Essential Concepts

- **simple graph:** graph without multi-edges and self-loops (as opposed to a **multi-graph**)
- graphs can be **directed/undirected**, **weighted/unweighted** and have other **attributes** on nodes/edges
- **multiplex network:** graph with different layers (but with *equivalent* nodes on each layer)
- **temporal network:** graph that evolves over time (nodes/edges can change)
- **degree**: number of connections (**indegree/outdegree** on directed networks)
- **degree sequence:** ordered list of degrees | **degree distribution:** frequency count of degree occurrences
- networks can be **sparse** or **dense** (a **complete graph/clique** is a graph with all possible connections)
- **component:** connected set of nodes | **giant component:** largest component (with high fraction of nodes)
- **strongly connected graph:** every vertex is reachable from every other vertex
- **strongly connected components:** partition of a directed graph into (maximal) strongly connected subgraphs
- **DAG - directed acyclic graph:** directed graph without cycles (paths that begin and end on same node)
- **bipartite graph:** graph with two disjoint sets of nodes U and V with edges only from U to V
- **distance:** number of edges connecting the shortest path between two nodes (sum weights on weighted networks)
- **diameter:** maximum shortest path between any pair of nodes
- **clustering coefficient:** fraction of neighbors that are connected

Graph Models

- **Erdös-Renyi** model: $G_{n,p}$ - graph with n nodes and each edge with probability p
  - degree distribution: binomial; clustering coef.: low; path length: small; emergence of a giant component
- **Small-World** model (Watts-Strogatz): regular lattice with some randomness introduced (*"shortcuts"*)
  - degree distribution: regular; clustering coef.: high; path length: small
- **Scale-Free** model (Barabasi-Albert): preferential-attachment growth as nodes arrive
  - degree distribution: power-law;

Node Centrality and Link Analysis

- **degree centrality:** nodes with higher degree are more central
- **betweenness centrality:** fraction of shortest paths the node is in
- **closeness centrality:** inverse of sum of path lengths to all other nodes (**harmonic:** sum of inverse of distances)
- **eigenvector centrality:** how central a node is depends on how central its neighbors are
- **hits algorithm:** two scores: hub (sum of votes that we point to), authority (sum of votes that point to us)
- **page rank:** sum of edges that point to us (normalized by degree of outgoing node) - **power iterations**
  - interpretation as **random walk**: probability that a random surfer ends up on a node
  - problems might arise with **dead ends** (no out links) or **spider traps** (outlinks within a group)
  - **teleporting** (with a certain probability) to solve these possible problems
  - **personalized page rank:** teleport to a specific *"relevant"* group of pages

Roles and Community Structure

- **roles:** partition of nodes into structural positions in the network
- **communities:** partition of nodes into sets with high nr of internal connections and low nr of external connections
  - motivation: *triadic closure* (chains tend to close) and *strength of weak ties*
  - **hierarchical clustering:** greedy approach to iteratively modify successive candidate partitions
  - **divise method:** start with all nodes in one community and refine by *splitting*
  - **agglomerative method:** start with all nodes in individual communities and improve by *merging*
  - **girvan-newman method:** divise - remove edges with highest edge betweenness centrality
  - **louvain algorithm:** agglomerative - perform merge with highest gains in modularity; contract graph into super-nodes when no more gains are achievable and repeat
  - **modularity:** measures quality of partition (compare with null model preserving degree distribution)

Subgraph Patterns

- **network motifs:** induced subgraphs with higher frequency than expected in similar networks (same degree seq.)
- **orbit:** structural position respecting symmetries (nodes in the same orbit map into each other on an automorphism)
- **graphlet degree vector:** feature vector with the frequency of the node in each orbit position
- **counting subgraphs (and orbits)** is computationally hard (*subgraph census*)
  - **network-centric** approach: count occurrences of all k-sized subgraphs
  - **subgraph-centric** approach: count occurrences of one subgraph at a time
  - **set-centric** approach: count occurrences of custom set of subgraphs
- **g-trie:** data structure to store and count subgraphs (*"prefix tree"* of graphs)
  - *flexible* (e.g. incorporate orbits, undirected/directed graphs, uncolored/colored graphs, use with *sampling*, ...)
  - *iterative insertion* using *canonical ordering*
  - *backtracking* procedure to match subgraphs with *symmetry* breaking conditions

Network Construction

- **multipartite network:** project into one mode (e.g. *common neighbors* or *jaccard index* [ratio of shared neighbors])
- **graph contraction:** shrink the graph by contracting into supernodes and repeat recursively
- **network deconvolution:** reversing the effects of transitivity ("recover" original network from observed one)
- **k-nearest neighbor graph:** graph with edges to k most *similar* nodes (e.g. cosine similarity)
- **from time series to networks:** convert time series into network and analyze network to understand time series
  - **correlation networks:** nodes are time series, edges represent correlation
  - **visibility graphs:** nodes are observations, edges represent *"visibility"* (can nodes *see* each other?)
  - **quantile graphs:** nodes are quantiles in values, edges represent amount of transitions