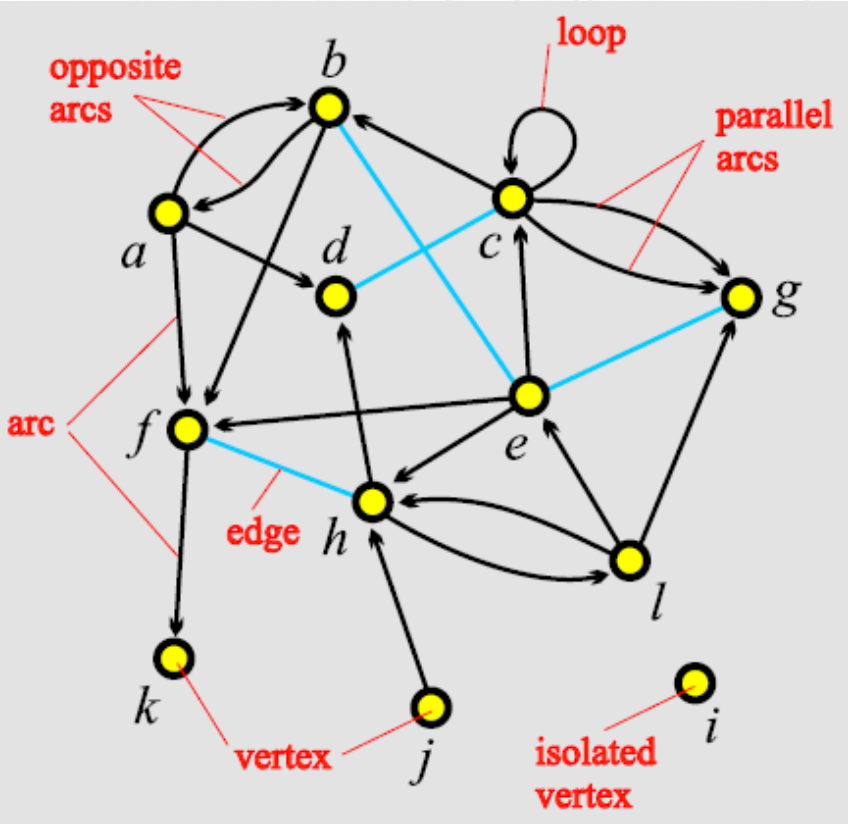
# *A short introduction to graph theory for SNA*

**Jožef Stefan International Postgraduate school**

Vid Podpečan, Jure Ferlež
{vid.podpecan, jure.ferlez} @ ijs.si

# Graphs




- node, vertex: actor / unit
- line, edge, arc: tie / link
- arc: directed line (j, h)
- edge: undirected (h : f)
- simple graph:
  - no loops
  - no parallel connections
- ego-centered networks:
  - impossible to measure the complete network
  - **only selected** nodes and their neighbours

# Formal notation

- network $N=(V,L,P,W)$
  - graph $N=(V,L)$
    - V = set of vertices
    - L = set of lines $L = \text{Arcs} \cup \text{Edges}$
  - P: vertex value functions
  - W: line value functions

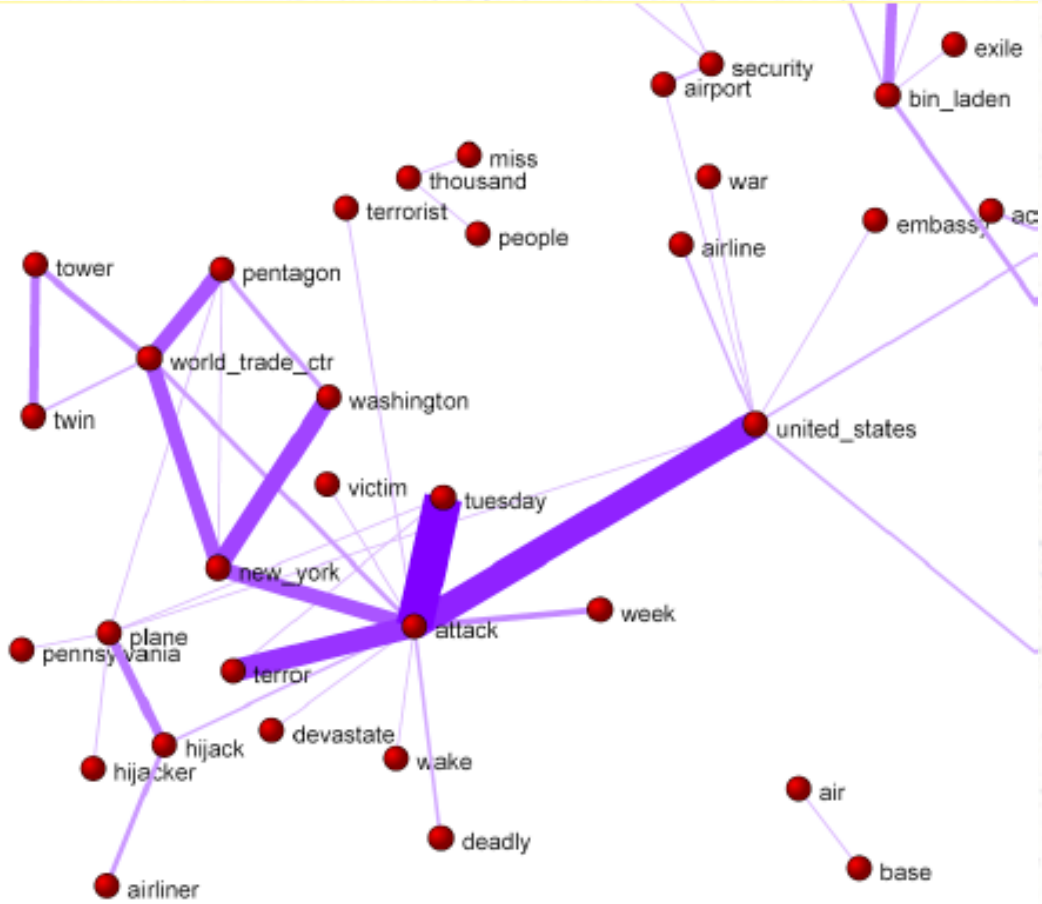
# Size of network

- n = |V|, m= |L|
- for simple graphs: $m \leq \frac{n*(n-1)}{2}$
- density: $\gamma = \frac{m}{m_{max}}$
- sparse networks: $m \ll n^2$
  - typically: $m = O(n)$ or $m = O(n*\log(n))$
- examples:
  - ODLIS dictionary: n = 2909, m = 18.419
  - IMDB: n = 1.324.748, m = 3.792.390
  - US patents: n = 3.774.768, m = 16.522.438
  - SI internet: n = 5.547.916, m = 62.259.968

# *Types of networks*

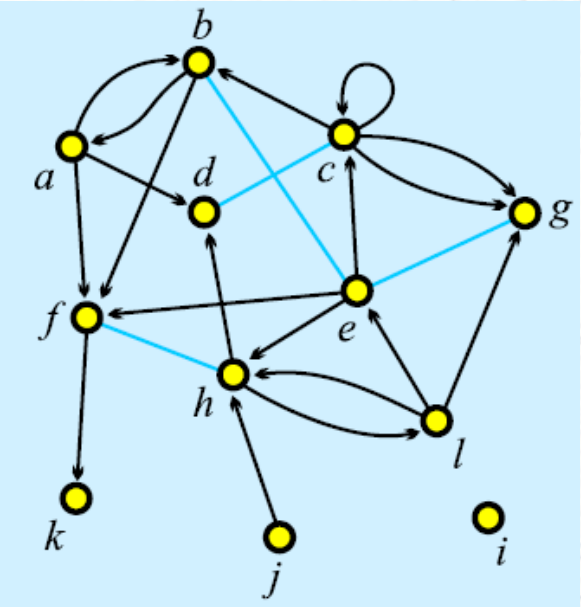
- k-mode:
  - between k disjoint sets of vertices
- multi-relational:
  - WordNet: semantic relations between words
    (synonymy, antonymy, hyponymy, ...)
- temporal:
  - dynamic, change over time (nodes, lines)
  - N(t),  t = time point

# Example: Terror news




- Reuters news for 66 days about September 11th
- temporal network of word coappearance

# Degrees

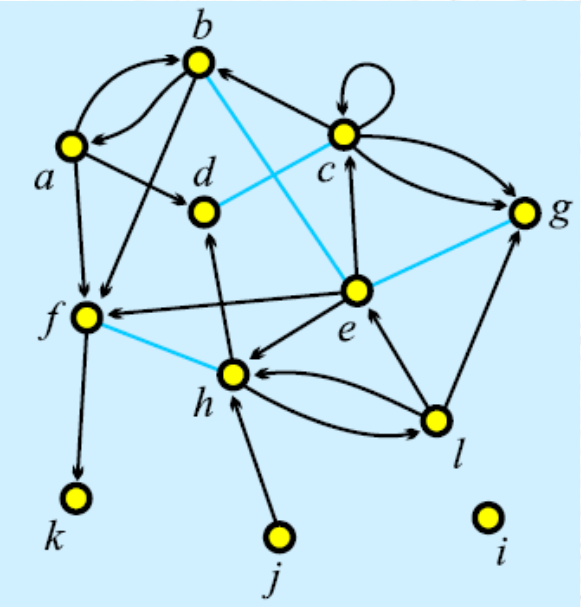



- degree of vertex v: deg(v) = number of lines with v as end-vertex
- indegree: indeg(v) = number of incomming lines
- outdegree: outdeg(v) = number of outgoing lines
- initial vertex: indeg(v) = 0
- terminal vertex: outdeg(v) = 0

Example:

deg(e) = ?,  indeg(e) = ?, outdeg(e) = ?
isolated vertex: ?
initial: ?, terminal: ?

# Degrees

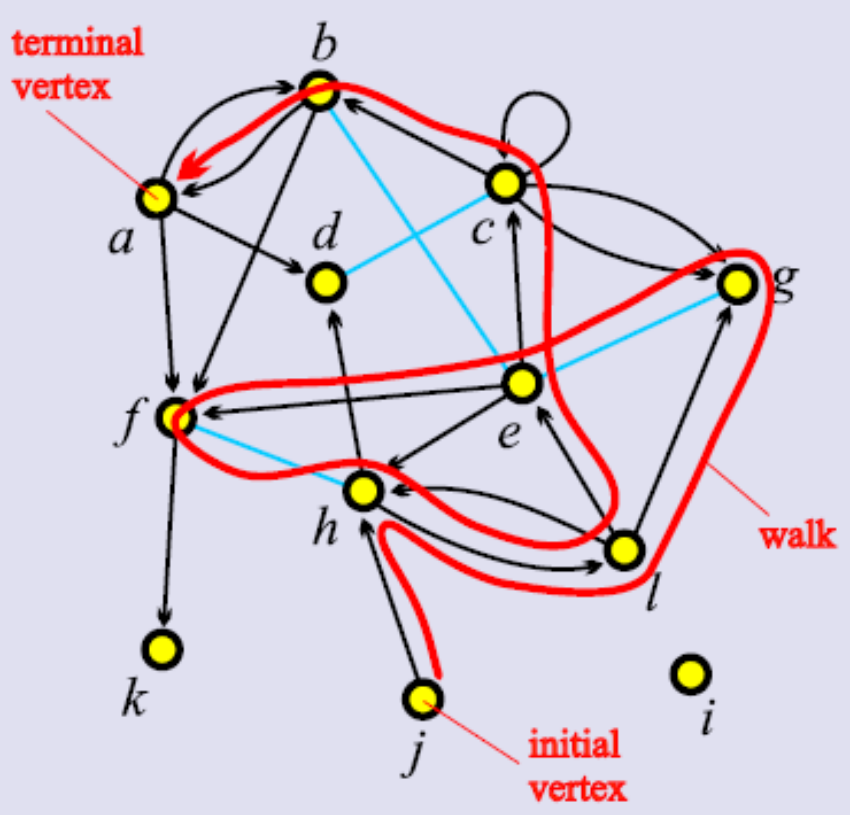



- degree of vertex v: deg(v) = number of lines with v as end-vertex
- indegree: indeg(v) = number of incomming lines
- outdegree: outdeg(v) = number of outgoing lines
- initial vertex: indeg(v) = 0
- terminal vertex: outdeg(v) = 0

Example:

deg(e) = **6**, indeg(e) = **3**, outdeg(e) = **5**
isolated vertex: **i**
initial: **j**, terminal: **k**

# Walks

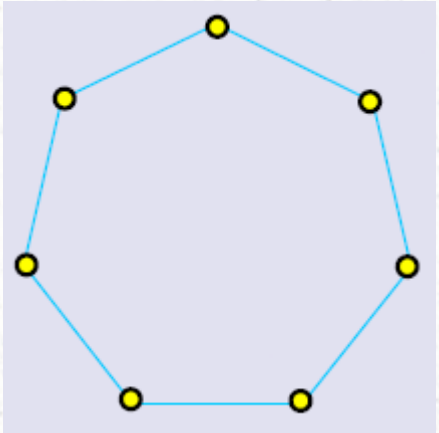
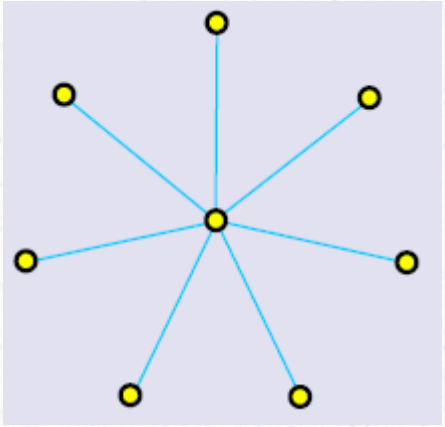
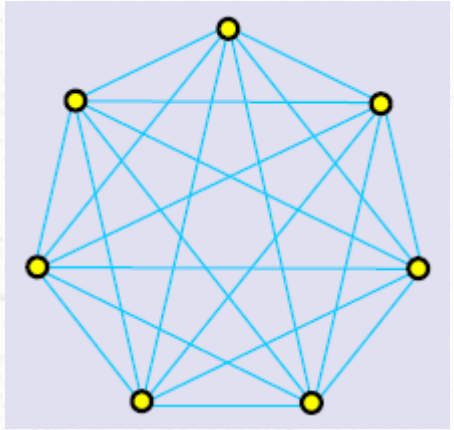


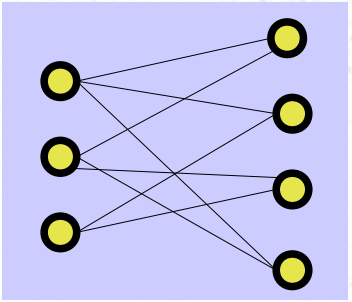

- walk: s = (j,h,l,g,e,f,h,l,e,c,b,a)
- length of walk: |s| = 11
- closed walk: start = end
- semiwalk: ignore line direction
- trail: all lines different
- **path:** all vertices different
- **cycle:** closed path
- **acyclic graph:** no cycles

# Subgraphs

- H(U, K) is a subgraph of G(V, L) if:
  - $U \subseteq V$
  - $K \subseteq L$
- spanning subgraph: U = V
- spanning tree:
  - spanning subgraph
  - no cycles (tree)

# Some special graphs



Path $P_5$

Circle $C_7$

Star $S_8$

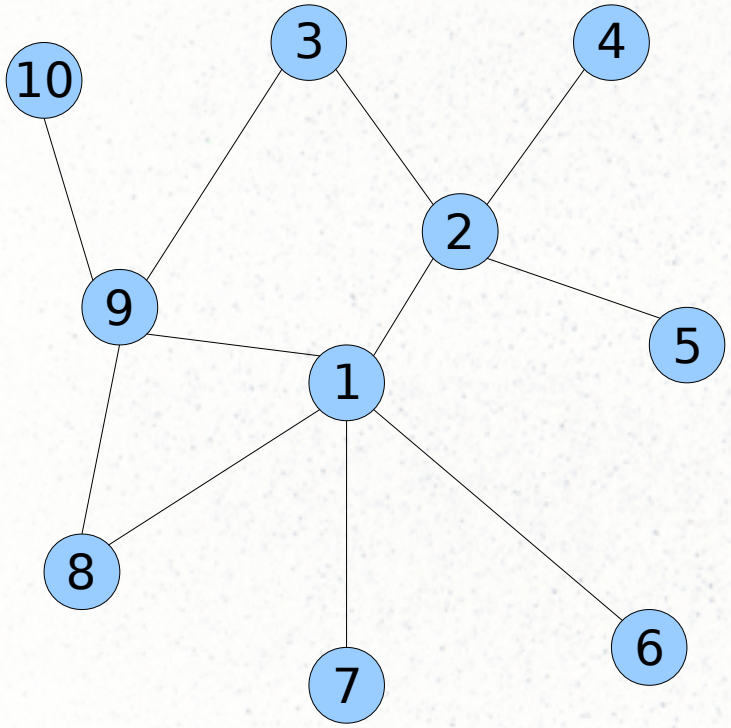
Complete graph $K_7$

Bipartite $B_{2,3}$

# Distances

- distance between node *u* and *v*
  - the length of the shortest path (geodesic distance)
- shortest path:
  - sum of the weights on the path is minimal
- eccentricity of vertex *v:*
  - greatest distance between *v* and any other vertex
- diameter: distance between the most distant vertices
  - also: maximal eccentricity
- radius of a graph:
  - minimal eccentricity of any vertex

# Example

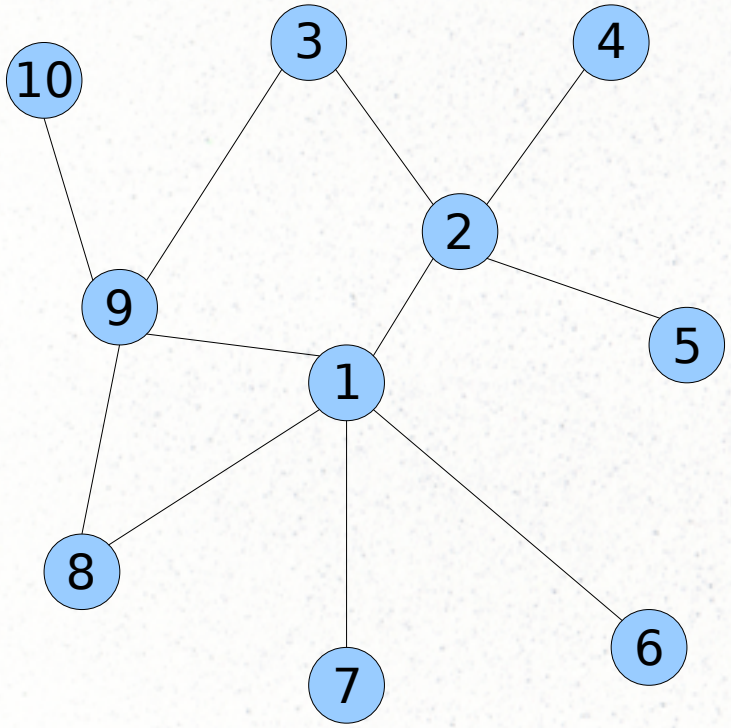



- distance (v6, v10) =
- shortest path (v5, v8) =
- diameter =
- radius =

Eccentricity

| v1 | v2 | v3 | v4 | v5 | v6 | v7 | v8 | v9 | v10 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |

# Example



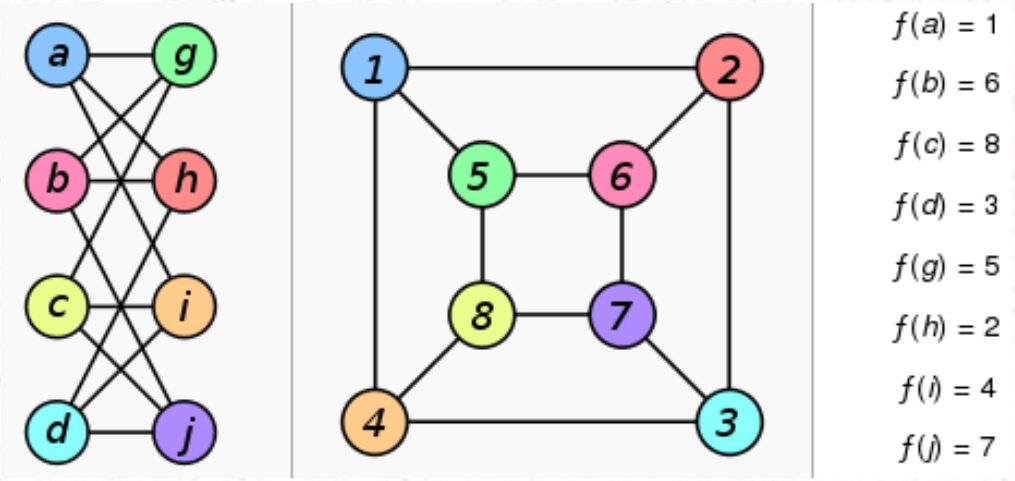

- distance (v6, v10) = **3**
- shortest path (v5, v8) = **4**
- diameter: **4**
- radius: **2**
- star-like graph (+ 3 cycles)

Eccentricity

| v1 | v2 | v3 | v4 | v5 | v6 | v7 | v8 | v9 | v10 |
|---|---|---|---|---|---|---|---|---|---|
| **2** | **3** | **3** | **4** | **4** | **3** | **3** | **3** | **3** | **4** |

# Isomorphism

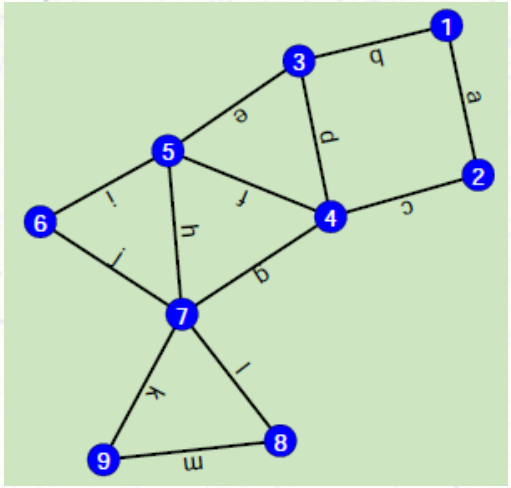
- G1 and G2 are isomorphic iff:
  - exists one-to-one correspondence $f : G_1 \rightarrow G_2$
  - regarding the structure, G1 and G2 are the same

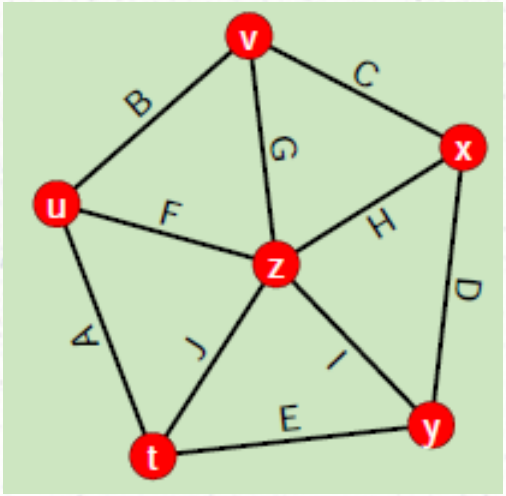



Interesting problem: polynomial time or NP-complete ?

# Homomorphism

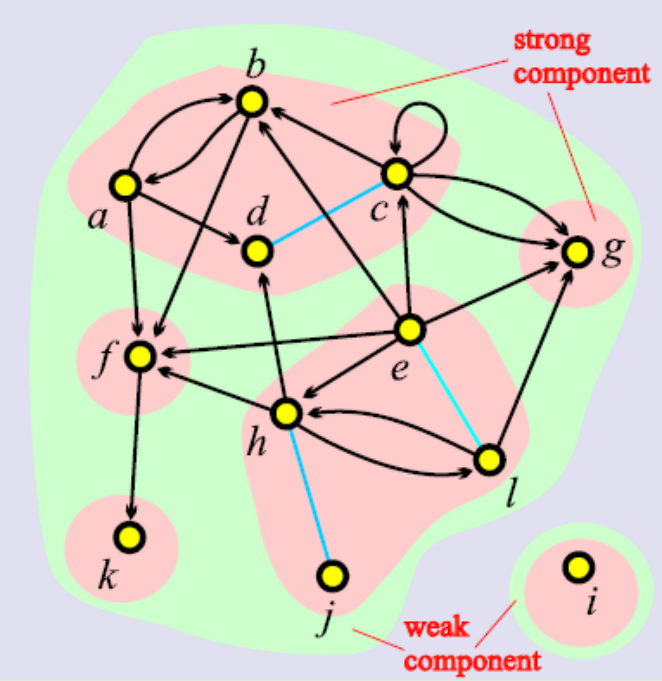
- weaker that isomorphism:
  - each line in G1 has a „picture“ in G2
  - the structure can now be different



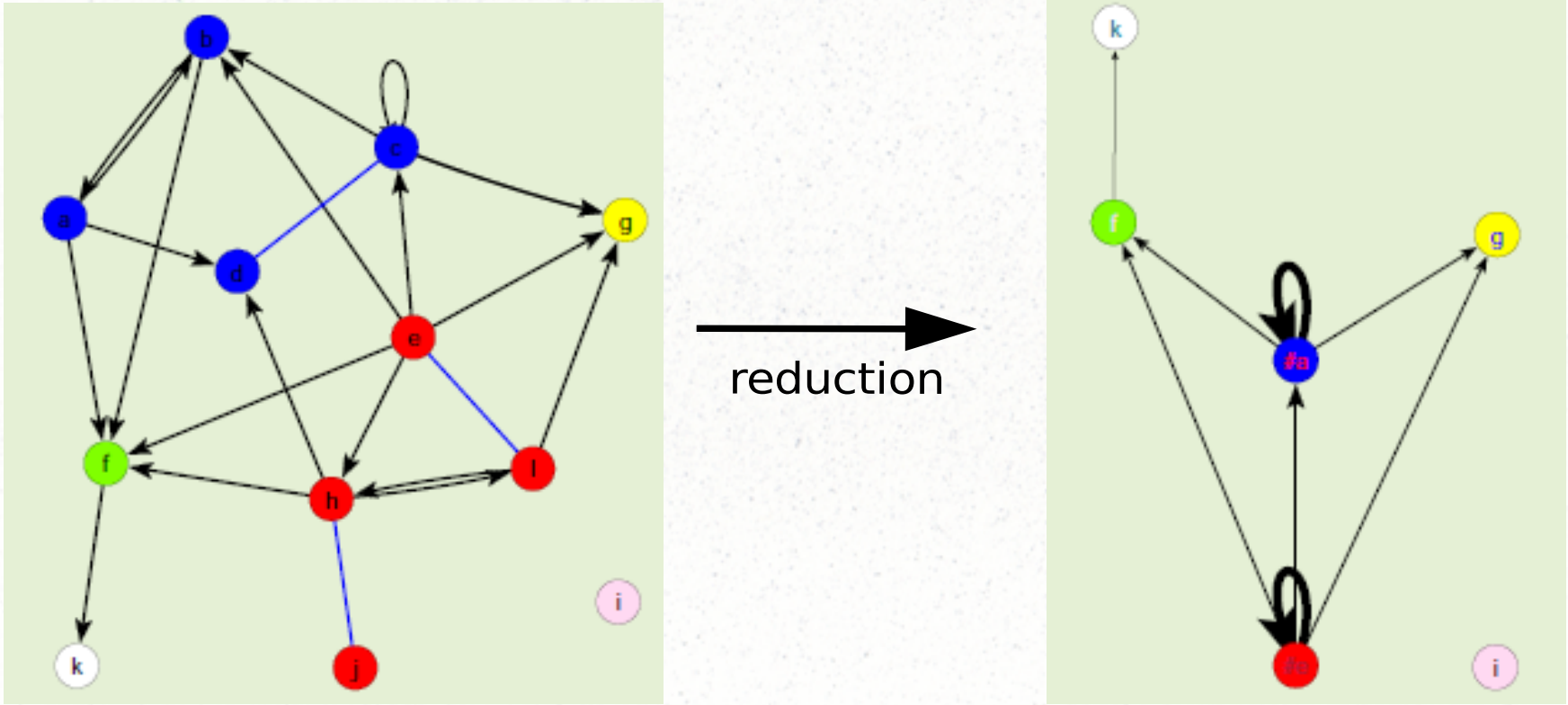

| a | b | c | d | e | f | g | h | i | j | k | l | m |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| E | J | D | H | G | C | H | G | B | F | J | I | E |

# Connectivity




- weak component:
  - connected subgraph
  - ignore line directions
- strong component:
  - connected subgraph
  - all vertices must be mutually reachable
- components can be **reduced**

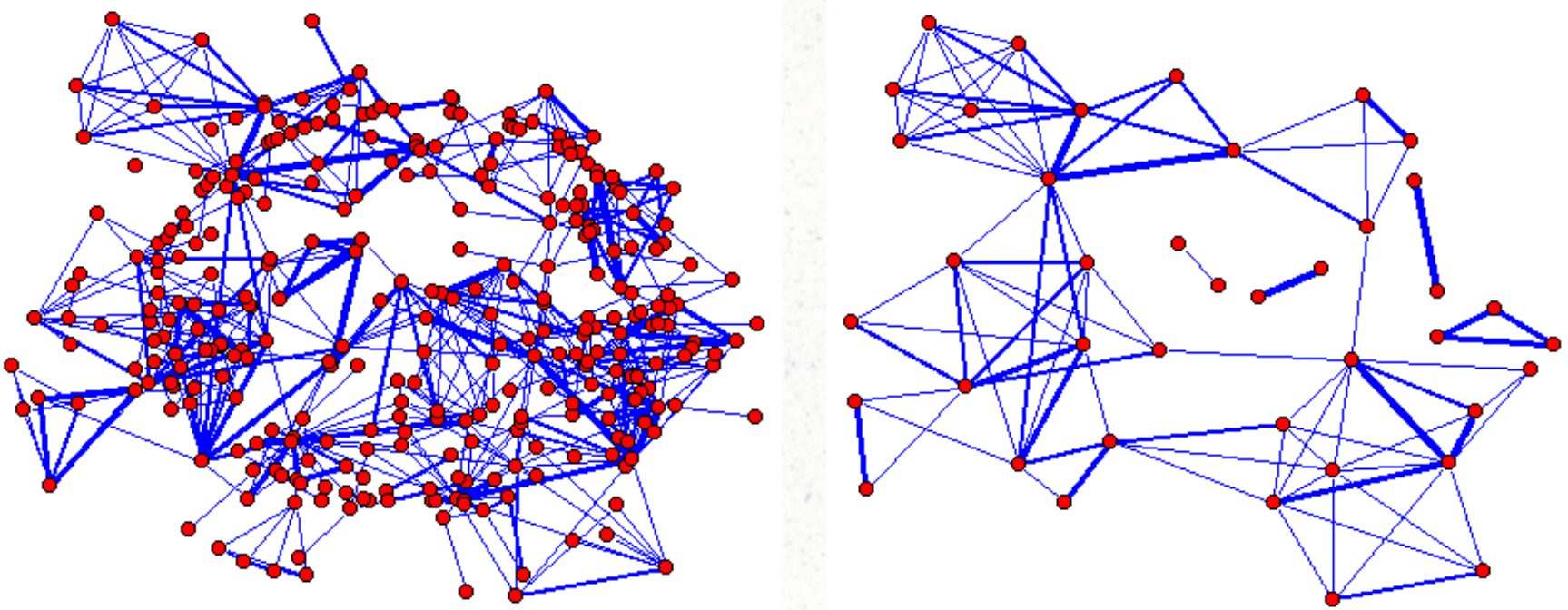
# Example:
## reduction of strong components



# Cuts

- interesting groups:
  - remove all vertices/lines with value less than ***p***
  - vertex-cut
  - line-cut
- important elements:
  - articulation vertex (also cut vertex)
  - bridge
  - deletion increases the number of weak components

# Example: Cuts



- remove all lines with value less than 6
- remove all isolated vertices

# *Important vertices*

- directed network: **importance measure**
  - influence: outgoing arcs
  - support: incomming arcs
- undirected network: **centrality**
- importance depends on the relation!
  - if relation = “doesn't like“ then
  - most important = “least popular“

# Measures of importance: Degree based

$$\text{centrality} = \frac{deg(v)}{n-1}$$

$$\text{support} = \frac{indeg(v)}{n}$$

$$\text{influence} = \frac{outdeg(v)}{n}$$

(all measures are normalized)

# Measures of importance: Distance based

$$\text{radius}(v) = \text{maximal eccentricity of v}$$

$$\text{total closeness}(v) = \sum_{u \in V} dist(v, u)$$

$$\text{normalized closeness}(v) = \frac{n-1}{\sum_{u \in V} dist(v, u)}$$

betweeness of a vertex:

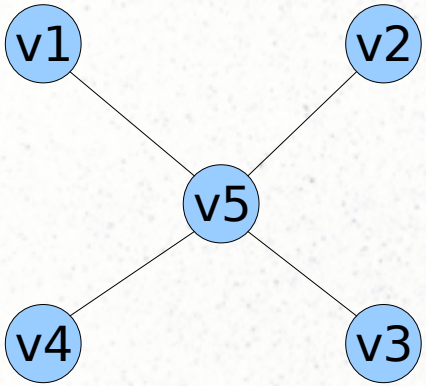
number of all shortest paths that go through this vertex

# *Network centralization*

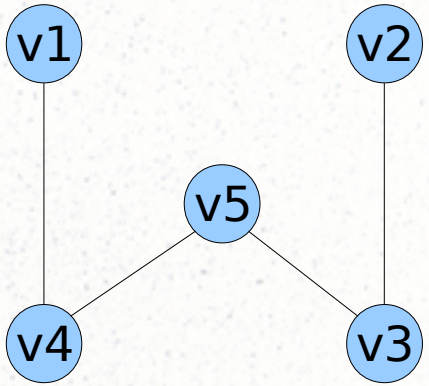
- is the variation of the measure of centrality for vertices
  - divided by maximum possible variation for such network
- degree, closeness, betweeness centralization
- if a network is centralized:
  - clear boundary between center and periphery

# Example: degree centralization

$$\text{degreee centralization} = \frac{\sum_{v \in V} (maxDeg_V - deg(v))}{\text{max. variation on these vertices}}$$



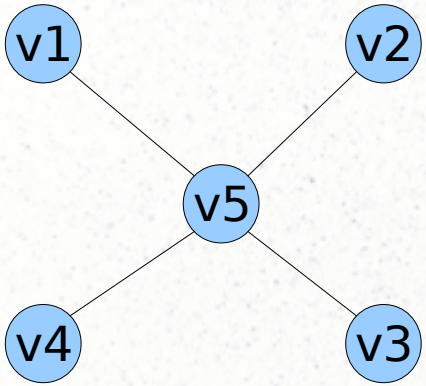

| vertex | v1 | v2 | v3 | v4 | v5 |
|---|---|---|---|---|---|
| degree | | | | | |

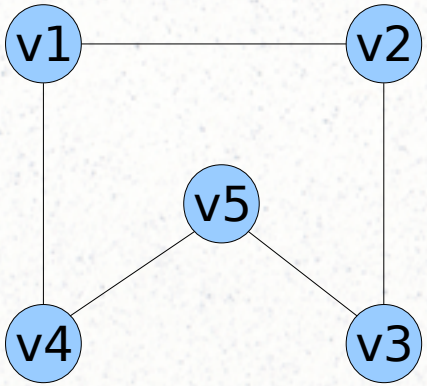


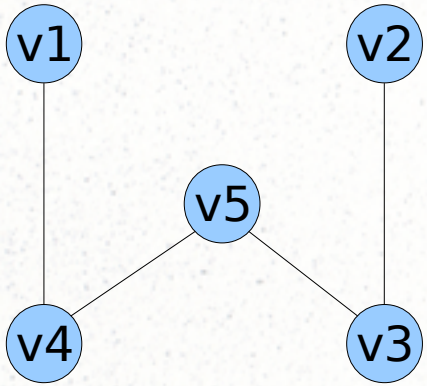

| vertex | v1 | v2 | v3 | v4 | v5 |
|---|---|---|---|---|---|
| degree | | | | | |

# Example: degree centralization

$$\text{degreee centralization} = \frac{\sum_{v \in V} (maxDeg_V - deg(v))}{\text{max. variation on these vertices}}$$




| vertex | v1 | v2 | v3 | v4 | v5 |
|---|---|---|---|---|---|
| degree | 1 | 1 | 1 | 1 | 4 |

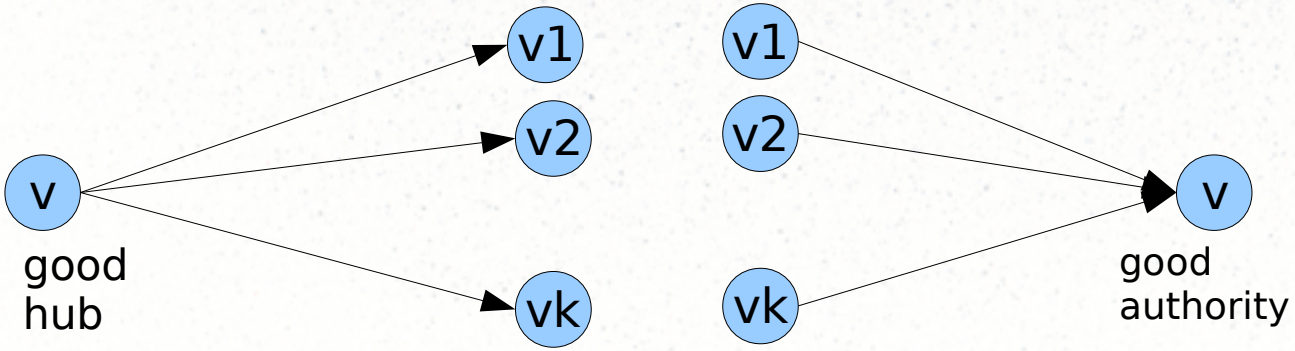
| vertex | v1 | v2 | v3 | v4 | v5 |
|---|---|---|---|---|---|
| degree | 1 | 1 | 2 | 2 | 2 |

$$\frac{(4-1)+(4-1)+(4-1)+(4-1)+(4-4)}{12} = 1.0$$

$$\frac{(2-1)+(2-1)+(2-2)+(2-2)+(2-2)}{12} = 0.17$$

# Hubs and authorities

- to each vertex *v* assign two values:
  - quality of its contents: authority $a_v$
  - quality of its references: hub $h_v$
- good hub points to good authorities
- good authority is selected by good hubs




$$a_v = \sum_{u:(u,v)\in L} h_u$$

$$h_v = \sum_{u:(v,u)\in L} a_u$$

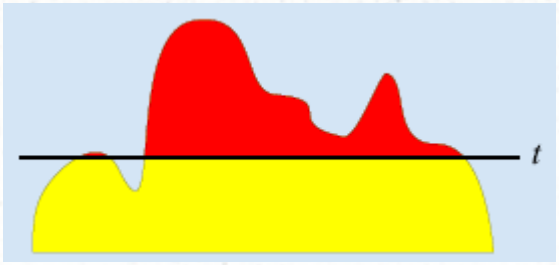
# Example: Google PageRank

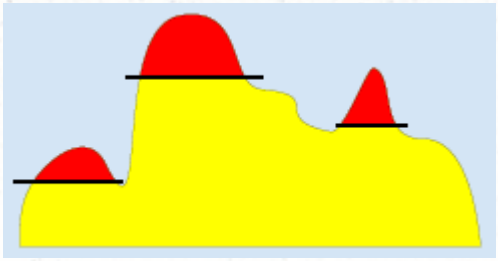
- based on citation analysis
- the probability to reach a page after many clicks
- favors older pages
- recalculated after each index rebuild
- can be fooled (e.g. „link farms“)
- recursive formula:

$$PR(p) = \sum_{v \in S_p} \frac{PR(v)}{|Links(v)|}$$

# *Important subnetworks: Islands*

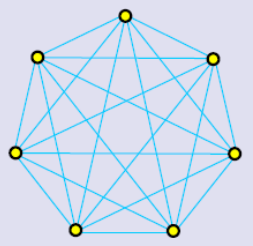
- we need:
  - numerical property of vertices/lines
  - min and max size
- result:
  - **locally important subnetworks** on different levels
- efficient algorithms for island discovery



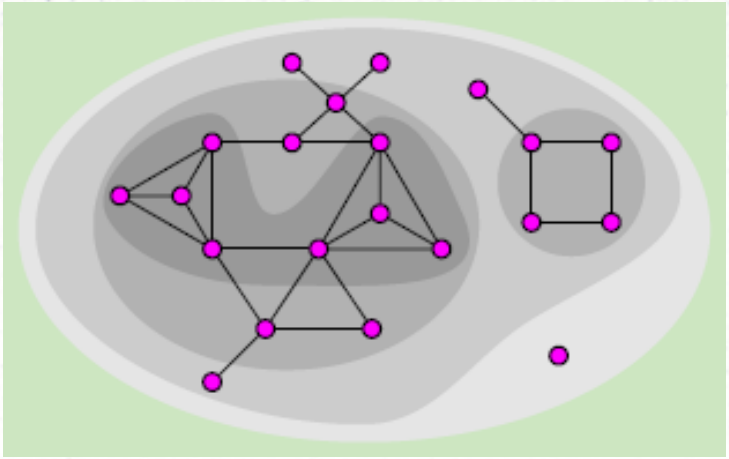
# Dense groups: cliques

- clique of order k:
  - complete subgraph (all posible lines)
- all actors are in relation with all other actors
- hard to compute (NP-complete problem)
- examples:
  - very good friends in a friend network
  - group of tighty connected scientists who published papers together with all others



# Dense groups: cores

- k-core on n vertices:
  - minimal degree is k
  - generally: set of subgraphs
- cores are nested: core hierarchy
- efficient algorithms exist

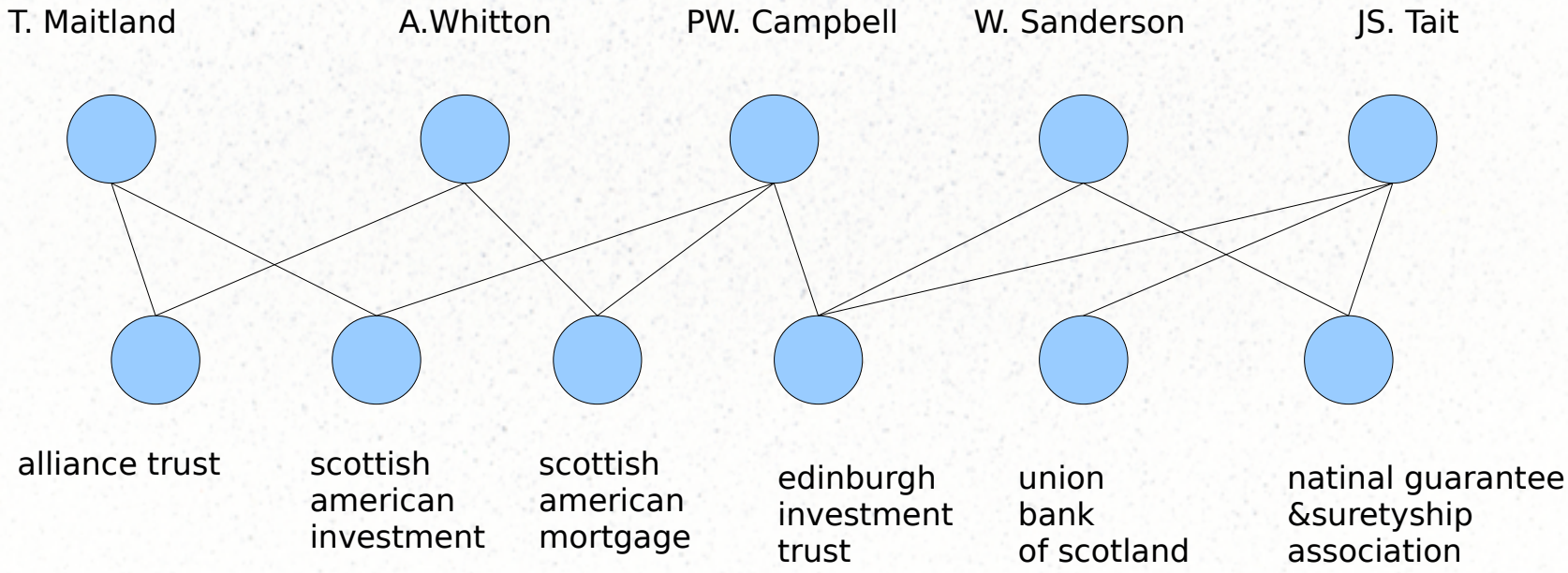


# *Two-mode networks*

- two disjoint sets of vertices
- how to analyze:
  - convert to one-mode or
  - use specialized algorithms
- analysis:
  - two-mode cores
  - clustering
  - blockmodelling
  - after conversion to one-mode: all std. algs.

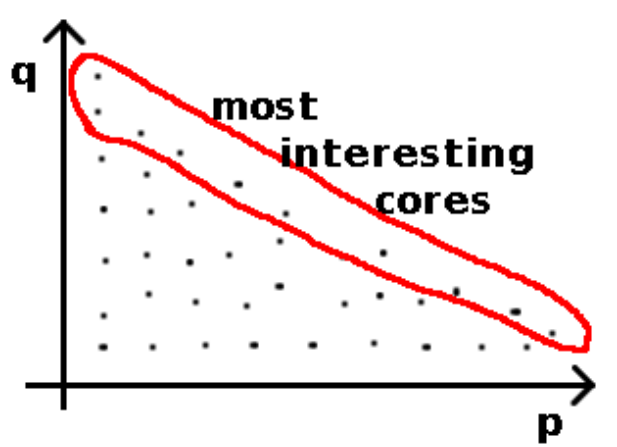
# *Example: two-mode network*

- directors and stock companies in Scotland (one small part of the whole network)



# Two-mode cores

- (p,q) core:
  - minDegree >= p in the first set
  - minDegree >= q in the second set
  - maximal subset with these conditions
- not always connected



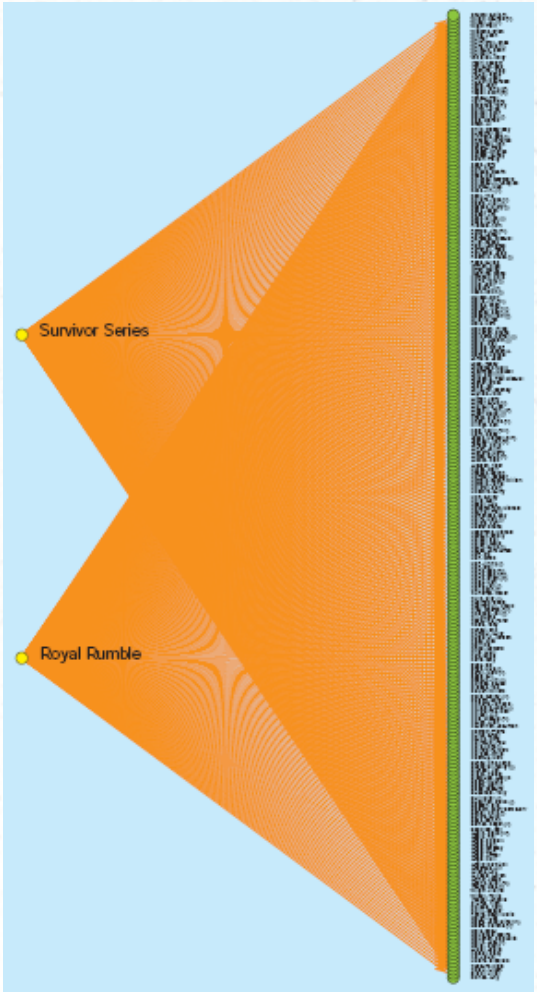
# *Example: Two-mode cores of IMDB*

- network: actors and films

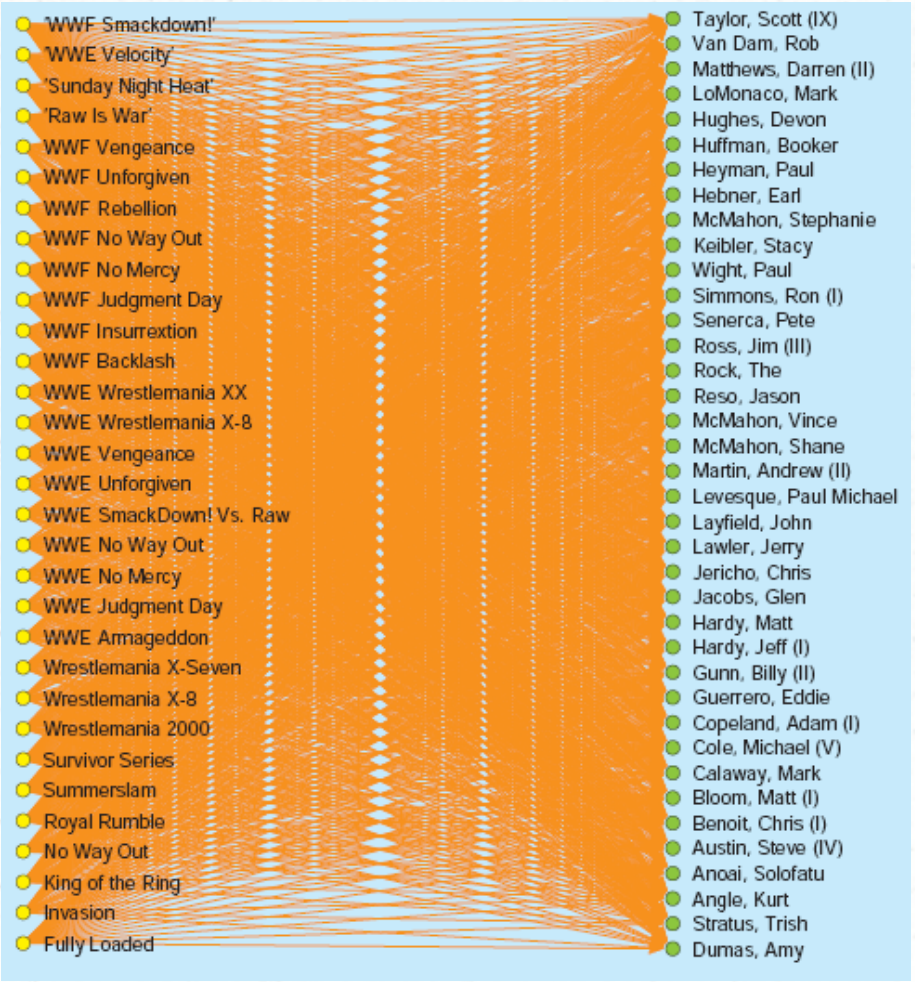
| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1590: | 1590 | 1 | 16 | 39: | 2173 | 678 | | 44 | 14: | 29 | 83 |
| 2 | 516: | 788 | 3 | 17 | 35: | 2791 | 995 | | 46 | 13: | 29 | 94 |
| 3 | 212: | 1705 | 18 | 18 | 32: | 2684 | 1080 | | 49 | 12: | 26 | 95 |
| 4 | 151: | 4330 | 154 | 19 | 30: | 2395 | 1063 | | 52 | 11: | 16 | 79 |
| 5 | 131: | 4282 | 209 | 20 | 28: | 2216 | 1087 | | 56 | 10: | 34 | 162 |
| 6 | 115: | 3635 | 223 | 21 | 26: | 1988 | 1087 | | 62 | 9: | 31 | 177 |
| 7 | 101: | 3224 | 244 | 22 | 24: | 1854 | 1153 | | 66 | 8: | 29 | 198 |
| 8 | 88: | 2860 | 263 | 24 | 23: | 34 | 39 | | 72 | 7: | 22 | 203 |
| 9 | 77: | 3467 | 393 | 27 | 22: | 31 | 38 | | 96 | 6: | 7 | 114 |
| 10 | 69: | 3150 | 428 | 29 | 20: | 35 | 52 | | 119 | 5: | 6 | 137 |
| 11 | 63: | 2442 | 382 | 32 | 19: | 34 | 57 | | 141 | 4: | 8 | 258 |
| 12 | 56: | 2479 | 454 | 35 | 18: | 33 | 61 | | 186 | 3: | 3 | 186 |
| 13 | 50: | 3330 | 716 | 36 | 17: | 33 | 65 | | 247 | 2: | 2 | 247 |
| 14 | 46: | 2460 | 596 | 39 | 16: | 29 | 70 | | 1334 | 1: | 1 | 1334 |
| 15 | 42: | 2663 | 739 | 42 | 15: | 28 | 76 | | | | | |

# Wrestling movies

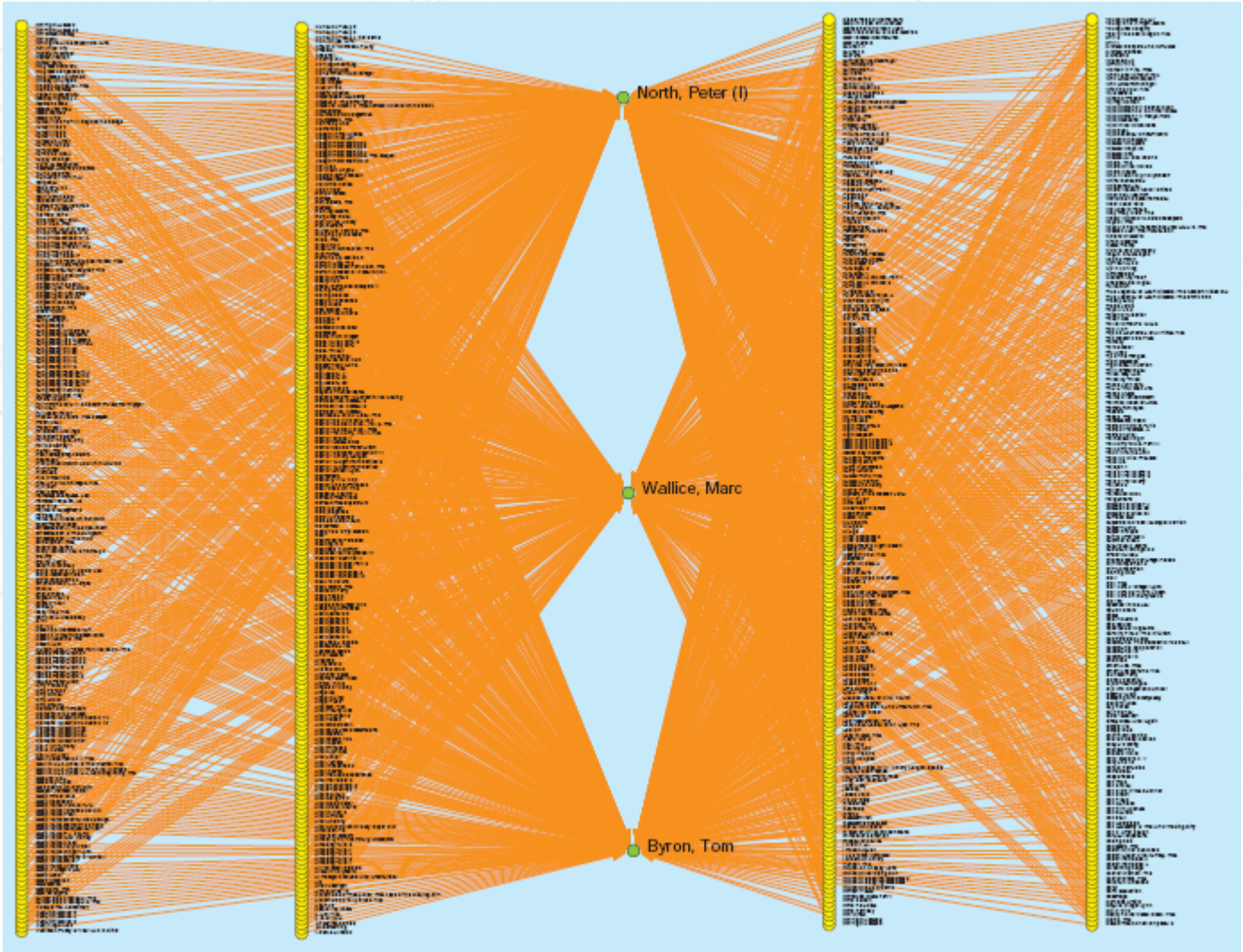
(247,2)




(27,22)



# (2,516) core: Hard core (core of films for adults...)

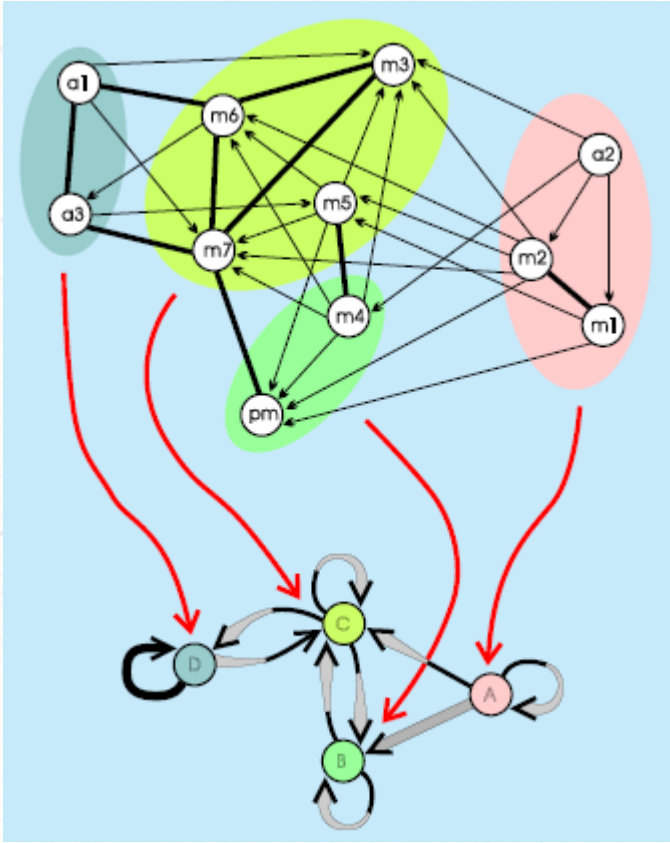


# *Clusters, partitions, hierarchies*

- cluster: nonempty subset of V: $C \subseteq V$
- clustering: nonempty set of clusters: $\boldsymbol{C} = \{C_i\}$
- clustering $\boldsymbol{C} = \{C_i\}$ is a partition iff:
  - $\bigcup_i C_i = V$ and $C_i \cap C_j = \emptyset$
- clustering $\boldsymbol{C} = \{C_i\}$ is a hierarchy iff:
  - $C_i \cap C_j = \{\emptyset, C_i, C_j\}$
- **why clustering:**
  - group similar vertices
  - gain insight into the structure

# *Blockmodelling*

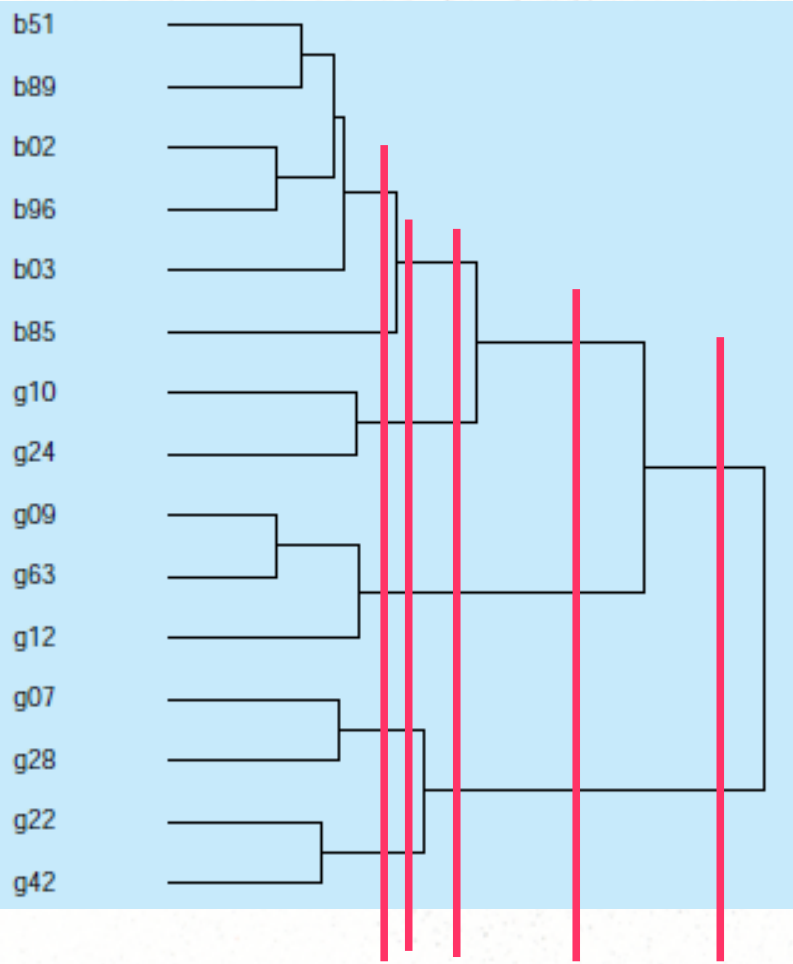
- reduce a large network
- identify clusters of units:
  - those who have smilar structural characteristic
- within cluster:
  - the same or similar connection patterns to other units (equivalency)



# *Equivalence*

- two types:
  - structural: the same connection pattern to the same neighbours
  - regular: the same or similar pattern to (possibly) different neighbours
- finding partitions in terms of equivalence is a special case of clustering
  - **indirect** approach
    - compute dissimilarities
  - direct approach
    - construct fit function and do local optimization

# *Indirect approach*

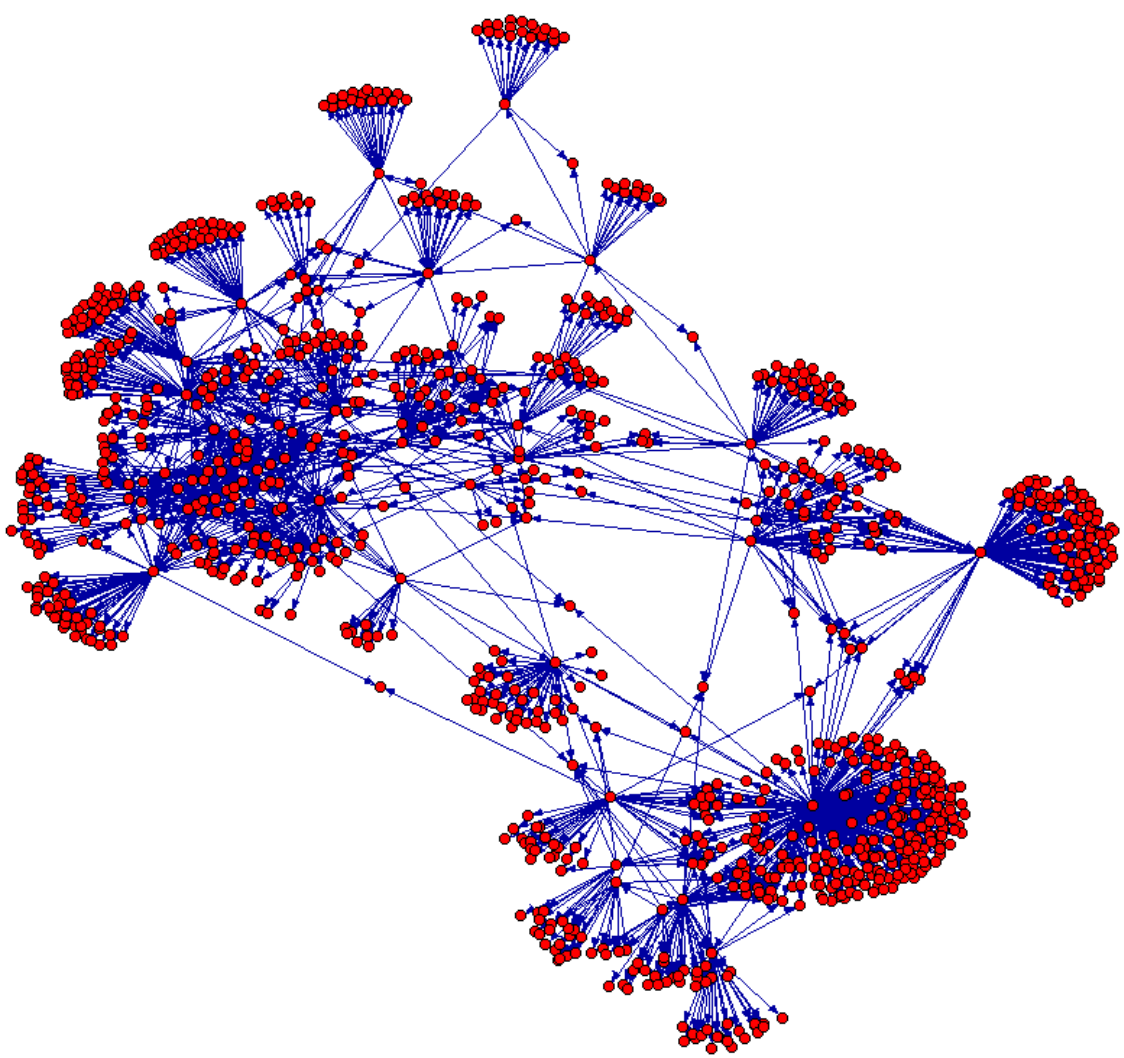



- compute dissimilarity (e.g. corrected Euclidean)
- use hierarchical clustering to obtain partitioning
- **now we can manually select suitable number of clusters**

# *Graph visualization*

- first step in any SNA task:
  - general overview of the network
  - get some ideas what analysis to perform
  - find some patterns manually
  - inspect individuals
- many algorithms:
  - exact: simulation of springs and electrical charges
  - approximate: identify and draw groups
- Pajek: up to ~5000 vertices

# *Visualization of a citation network: Literature on graph drawing*



# Springer ILP citation network