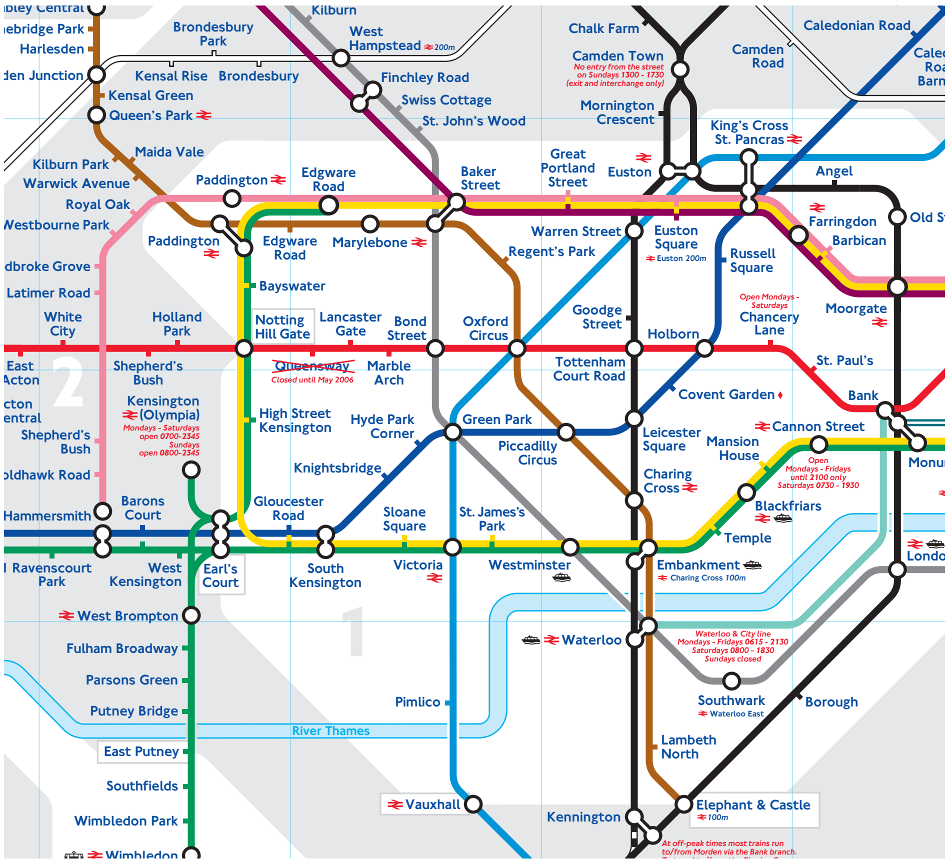


London Tube

Network Analysis

Structure of Networks 2

Connectivity

Vladimir Batagelj

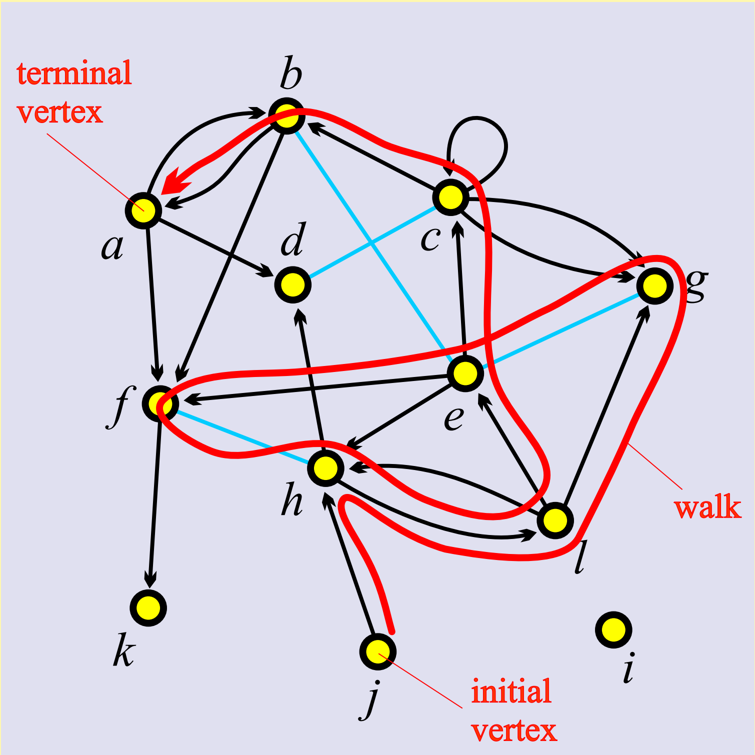
University of Ljubljana

ECPR Summer School, July 30 – August 16, 2008

Faculty of Social Sciences, University of Ljubljana

Outline

Walks



length $|s|$ of the walk s is the number of lines it contains.

$s = (j, h, l, g, e, f, h, l, e, c, b, a)$

$|s| = 11$

A walk is *closed* iff its initial and terminal vertex coincide.

If we don't consider the direction of the lines in the walk we get a *semiwalk* or *chain*.

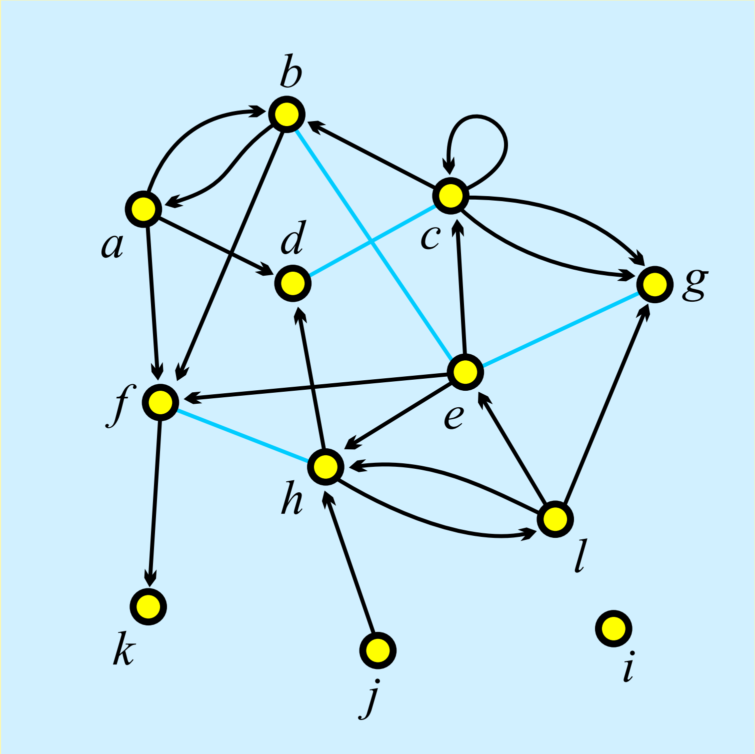
trail – walk with all lines different

path – walk with all vertices different

cycle – closed walk with all internal vertices different

A graph is *acyclic* if it doesn't contain any cycle.

Shortest paths



A shortest path from u to v is also called a *geodesic* from u to v. Its length is denoted by $d(u, v)$.

If there is no walk from u to v then $d(u, v) = \infty$.

$d(j, a) = |(j, h, d, c, b, a)| = 5$

$d(a, j) = \infty$

$\hat{d}(u, v) \;=\; \max(d(u, v), d(v, u))$

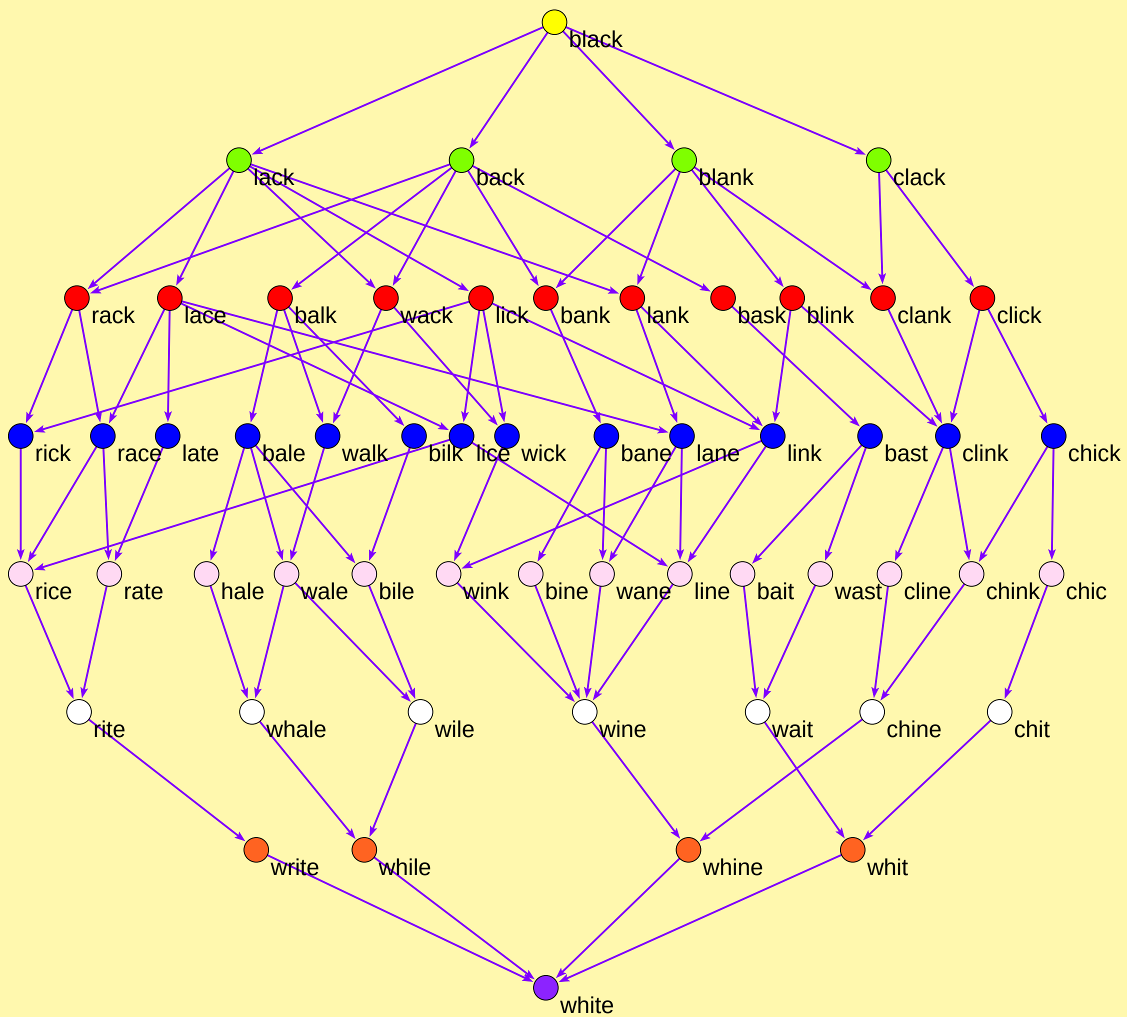
is a *distance*:

$\hat{d}(v, v) = 0$, $\hat{d}(u, v) = \hat{d}(v, u)$,

$\hat{d}(u, v) \leq \hat{d}(u, t) + \hat{d}(t, v)$.

The *diameter* of a graph equals to the distance between the most distant pair of vertices: $D = \max_{u,v \in \mathcal{V}} d(u, v)$.

Shortest paths



DICT28.

Equivalence relations and Partitions

A relation R on $\mathcal{V}$ is an *equivalence* relation iff it is reflexive $\forall v \in \mathcal{V} : vRv$, symmetric $\forall u, v \in \mathcal{V} : uRv \Rightarrow vRu$, and transitive $\forall u, v, z \in \mathcal{V} : uRz \wedge zRv \Rightarrow uRv$.

Each equivalence relation determines a partition into *equivalence classes* $[v] = \{u : vRu\}$.

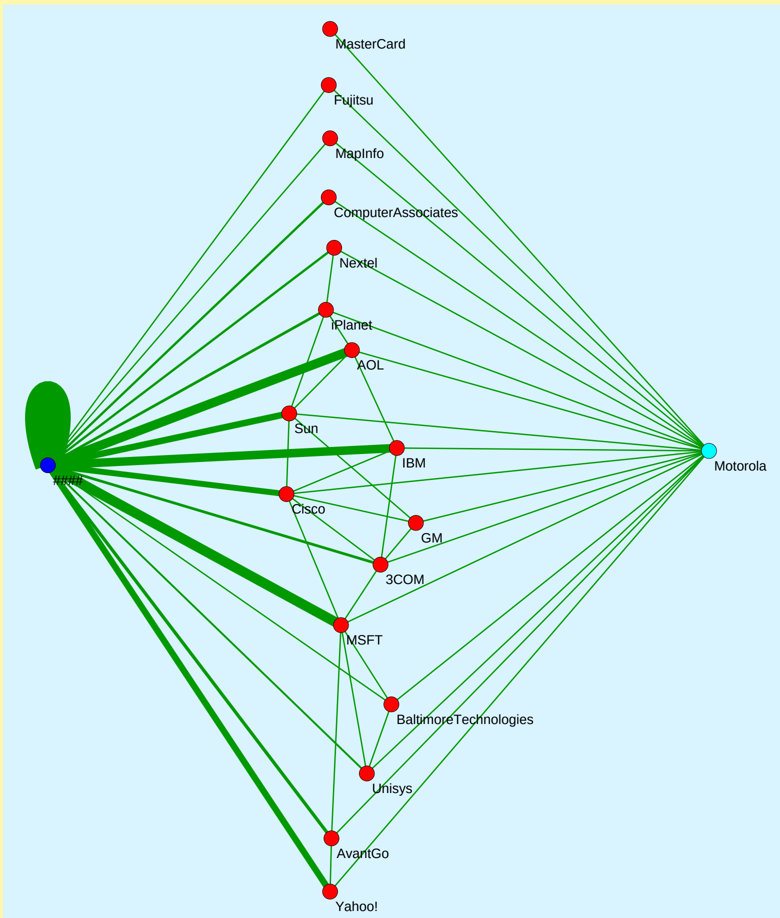
Each partition $\mathbf{C}$ determines an equivalence relation $uRv \Leftrightarrow \exists C \in \mathbf{C} : u \in C \wedge v \in C$.

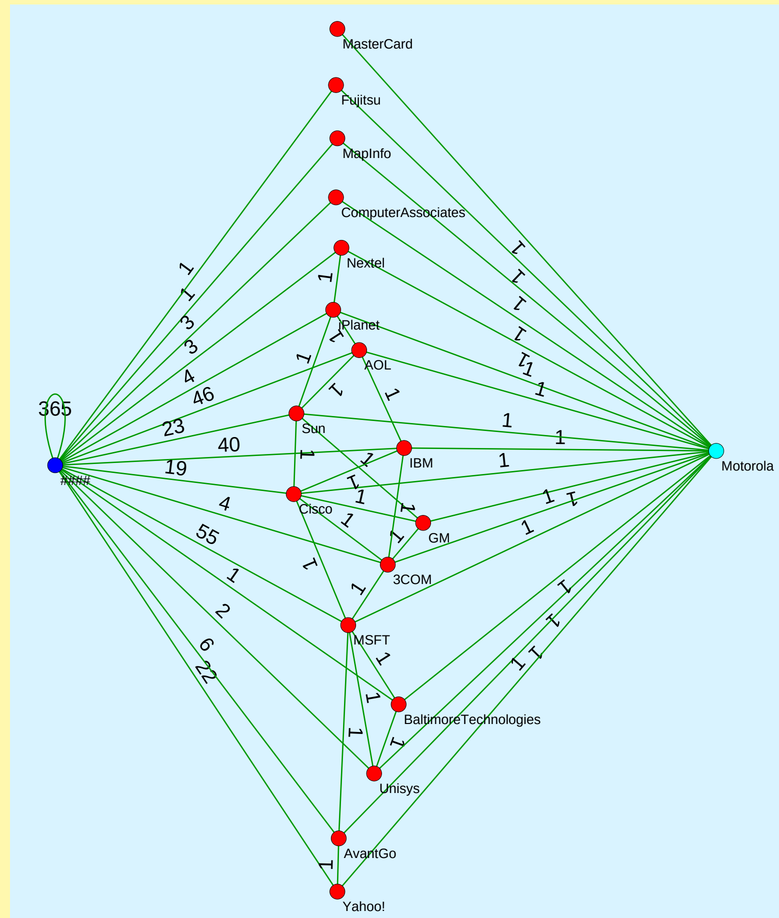
k-neighbors of v is the set of vertices on 'distance' k from v, $N^k(v) = \{u \in v : d(v, u) = k\}$.

The set of all k-neighbors, $k = 0, 1, \ldots$ of v is a partition of $\mathcal{V}$.

k-neighborhood of v, $N^{(k)}(v) = \{u \in v : d(v, u) \leq k\}$.

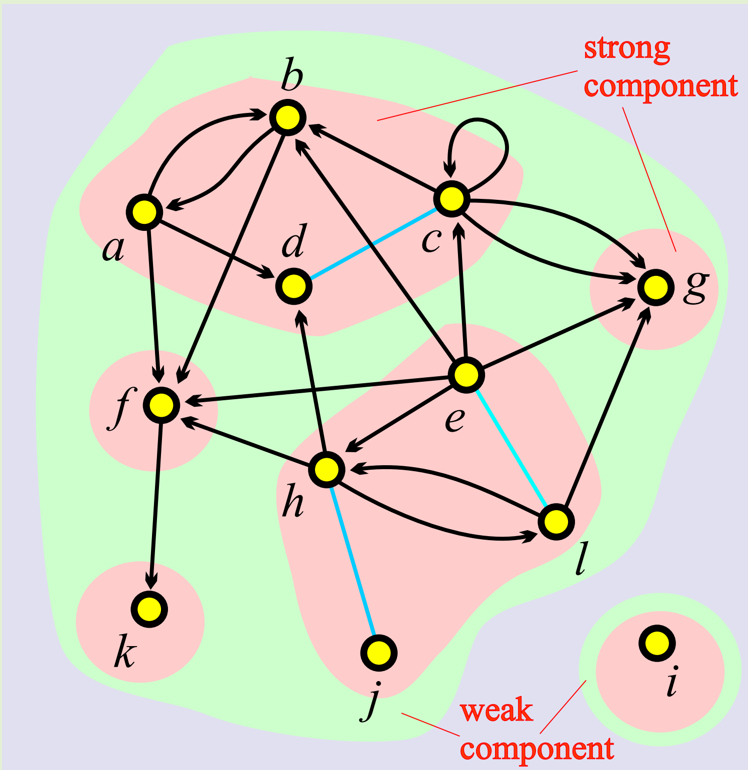
Motorola's neighborhood





The thickness of edges is a square root of its value.

Connectivity



Vertex u is *reachable* from vertex v iff there exists a walk with initial vertex v and terminal vertex u.

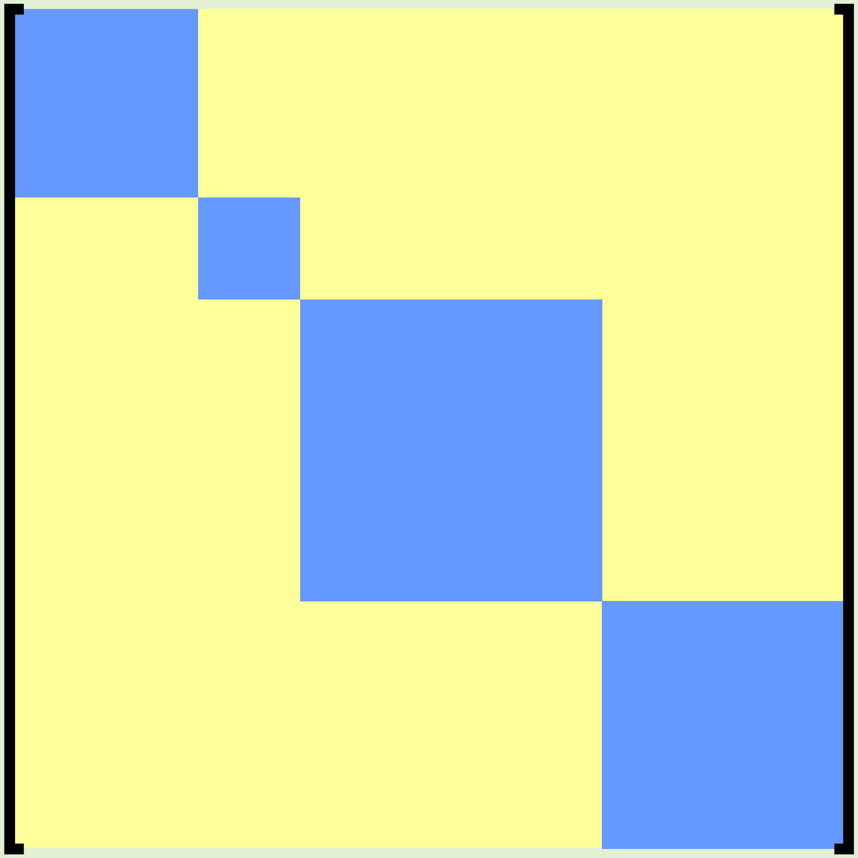
Vertex v is *weakly connected* with vertex u iff there exists a semiwalk with v and u as its end-vertices.

Vertex v is *strongly connected* with vertex u iff they are mutually reachable.

Weak and strong connectivity are equivalence relations.

Equivalence classes induce weak/strong *components*.

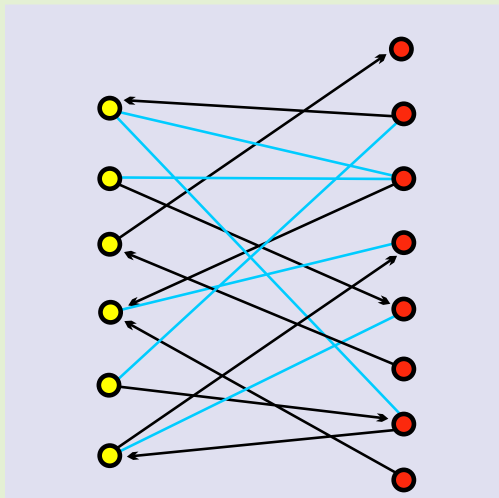
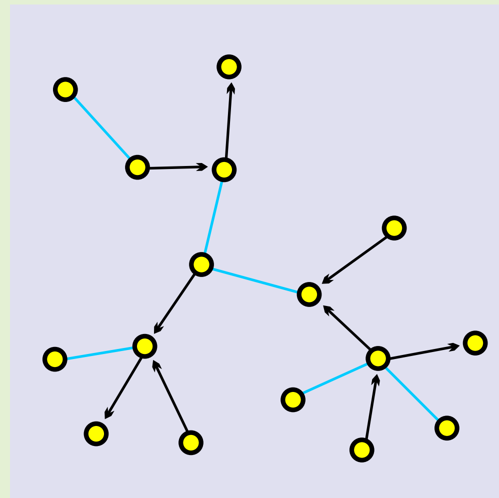
Weak components



Reordering the vertices of network such that the vertices from the same class of weak partition are put together we get a matrix representation consisting of diagonal blocks – weak components.

Most problems can be solved separately on each component and afterward these solutions combined into final solution.

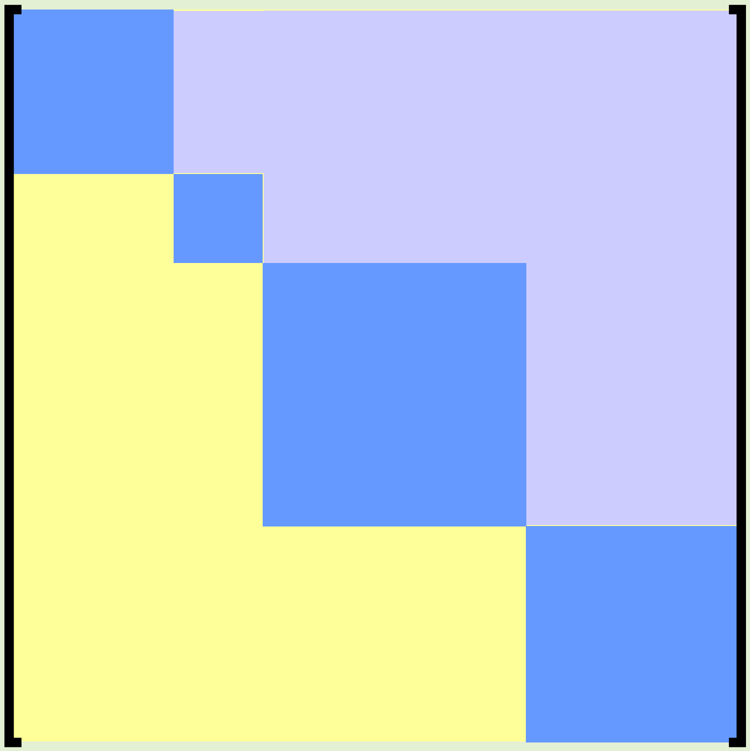
Special graphs – bipartite, tree

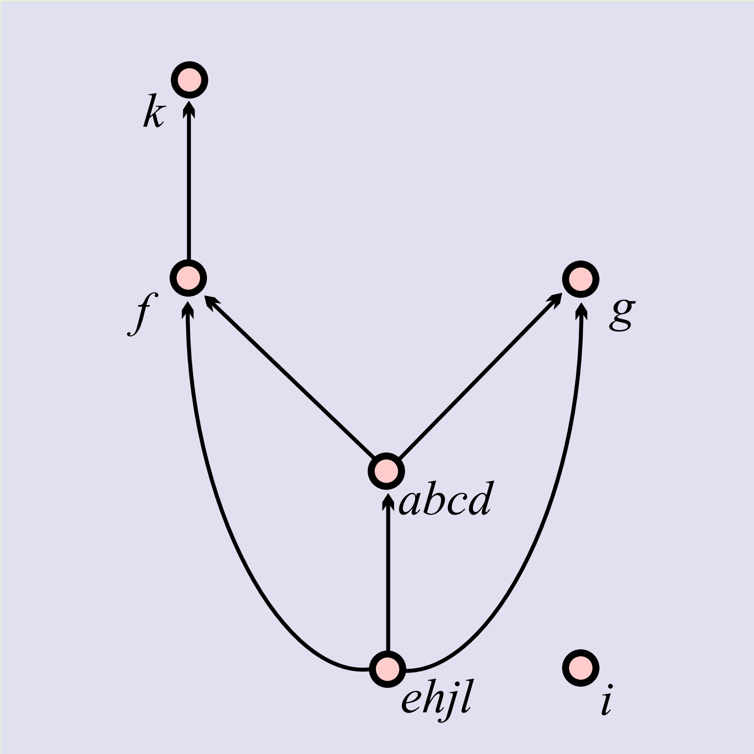



A graph $\mathcal{G} = (\mathcal{V}, \mathcal{L})$ is *bipartite* iff its set of vertices $\mathcal{V}$ can be partitioned into two sets $\mathcal{V}_1$ and $\mathcal{V}_2$ such that every line from $\mathcal{L}$ has one end-vertex in $\mathcal{V}_1$ and the other in $\mathcal{V}_2$.

A weakly connected graph $\mathcal{G}$ is a *tree* iff it doesn't contain loops and semicycles of length at least 3.

Reduction (condensation)

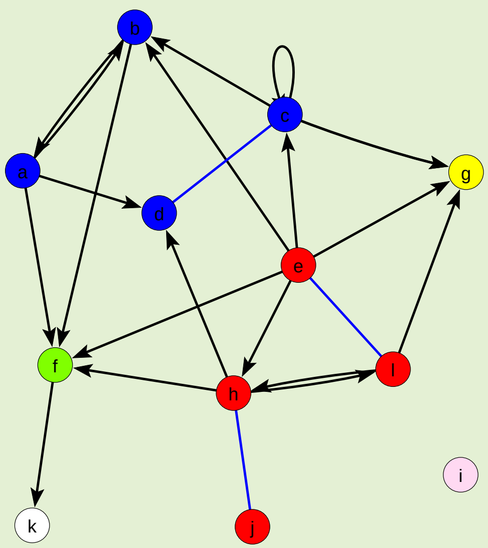


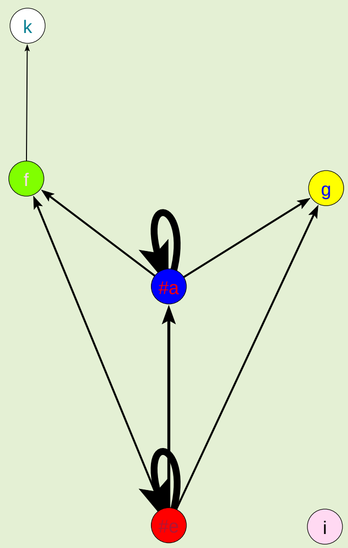


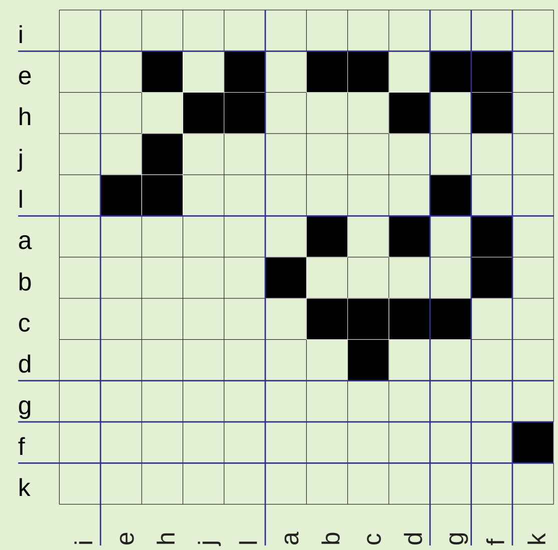
If we shrink every strong component of a given graph into a vertex, delete all loops and identify parallel arcs the obtained *reduced* graph is acyclic. For every acyclic graph an *ordering* / *level* function $i : \mathcal{V} \rightarrow \mathbb{N}$ exists s.t. $(u, v) \in \mathcal{A} \Rightarrow i(u) < i(v)$.

Reduction – Example

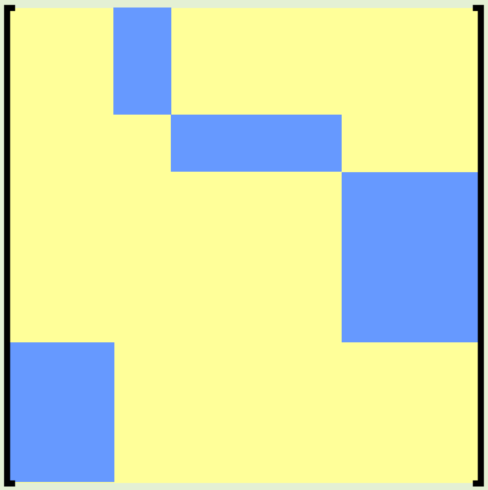
```
Net / Components / Strong [1]
Operations / Shrink Network / Partition [1][0]
Net / Transform / Remove / Loops [yes]
Net / Partitions / Depth / Acyclic
Partition / Make Permutation
Permutation / Inverse
select partition [Strong Components]
Operations / Functional Composition / Partition*Permutation
Partition / Make Permutation
select [original network]
File / Network / Export Matrix to EPS / Using Permutation
```

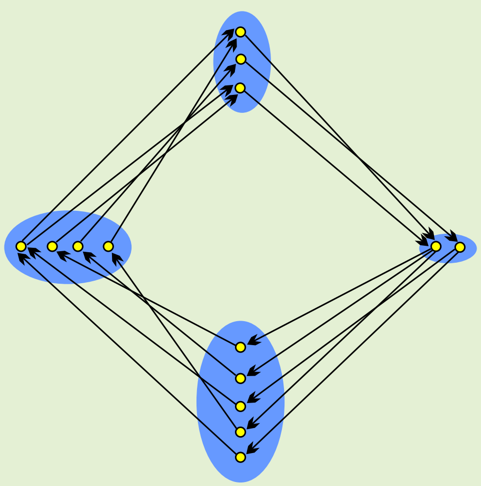






…internal structure of strong components



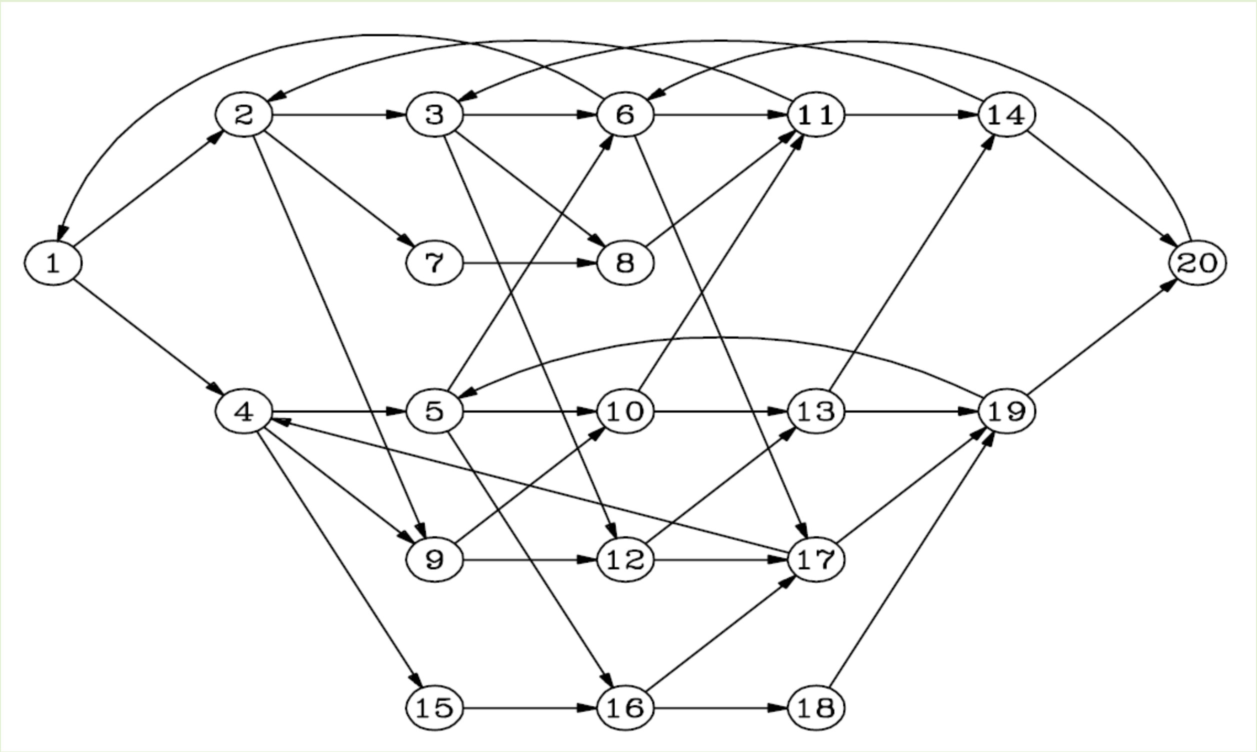


Let d be the largest common divisor of lengths of closed walks in a strong component.

The component is said to be *simple*, iff $d = 1$; otherwise it is *periodical* with a period d.

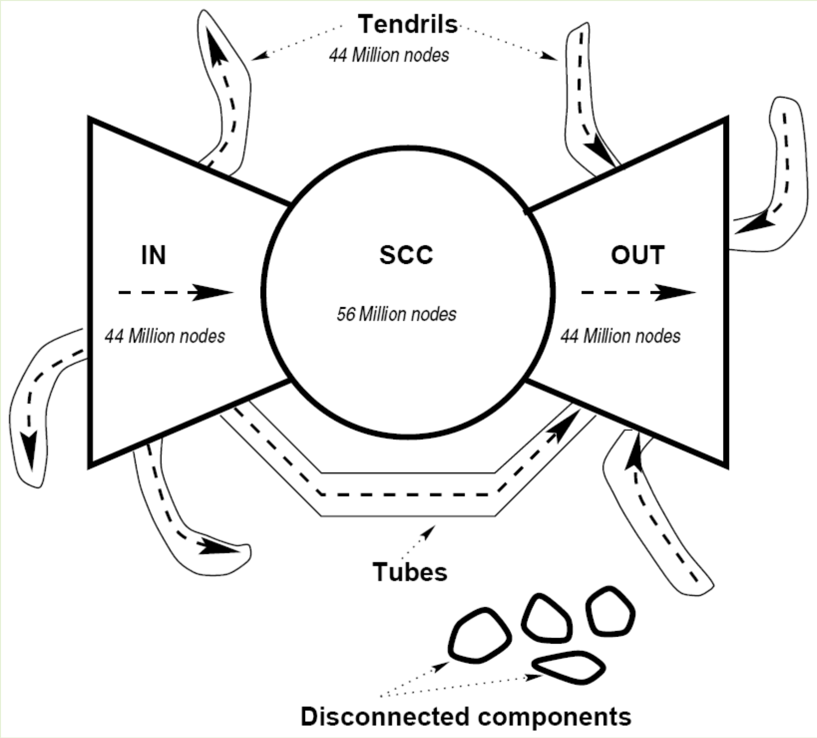
The set of vertices $\mathcal{V}$ of strongly connected directed graph $\mathcal{G} = (\mathcal{V}, R)$ can be partitioned into d clusters $\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_d$, s.t. for every arc $(u, v) \in R$ holds $u \in \mathcal{V}_i \Rightarrow v \in \mathcal{V}_{(i \bmod d)+1}$.

. . . internal structure of strong components



Bonhoure's periodical graph. Pajek data

Bow-tie structure of the Web graph



Kumar &: The Web as a graph

Let $\mathcal{S}$ be the *largest strong component* in network $\mathcal{N}$; $\mathcal{W}$ the weak component containing $\mathcal{S}$; $\mathcal{I}$ the set of vertices from which $\mathcal{S}$ can be reached; $\mathcal{O}$ the set of vertices reachable from $\mathcal{S}$; $\mathcal{T}$ (tubes) set of vertices (not in $\mathcal{S}$) on paths from $\mathcal{I}$ to $\mathcal{O}$; $\mathcal{R} = \mathcal{W} \setminus (\mathcal{I} \cup \mathcal{S} \cup \mathcal{O} \cup \mathcal{T})$ (tendrils); and $\mathcal{D} = \mathcal{V} \setminus \mathcal{W}$. The partition

$$\{\mathcal{I}, \mathcal{S}, \mathcal{O}, \mathcal{T}, \mathcal{R}, \mathcal{D}\}$$

is called the *bow-tie* partition of $\mathcal{V}$.

Cuts

The standard approach to find interesting groups inside a network was based on properties/weights – they can be *measured* or *computed* from network structure (for example Kleinberg's hubs and authorities).

The *vertex-cut* of a network $\mathcal{N} = (\mathcal{V}, \mathcal{L}, p)$, $p : \mathcal{V} \to \mathbb{R}$, at selected level t is a subnetwork $\mathcal{N}(t) = (\mathcal{V}', \mathcal{L}(\mathcal{V}'), p)$, determined by the set

$$\mathcal{V}' = \{v \in \mathcal{V} : p(v) \geq t\}$$

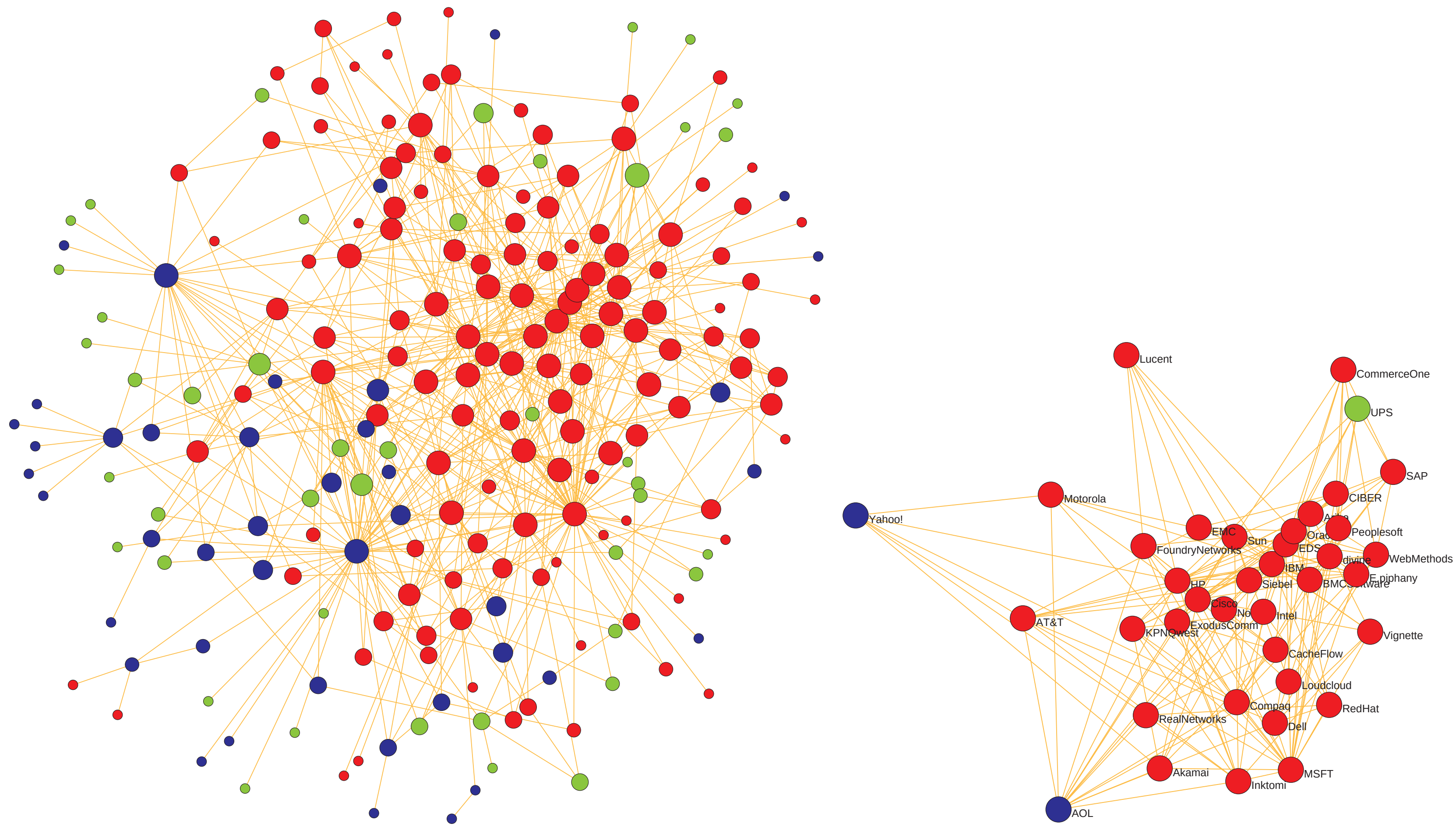
and $\mathcal{L}(\mathcal{V}')$ is the set of lines from $\mathcal{L}$ that have both endpoints in $\mathcal{V}'$.

The *line-cut* of a network $\mathcal{N} = (\mathcal{V}, \mathcal{L}, w)$, $w : \mathcal{L} \to \mathbb{R}$, at selected level t is a subnetwork $\mathcal{N}(t) = (\mathcal{V}(\mathcal{L}'), \mathcal{L}', w)$, determined by the set

$$\mathcal{L}' = \{e \in \mathcal{L} : w(e) \geq t\}$$

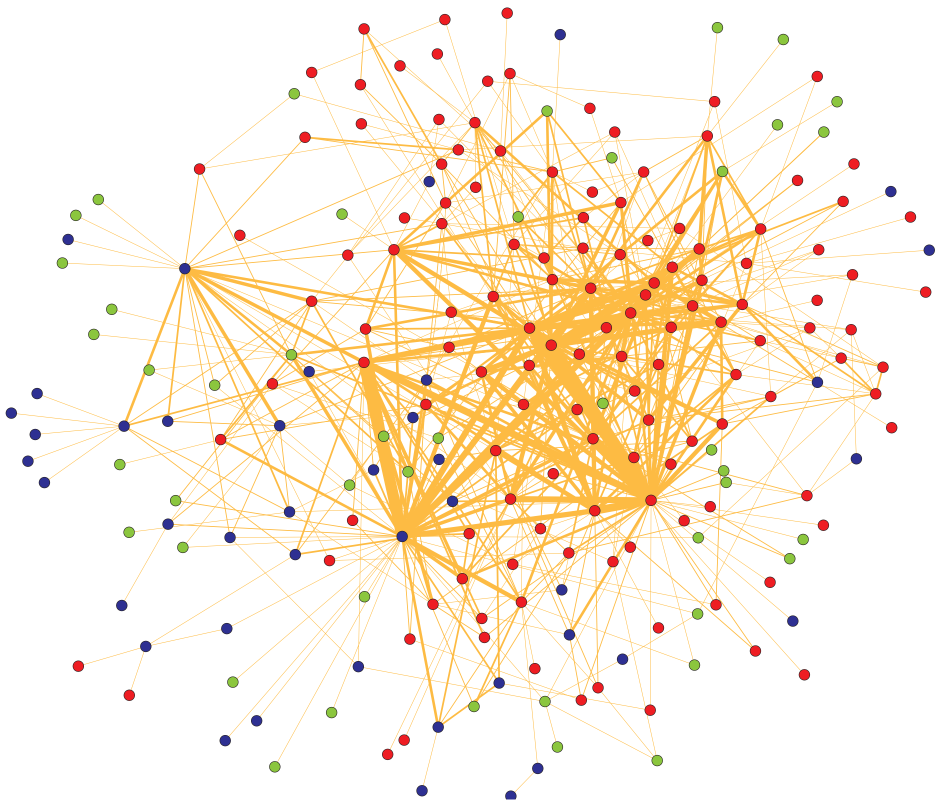
and $\mathcal{V}(\mathcal{L}')$ is the set of all endpoints of the lines from $\mathcal{L}'$.

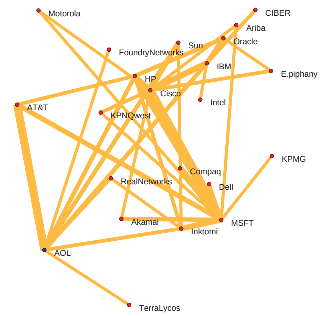
Vertex-cut: Krebs Internet Industries, core=6

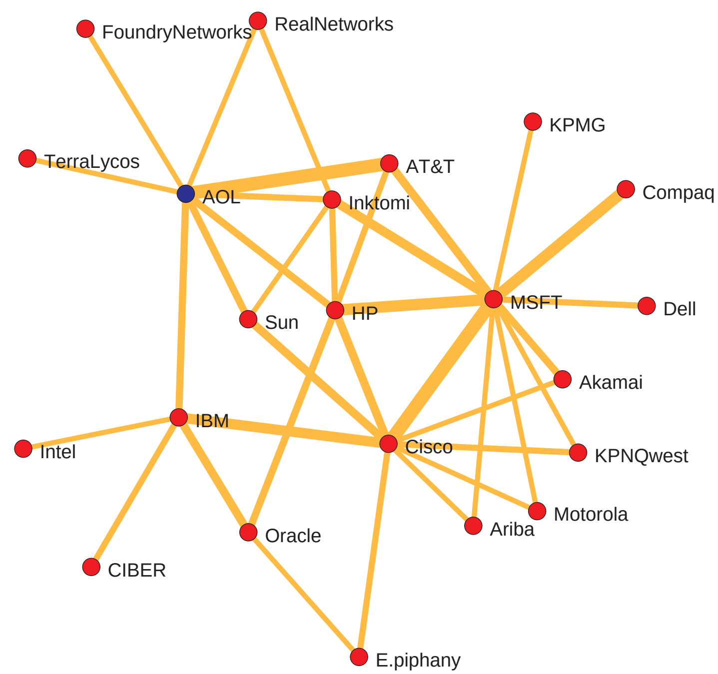


Each vertex represents a company that competes in the Internet industry, 1998 do 2001. $n = 219$, $m = 631$. red – content, blue – infrastructure, green – commerce. Two companies are linked with an edge if they have announced a joint venture, strategic alliance or other partnership.

Line-cut: Krebs Internet Industries, $w_3 \geq 5$







Line-cut in EAT

```
File/Network/read  eatRS.net
Info/Network/Line values          ... >= 70
Net/Transform/Remove/Lines with Value/lower than   70
Net/Partitions/Degree/All
Operations/Extract from Network/Partition   1-*
Net/Components/Weak
Draw/Draw-Partition
```

Simple analysis using cuts

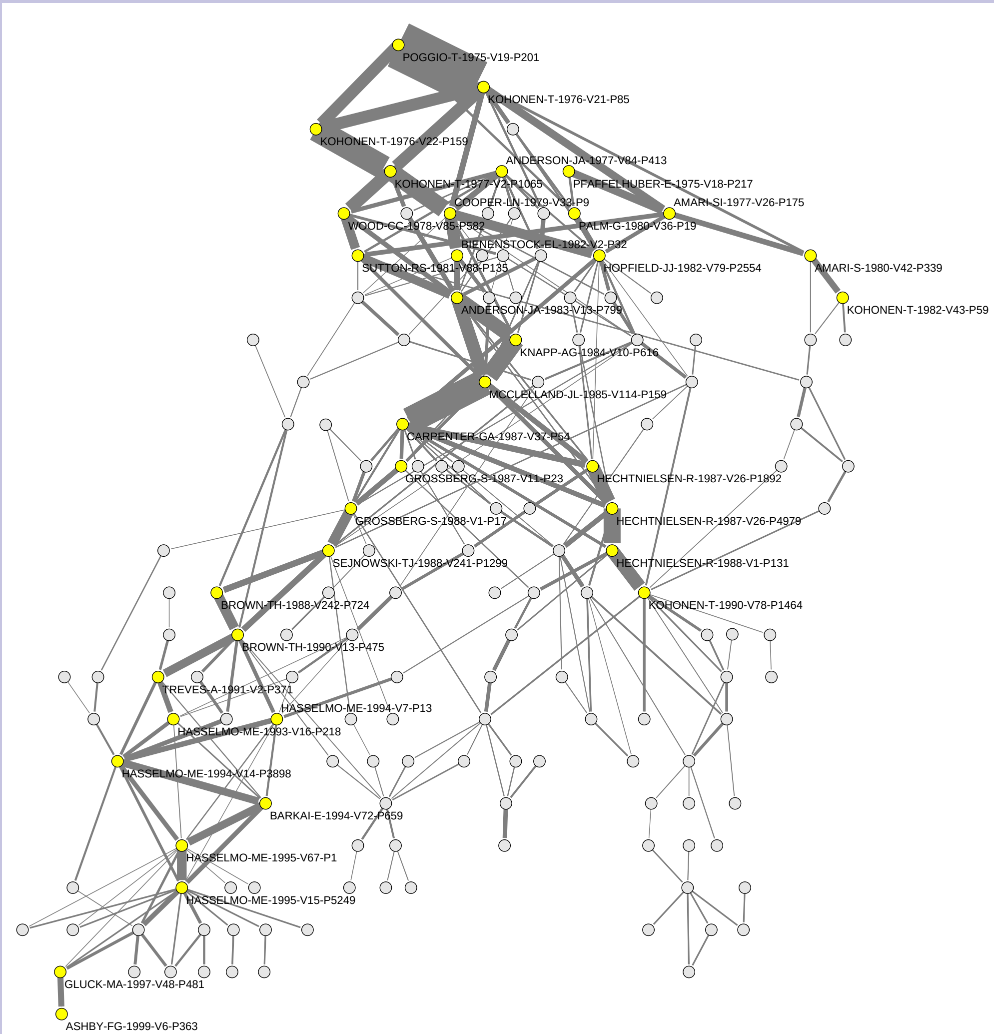
We look at the components of $\mathcal{N}(t)$.

Their number and sizes depend on t. Usually there are many small components. Often we consider only components of size at least k and not exceeding K. The components of size smaller than k are discarded as 'noninteresting'; and the components of size larger than K are cut again at some higher level.

The values of thresholds t, k and K are determined by inspecting the distribution of vertex/arc-values and the distribution of component sizes and considering additional knowledge on the nature of network or goals of analysis.

We developed some new and efficiently computable properties/weights.

Citation weights



The citation network analysis started in 1964 with the paper of Garfield et al. In 1989 Hummon and Doreian proposed three indices – weights of arcs that are proportional to the number of different source-sink paths passing through the arc. We developed algorithms to efficiently compute these indices.

Main subnetwork (arc cut at level 0.007) of the SOM (selforganizing maps) citation network (4470 vertices, 12731 arcs).

See paper.

Biconnectivity

Vertices u and v are *biconnected* iff they are connected (in both directions) by two independent (no common internal vertex) paths.

Biconnectivity determines a partition of the set of lines.

A vertex is an *articulation* vertex iff its deletion increases the number of weak components in a graph.

A line is a *bridge* iff its deletion increases the number of weak components in a graph.

k-connectivity

Vertex connectivity κ of graph $\mathcal{G}$ is equal to the smallest number of vertices that, if deleted, induce a disconnected graph or the trivial graph K_1.

Line connectivity λ of graph $\mathcal{G}$ is equal to the smallest number of lines that, if deleted, induce a disconnected graph or the trivial graph K_1.

Whitney's inequality: $\kappa(\mathcal{G}) \leq \lambda(\mathcal{G}) \leq \delta(\mathcal{G})$.

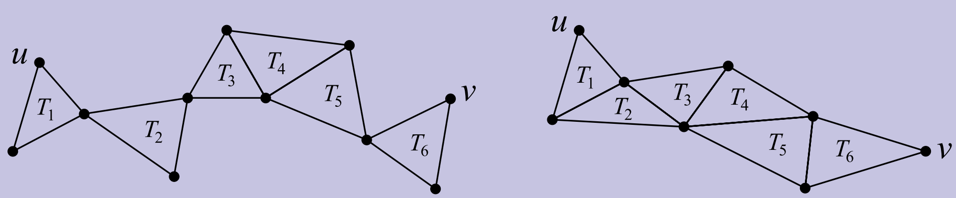
Graph $\mathcal{G}$ is *(vertex) $k-$connected*, if $\kappa(\mathcal{G}) \geq k$ and is *line $k-$connected*, if $\lambda(\mathcal{G}) \geq k$.

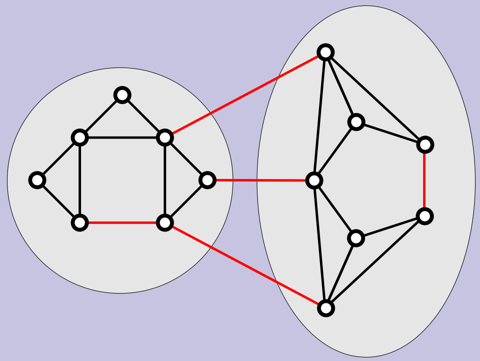
Whitney / Menger theorem: Graph $\mathcal{G}$ is vertex/line $k-$connected iff every pair of vertices can be connected with k vertex/line internally disjoint (semi)walks.

Triangular and short cycle connectivities

In an undirected graph we call a *triangle* a subgraph isomorphic to K_3.

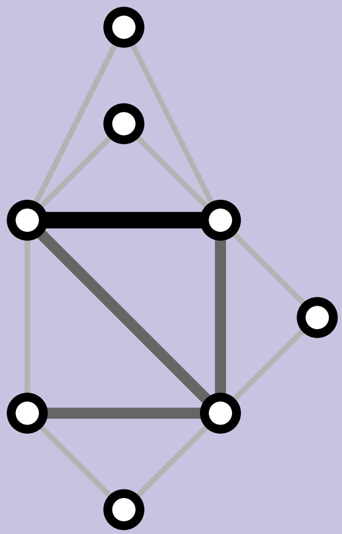
A sequence $(T_1, T_2, \ldots, T_s)$ of triangles of $\mathcal{G}$ *(vertex) triangularly connects* vertices $u, v \in \mathcal{V}$ iff $u \in T_1$ and $v \in T_s$ or $u \in T_s$ and $v \in T_1$ and $\mathcal{V}(T_{i-1}) \cap \mathcal{V}(T_i) \neq \emptyset$, $i = 2, \ldots s$. It *edge triangularly connects* vertices $u, v \in \mathcal{V}$ iff a stronger version of the second condition holds $\mathcal{E}(T_{i-1}) \cap \mathcal{E}(T_i) \neq \emptyset$, $i = 2, \ldots s$.





Vertex triangular connectivity is an equivalence on $\mathcal{V}$; and edge triangular connectivity is an equivalence on $\mathcal{E}$. See the paper.

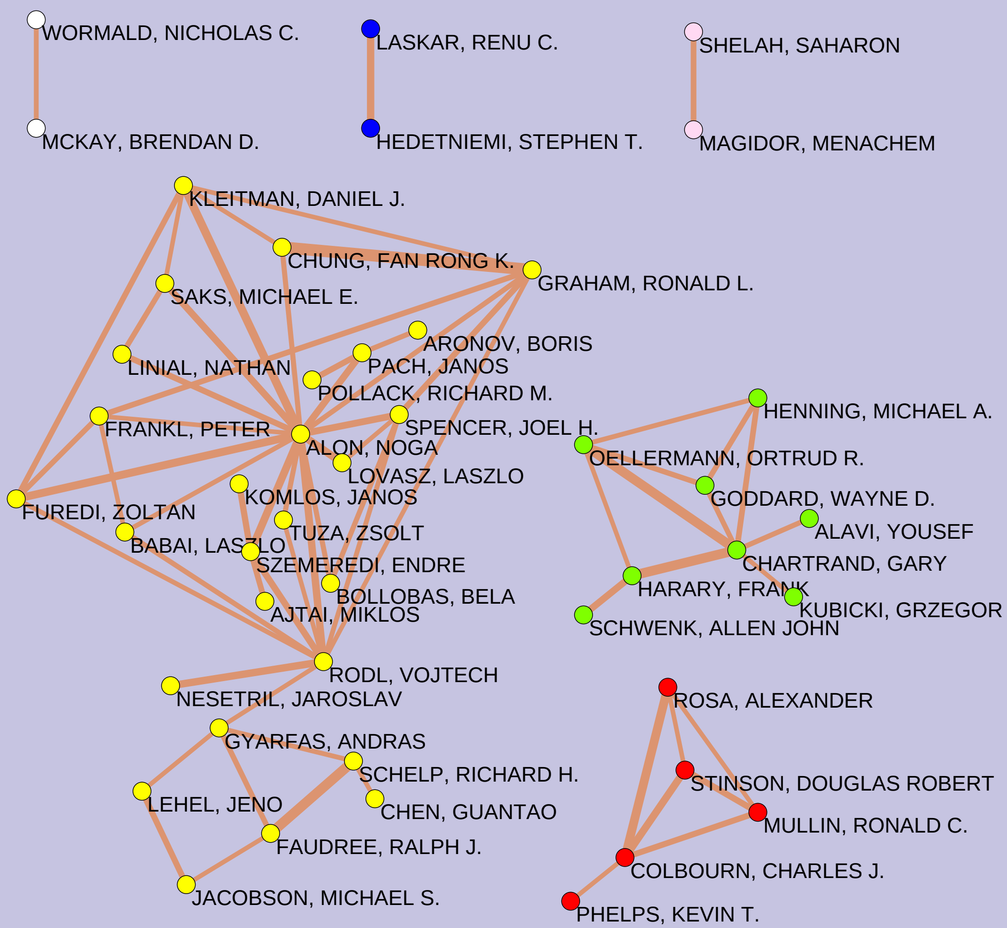
Triangular network



Let $\mathcal{G}$ be a simple undirected graph. A *triangular network* $\mathcal{N}_T(\mathcal{G}) = (\mathcal{V}, \mathcal{E}_T, w)$ determined by $\mathcal{G}$ is a subgraph $\mathcal{G}_T = (\mathcal{V}, \mathcal{E}_T)$ of $\mathcal{G}$ which set of edges $\mathcal{E}_T$ consists of all triangular edges of $\mathcal{E}(\mathcal{G})$. For $e \in \mathcal{E}_T$ the weight $w(e)$ equals to the number of different triangles in $\mathcal{G}$ to which e belongs.

Triangular networks can be used to efficiently identify dense clique-like parts of a graph. If an edge e belongs to a k-clique in $\mathcal{G}$ then $w(e) \geq k - 2$.

Edge-cut at level 16 of triangular network of Erdős collaboration graph



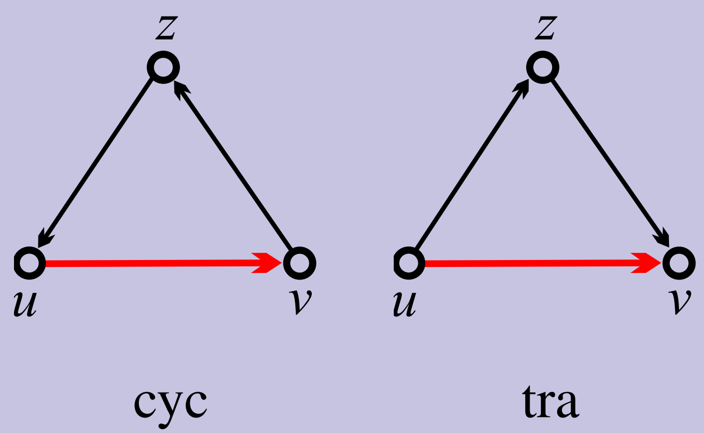
without Erdős,

$n = 6926$,

$m = 11343$

Triangular connectivity in directed graphs

If the graph $\mathcal{G}$ is mixed we replace edges with pairs of opposite arcs. In the following let $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ be a simple directed graph without loops. For a selected arc $(u, v) \in \mathcal{A}$ there are only two different types of directed triangles: **cyc**lic and **tra**nsitive.



For each type we get the corresponding triangular network $\mathcal{N}_{cyc}$ and $\mathcal{N}_{tra}$.

The notion of triangular connectivity can be extended to the notion of *short (semi) cycle connectivity*.

Arc-cut at level 11 of transitive triangular network of ODLIS dictionary