Social Network Analysis for Computer Scientists

Frank Takes

LIACS, Leiden University

https://liacs.leidenuniv.nl/~takesfw/SNACS

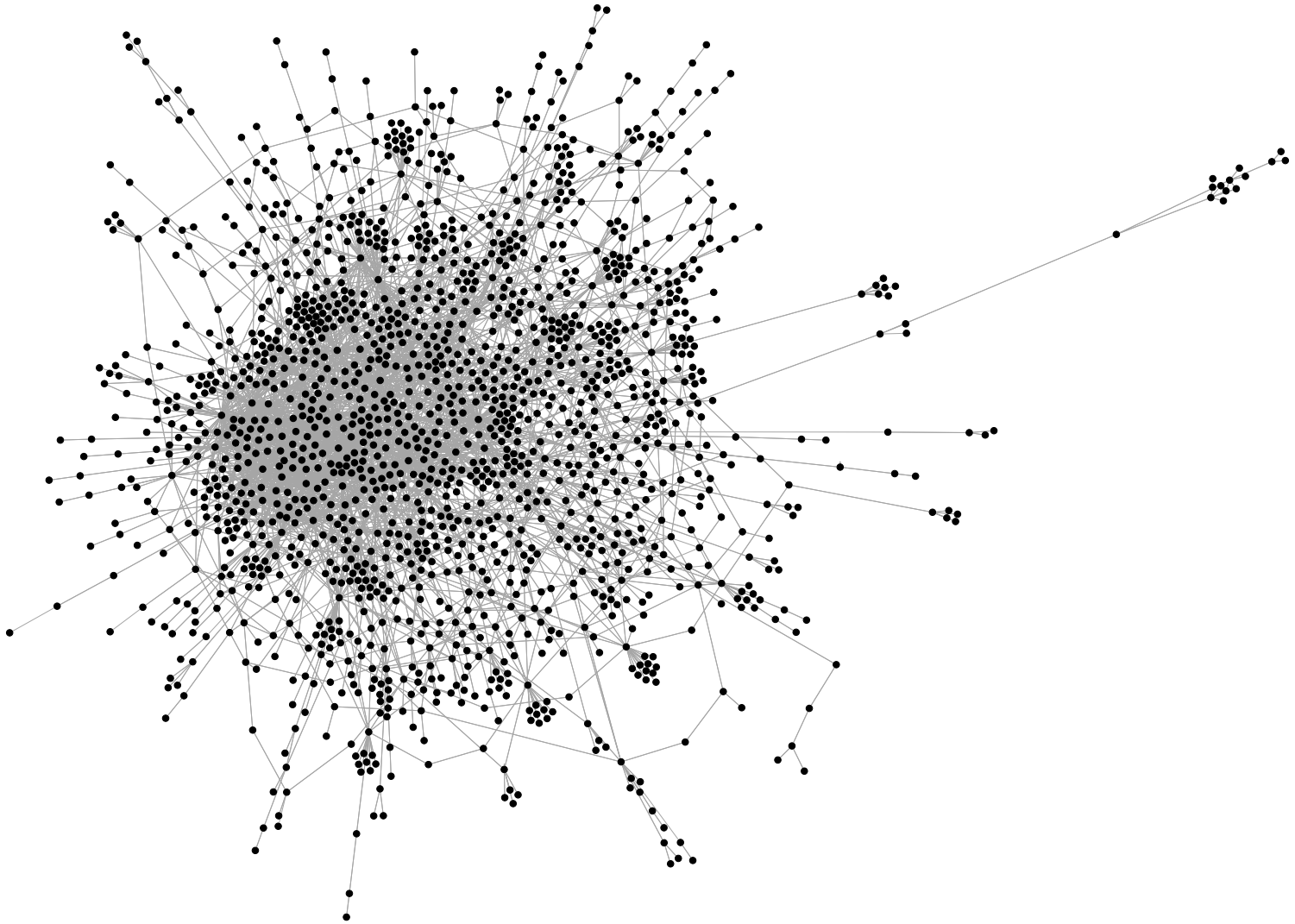
Lecture 11 — Network model extensions and visualization

Today

- Recap
- Weighted networks
- Signed networks
- Multilayer networks
- Higher-order networks
- Network visualization

Recap

Networks



Real-world networks

1. Sparse networks — density
2. Fat-tailed power-law degree distribution — degree
3. Giant component — components
4. Low pairwise node-to-node distances — distance
5. Many triangles — clustering coefficient

- Many examples: social networks, communication networks, citation networks, collaboration networks, protein interaction networks, information networks, webgraphs, financial networks, ...

Centrality measures

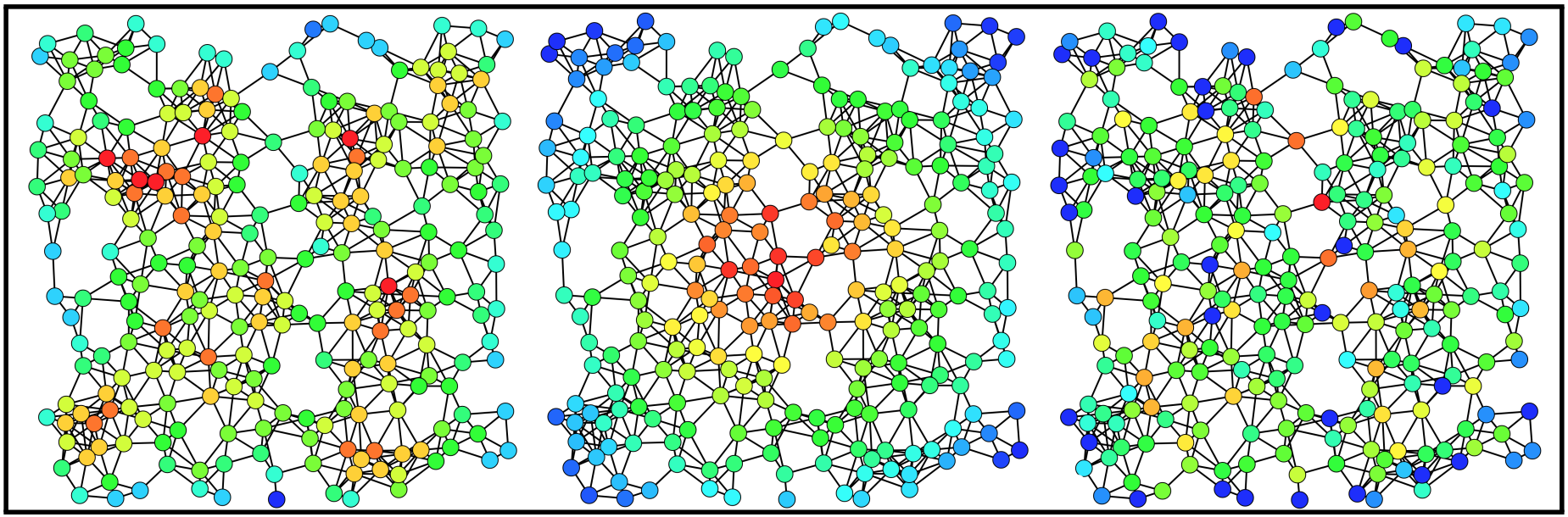
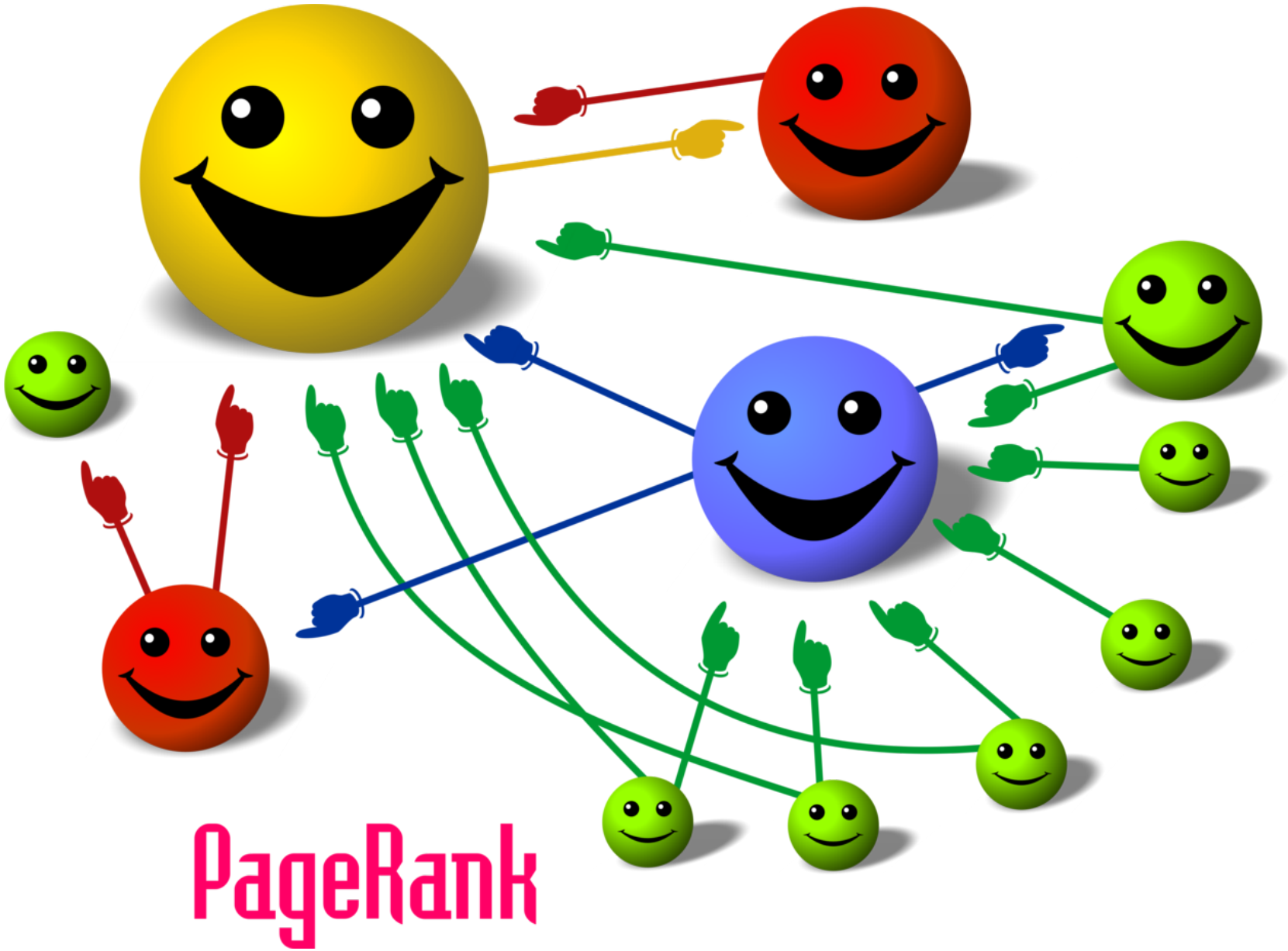


Figure: Degree, closeness and betweenness centrality.

Source: "Centrality" by Claudio Rocchini, Wikipedia `File:Centrality.svg`


PageRank

Projection

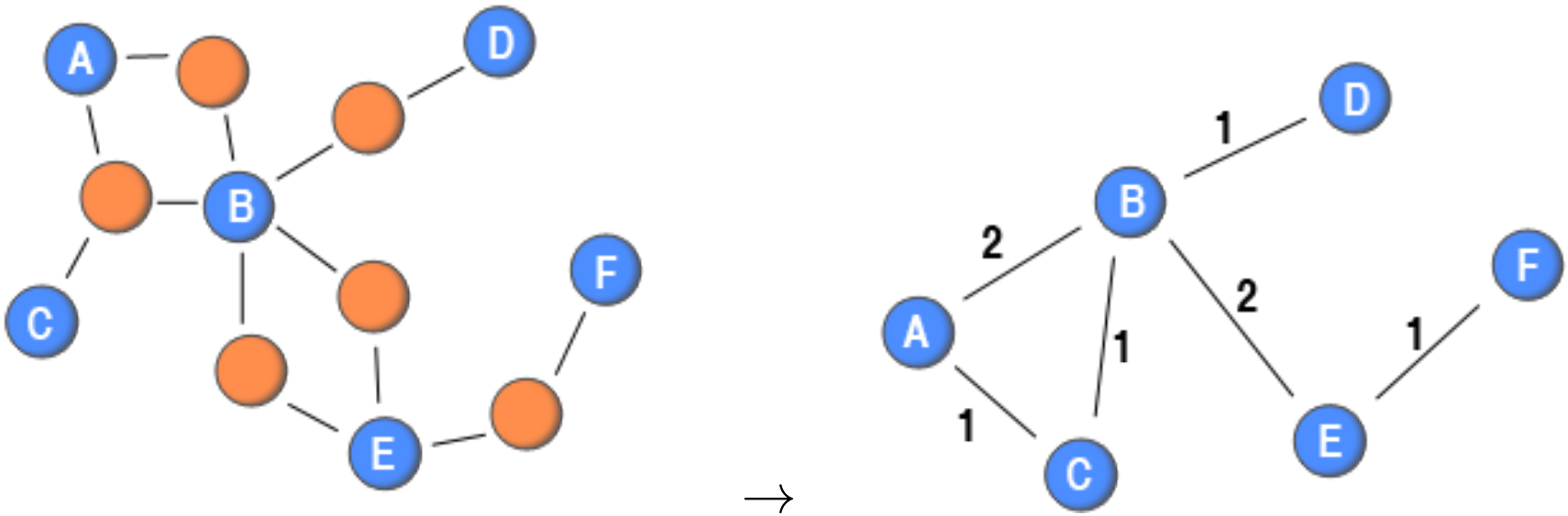


Image: `http://toreopsahl.com`

Community detection

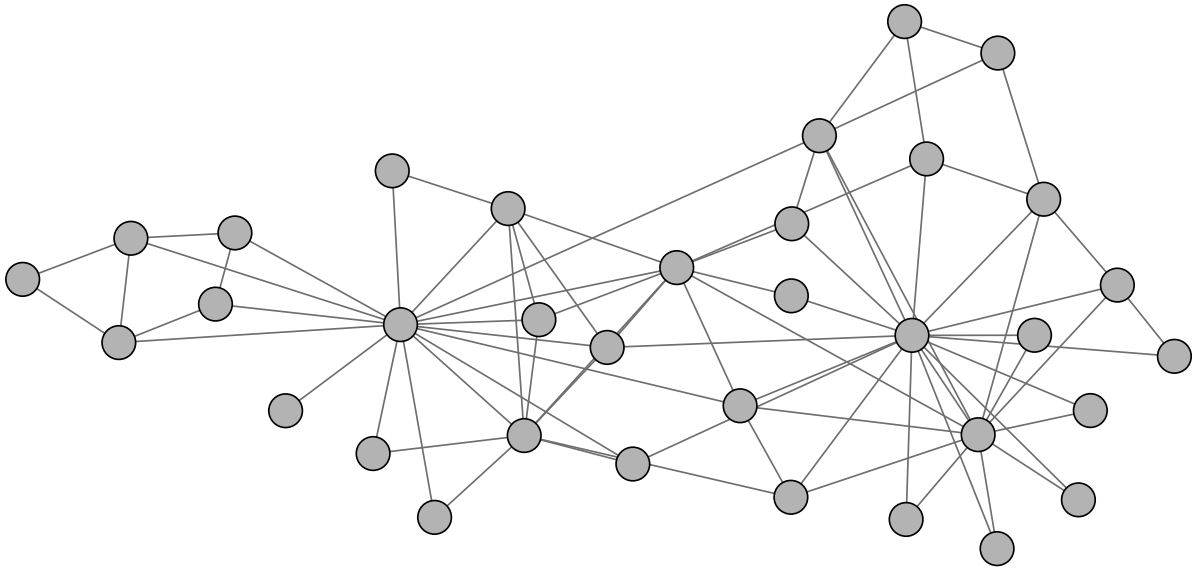


Figure: Communities: node subsets connected more strongly with each other.

- Partitioning a given graph into densely connected groups of nodes

Community detection

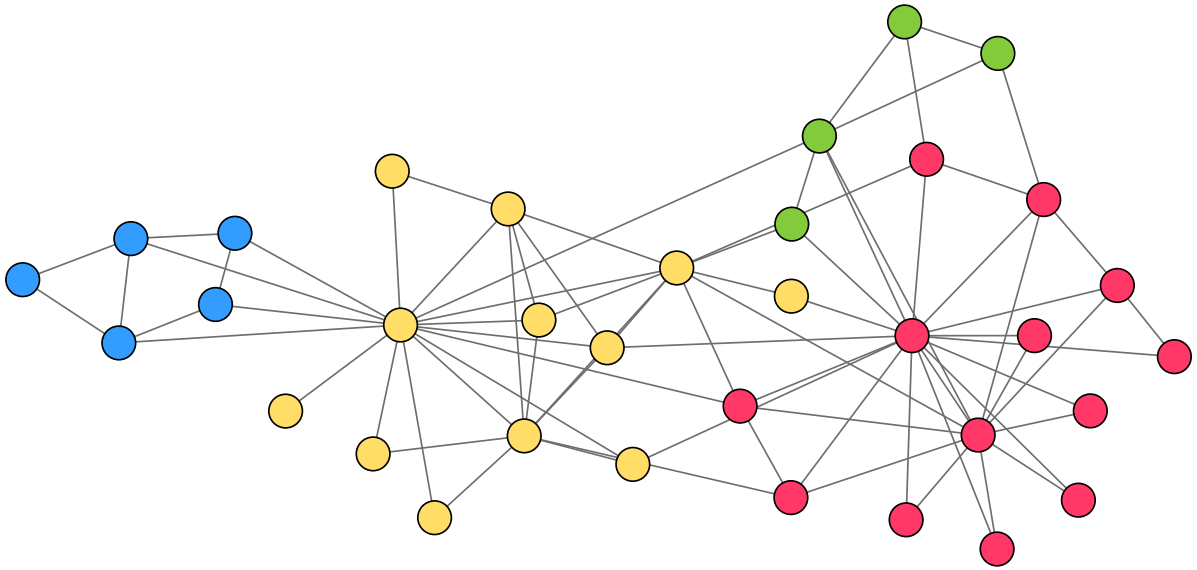


Figure: Communities: node subsets connected more strongly with each other.

- Divisive algorithms (Girvan–Newman), local search algorithms (Louvain) optimizing a quality function (e.g., modularity), label propagation and random walk based approaches.

Graph evolution

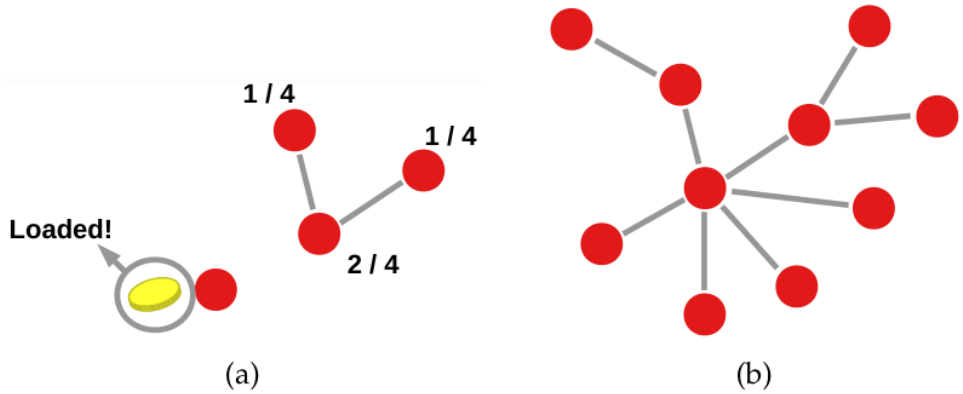


Figure: PA: target node based on degree (a), graph formed through PA (b).

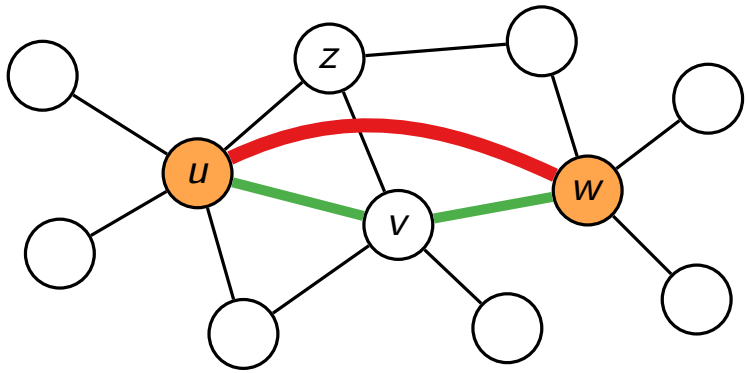
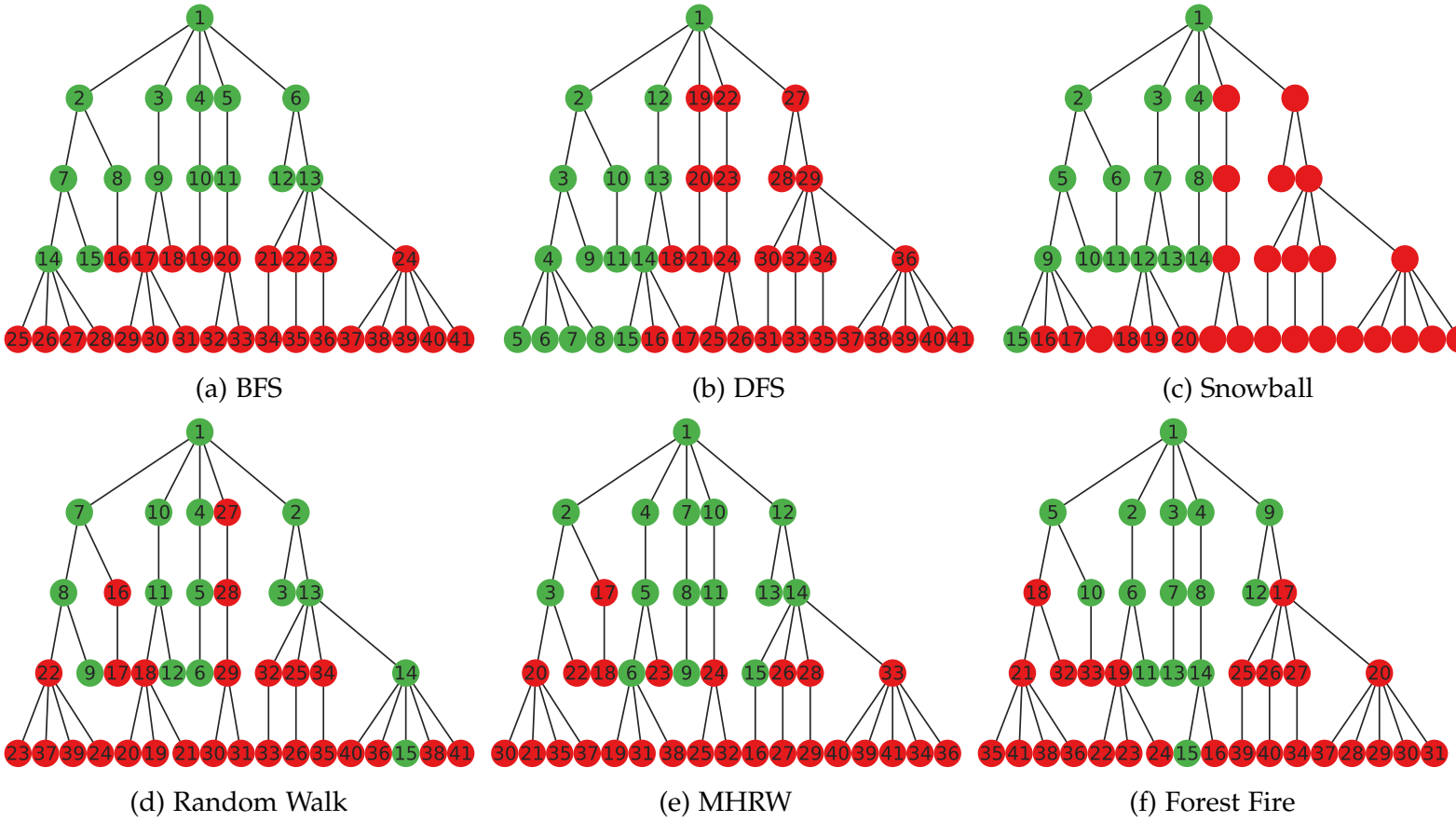


Figure: Triadic closure.

Network sampling



(a) BFS (b) DFS (c) Snowball

(d) Random Walk (e) MHRW (f) Forest Fire

Figure: Six sampling strategies on the same network with a budget of 15 nodes. Labels give the exploration order; nodes are sampled (green) or not (red).

Machine learning and networks

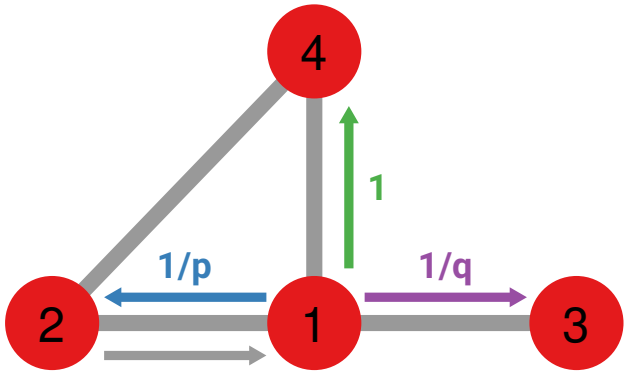


Figure: Node embeddings based on random walks.

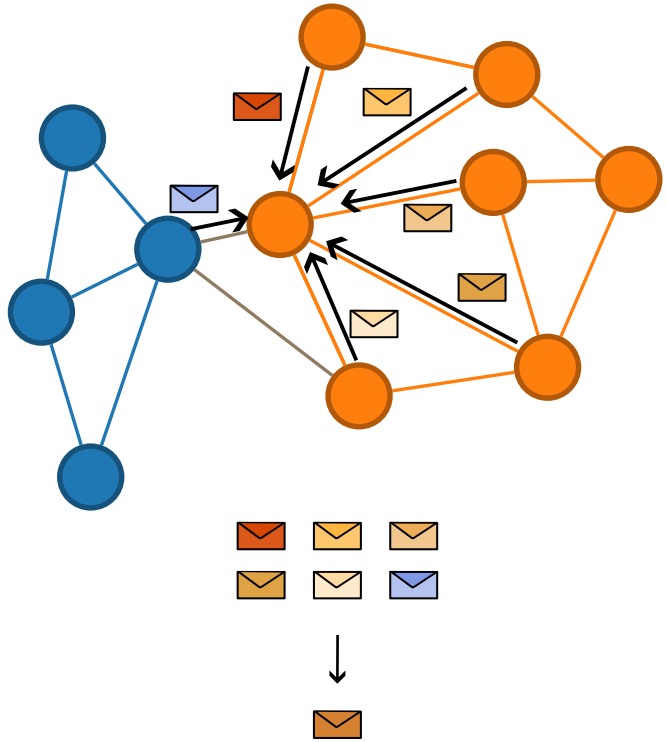
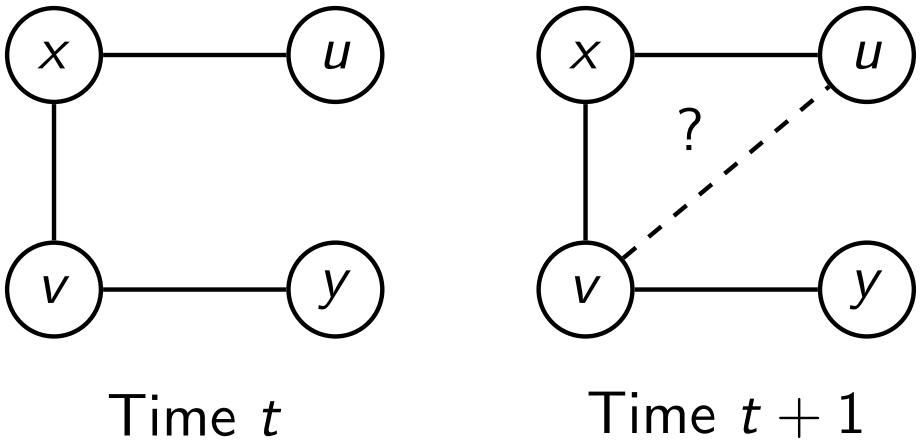


Figure: Message passing in GNNs.

Link prediction

- **Link prediction**: given a network $G = (V, E)$, predict which of the node pairs that are not connected are most likely to be connected
  - **Dynamic**: given snapshot G^t, predict new edges of G^{t+1}
  - **Static**: given an incomplete observation of G, predict **missing** edges
- **Approach**: Formulate a hypothesis on how nodes link, turn it into a **link prediction score** $score(u, v)$ for each unconnected pair, and rank the pairs by score



Network motifs

- **Network motif**: small subgraph with a given topology
- Real-world networks have **overexpressed** motifs, such as triangles
- Is a motif significant? Compare its count with the count in a null model, e.g., the configuration model
- Counting motifs requires deciding whether two graphs are the same

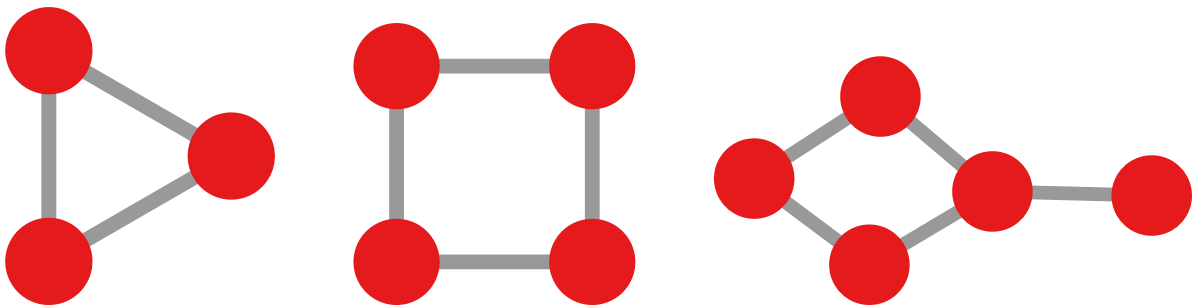
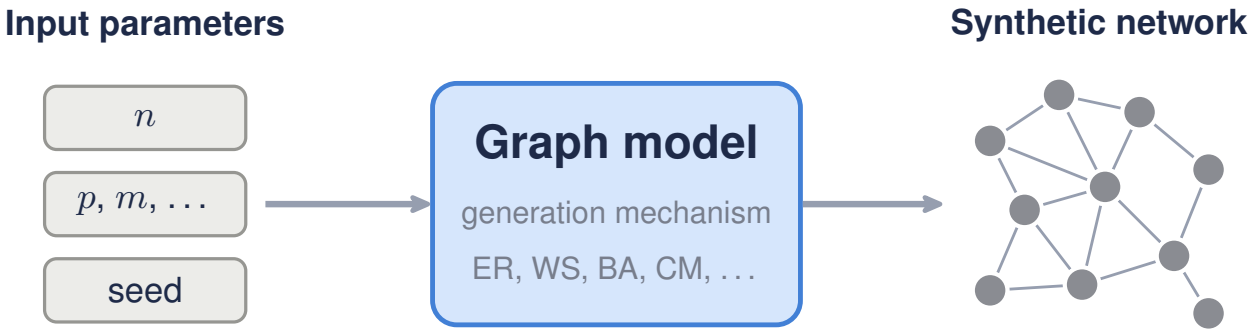


Figure: Three motifs: (a) a triangle, (b) a square and (c) a square with a dangling edge.

Network models



Network resilience

- Random failures versus targeted attacks on hubs
- Networks with a skewed degree distribution are robust to random failures, but fragile to targeted attacks
- Failures can cascade, also across interdependent networks

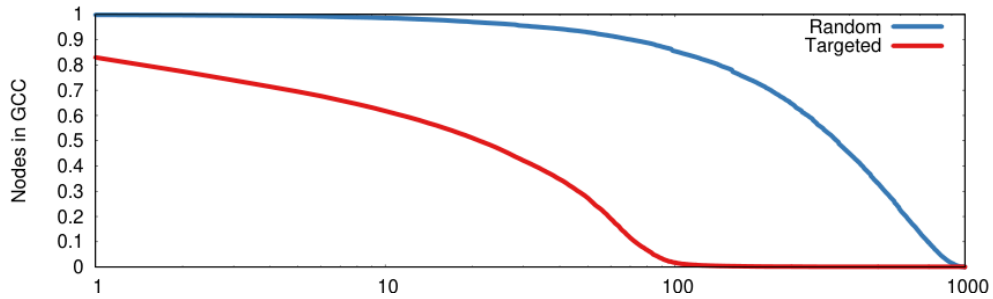


Figure: Probability of being in the largest connected component as a function of the number of failing nodes, for random (blue) and targeted (red) failures.

Spreading processes

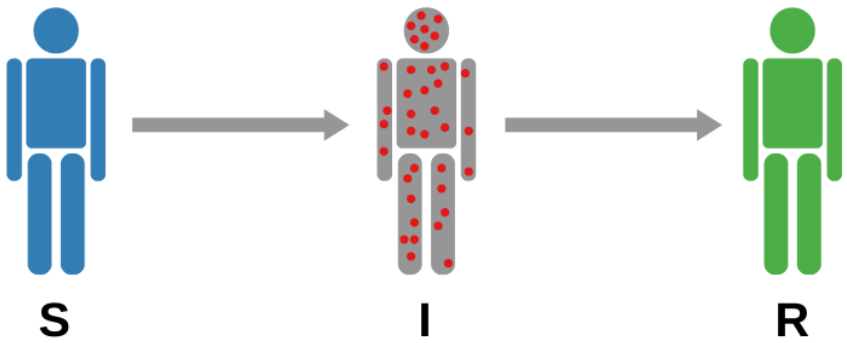
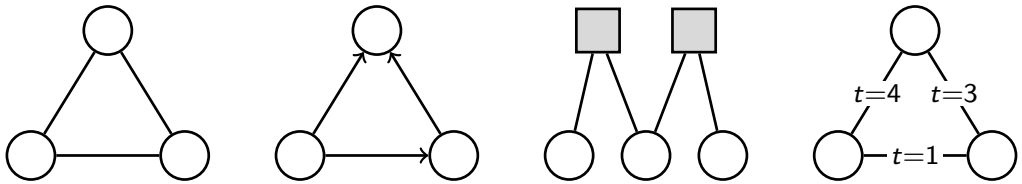


Figure: The SIR model: three states and two transitions.

Network model extensions

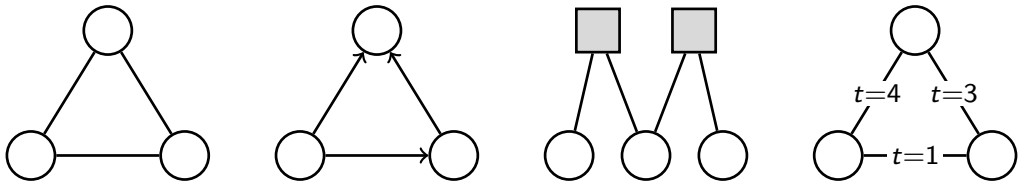
Today: beyond simple graphs



(a) undirected (b) directed (c) two-mode (d) temporal

Figure: Network models discussed so far.

Today: beyond simple graphs



(a) undirected (b) directed (c) two-mode (d) temporal

Figure: Network models discussed so far.

- Network types covered today:
  - **Weighted**: weighted centrality and modularity, backboning
  - **Signed**: structural balance, frustration, predicting signs
  - **Multilayer**: layer relevance, heterogeneous networks, density, clustering
  - **Higher-order**: hypergraphs, simplicial complexes, higher-order ties with memory

Weighted networks

Weighted networks

- Weighted network $G = (V, E, W)$: edge (u, v, w) with weight w, usually $W \subseteq \mathbb{R}^+$

Weighted networks

- Weighted network $G = (V, E, W)$: edge (u, v, w) with weight w, usually $W \subseteq \mathbb{R}^+$
- A weight can represent
  - **Proximity** (high weight: nodes are closer, e.g., many interactions)
  - **Distance** (high weight: nodes are further apart, e.g., length, cost)

Weighted networks

- Weighted network $G = (V, E, W)$: edge (u, v, w) with weight w, usually $W \subseteq \mathbb{R}^+$
- A weight can represent
  - **Proximity** (high weight: nodes are closer, e.g., many interactions)
  - **Distance** (high weight: nodes are further apart, e.g., length, cost)
- **Correlation network**: special type of weighted network in which the weight is the similarity of two node attribute vectors; every node pair gets a weight, so edges must be filtered

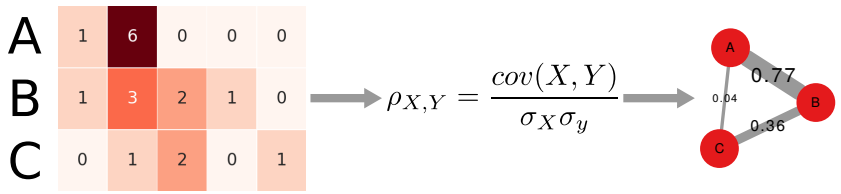


Figure: Correlation networks based on attribute similarity.

Weighted adjacency matrix and weighted degree

- In the adjacency matrix of a weighted network, cell A_{uv} contains weight w_{uv} (instead of 0 or 1)
- Weighted degree (node strength): $s(u) = \sum_{v \in N(u)} w_{uv}$, which equals $deg(u)$ if all weights are 1

0	5	0	2	2	0	6	0	0
5	0	0	0	3	4	5	3	0
0	0	0	3	2	5	0	0	2
2	0	3	0	0	0	0	0	5
2	3	2	0	0	5	0	0	0
0	4	5	0	5	0	4	0	0
6	5	0	0	0	4	0	0	0
0	3	0	0	0	0	0	0	0
0	0	2	5	0	0	0	0	0

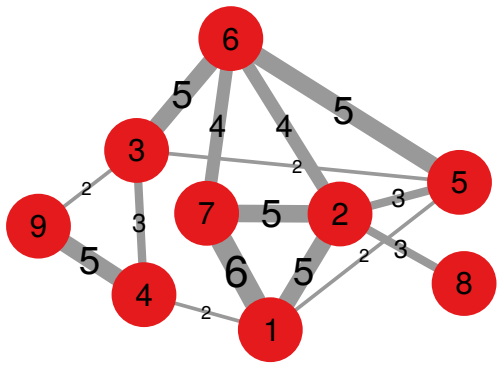


Figure: Non-binary adjacency matrix and the corresponding weighted graph.

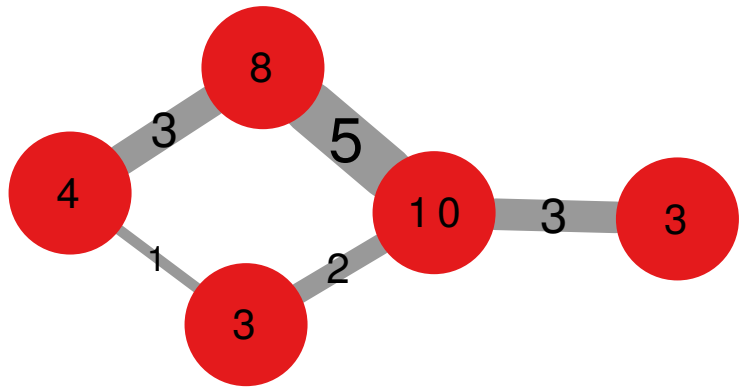


Figure: Weighted degree as node label in a weighted network.

Weight distribution

- Recall that the degree distribution of real-world networks is fat-tailed, often a power law
- **Weight distribution**: frequency of each edge weight value
- Also **fat-tailed**: most edges are weak, a few are very strong
- As for degrees, plot it with logarithmic horizontal and vertical axes

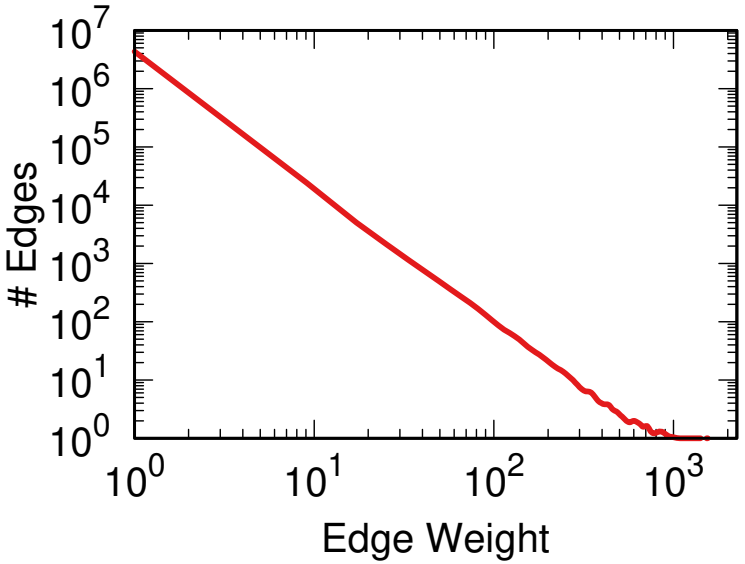


Figure: Number of edges (vertical axis) per edge weight (horizontal axis) in a projected Twitter network: 82% of the edges have weight 1.

Weighted centrality: degree, closeness and betweenness

- **Degree**: weighted degree (strength) $s(u)$; or combine both with a parameter α:

$$C_D^{\alpha}(u) = deg(u)^{1-\alpha} \cdot s(u)^{\alpha}$$

with $\alpha = 0$ the degree and $\alpha = 1$ the strength

Weighted centrality: degree, closeness and betweenness

- **Degree**: weighted degree (strength) $s(u)$; or combine both with a parameter α:

$$C_D^{\alpha}(u) = deg(u)^{1-\alpha} \cdot s(u)^{\alpha}$$

  with $\alpha = 0$ the degree and $\alpha = 1$ the strength
- **Closeness and betweenness** use shortest paths: in a weighted network, the length of a path is the sum of its edge lengths, found with Dijkstra instead of BFS
- Weights are usually proximities (tie strength), so first turn them into lengths, most commonly $\ell_{uv} = 1/w_{uv}$, or more generally $\ell_{uv} = 1/w_{uv}^{\alpha}$
- Then as before, e.g., closeness $C_C(u) = 1/\sum_{v \in V} d_w(u, v)$ with d_w the weighted distance

T. Opsahl, F. Agneessens and J. Skvoretz, Node Centrality in Weighted Networks: Generalizing Degree and Shortest Paths, *Social Networks* 32(3): 245–251, 2010.

Weighted centrality: an example and other measures

- Which path from u to v is shortest, with lengths $\ell = 1/w^{\alpha}$?

Weighted centrality: an example and other measures

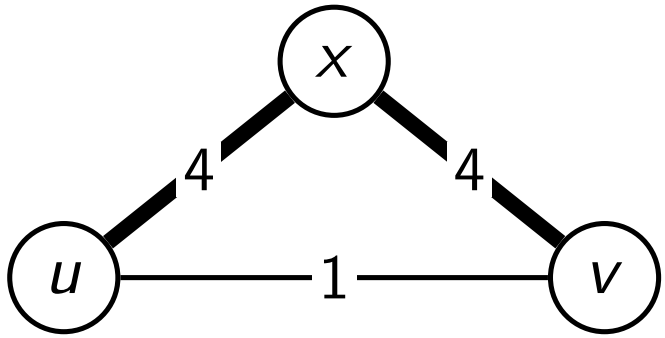
- Which path from u to v is shortest, with lengths $\ell = 1/w^{\alpha}$?
- Direct: length 1; via x: $2 \cdot 4^{-\alpha}$
- $\alpha = 0$ (hops only): direct, $1 < 2$
- $\alpha = 1$

Weighted centrality: an example and other measures

- Which path from u to v is shortest, with lengths $\ell = 1/w^{\alpha}$?
- Direct: length 1; via x: $2 \cdot 4^{-\alpha}$
- $\alpha = 0$ (hops only): direct, $1 < 2$
- $\alpha = 1$ (weights only): via x, $0.5 < 1$
- $\alpha = 0.5$

Weighted centrality: an example and other measures

- Which path from u to v is shortest, with lengths $\ell = 1/w^{\alpha}$?
- Direct: length 1; via x: $2 \cdot 4^{-\alpha}$
- $\alpha = 0$ (hops only): direct, $1 < 2$
- $\alpha = 1$ (weights only): via x, $0.5 < 1$
- $\alpha = 0.5$: a tie, both have length 1



- **PageRank, HITS** and eigenvector centrality can use the weighted adjacency matrix directly; the random walker follows edge (u, v) with probability $w_{uv}/s(u)$
- Always check whether a weight is a proximity or a distance!

Community detection

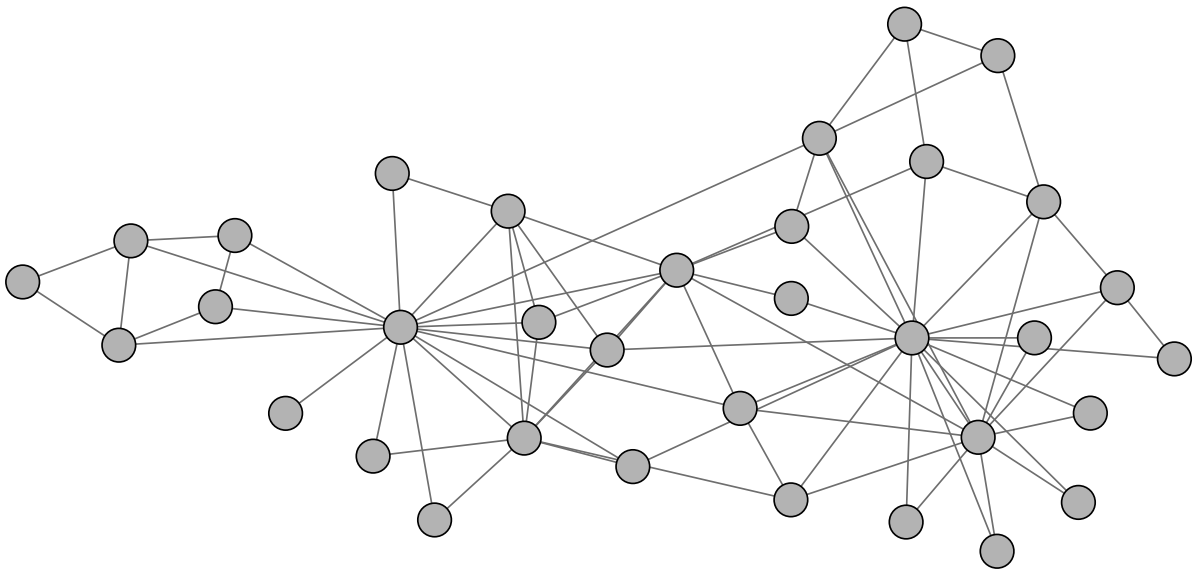


Figure: Communities: node subsets connected more strongly with each other.

Community detection

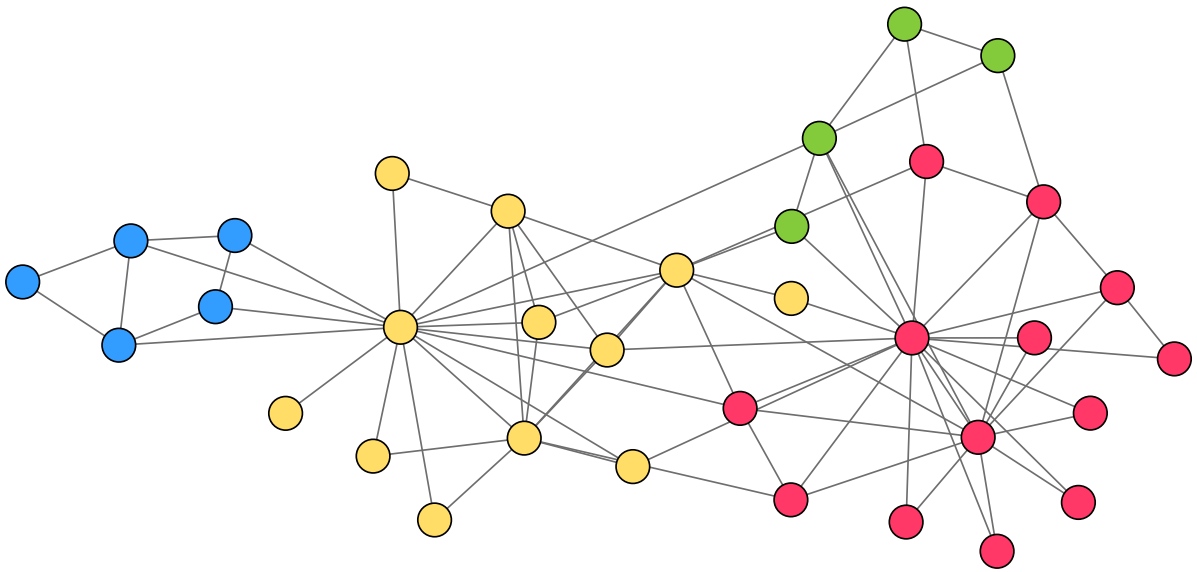


Figure: Communities: node subsets connected more strongly with each other.

Modularity

- **Modularity** compares the observed network with that of a network with the same node and edge count and degrees, rewired at random

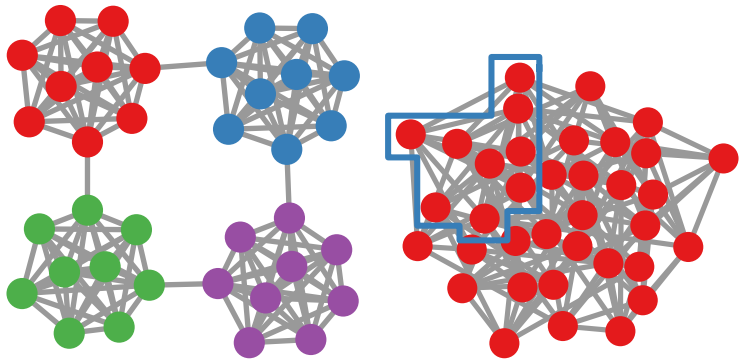


Figure: Network with communities (left) and rewired network (right).

- Resolution parameter r indicating how "tough" the algorithm should look for communities
- Algorithms for community detection optimize (maximize) modularity score Q given some r (using local search, heuristics, hill climbing, genetic algorithms or other optimization techniques)

Weighted modularity

- Modularity Q: the edges inside communities minus the expected number in the rewired network

$$Q = \frac{1}{2m} \sum_{u,v \in V} \left(A_{uv} - \frac{deg(u) \cdot deg(v)}{2m} \right) \delta(c_u, c_v)$$

with $\delta(c_u, c_v) = 1$ if u and v are in the same community, 0 otherwise

Weighted modularity

- Modularity Q: the edges inside communities minus the expected number in the rewired network

$$Q = \frac{1}{2m} \sum_{u,v \in V} \left(A_{uv} - \frac{deg(u) \cdot deg(v)}{2m} \right) \delta(c_u, c_v)$$

  with $\delta(c_u, c_v) = 1$ if u and v are in the same community, 0 otherwise
- Weighted modularity: replace A_{uv} by the weight w_{uv}, degree by strength $s(u)$ and m by the total weight W

$$Q_w = \frac{1}{2W} \sum_{u,v \in V} \left(w_{uv} - \frac{s(u) \cdot s(v)}{2W} \right) \delta(c_u, c_v)$$

- Louvain algorithm from lecture 4 already works with weights: after coarsening, the network is weighted

Network backboning

- **Network backboning**: remove the edges that are not significant, while keeping all nodes
- Solves problems if the network is noisy, or too large to compute on
- Most backboning methods are for weighted networks

Network backboning

- **Network backboning**: remove the edges that are not significant, while keeping all nodes
- Solves problems if the network is noisy, or too large to compute on
- Most backboning methods are for weighted networks
- **Naive thresholding**: keep only the edges with a weight above a threshold

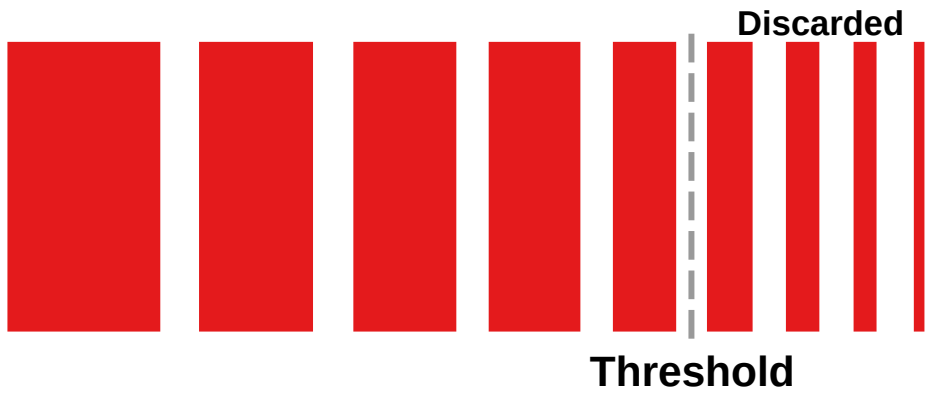
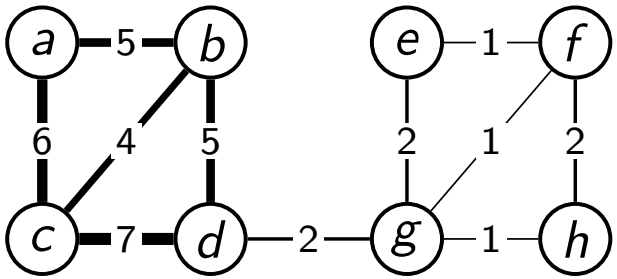


Figure: Naive thresholding: edges (bars, width is weight) sorted by decreasing weight; everything below the threshold is discarded.

Naive thresholding: an example

- What happens if we keep only the edges with weight at least 3?

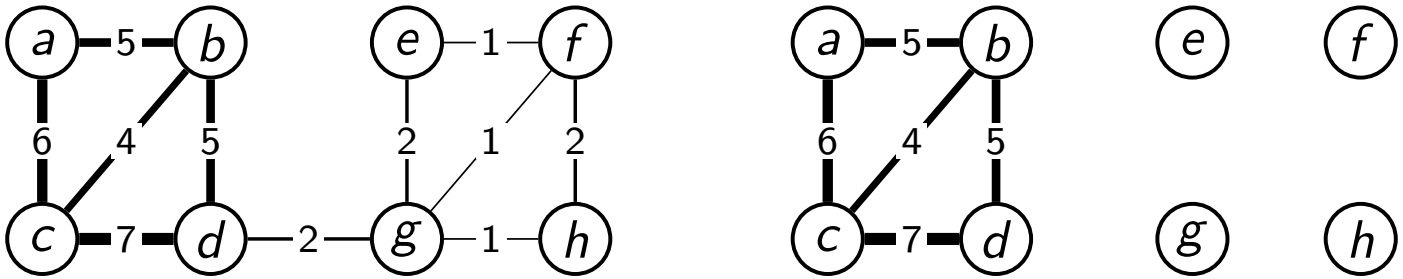


(a) weighted network

Figure: (a) A weighted network, edge width is weight. (b) The same network after naive thresholding.

Naive thresholding: an example

- What happens if we keep only the edges with weight at least 3?
- Subgraph a, b, c, d keeps all its edges, subgraph e, f, g, h loses all of them
- Strong edges cluster, so one global threshold is not always optimal



(a) weighted network

(b) threshold $w \geq 3$

Figure: (a) A weighted network, edge width is weight. (b) The same network after naive thresholding.

Signed networks

Signed networks

- **Signed network**: edges are positive (friend, trust) or negative (enemy, distrust)

Signed networks

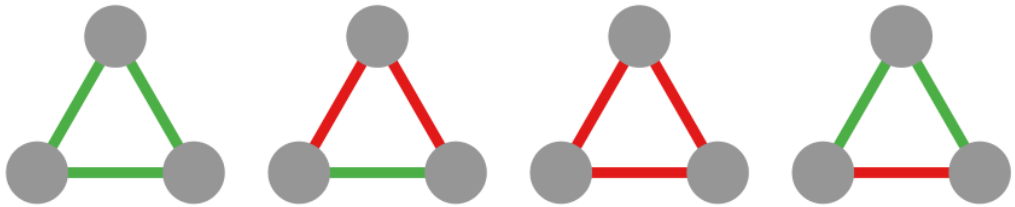
- **Signed network**: edges are positive (friend, trust) or negative (enemy, distrust)
- Example: on Slashdot, users tag other users as **friend** or **foe**; most tags are positive, but a notorious troll collects thousands of foes
- What do you expect the positive and negative degree distributions to look like?

Signed networks

- **Signed network**: edges are positive (friend, trust) or negative (enemy, distrust)
- Example: on Slashdot, users tag other users as **friend** or **foe**; most tags are positive, but a notorious troll collects thousands of foes
- What do you expect the positive and negative degree distributions to look like?
  - There is a limit to the number of friends you can have, but not to the number of people that can be mad at you

Structural balance

- **Structural balance** theory: the friend of my friend is my friend, the enemy of my friend is my enemy
- Rule of thumb: a triangle is balanced if it has an **odd number of positive edges**
- Balanced triangles are overexpressed in real networks, unbalanced ones are expected to change



Structural balance

- **Structural balance** theory: the friend of my friend is my friend, the enemy of my friend is my enemy
- Rule of thumb: a triangle is balanced if it has an **odd number of positive edges**
- Balanced triangles are overexpressed in real networks, unbalanced ones are expected to change

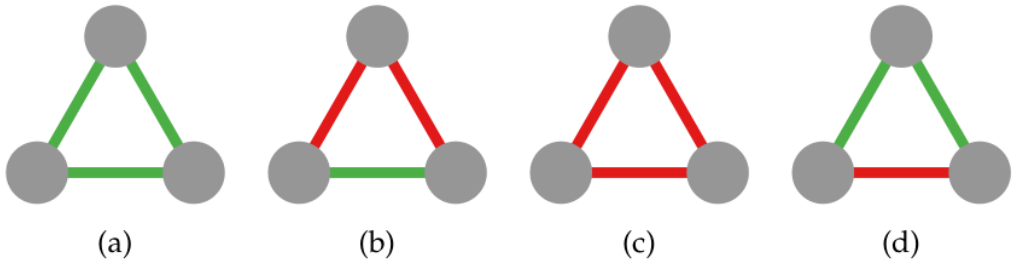


Figure: Triangles with positive (green) and negative (red) edges. (a) and (b) are balanced, (c) and (d) unbalanced.

Structural balance

- **Structural balance** theory: the friend of my friend is my friend, the enemy of my friend is my enemy
- Rule of thumb: a triangle is balanced if it has an **odd number of positive edges**
- Balanced triangles are overexpressed in real networks, unbalanced ones are expected to change

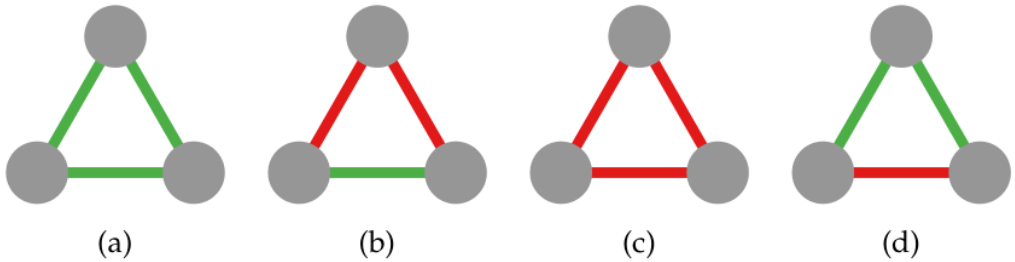


Figure: Triangles with positive (green) and negative (red) edges. (a) and (b) are balanced, (c) and (d) unbalanced.

- The all-negative triangle is sometimes considered balanced . . .

Balanced networks

- A signed network is **balanced** if all its triangles are balanced, i.e., each has an odd number of positive edges

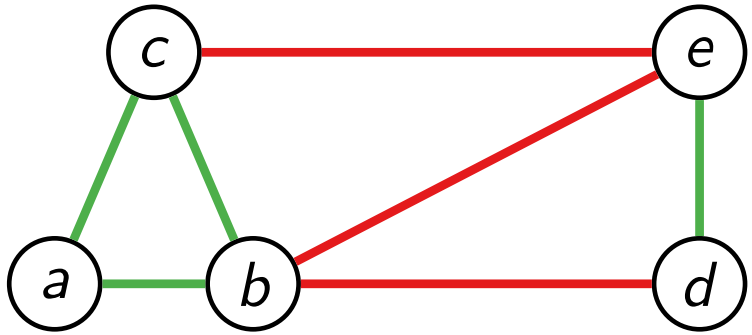


Figure: A balanced network with positive (green) and negative (red) edges.

Balanced networks

- A signed network is **balanced** if all its triangles are balanced, i.e., each has an odd number of positive edges
- Is the network shown on the right balanced?
- Most real networks are not perfectly balanced: how far are they from balance?

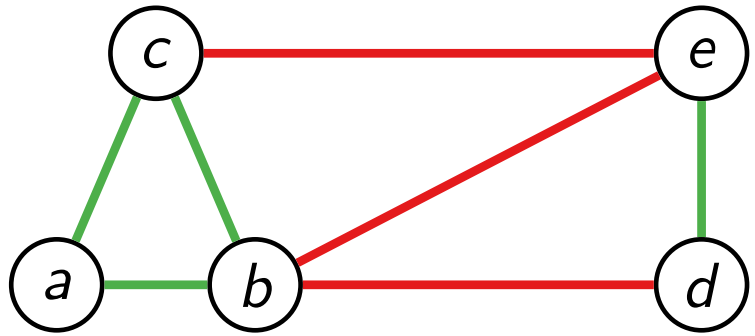


Figure: A balanced network with positive (green) and negative (red) edges.

Frustration

- **Frustration**: smallest number of edges whose removal or sign flip makes the network balanced, divided by the number of edges
- What is the frustration of the network below?

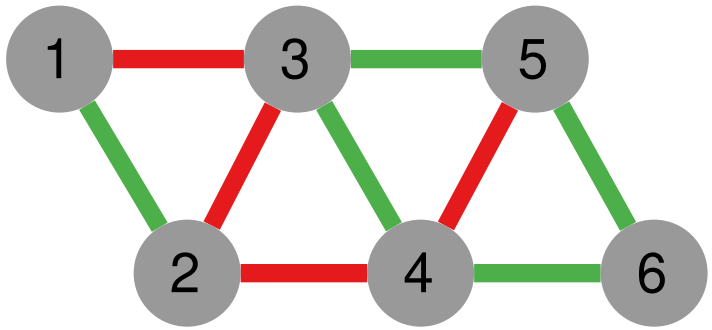


Figure: A signed network with positive (green) and negative (red) edges.

Frustration

- **Frustration**: smallest number of edges whose removal or sign flip makes the network balanced, divided by the number of edges
- What is the frustration of the network below?
- Triangles $(3, 4, 5)$ and $(4, 5, 6)$ are unbalanced; flipping edge $(4, 5)$ balances both, so the frustration is $1/9$

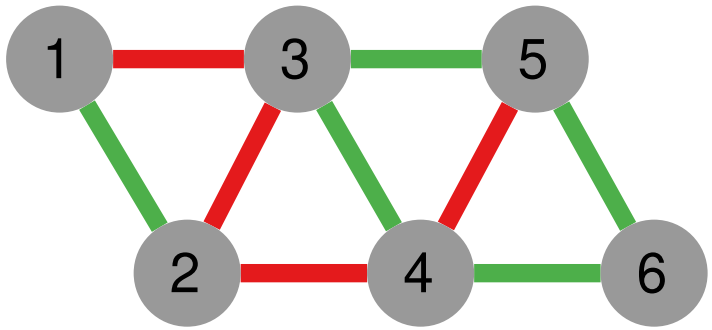


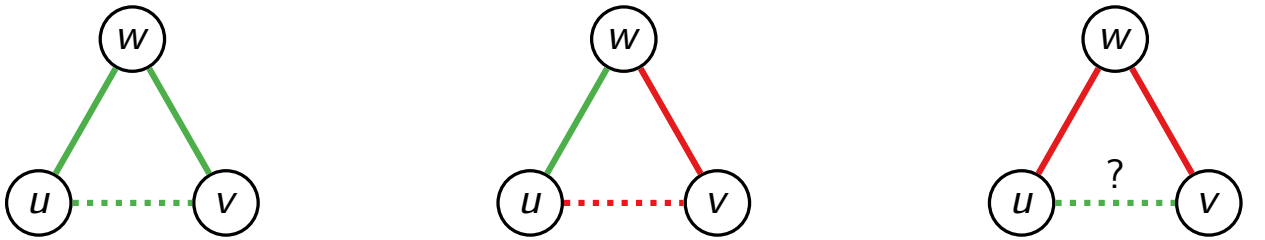
Figure: A signed network with positive (green) and negative (red) edges.

Predicting signs

- Signed link prediction: first predict **which** pair connects with $score(u, v)$, then predict the **sign**
- In triadic closure, balance theory predicts the sign that closes a triangle:
  - Two positive edges: close with a positive edge
  - One positive and one negative edge: close with a negative edge
  - Two negative edges: harder to predict, balance theory says positive

Predicting signs

- Signed link prediction: first predict **which** pair connects with $score(u, v)$, then predict the **sign**
- In triadic closure, balance theory predicts the sign that closes a triangle:
  - Two positive edges: close with a positive edge
  - One positive and one negative edge: close with a negative edge
  - Two negative edges: harder to predict, balance theory says positive



(a) two positive: close positive

(b) mixed: close negative

(c) two negative: positive preferred

Figure: The three cases, with positive (green) and negative (red) edges; the dotted edge (u, v) is the one to predict.

Multilayer networks

Multilevel networks

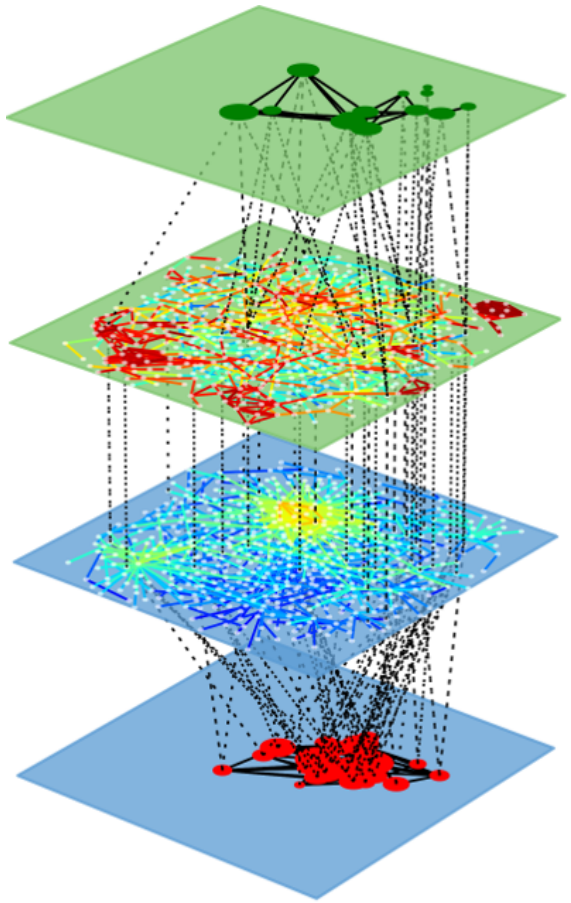


Figure: This is a multilevel network, not to be confused with a multilayer network.

Multilayer networks

- Multilayer network $G = (V, E, L)$: edge (u, v, l) connects u and v in layer $l \in L$, often modelled as a multigraph
- Each layer is a simple graph, e.g., friendship on Facebook, Twitter and LinkedIn
- If there are both weights and layers: (u, v, l, w)

Multilayer networks

- Multilayer network $G = (V, E, L)$: edge (u, v, l) connects u and v in layer $l \in L$, often modelled as a multigraph
- Each layer is a simple graph, e.g., friendship on Facebook, Twitter and LinkedIn
- If there are both weights and layers: (u, v, l, w)
- **Multiplex network**: every node is the same entity in every layer (one-to-one mapping)

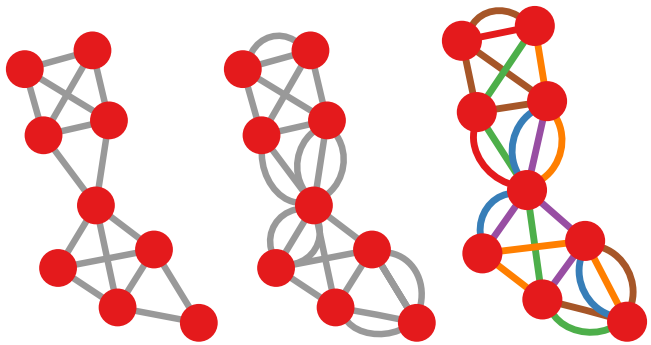


Figure: A simple graph, a multigraph and a multilayer network, where the edge color is the layer.

Layer relevance

- $N(u, l)$: neighbors of u in layer l
- **Exclusive neighbors** $N^{XOR}(u, l)$: neighbors connected to u **only** in layer l; lost if layer l disappears
- **Layer relevance**: $|N(u, l)|/|N(u)|$
- Exclusive relevance: $|N^{XOR}(u, l)|/|N(u)|$

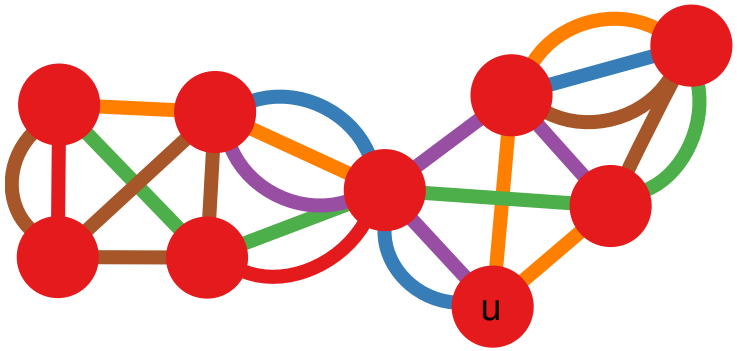


Figure: A multiplex network as a multigraph.

Layer relevance

- $N(u, l)$: neighbors of u in layer l
- **Exclusive neighbors** $N^{XOR}(u, l)$: neighbors connected to u **only** in layer l; lost if layer l disappears
- **Layer relevance**: $|N(u, l)|/|N(u)|$
- Exclusive relevance: $|N^{XOR}(u, l)|/|N(u)|$
- Example: $deg(u) = 4$ and $|N(u)| = 3$
  - Orange: relevance $2/3$, exclusive relevance $2/3$
  - Blue and purple: relevance $1/3$, exclusive relevance 0

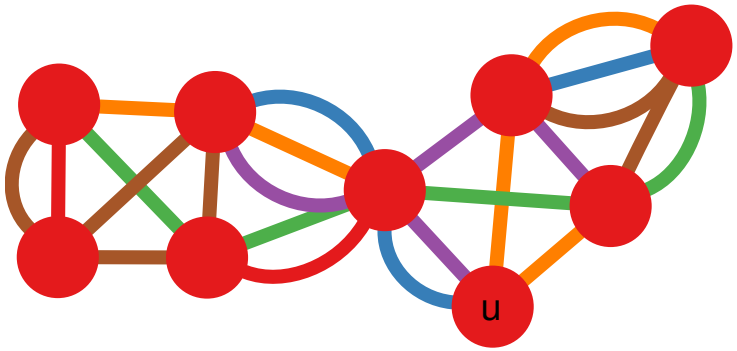


Figure: A multiplex network as a multigraph.

Multilayer density

- Is a group dense if it is connected in **all** layers, or in **any** layer?
- **Redundancy** of a group of nodes c (e.g., a community) with node pairs P_c: share of node pairs and layers that are connected

$$\rho_c = \frac{\sum_{(u,v)\in P_c} |\{l : (u, v, l) \in E\}|}{|L| \cdot |P_c|}$$

- **Complementarity**: many layers, mostly one layer per node pair, edges spread evenly over layers

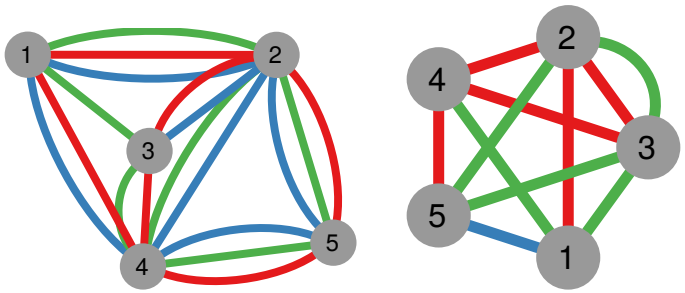


Figure: Three layers: (a) high redundancy, $\rho_c = 18/(3 \cdot 10) = 0.6$, (b) high complementarity.

The anatomy of a population-scale social network

E. Bokányi, E.M. Heemskerk and F.W. Takes, The Anatomy of a Population-Scale Social Network, *Scientific Reports* 13: 9209, Nature Publishing Group, 2023.

A population-scale social network

- Nodes: all 17.2 million people registered in the Netherlands (2018), from the registers of Statistics Netherlands
- Edges: 827 million edges in layers, being family, household, work, school and next-door neighbors
- A multiplex network of formal ties: the social opportunity structure of an entire population

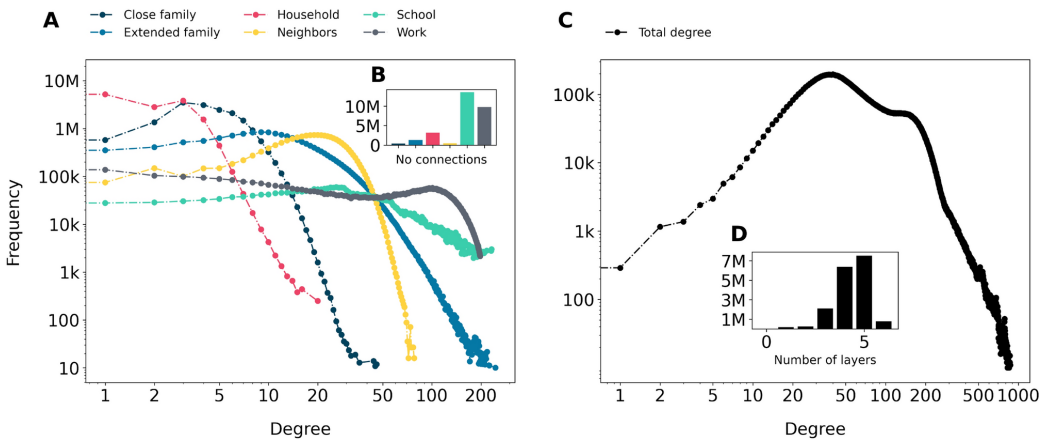


Figure: Degree distribution per layer (A), people without connections per layer (B), total degree (C), and number of layers with connections (D).

Ego network population-scale social network

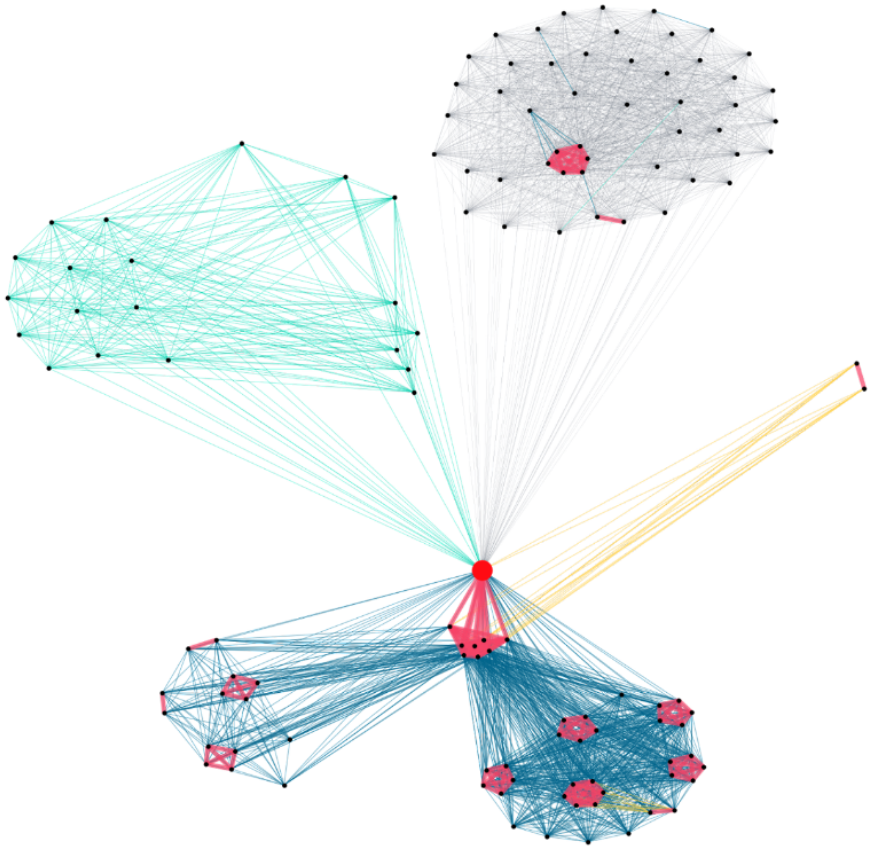


Figure: Ego network of one person (large red node) with its family, school, household, next-door neighbor and work layers.

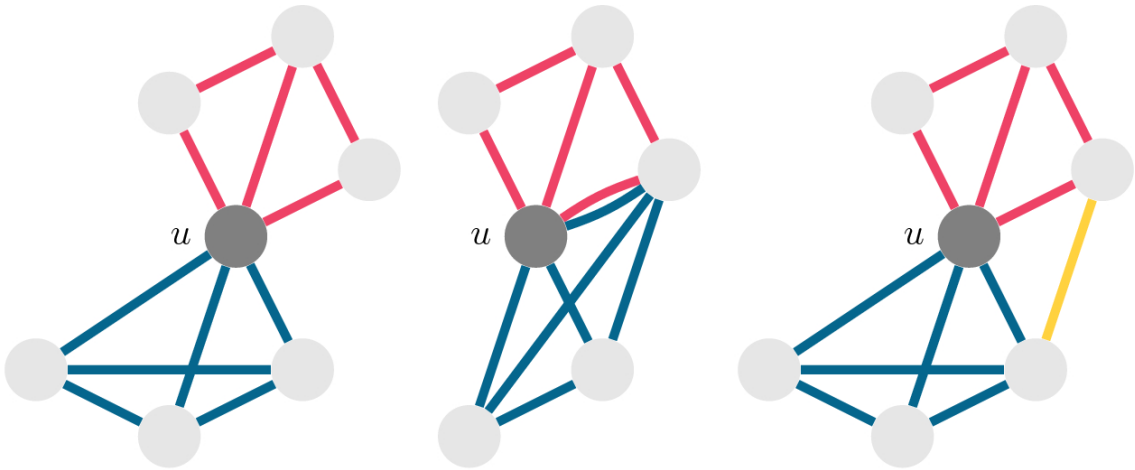
Excess closure

- The regular clustering coefficient is very high (average 0.40): work, school and household layers are projections (cliques), families are clustered by nature
- **Excess closure**: closure beyond the triangles inside single layers

$$c_{excess}(u) = \frac{c_{actual}(u) - c_{pure}(u)}{1 - c_{pure}(u)}$$

- $c_{pure}(u)$: share of tie pairs of u closed within one layer
- $c_{actual}(u)$: share closed in any way

Excess closure: examples



T_pure_u	5	5	5
T_unique_u	5	8	6
P_u	$\binom{6}{2} = 15$	$\binom{6}{2} - 1 = 14$	$\binom{6}{2} = 15$
$c_pure_u = \frac{T_pure_u}{P_u}$	$\frac{5}{15}$	$\frac{5}{14}$	$\frac{5}{15}$
$c_unique_u = \frac{T_unique_u}{P_u}$	$\frac{5}{15}$	$\frac{8}{14}$	$\frac{6}{15}$
$c_excess_u = \frac{c_unique_u - c_pure_u}{1 - c_pure_u}$	0	$\frac{1}{3}$	$\frac{1}{10}$

Clustering coefficient versus excess closure

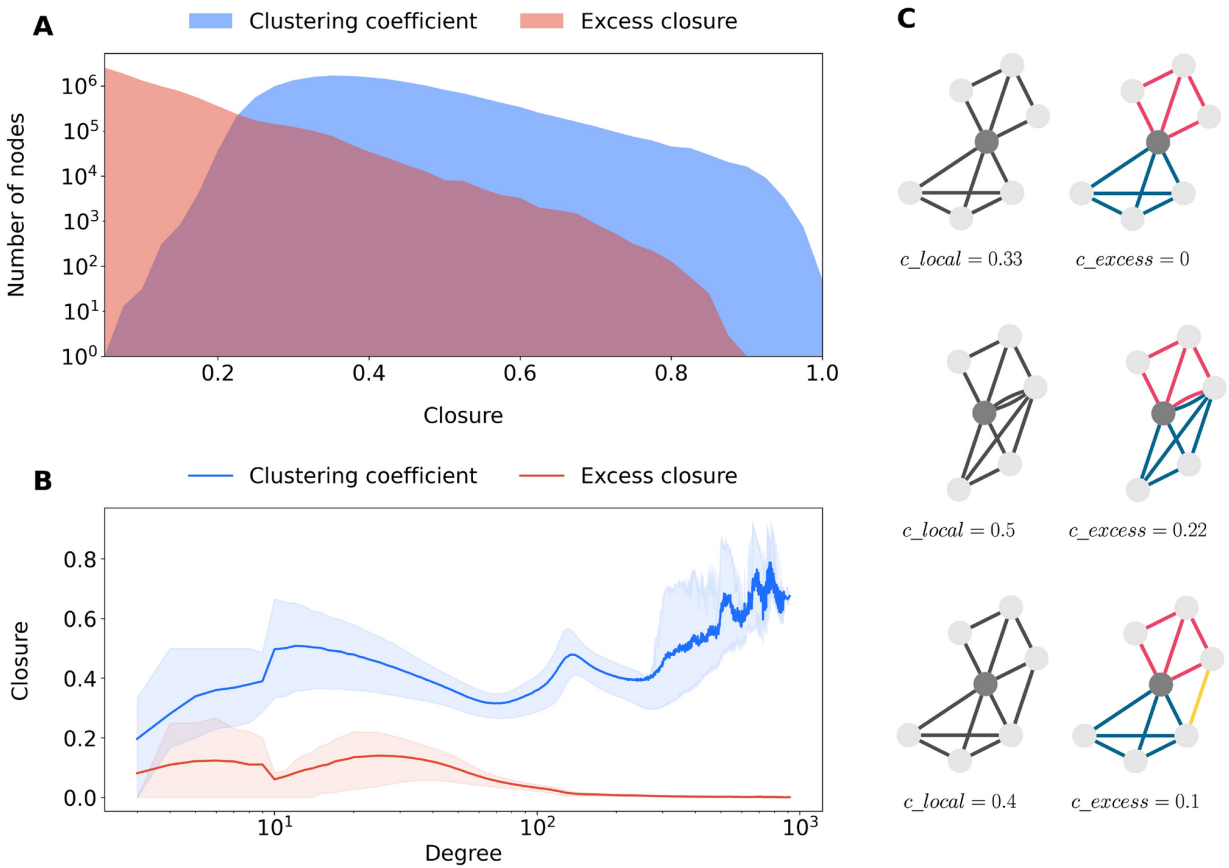


Figure: Clustering coefficient and excess closure: distribution (A), average per degree (B), and examples (C).

Network position over the life course

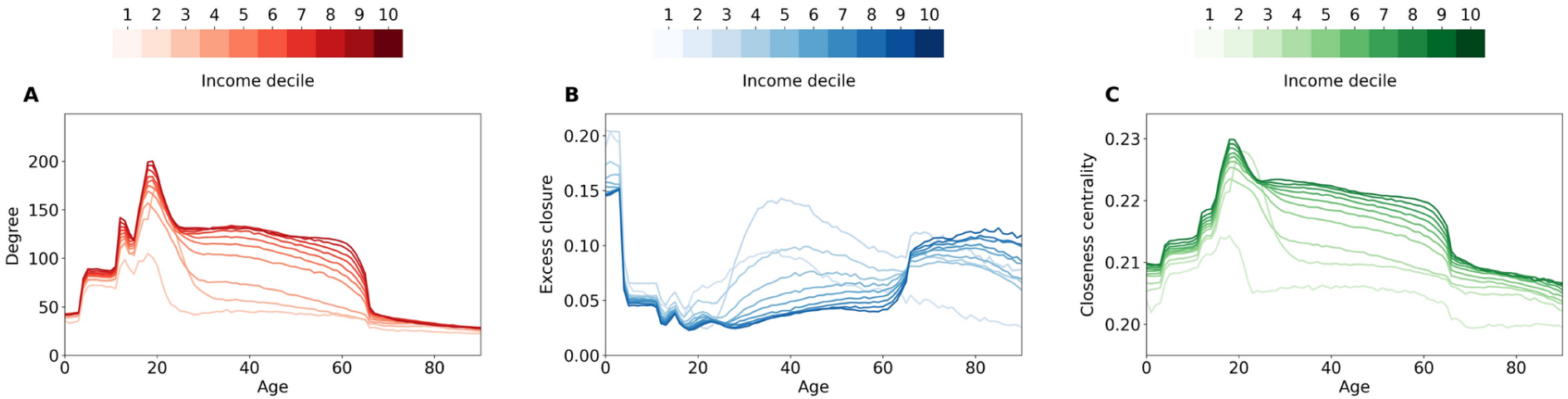


Figure: Average degree (A), excess closure (B) and closeness centrality (C) by age, per income decile (darker: higher income).

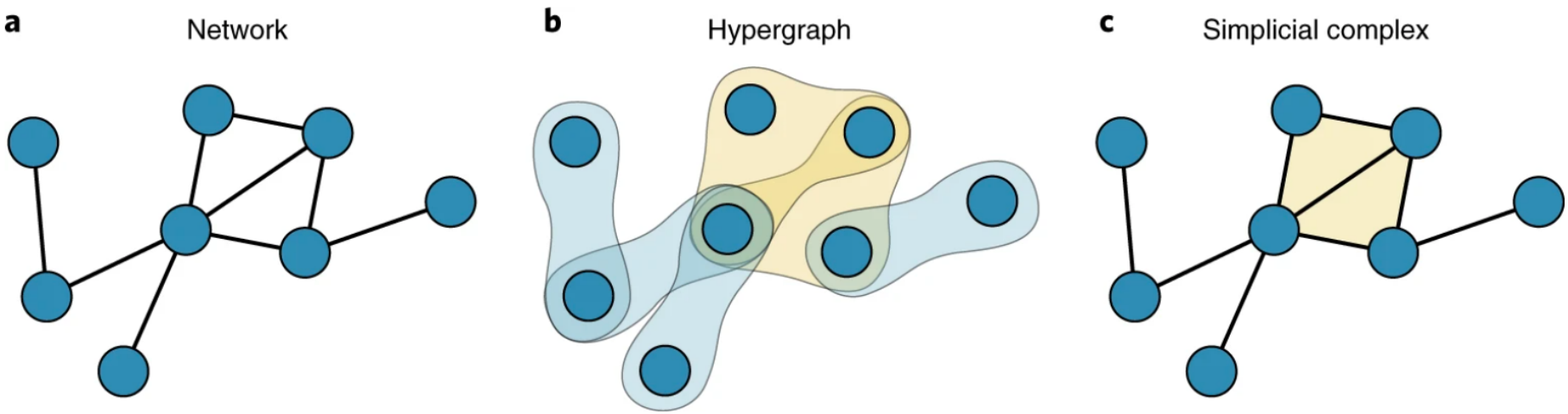
- School, work and retirement shape the network position of people
- Lower incomes: lower degree and closeness, but higher excess closure in midlife

Higher-order networks

Higher-order networks: terminology

- "Higher-order network" is used for two different things:
  1. **Group interactions**: ties between more than two nodes at once
     - Hypergraph: ties of any size; the **order** of a tie is its number of nodes
     - Simplicial complex: a hypergraph in which every subset of a tie is also a tie
  2. **Memory**: the next step of a flow depends on the previous steps; the order is the length of the memory
- To make it more confusing: in graph theory, the order of a graph is its number of nodes
- Today: first group interactions, then memory

Higher-order networks / Simplicial complexes



Battiston et al. "The physics of higher-order interactions in complex systems." Nature Physics 17 (2021): 1093–1098.

Hypergraphs

- Some relations connect more than two nodes at once: project teams, co-authors of a paper, group chats

- **Hyperedge**: edge that connects any number of nodes; a **hypergraph** is a set of nodes and hyperedges
- **Uniform hypergraph**: all hyperedges have the same size, e.g., soccer teams

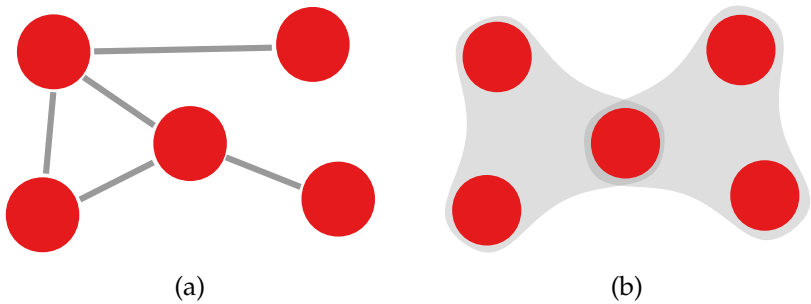


Figure: (a) A graph. (b) A hypergraph with one hyperedge.

Hypergraphs

- Some relations connect more than two nodes at once: project teams, co-authors of a paper, group chats

- **Hyperedge**: edge that connects any number of nodes; a **hypergraph** is a set of nodes and hyperedges
- **Uniform hypergraph**: all hyperedges have the same size, e.g., soccer teams
- Degree: the number of hyperedges a node is part of (unrelated to its number of neighbors)
- Average degree is no longer $2|E|/|V|$: one hyperedge can contain all nodes

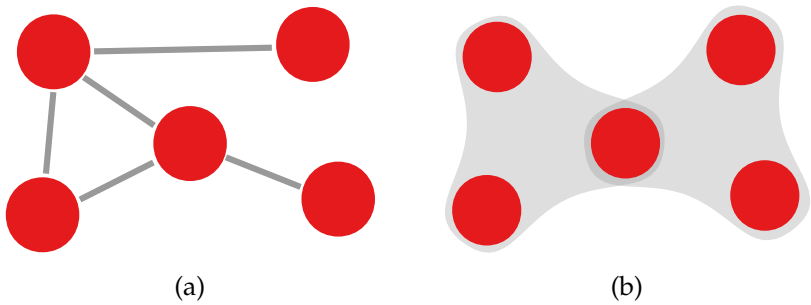


Figure: (a) A graph. (b) A hypergraph with one hyperedge.

Simplifying hypergraphs

- Algorithms for hypergraphs get complicated quickly, so hypergraphs are often simplified
- **Clique in undirected network**: connect all nodes of a hyperedge
- **Bipartite network**: one node per hyperedge, connected to its members

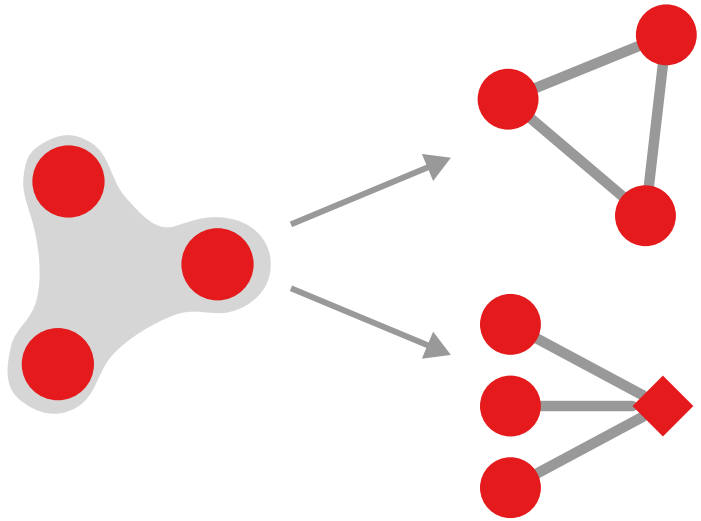


Figure: A hyperedge as a triangle (top) or as a bipartite network (bottom).

Simplicial complexes

- **Simplex**: an n-simplex connects $n+1$ nodes: node (0), edge (1), filled triangle (2), filled tetrahedron (3)
- An n-simplex contains all its **faces**, the smaller simplices of its nodes; a hyperedge does not contain its subsets
- **Simplicial complex**: a network with simplices; its dimension is that of its largest simplex

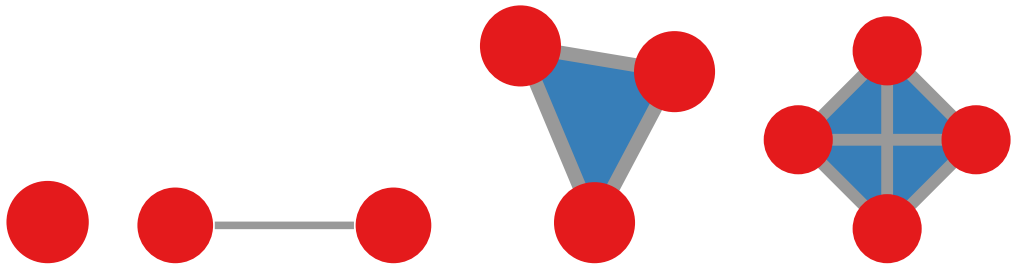


Figure: The smallest simplices: (a) 0-simplex, (b) 1-simplex, (c) 2-simplex, (d) 3-simplex.

The relevance of higher-order ties

A. Ceria and F.W. Takes, The Relevance of Higher-Order Ties, *EPJ Data Science* 14: 62, 2025.

The relevance of higher-order ties: data

- The **order** d of a hyperedge is the number of nodes it connects
- In a weighted projection of a hypergraph, weight is the number of hyperedges two nodes share. But does each order matter equally much?
- **Board interlock network** (Orbis, 2017): nodes are companies, a hyperedge is a director connecting the 2 to 50 companies on whose boards they sit
- **Scientific collaboration network** (Web of Science, 2008–2023): nodes are authors, a hyperedge is a publication with 2 to 25 authors
- Node label: the country of the company or of the author's affiliation

Network	Nodes	Hyperedges	Links	Largest comp.
Board interlock	38,305,725	13,338,023	176,789,812	16,124,492
Collaboration	30,861,046	20,067,279	189,969,041	28,203,800

The relevance of higher-order ties: method

- Contribution of small and large hyperedges to a measure M of the projection: link weights, triangles, **largest component size**
- **Order contribution** $M^-(d)$: value of M using only hyperedges of order at most d
- Complementary: $\overline{M^-(d)} = M_{max} - M^+(d)$, with $M^+(d)$ using only orders above d

The relevance of higher-order ties: method

- Contribution of small and large hyperedges to a measure M of the projection: link weights, triangles, **largest component size**
- **Order contribution** $M^-(d)$: value of M using only hyperedges of order at most d
- Complementary: $\overline{M^-(d)} = M_{max} - M^+(d)$, with $M^+(d)$ using only orders above d
- **Order relevance**: close to 1 if small orders matter, close to 0 if large orders matter

$$\Gamma_H(M) = \frac{\sum_{d=d_{min}}^{d_{max}} M^-(d)/M_{max} - 1}{d_{max} - d_{min}}$$

The relevance of higher-order ties: method

- Contribution of small and large hyperedges to a measure M of the projection: link weights, triangles, **largest component size**
- **Order contribution** $M^-(d)$: value of M using only hyperedges of order at most d
- Complementary: $\overline{M^-(d)} = M_{max} - M^+(d)$, with $M^+(d)$ using only orders above d
- **Order relevance**: close to 1 if small orders matter, close to 0 if large orders matter

$$\Gamma_H(M) = \frac{\sum_{d=d_{min}}^{d_{max}} M^-(d)/M_{max} - 1}{d_{max} - d_{min}}$$

- Gap $\Delta_H(M) = \Gamma_H(M) - \overline{\Gamma_H(M)}$: below 0, orders are **synergistic**; above 0, **redundant**
- Triangles come mostly from large hyperedges, as each becomes a clique: take clustering in projected networks with a grain of salt

The relevance of higher-order ties: largest component

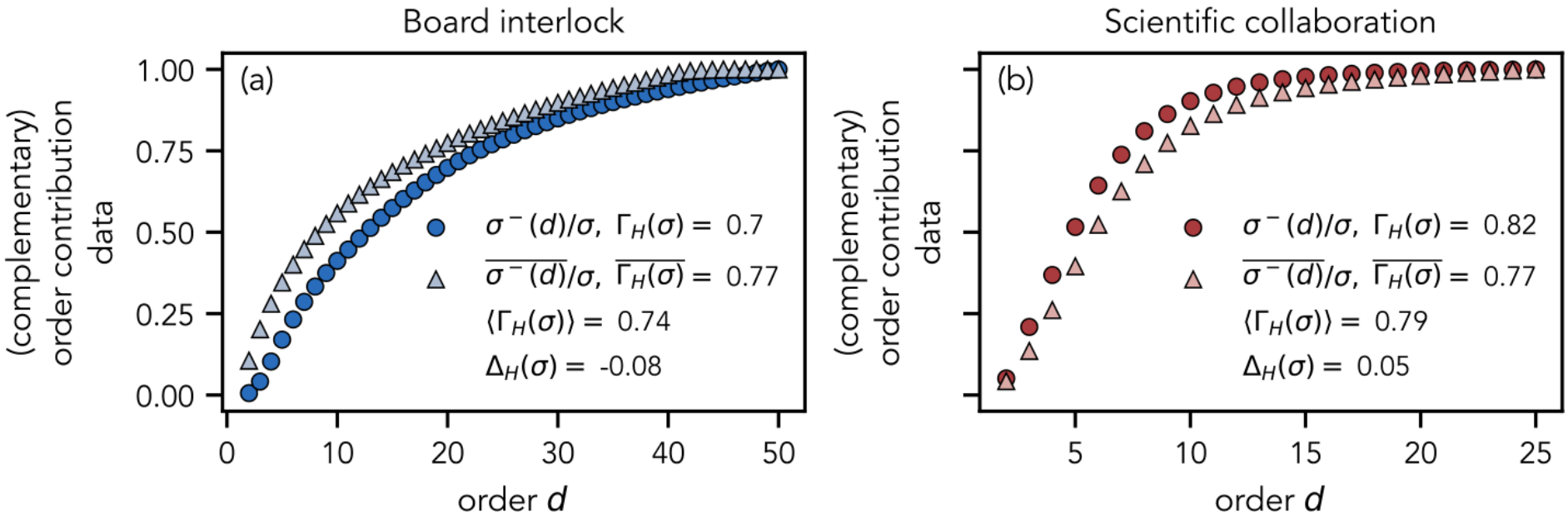


Figure: Order contribution (circles) and complementary order contribution (triangles) to the size of the largest connected component, as a function of the order d (horizontal axis). Board interlock networks (a): directors with few and with many board seats reinforce each other in connecting companies (**synergy**). Collaborations (b): the circles lie above the triangles ($\Delta > 0$), so small and large author teams largely connect the same researchers (**redundancy**).

Memory higher-order network

- Air travel: from the airport network you only know where a passenger can fly next
- But the next step depends on where the passenger **came from**! So we need **memory**
- Random walks so far are memoryless: the next step only depends on the current node

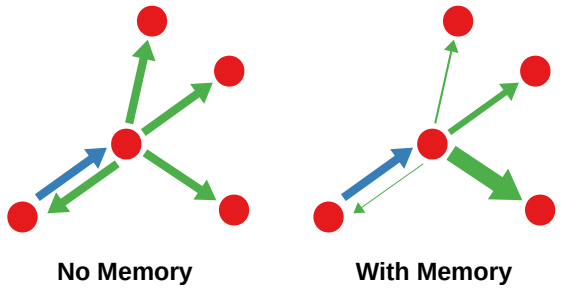


Figure: After a first step (blue), all second steps (green) are equally likely without memory, but not with memory.

Higher-order network

- **Higher-order network**: split a node into conditional nodes, one per origin of the flow, e.g., for an airport network NY|BOS and NY|LON
- On this network, run any standard algorithm, such as PageRank or Infomap, and aggregate back to the original node
- Catch: the network grows quickly and can fall apart into components

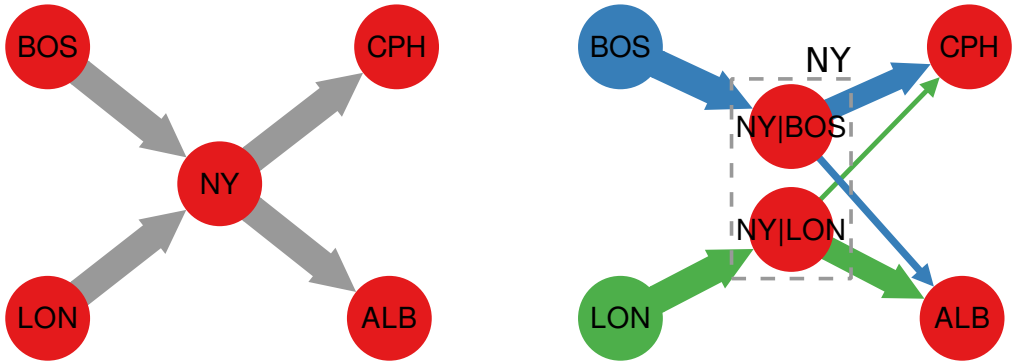


Figure: (a) A directed air travel network. (b) Its higher-order version, with conditional nodes for flows into New York from Boston and from London.

Memory: more examples

- **Global shipping**: where a cargo ship goes next depends on the previous port, since ships follow fixed routes
  - A higher-order network better predicts the spread of invasive species in ballast water
- **Web browsing**: the next page a user clicks depends on the previous pages
- **Patients**: the next hospital ward depends on the ward the patient came from
- In all cases, a memoryless random walk takes paths that real flows rarely take

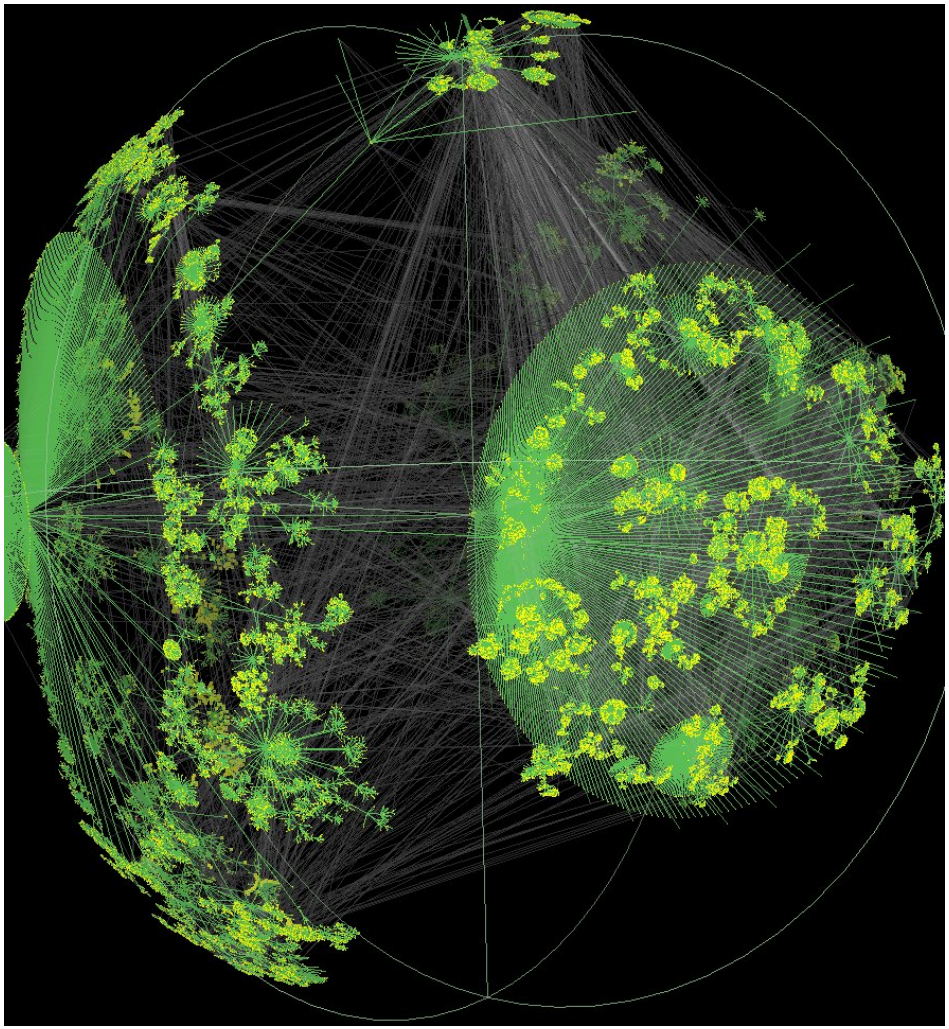
J. Xu, T.L. Wickramarathne and N.V. Chawla, Representing higher-order dependencies in networks, *Science Advances* 2(5): e1600028, 2016.

Network visualization

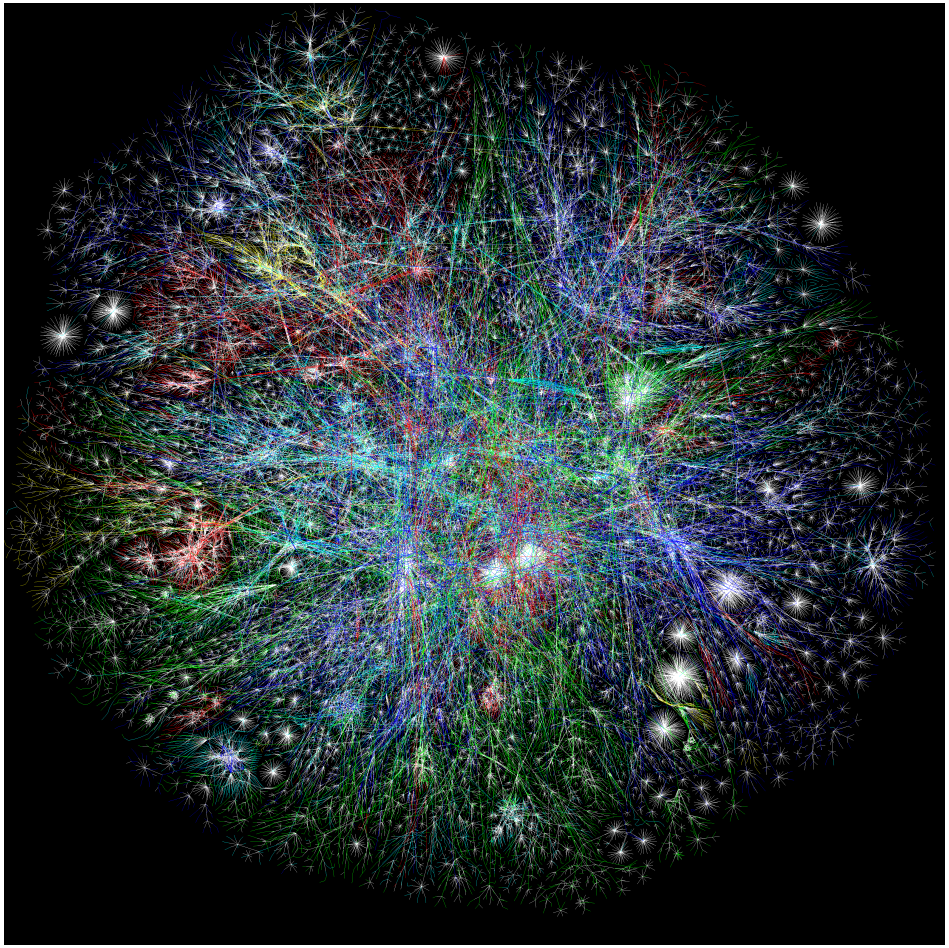
Network visualization: overview

1. Network layouts
2. Force-directed layouts: Fruchterman–Reingold
3. Multilevel layouts, the grid variant and the Barnes–Hut approximation
4. ForceAtlas2
5. Other layouts
6. GPU acceleration

Webgraph: $\sim$500,000 nodes



Webgraph: $\sim$500,000 nodes



Opte, Internet visualization (2005)

Network layout

- **Network layout algorithms** map nodes to coordinates in the plane
- In a network only the **relative** positions matter
- So we may move nodes around, as long as the same rules apply to all nodes

What makes a good layout?

- Readability: few edge crossings, uniform edge lengths, no overlapping nodes, nodes spread evenly, symmetry visible
- The layout influences the interpretation of the network
- What does the right drawing suggest about node 0?

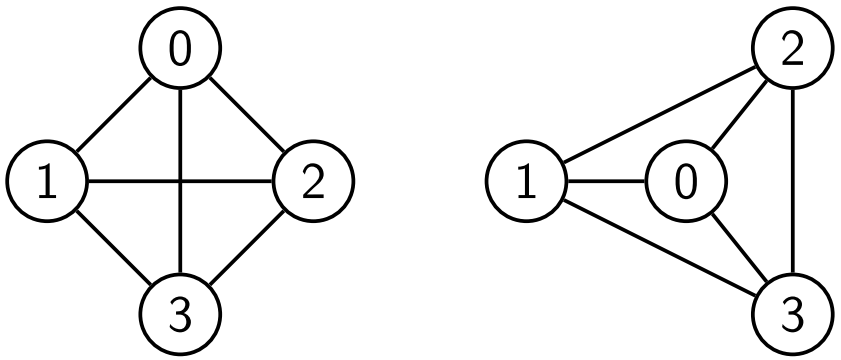


Figure: Two layouts of the complete graph K_4.

What makes a good layout?

- Readability: few edge crossings, uniform edge lengths, no overlapping nodes, nodes spread evenly, symmetry visible
- The layout influences the interpretation of the network
- What does the right drawing suggest about node 0?
- Node 0 looks central, but all nodes of K_4 are equivalent: the left drawing is more truthful

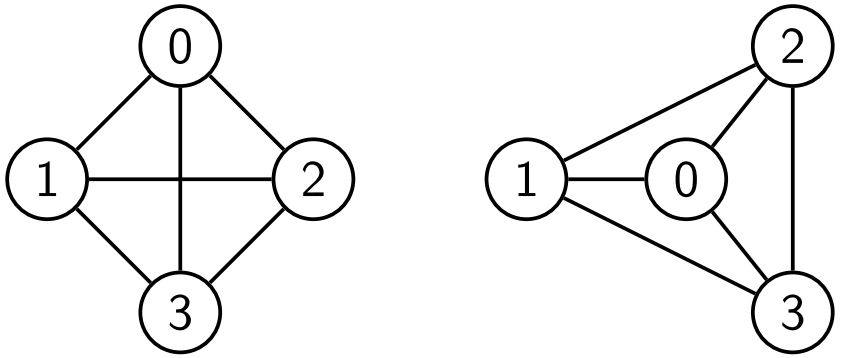


Figure: Two layouts of the complete graph K_4.

Random versus force-directed layout

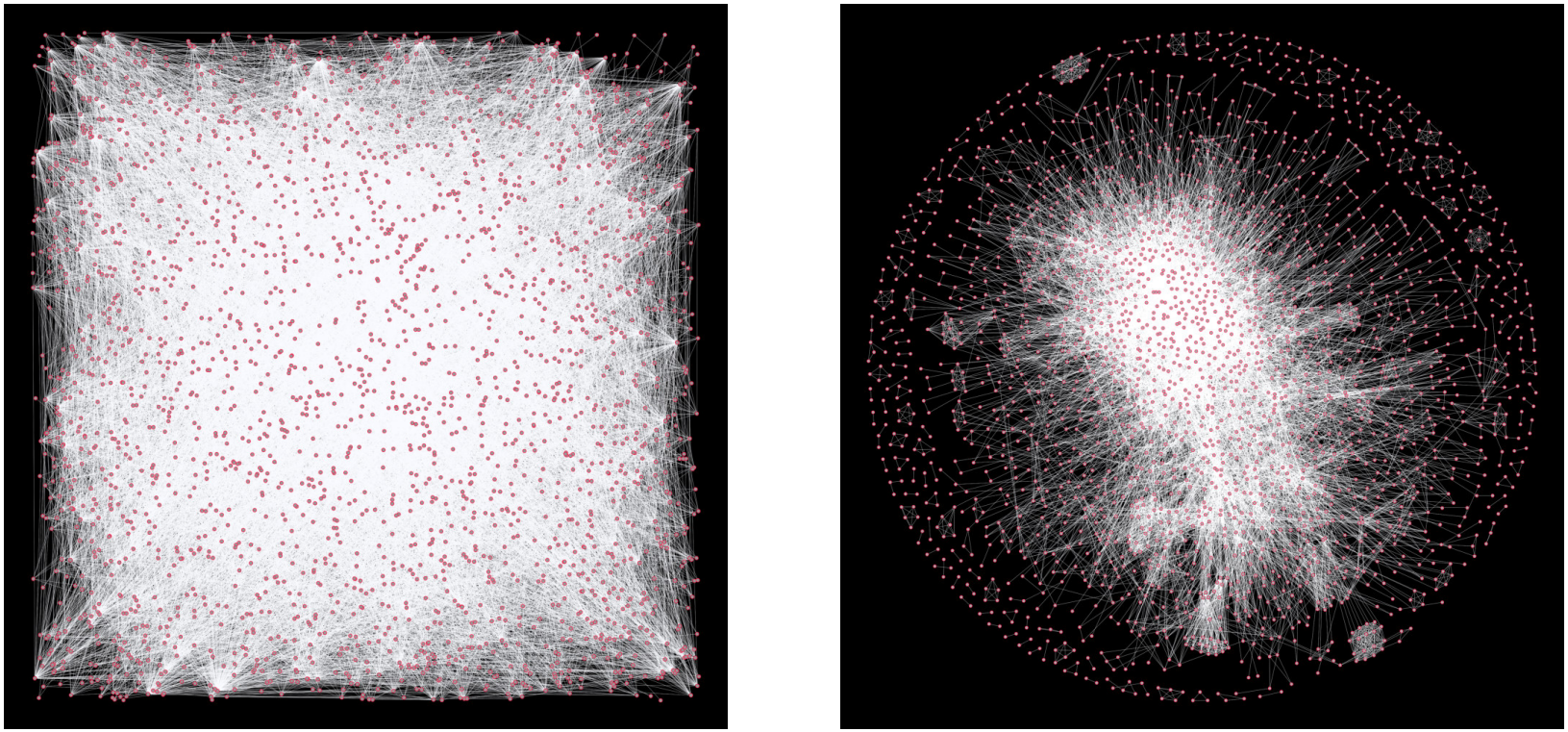


Figure: The Hamsterster network ($n = 2{,}426$, $m = 16{,}631$) with a random layout (left) and a force-directed Fruchterman–Reingold layout (right).

Images: G. Brinkmann

Force-directed layout

- **Force-directed layout**: nodes are particles with the same charge that repel each other, edges are springs that pull connected nodes together
- Start from random positions and iterate: compute the net force on every node, move every node a bit
- Stop when the system reaches a state of low energy, or after a fixed number of iterations
- First with real springs (Eades, 1984); later methods use artificial forces that just give readable layouts

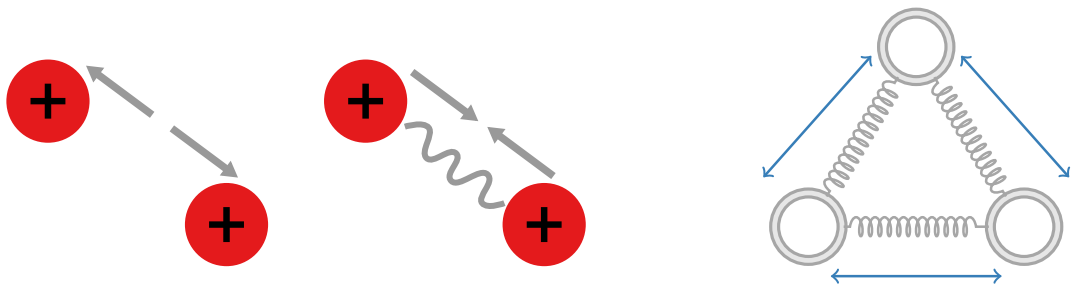


Figure: Charged particles repel, springs attract (left); rings and springs (right, after G. Brinkmann).

Fruchterman–Reingold

- Ideal edge length $k = C\sqrt{area/n}$; d is the distance between two nodes in the drawing
- **Attraction** between neighbors: $f_a(d) = d^2/k$
- **Repulsion** between all pairs: $f_r(d) = -k^2/d$

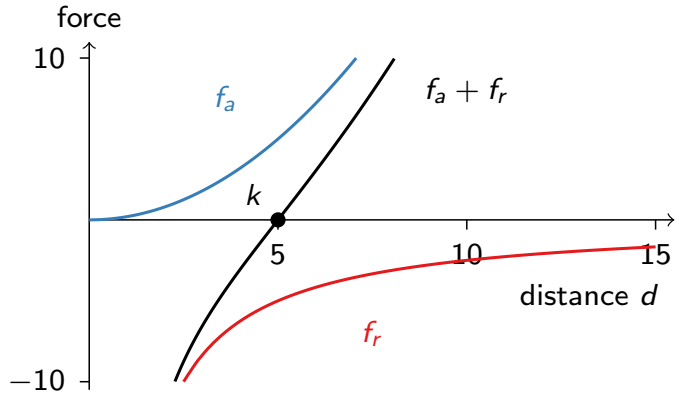


Figure: Forces for $k = 5$; they cancel at $d = k$ (after G. Brinkmann).

Fruchterman–Reingold

- Ideal edge length $k = C\sqrt{area/n}$; d is the distance between two nodes in the drawing
- **Attraction** between neighbors: $f_a(d) = d^2/k$
- **Repulsion** between all pairs: $f_r(d) = -k^2/d$

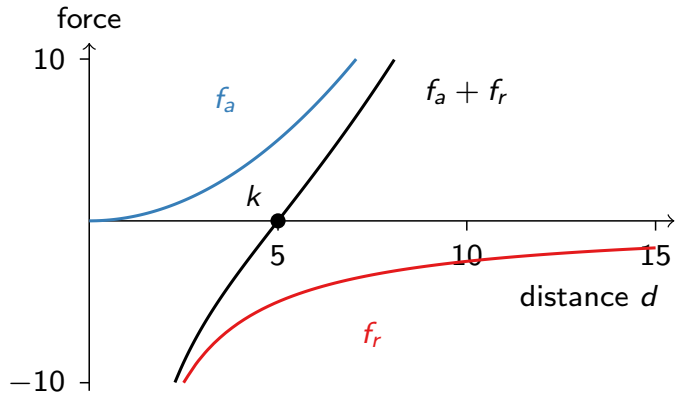


Figure: Forces for $k = 5$; they cancel at $d = k$ (after G. Brinkmann).

- Each iteration:
  1. Repulsive forces between all node pairs: $\Theta(n^2)$
  2. Attractive forces along all edges: $\Theta(m)$
  3. Move each node in the direction of its net force, at most a distance equal to the **temperature**
  4. Lower the temperature (cooling)

T.M.J. Fruchterman and E.M. Reingold, Graph Drawing by Force-directed Placement, *Software: Practice and Experience* 21(11): 1129–1164, 1991.

Local minima and multilevel layouts

- Large networks have many **local minima**: the layout gets stuck, e.g., with a node on the wrong side of a cluster
- A high initial temperature lets nodes escape; cooling too fast freezes a bad layout (simulated annealing)

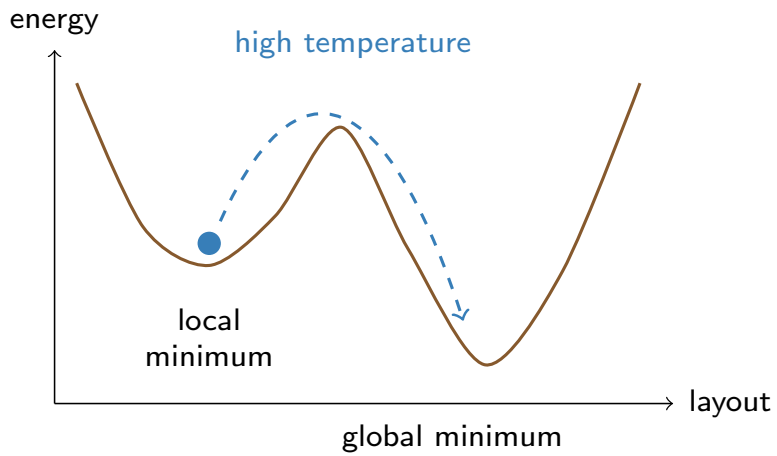


Figure: A node stuck in a local minimum of the energy can escape at a high temperature (after G. Brinkmann).

Local minima and multilevel layouts

- Large networks have many **local minima**: the layout gets stuck, e.g., with a node on the wrong side of a cluster
- A high initial temperature lets nodes escape; cooling too fast freezes a bad layout (simulated annealing)
- **Multilevel layout**: coarsen the network by merging nodes, lay out the small network, then refine it level by level
- Same idea as the coarsening in Louvain (lecture 4); used in, e.g., OpenOrd

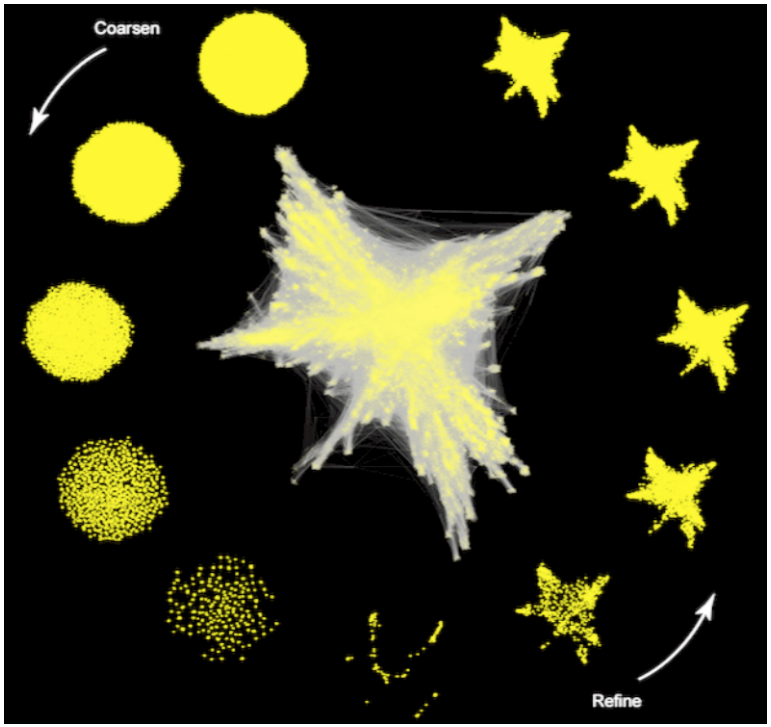


Figure: Multilevel layout: coarsen the network, lay out the smallest version, and refine it again.

Speeding up repulsion

- Repulsion between all pairs costs $O(n^2)$ per iteration: infeasible for millions of nodes
- **Grid variant**: cells of size $2k$; only nodes in neighboring cells within distance $2k$ repel each other

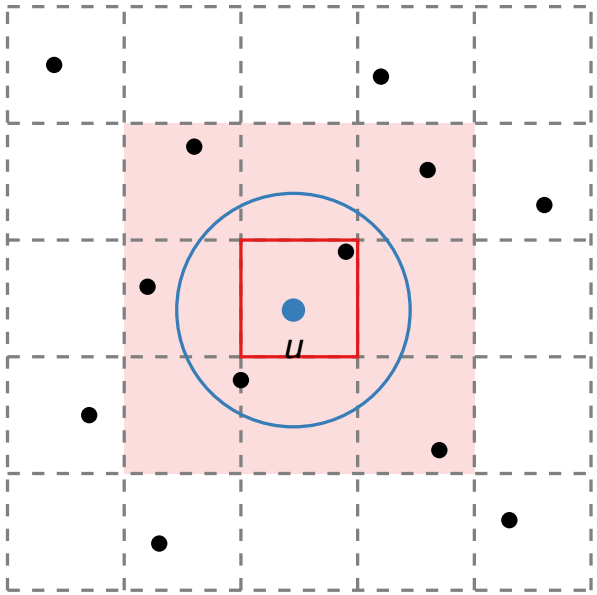


Figure: Grid variant: only the nodes in the shaded cells and inside the circle repel u (after G. Brinkmann).

Speeding up repulsion

- Repulsion between all pairs costs $O(n^2)$ per iteration: infeasible for millions of nodes
- **Grid variant**: cells of size $2k$; only nodes in neighboring cells within distance $2k$ repel each other
- **Barnes–Hut approximation**: recursively split the drawing into a quadtree, until each cell holds at most one node
- A cell that is far away acts as one body at its center of mass; parameter θ sets the accuracy
- Repulsion in $O(n \log n)$ per iteration

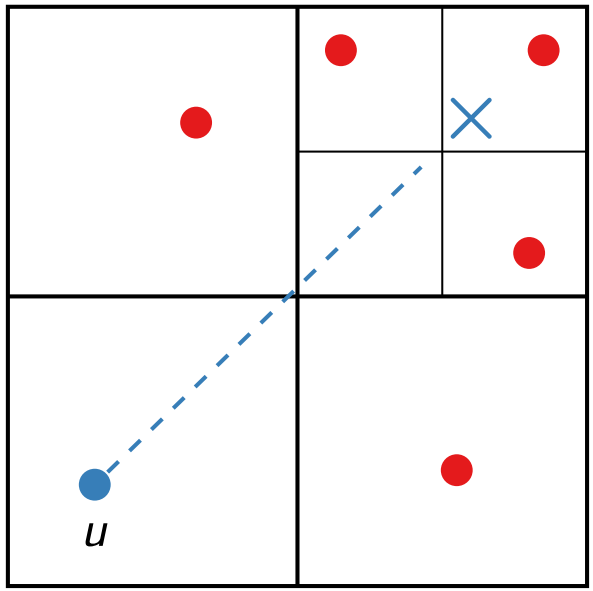


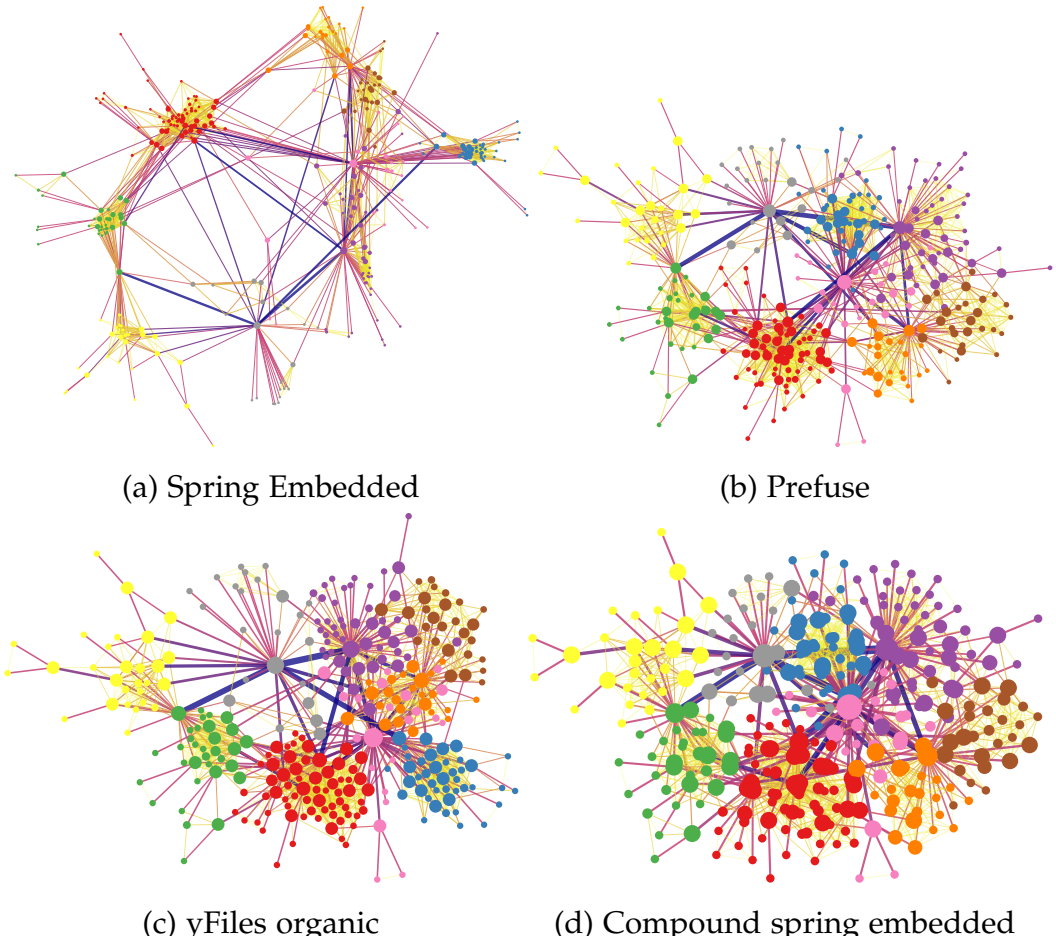
Figure: The three nodes in the top right cell are far from u, so they repel u as one body at their center of mass ($\times$).

ForceAtlas2

- Force-directed layout of Gephi, designed for social networks
- Attraction between neighbors u and v grows linearly with their distance d: $F_a = d$
- **Degree-dependent repulsion**: $F_r = k_r \frac{(deg(u)+1)\cdot(deg(v)+1)}{d}$, so hubs push each other away and low-degree nodes end up around their hubs
- **Gravity** towards the center, $F_g = k_g(deg(u)+1)$, keeps disconnected components in view
- **Adaptive speed**: nodes that oscillate (swing) between iterations move slower
- Barnes–Hut for the repulsion; LinLog mode (logarithmic attraction) gives tighter clusters

M. Jacomy, T. Venturini, S. Heymann and M. Bastian, ForceAtlas2, a Continuous Graph Layout Algorithm for Handy Network Visualization Designed for the Gephi Software, *PLoS ONE* 9(6): e98679, 2014.

Flavors of force-directed layouts



(a) Spring Embedded

(b) Prefuse

(c) yFiles organic

(d) Compound spring embedded

Figure: Same network, 4 algorithms; node color is community, node size is degree.

Edge bundling and ghost edges

- **Edge bundling**: bend edges between the same regions of the drawing into bundles, reducing clutter
- **Ghost edge**: in a 2D drawing, a node can end up on top of an edge it is not part of, suggesting connections that do not exist
- Edge routing that bends edges around nodes avoids this

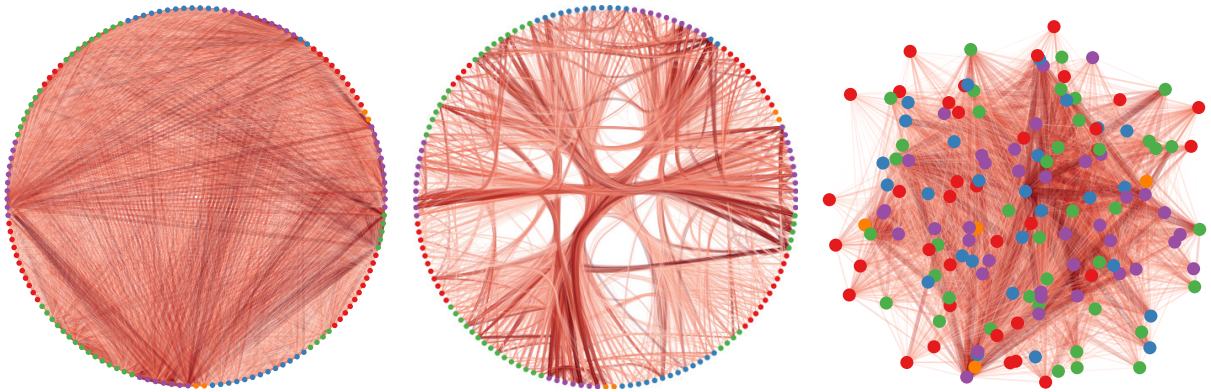


Figure: Circular layout without and with edge bundling, and a force-directed layout (same data).

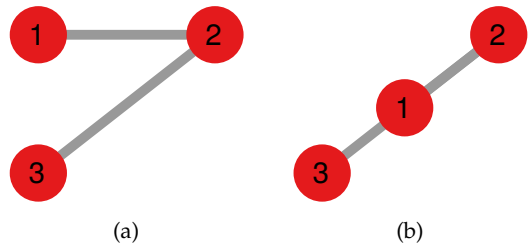


Figure: (a) A path $1, 2, 3$. (b) Drawn such that 1 seems connected to 2 and 3.

Layouts of networks with millions of nodes

- Idea: force computations per node are independent, so can be done **in parallel** on a GPU
- ForceAtlas2 for networks with millions of nodes and edges
- Average speedup of 45× over a sequential CPU implementation
- The Barnes–Hut approximation takes most of the time and limits the speedup

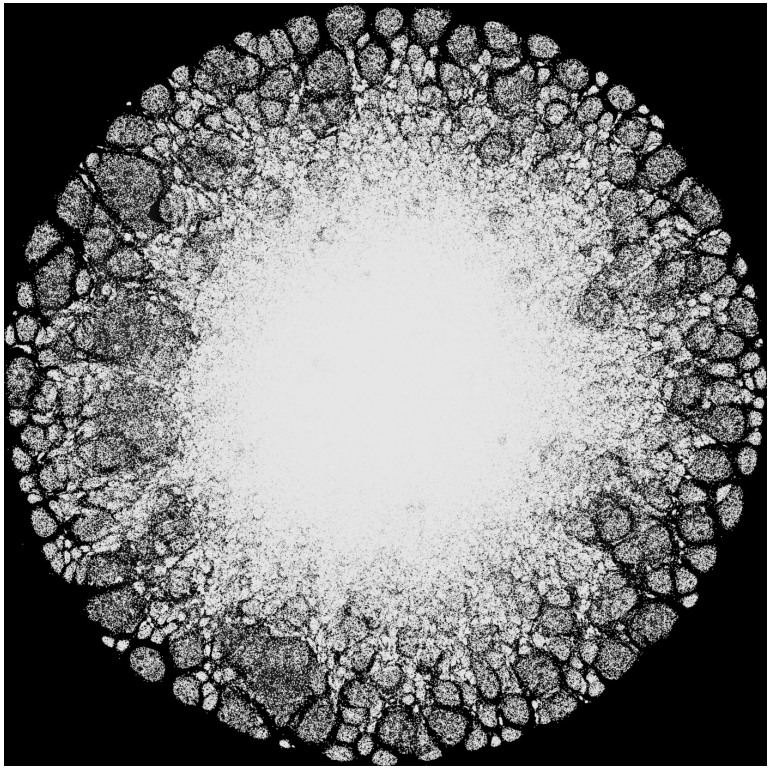


Figure: GPU ForceAtlas2 layout of a corporate network.

G.G. Brinkmann, K.F.D. Rietveld and F.W. Takes, Exploiting GPUs for Fast Force-Directed Visualization of Large-Scale Networks, in *Proceedings of the 46th International Conference on Parallel Processing (ICPP)*, IEEE, 2017.

What is left to do?

What is left to do?

- Study for the (practice) exam!
- Next two lab sessions are mere question hours.

Credits

- Several images are gratefully reused from
  Michele Coscia, The Atlas for the Aspiring Network Scientist, v3 of
  `https://arxiv.org/abs/2101.00863`
- Force-directed layout illustrations and Hamsterster layouts after the SNACS presentation by G. Brinkmann