Lecture 19. Introduction to Network Analysis

COMP90051 Statistical Machine Learning

Semester 2, 2015
Lecturer: Andrey Kan

THE UNIVERSITY OF MELBOURNE

Networks in real life: the Internet

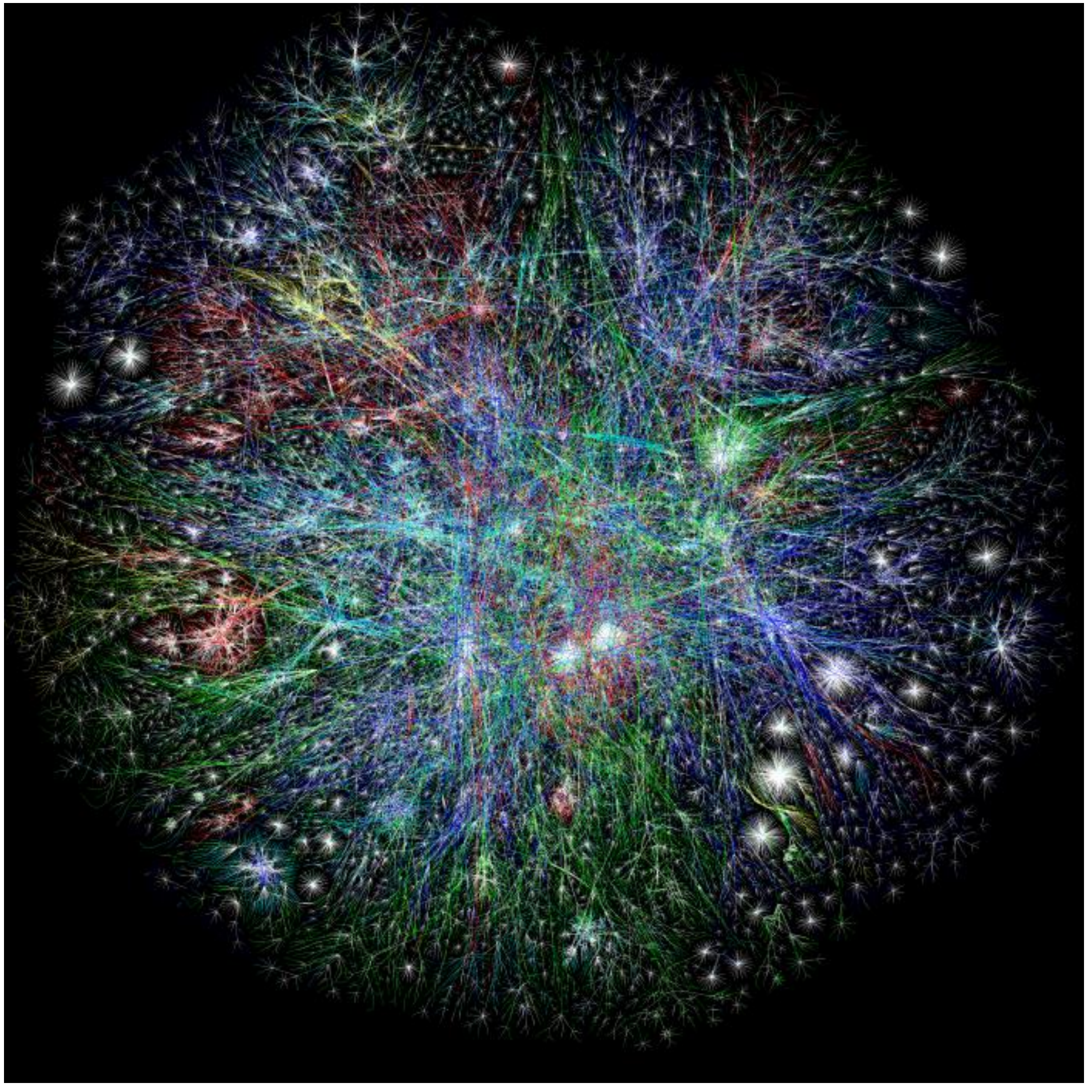
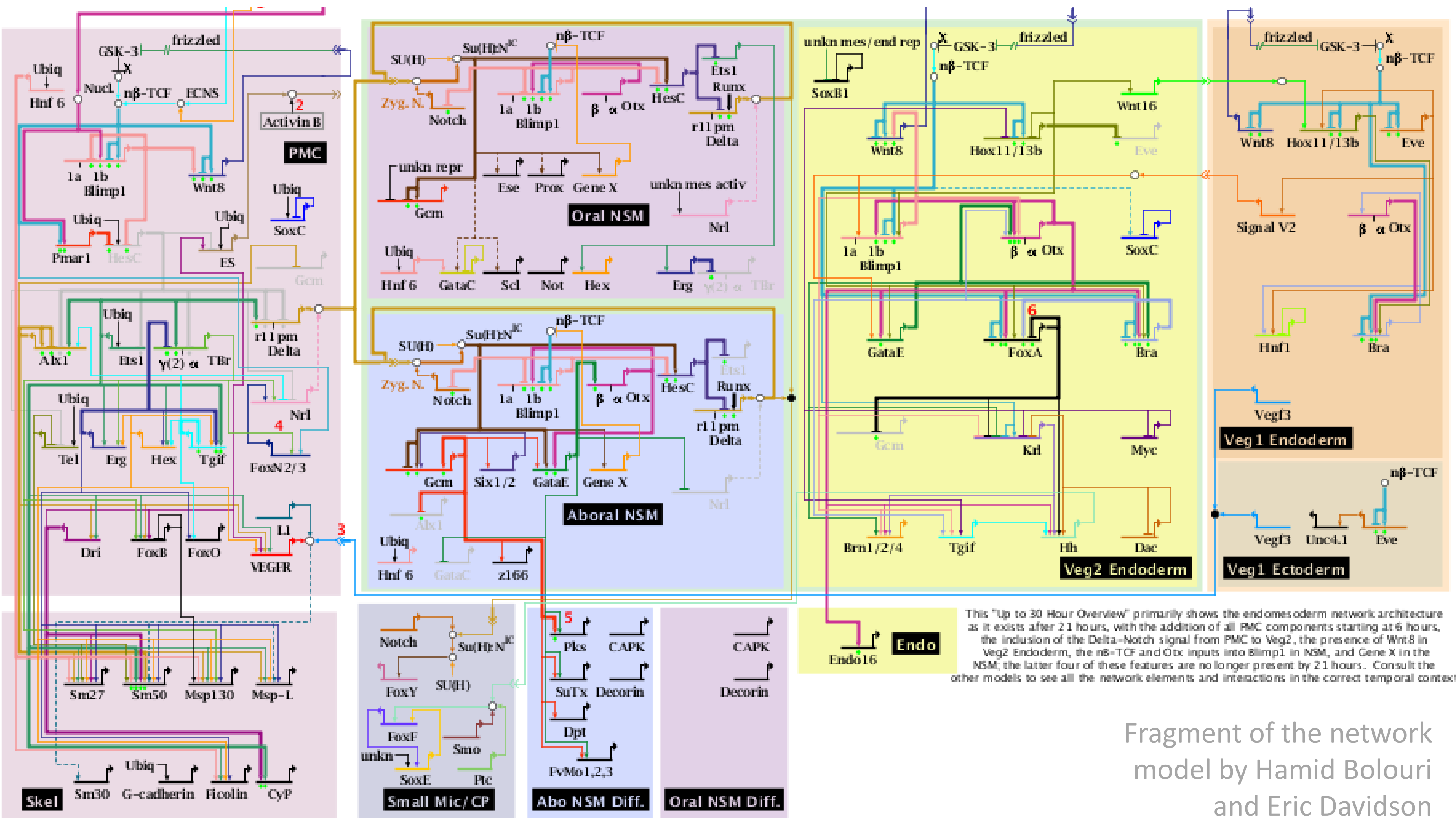


Image: OPTE Project Map (CC2)

Networks in real life: gene regulatory network



Fragment of the network model by Hamid Bolouri and Eric Davidson

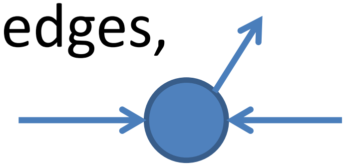
Networks in real life: transport map

Graph as a mathematical abstraction

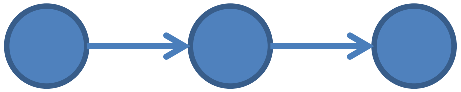
- Network = graph
- Graph is a tuple $G = \{V, E\}$, where V is a set of vertices, and $E \subseteq V \times V$ is a set of pairs of vertices (edges)
  - Undirected graph: unordered pair
  - Directed graph: ordered pair
- Graphs model pairwise relations between objects
- *Graph is a major type of data*
  - Other types of data: feature sets, sequences, images, distributions
  - Mixed types, e.g., graph where each vertex is a sequence

Basic definitions (refresher)

- *Vertex degree* is the number of incident edges
  - For directed graphs, in-degree and out-degree denote the number of adjacent incoming and outgoing edges, respectively



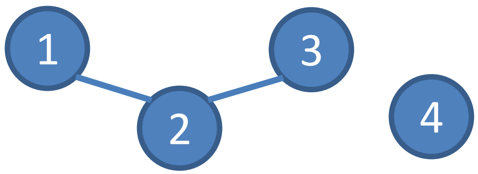
- A *path* is a sequence of vertices, such that each two consecutive vertices are connected
  - For directed graphs, edges in path must point in the same direction



- A *subgraph* is a graph with a subset of vertices and edges from the original graph
  - For graph $G = \{V, E\}$, H is a subgraph if $H = \{V_H, E_H\}$, where $V_H \subset V$, $E_H \subset E$ and $E_H \subseteq V_H \times V_H$

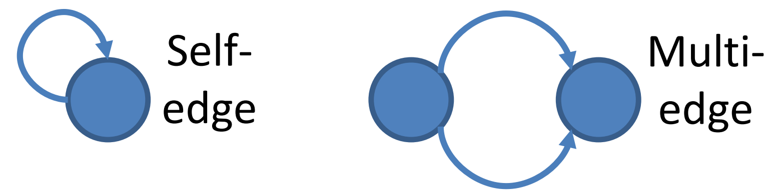
Basic definitions (refresher)

- *Connected component* is a maximal subgraph where each vertex is reachable from each other vertex via a path
  - *Reachable* means there exists a path
  - *Maximal* means that after adding any additional vertices, the new subgraph is not a connected component anymore



- *Clique* is a subgraph where each vertex is connected to each other vertex (for undirected graphs)

Types of graphs

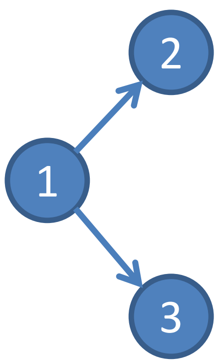


- Directed vs undirected
- Allowing self-edges or not
- Allowing multi-edges or not
- Weighted or unweighted
  * Weights on edges or on vertices
- Unlabeled vs labelled
  * Labels on edges or vertices

- In graphs (especially unlabeled and unweighted) most of the information is contained in the way the vertices are connected (connectivity structure *aka* topology)

Adjacency matrix for directed graph

- Each graph $G = \{V, E\}$ can be represented with an adjacency matrix A
  - Size of A is $|V| \times |V|$
  - $A_{ij} = 1 \Leftrightarrow (i \rightarrow j) \in E$, otherwise $A_{ij} = 0$

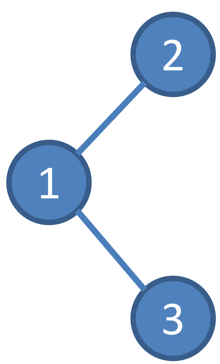


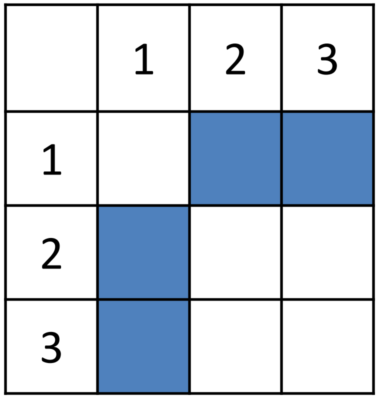
	1	2	3
1			
2			
3			

From now on we assume no multi-edges and no weights

Adjacency matrix for undirected graphs

- For undirected graphs, adjacency matrix is symmetric
- Diagonal elements are zeros unless self-edges are allowed

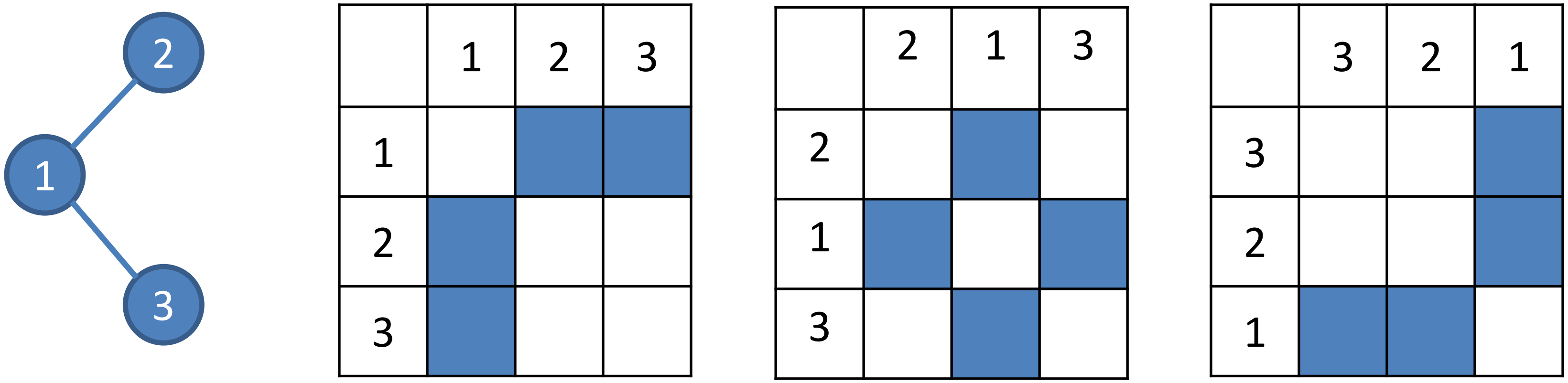




It's like a binarized kernel matrix or a pairwise similarity matrix!

Adjacency matrix

- Rows and columns of the adjacency matrix can be permuted (simultaneously)
  - This is also true for directed graphs



More network examples

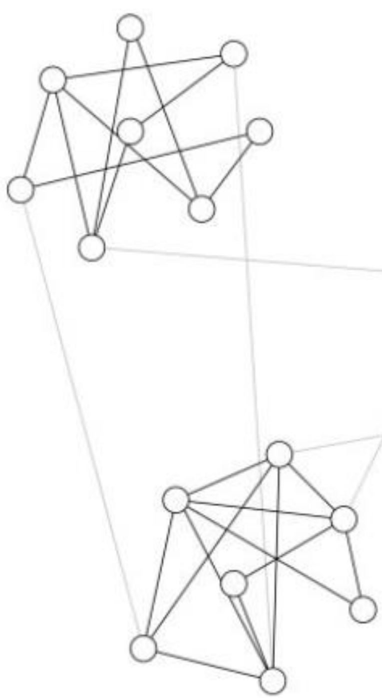
- Probabilistic Graphical Models
  * Vertices – variables
  * Directed edges – model dependencies
- Neural networks
  * Vertices – values (input, intermediate, output)
  * Directed edges – flow of computation
- Metro maps
  * Vertices – stations
  * Undirected edges – tunnels, rails
- Social relations
  * Vertices – individuals
  * Undirected edges – pairs of individuals often seen together

Learning from networks

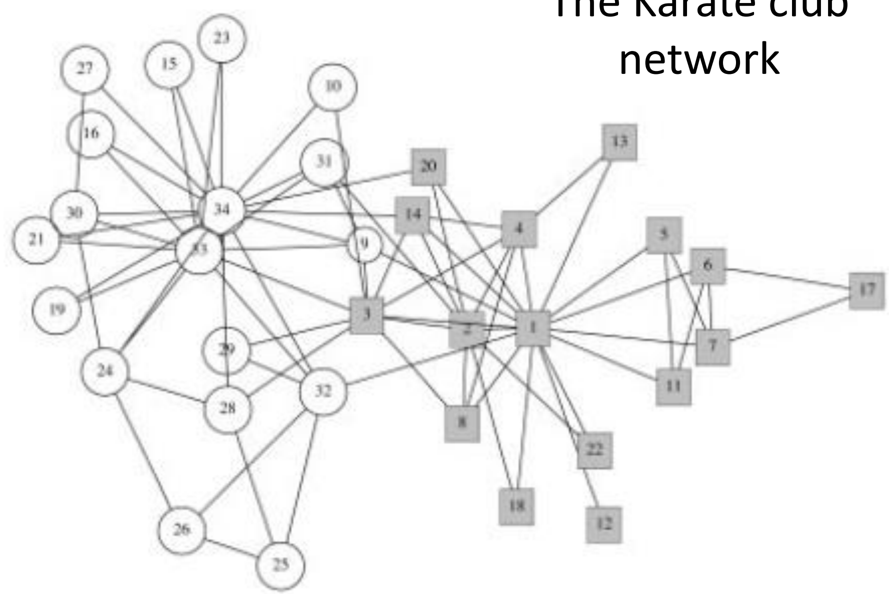
- In this course we focus on *real-world networks*
  - Naturally emerging networks
  - Emphasis on social and biological networks
  - Examples: the Internet, Facebook friendship, gene interaction
- Growing interest as more and more data becomes available
- Example problems / types of analysis
  - Link prediction
  - Identifying frequent subgraphs
  - Identifying influential vertices
  - Community finding

Properties of real-world networks

- Real-world networks are not homogeneous
  - Different vertices play different “roles”



A schematic representation of a network with community structure



The Karate club network

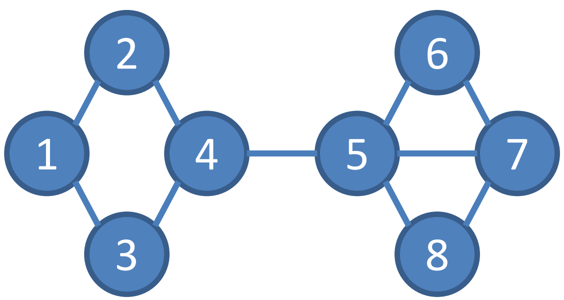
Images: Girvan and Newman, Community structure in social and biological networks, PNAS, 2002

Properties of real-world networks

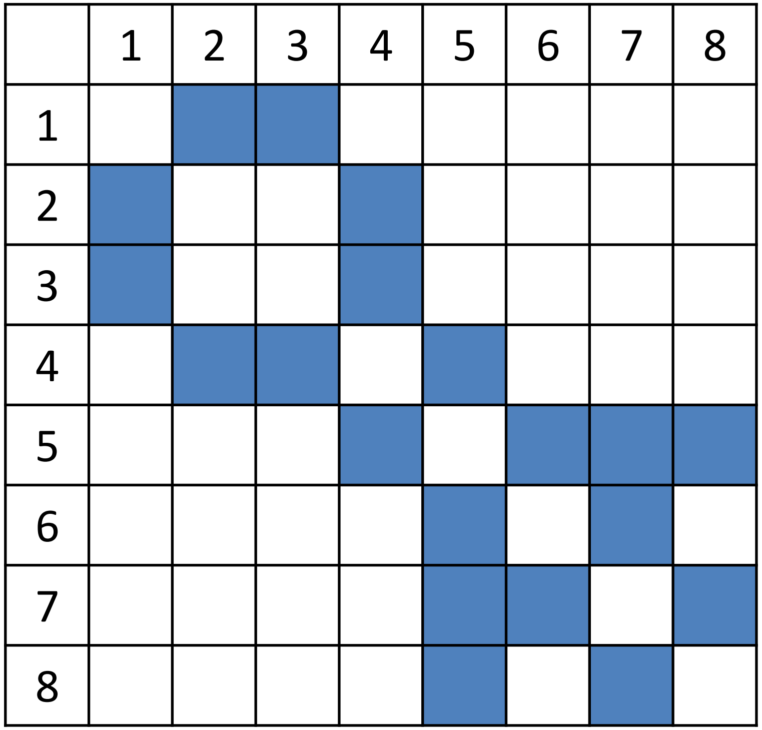
- Sparse adjacency matrix
- Small world phenomenon
- Right-skewed degree distribution
- Clustering (transitivity)

Properties of real-world networks: Sparsity

- Given $|V|$ vertices the maximum number of possible edges is $|V| \times |V|$
- However, many real-world networks have much fewer number of edges, often in the order of $|V|$



- The resulting adjacency matrix is sparse: most of its elements are zero



Properties of real-world networks: Small world

- Small world phenomenon: most vertices can be reached from any other vertex with a small number of hops
  - “Six degrees of separation”
  - Friends of friends chain

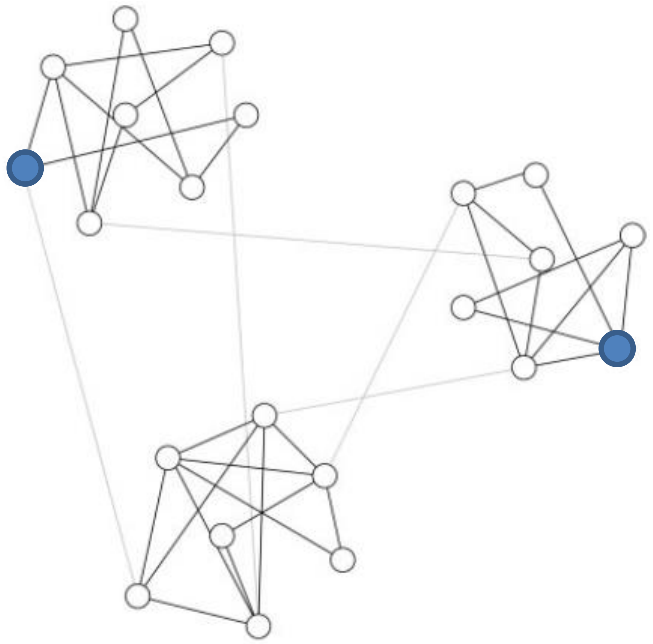
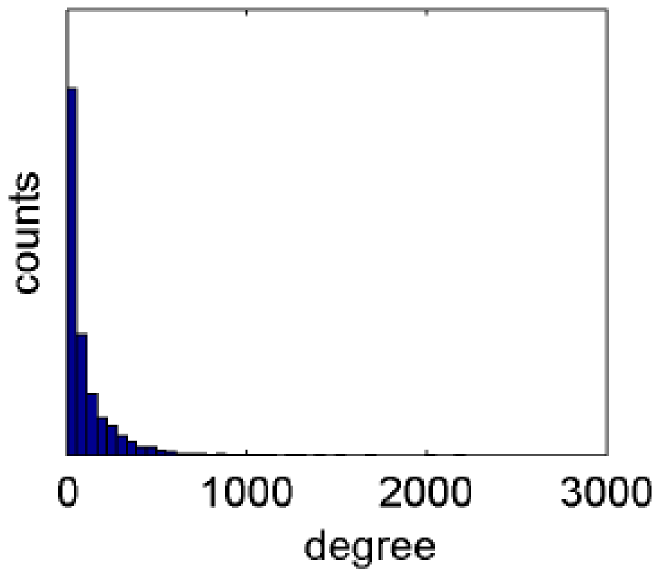


Image: adapted from Girvan and Newman, Community structure in social and biological networks, PNAS, 2002

Properties of real-world networks: Power law

- Right-skewed degree distribution is common
  - Few "hubs", and a large number of peripheral vertices
- Often asymptotically follows a power law $P(k) \sim k^{-\gamma}$
  - In many networks $2 < \gamma < 3$
- "The rich get richer" or

  Preferential attachment



Properties of real-world networks: Clustering

- If two vertices are both connected to the same third vertex, they are more likely to be connected
  - More likely compared to two arbitrarily chosen vertices
- This property is also called network transitivity
- Clustering coefficient

$$C = \frac{3 \times (\#triangles)}{(\#connected\ triples\ of\ vertices)}$$

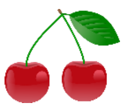
- In many networks $0.1 < C < 0.5$

Checkpoint

- Which of the following statements is true?

 There is a finite number of paths in a real-world network

 Maximum shortest path across all pairs of vertices tends to be small for real-world networks

 In a small network each vertex can be accessed from any other vertex via a path

art: OpenClipartVectors at pixabay.com (CC0)

The Google PageRank algorithm

- Rank webpages by some measure importance
- Consider a directed graph where vertices are webpages, and edges are links

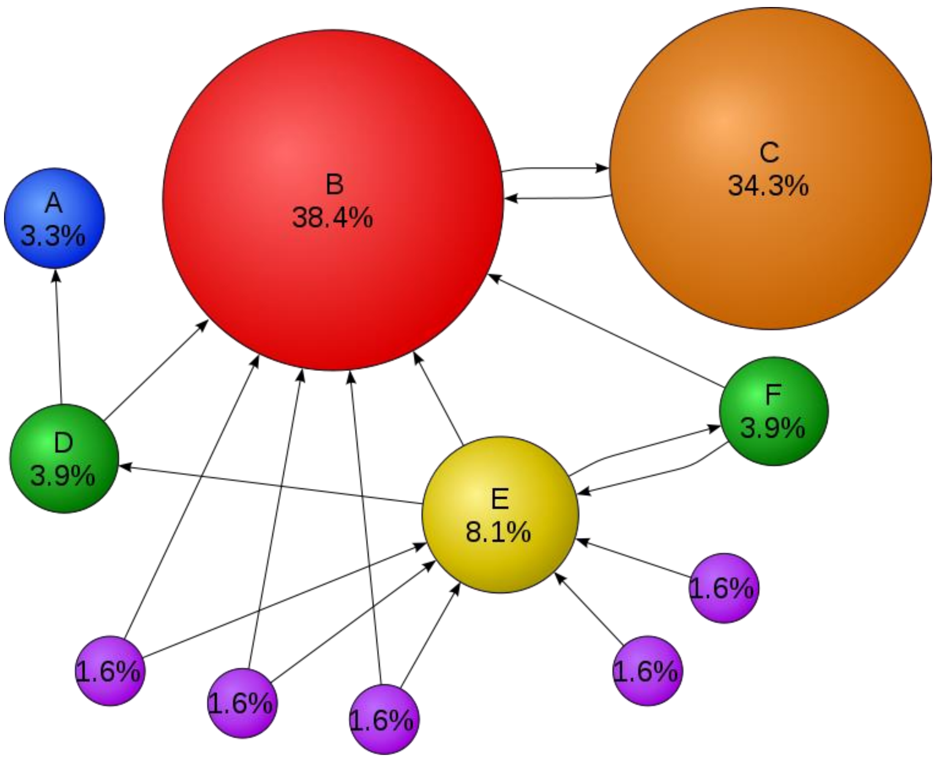


Image: 345Kai, Wikimedia Commons, Public Domain

PageRank: Ranking scheme revisited

- PageRank assigns a score of importance to each page (vertex)
- A recursive definition: a page is important if it is referred by important vertices

$$p_i = \frac{(1-d)}{N} + d\sum_{j=1}^{N} A_{ji}\frac{p_j}{c_j}$$

- A is the adjacency matrix:
  - A_{ji} equals to 1 if there is a link from page j to page i, otherwise A_{ji} is 0
- $(1-d)$ is the minimum guaranteed rank
- c_j is the number of pages linked from page j (out-degree of vertex j)
- N is the total number of pages

PageRank: Interpretation

- A recursive definition: a page is important if it is referred by important vertices

$$p_i = \frac{(1-d)}{N} + d \sum_{j=1}^{N} A_{ji} \frac{p_j}{c_j}$$

- PageRank p_i can be interpreted as a likelihood that a random surfer will land at page i
  - The surfer starts from a random page
  - Given a current page, the surfer follows a random link on this page
  - With a small probability the surfer does not follow any links from the current page, but jumps to a random page

PageRank: Iterative solution

- At time $t = 0$ assume $p_i(0) = \frac{1}{N}$
- At each subsequent time step

$$p_i(t+1) = \frac{1-d}{N} + d\sum_{j=1}^{N} A_{ji}\frac{p_j(t)}{c_j}$$

- In matrix form

$$\boldsymbol{p}(t+1) = \frac{1-d}{N}\boldsymbol{e} + dA^T \mathrm{D}_c^{-1}\boldsymbol{p}(\boldsymbol{t})$$

  * Here $\boldsymbol{e}$ is a vector of N ones
  * D_c is a diagonal matrix with elements $\frac{1}{c_j}$

- Stop when convergence is observed

$$|\boldsymbol{p}(t+1) - \boldsymbol{p}(t)| < \varepsilon$$

PageRank: Iterative solution

- Assume a steady state at $t \to \infty$
  - Steady state means that for some large t: $\boldsymbol{p}(t+1) = \boldsymbol{p}(t)$
- We have that

$$\boldsymbol{p} = \frac{1-d}{N}\boldsymbol{e} + dA^T \mathrm{D}_c^{-1}\boldsymbol{p}$$

  - Here $\boldsymbol{e}$ is a vector of N ones
  - D_c is a diagonal matrix with elements $\frac{1}{c_j}$
- After rearranging the terms one gets

$$\boldsymbol{p} = (I - dA^T \mathrm{D}_c^{-1})^{-1}\frac{1-d}{N}\boldsymbol{e}$$

  - Here I is the identity matrix
- Proofs of existence and uniqueness of solution are omitted here

PageRank: Solving using the power method

- A recursive definition: a page is important if it is referred by important vertices

$$p_i = (1-d) + d\sum_{j=1}^{N} A_{ji}\frac{p_j}{c_j}$$

- Let $\boldsymbol{e}$ be a vector of N ones and D_c be diagonal matrix with elements c_j. Also assume that PageRank is normalized $\boldsymbol{e}^T\boldsymbol{p} = N$. PageRank equation can then be rewritten in a matrix form

$$\boldsymbol{p} = (1-d)\boldsymbol{e} + A^T \mathrm{D}_c^{-1}\boldsymbol{p} = \left[\frac{1}{N}(1-d)\boldsymbol{e}\boldsymbol{e}^T + dA^T D_c^{-1}\right]\boldsymbol{p}$$

- The expression in the square braces contains known information, denote the expression as X. One gets $\boldsymbol{p} = X\boldsymbol{p}$

- Vector $\boldsymbol{p}$ (ranks) can be found using the power method
  * The proof of this statement involves relating the PageRank equation to Markov chains

Summary

- Recall basic definitions of graph theory (vertex degree, paths, connected components)
- How to construct an adjacency matrix?
- Give examples of real-world networks
- What are the properties of real-world networks?
- What is the aim of PageRank algorithm, and what is the intuition behind its main equation?