - A network for which the density ρ tends to zero as n→∞ is called <u>sparse</u>
    - ✓ This basically means that in a sparse network c tends to a constant when n increases arbitrarily

Degrees in directed networks

- **In a directed network each vertex is associated with two degrees**
  - In-degree is the number of incoming edges to a vertex
  - Out-degree is the number of outgoing edges from a vertex

$$k_i^{in} = \sum_{j=1}^{n} A_{ij} \qquad k_i^{out} = \sum_{j=1}^{n} A_{ji}$$

- **The total number of edges in a directed network is:**

$$m = \sum_{i=1}^{n} k_i^{in} = \sum_{i=1}^{n} k_i^{out} = \sum_{ij} A_{ij}$$

  - Thus, mean in (c_{in}) and out (c_{out}) degrees are equal

$$c_{in} = \frac{1}{n}\sum_{i=1}^{n} k_i^{in} = \frac{1}{n}\sum_{i=1}^{n} k_i^{out} = c_{out} = c = \frac{m}{n}$$

Walks

- **A sequence of vertices such that every consecutive pair of vertices in the sequence is connected by an edge in the network**
  - For directed graphs the edges traversed from the path needs to be traversed in the correct direction
  - Walks that do not intersect with themselves are called self-avoiding paths

- **Length of a walk is the number of edges traversed along the walk**
  - When a walk traverses the same edge e two times, e is counted twice

Walks in undirected networks

- **The product $A_{ik}A_{kj}$ is 1 iff there is a path between i and j through k (a path of length 2)**
  - Hence, if we want to find how many length 2 paths between i and j exist:

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

  - This can easily be generalized to: $N_{ij}^{(r)} = [A^r]_{ij}$

- **$[A^r]_{ii}$ gives the number of length r paths that originate and end at node i (cycles)**
  - Hence if we want to find the number L_r of length r cycles in a graph

$$L_r = \sum_{i=1}^{n} [A^r]_{ii} = TrA^r$$

Walks in undirected networks

- **A is symmetric → has n real, non-negative eigenvalues**
  - A = UKU^T
    - ✓U orthogonal matrix of eigenvectors
    - ✓K diagonal matrix with eigenvalues

$$A^r = (UKU^T)^r = UK^rU^T$$

$$L_r = TrA^r = Tr(UK^rU^T) = Tr(U^TUK^r) = TrK^r = \sum_i \kappa_i^r$$

κ_i is the i-th eigenvalue of matrix A

Paths in directed networks

- **The adjacency matrix of a directed network is in general non symmetric and hence not diagonalizable**

- **We will use the <u>Schur decomposition</u>**
  - $A=QTQ^T$
    - ✓ Q is orthogonal
    - ✓ T is upper triangular
      - Its diagonal elements are its eigenvalues
        - *They are equal with those of A (why?)

$$L_r = Tr A^r = Tr(QT^rQ^T) = Tr(Q^TQT^r) = Tr T^r = \sum_i k_i^r$$

Geodesic paths

- **A geodesic path (shortest path) is a path between two vertices such that no shorter path exists**
  - The length of this path is called geodesic (or shortest) distance
  - If two nodes are not connected with any path their geodesic distance is infinite

- **By definition shortest paths are self avoiding (why?)**

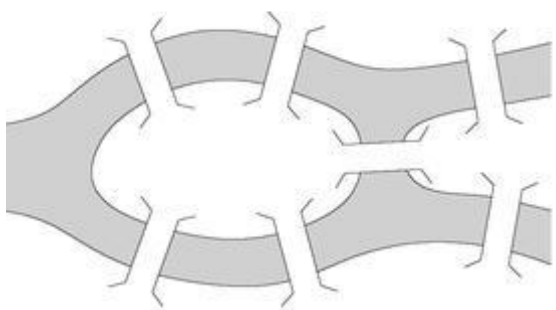
- **Diameter of a network is the length of the longest geodesic path between any pair of vertices in the network for which a path actually exists**
  - Other definitions are extensively used in the literature as well, such as, the average value of all geodesic paths in the network etc.

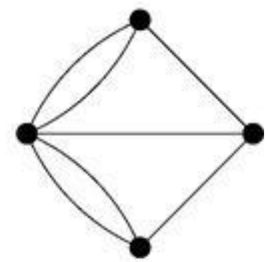
Eulerian and Hamiltonian paths

- 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
  - Self avoiding

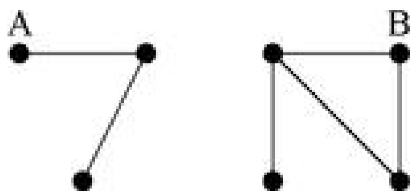
Konisberg Bridge Problem





Components

- **A network for which there exists pairs of vertices that there is no path between them is called <u>disconnected</u>**
  - If there exists a path between any possible pair of vertices in a network the latter is called <u>connected</u>



- **<u>Component</u> is a *maximal* subset of vertices of a network such that there exists at least one path from every vertex of the subgroup to any other**

How can the adjacency matrix of a network with more than one component be written?

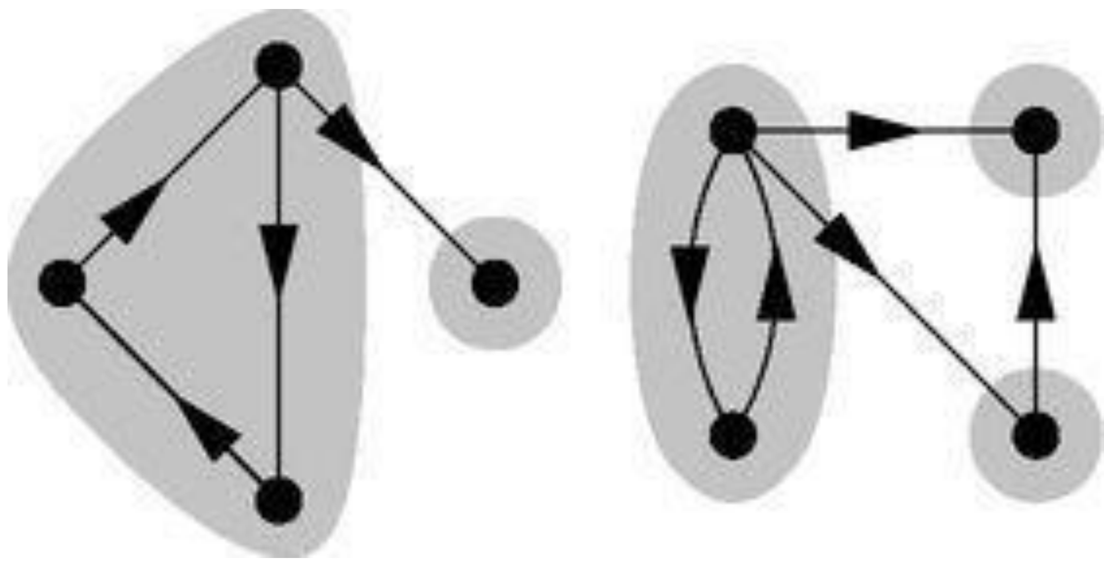
Components in directed networks

- **There are many different types of connected components that we can define for a directed network**

- **Assuming no direction on the edges, we can identify connected components as in an undirected graph**
  - <u>Weakly connected components</u>

- **Imposing a constrained on the direction of the edge we get the <u>strongly connected components</u>**
  - Maximal subsets of vertices such that there is a directed path in both directions between every pair in the subset
    - ✓ Every vertex belonging to a strongly connected component with more than one vertex must belong to at least one cycle (why?)

Visually



Out-components

- **The set of vertices that are reachable via directed paths from node A (including A), form its <u>out-component</u>**
  - Out-component is a property of both the network and the starting vertex
    - ✓ Hence, a vertex can belong to more than one out-components

- **All members of the strongly connected component that A belongs to, are members of its out-component**

- **All vertices that are reachable from A are also reachable from any other member of its strongly connected component**
  - Out-components really "belong" not to individual vertices but to strongly connected components

In-components

- The in-component of a specific vertex A is the set of all vertices (including A) from which there is a directed path to A

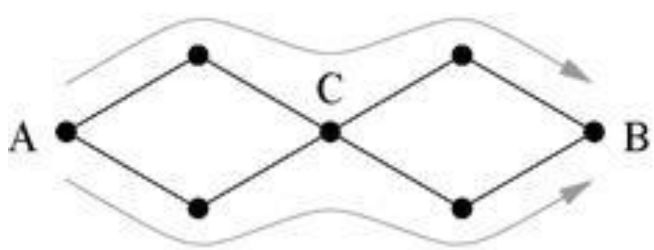
- All the members of a strongly connected component have the same in-component

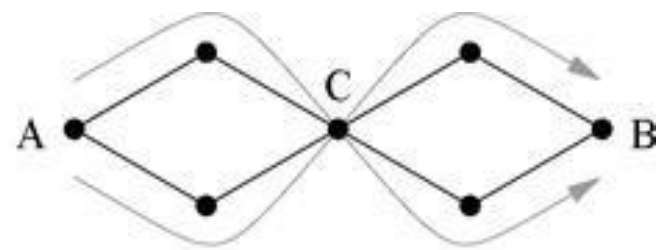
- A's strongly connected component is the intersection of its out- and in-components

Independent paths and connectivity

- **Two paths connecting a given pair of vertices are <u>edge (vertex)-independent</u> if they share no edges (vertices other than the starting and ending vertices)**
  - Two vertex-independent paths are also edge-independent (the opposite is not true)

- **The number of independent paths between a pair of vertices is called <u>connectivity</u> of the vertices**
  - <u>Edge-</u> and <u>vertex-connectivity</u>





Cut set

- **A <u>vertex cut set</u>, is a set of vertices whose removal will disconnect a specified pair of vertices**
  - This set can be thought as the *bottleneck* for the connectivity of this specific pair of nodes

- **An <u>edge cut set</u>, is a set of edges whose removal will disconnect a specified pair of vertices**

- **A <u>minimum cut set</u> is the smallest cut set that will disconnect a specified pair of vertices**

Menger’s theorem

- **The size of the minimum vertex cut set that disconnects a given pair of vertices in a network is equal to the vertex connectivity of the same vertices**
  - The same holds true for edges

- **The edge version of the theorem is important for the maximum flow problem**

Max-flow/min-cut theorem

- **In the general case we have weighted networks**
  - Minimum edge cut set is a cut set such that the sum of the weights on the edges has the minimum possible value
- **The maximum flow between a given pair of vertices in a network is equal to the sum of the weights on the edges of the minimum edge cut set that separates the same two vertices**
  - Intuitively, the “low” weight edges form bottlenecks that do not allow the flow between the two vertices to increase.