Introduction to Network Analysis using Pajek

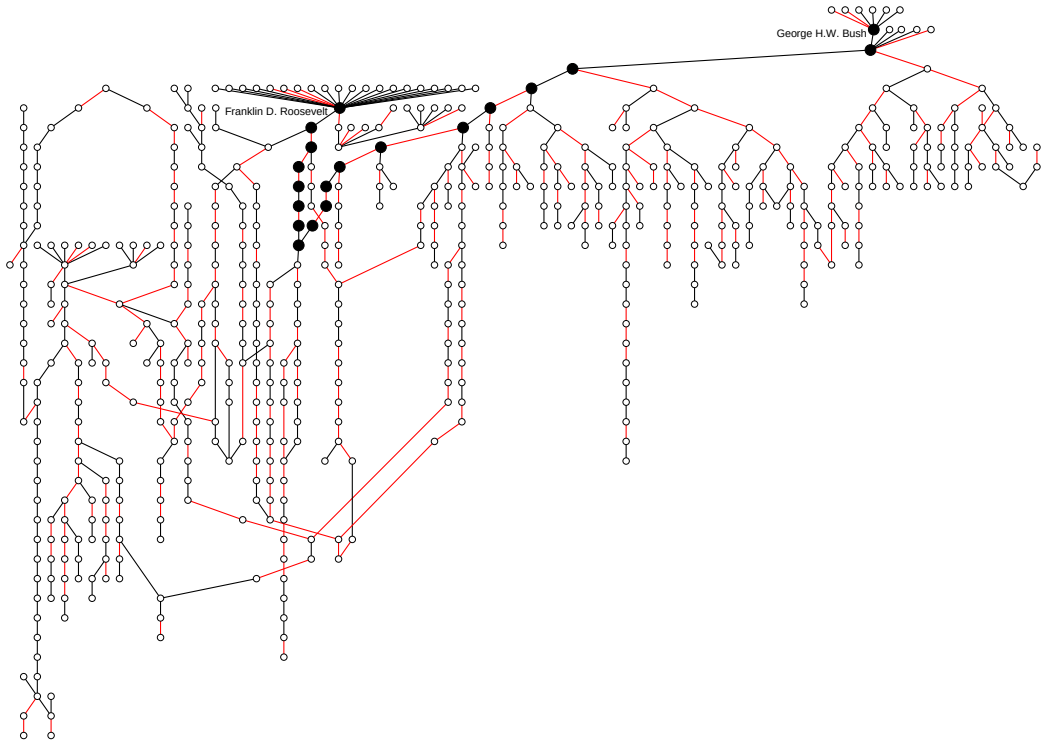
6. Structure of networks 4
Acyclic networks and Patterns search

Vladimir Batagelj

IMFM Ljubljana and IAM Koper

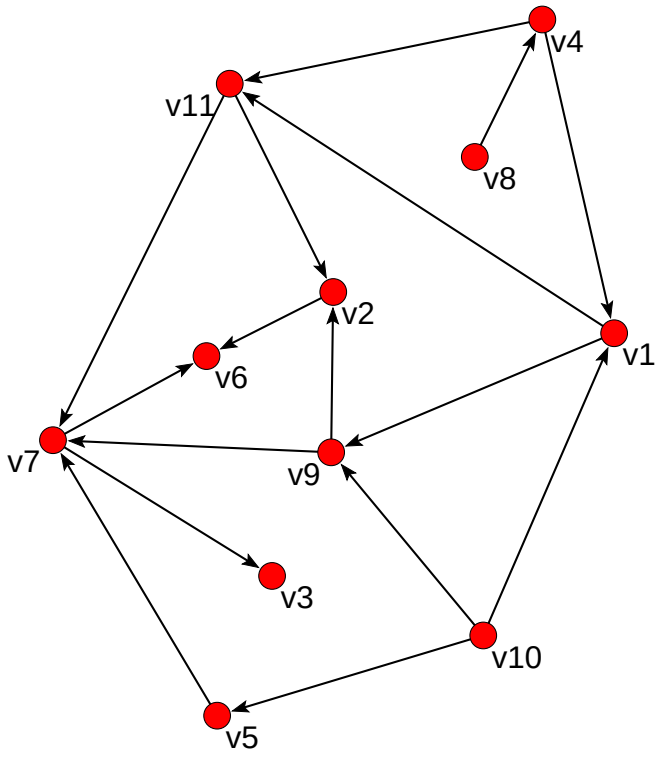
Phd program on Statistics
University of Ljubljana, 2018

Outline



e-mail: vladimir.batagelj@fmf.uni-lj.si
wiki: http://vladowiki.fmf.uni-lj.si/doku.php?id=pajek:ev:pde
version: May 8, 2018

Acyclic networks



Network $\mathcal{G} = (\mathcal{V}, R)$, $R \subseteq \mathcal{V} \times \mathcal{V}$ is acyclic, if it doesn't contain any (proper) cycle.

$$\overline{R} \cap I = \emptyset$$

In some cases we allow loops. Examples: citation networks, genealogies, project networks, ...

In real-life acyclic networks we usually have a node property $p : \mathcal{V} \rightarrow \mathbb{R}$ (most often time), that is compatible with arcs

$$(u, v) \in R \Rightarrow p(u) < p(v)$$

acyclic.paj

`Network/Create Partition/Components/Strong [2]`

Basic properties of acyclic networks

Let $\mathcal{G} = (\mathcal{V}, R)$ be acyclic and $\mathcal{U} \subseteq \mathcal{V}$, then $\mathcal{G}|\mathcal{U} = (\mathcal{U}, R|\mathcal{U})$, $R|\mathcal{U} = R \cap \mathcal{U} \times \mathcal{U}$ is also acyclic.

Let $\mathcal{G} = (\mathcal{V}, R)$ be acyclic, then $\mathcal{G}' = (\mathcal{V}, R^{-1})$ is also acyclic. Duality.

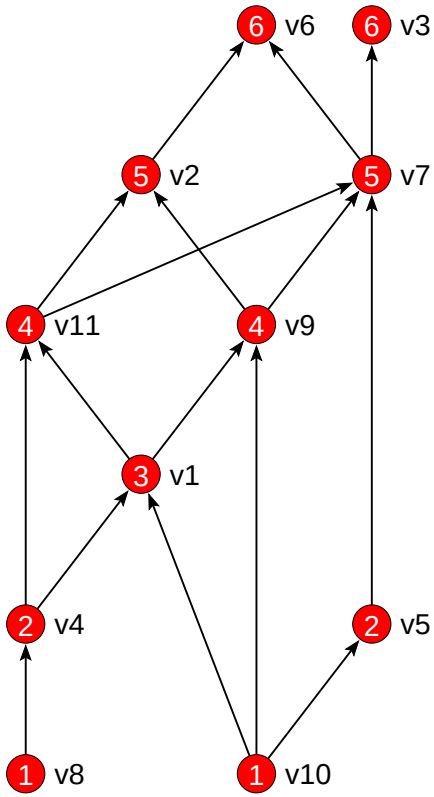
The set of sources $\mathrm{Min}_R(\mathcal{V}) = \{v : \neg\exists u \in \mathcal{V} : (u, v) \in R\}$ and the set of sinks $\mathrm{Max}_R(\mathcal{V}) = \{v : \neg\exists u \in \mathcal{V} : (v, u) \in R\}$ are nonempty (in finite networks).

Transitive closure $\overline{R}$ of an acyclic relation R is acyclic.

Relation Q is a skeleton of relation R iff $Q \subseteq R$, $\overline{Q} = \overline{R}$ and relation Q is minimal such relation – no arc can be deleted from it without destroying the second property.

A general relation (graph) can have several skeletons; but in a case of acyclic relation it is uniquely determined $Q = R \setminus R * \overline{R}$.

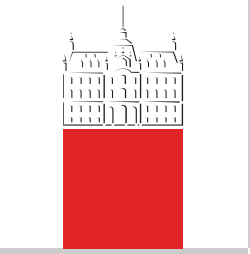
Depth



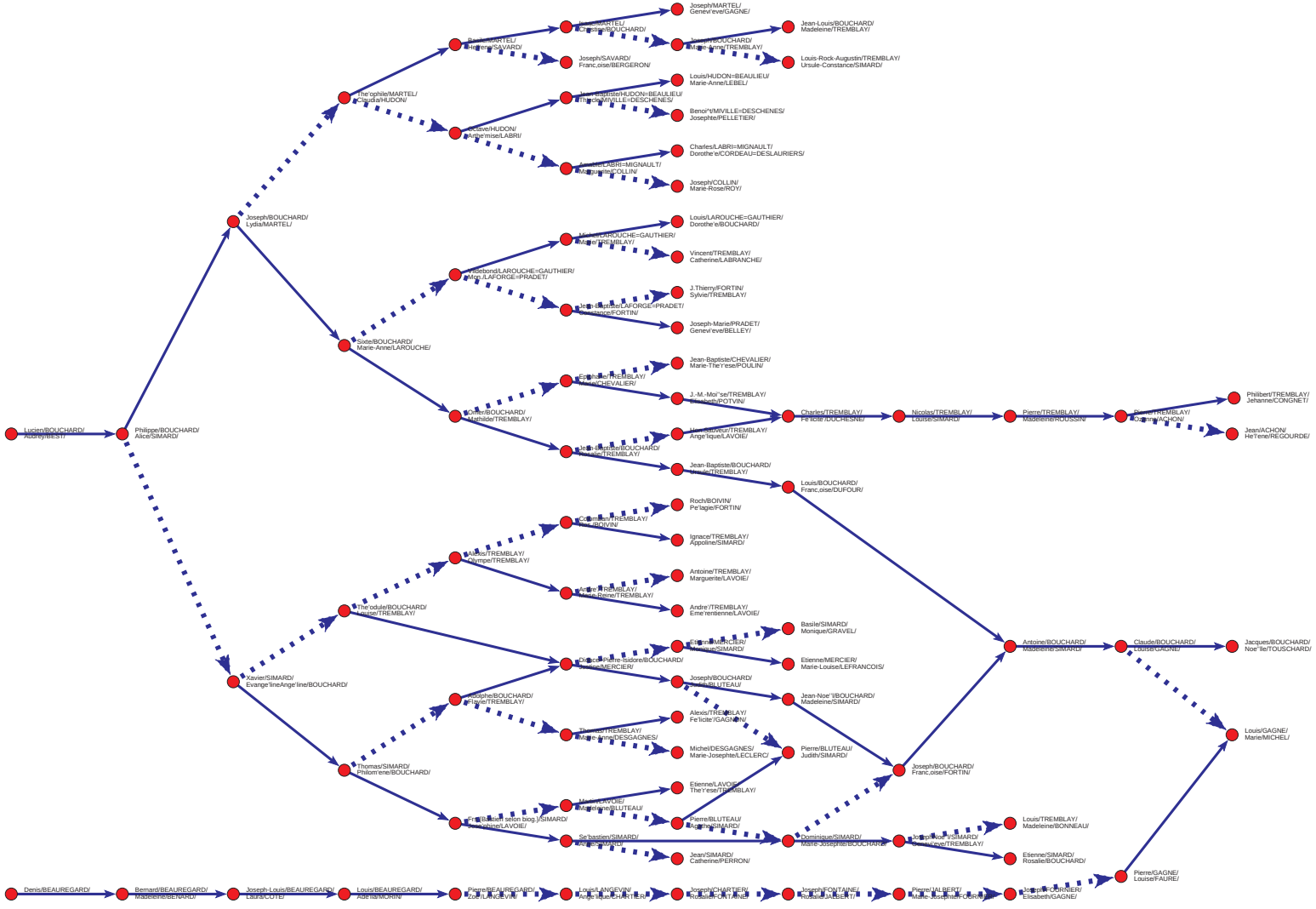
Mapping $h : \mathcal{V} \rightarrow \mathbb{N}^+$ is called depth or level if all differences on the longest path and the initial value equal to 1.

$\mathcal{U} \leftarrow \mathcal{V}$; $k \leftarrow 0$
while $\mathcal{U} \neq \emptyset$ **do**
  $\mathcal{T} \leftarrow \mathrm{Min}_R(\mathcal{U})$; $k \leftarrow k + 1$
  for $v \in \mathcal{T}$ **do** $h(v) \leftarrow k$
  $\mathcal{U} \leftarrow \mathcal{U} \setminus \mathcal{T}$

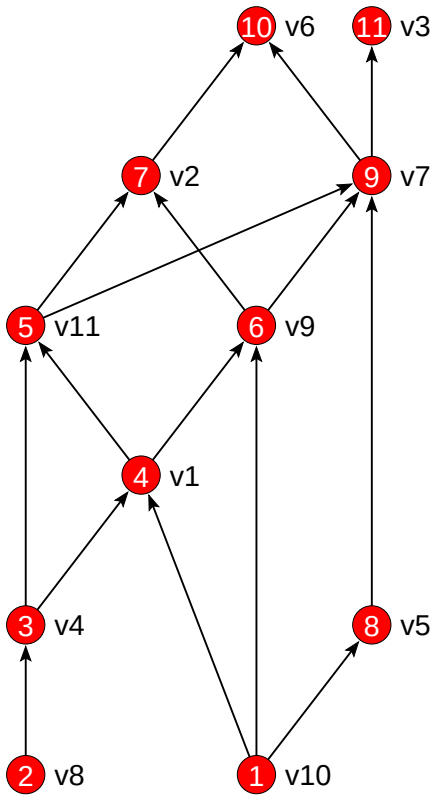
Drawing on levels. Macro `Layers`.



p-graph of Bouchard's genealogy



Topological numberings

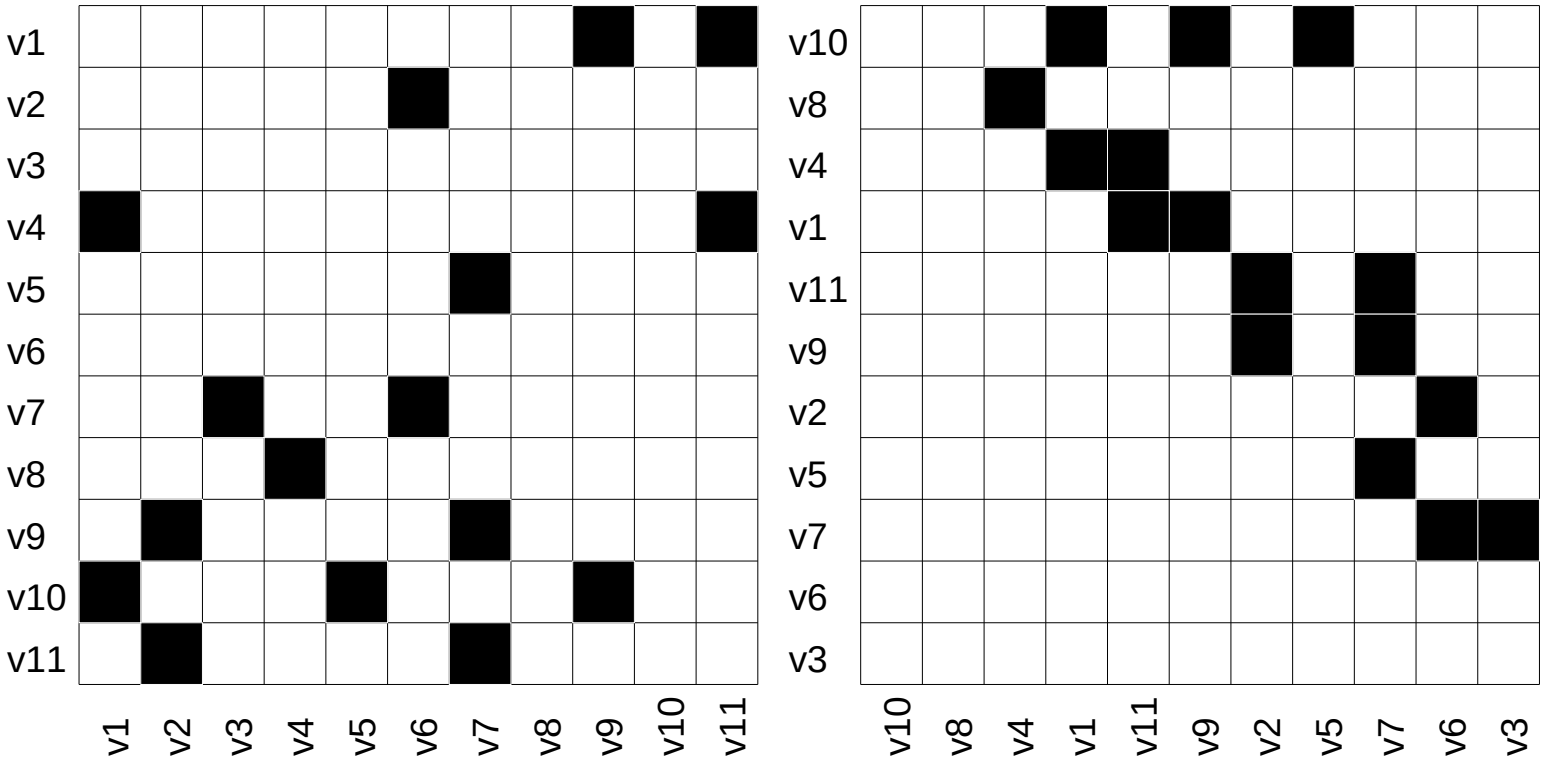


Injective mapping $h : \mathcal{V} \rightarrow 1..|\mathcal{V}|$ compatible with relation R is called a topological numbering.
'Topological sort'

$\mathcal{U} \leftarrow \mathcal{V}$; $k \leftarrow 0$
while $\mathcal{U} \neq \emptyset$ **do**
  select $v \in \mathrm{Min}_R(\mathcal{U})$; $k \leftarrow k + 1$
  $h(v) \leftarrow k$
  $\mathcal{U} \leftarrow \mathcal{U} \setminus \{v\}$

Matrix display of acyclic network with vertices reordered according to a topological numbering has a zero lower triangle.

. . . Topological numberings




```
read or select  [Acyclic.paj]
Network/Acyclic Natwork/Depth Partition/Acyclic
Partition/Make Permutation
File/Network/Export as Matrix to EPS/Using Permutation [acy.eps]
```

Topological numberings and functions on acyclic networks

Let the function $f : \mathcal{V} \rightarrow \mathbb{R}$ be defined in the following way:

- $f(v)$ is knownn in sources $v \in \mathrm{Min}_R(\mathcal{V})$
- $f(v) = F(\{f(u) : uRv\})$

If we compute the values of function f in a sequence determined by a topological numbering we can compute them in one pass since for each node $v \in \mathcal{V}$ the values of f needed for its computation are already known.

Topological numberings – CPM

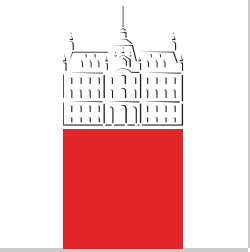
CPM (Critical Path Method): A project consists of tasks. Nodes of a project network represent states of the project and arcs represent tasks. Every project network is acyclic. For each task (u, v) its execution time $t(u, v)$ is known. A task can start only when all the preceeding tasks are finished. We want to know what is the shortest time in which the project can be completed.

Let $T(v)$ denotes the earliest time of completion of all tasks entering the state v.

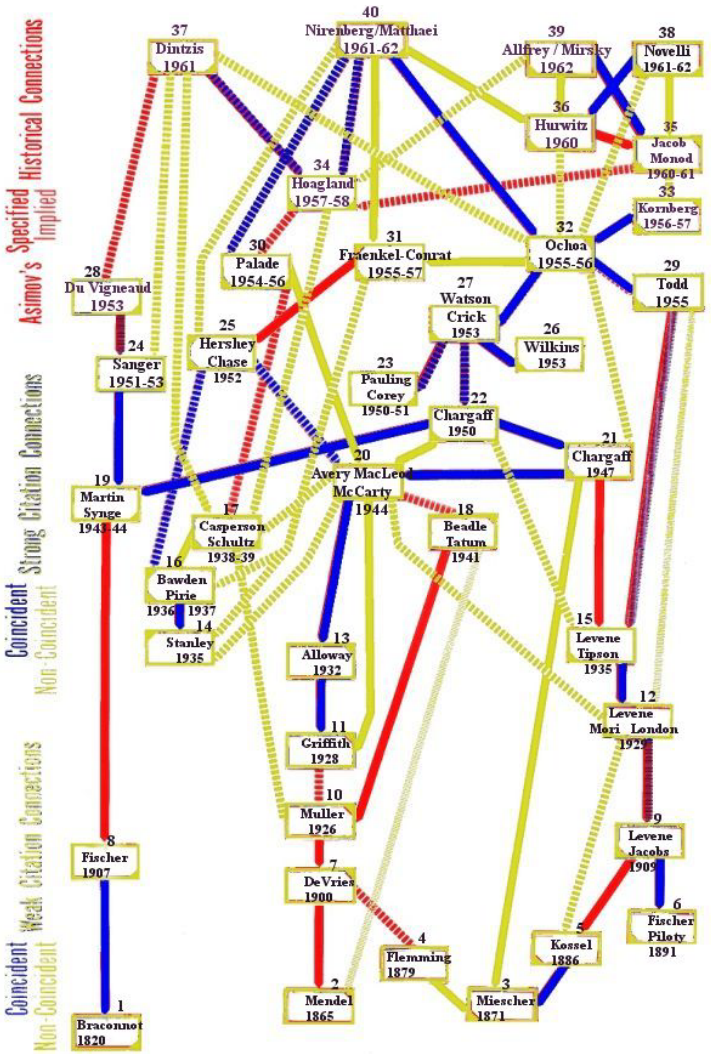
$$T(v) = 0, \qquad v \in \mathrm{Min}_R(\mathcal{V})$$

$$T(v) = \max_{u:uRv}(T(u) + t(u, v))$$

`Network/Acyclic Network/Critical Path Method-CPM`



Citation networks



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 provide us with automatic way to identify the (most) important part of the citation network. For two of these indices we developed algorithms to efficiently compute them.

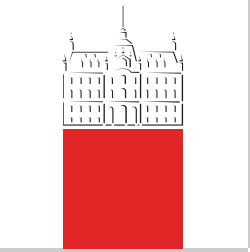
. . . Citation networks

In a given set of units/nodes $\mathcal{U}$ (articles, books, works, etc.) we introduce a citing relation/set of arcs $R \subseteq \mathcal{U} \times \mathcal{U}$

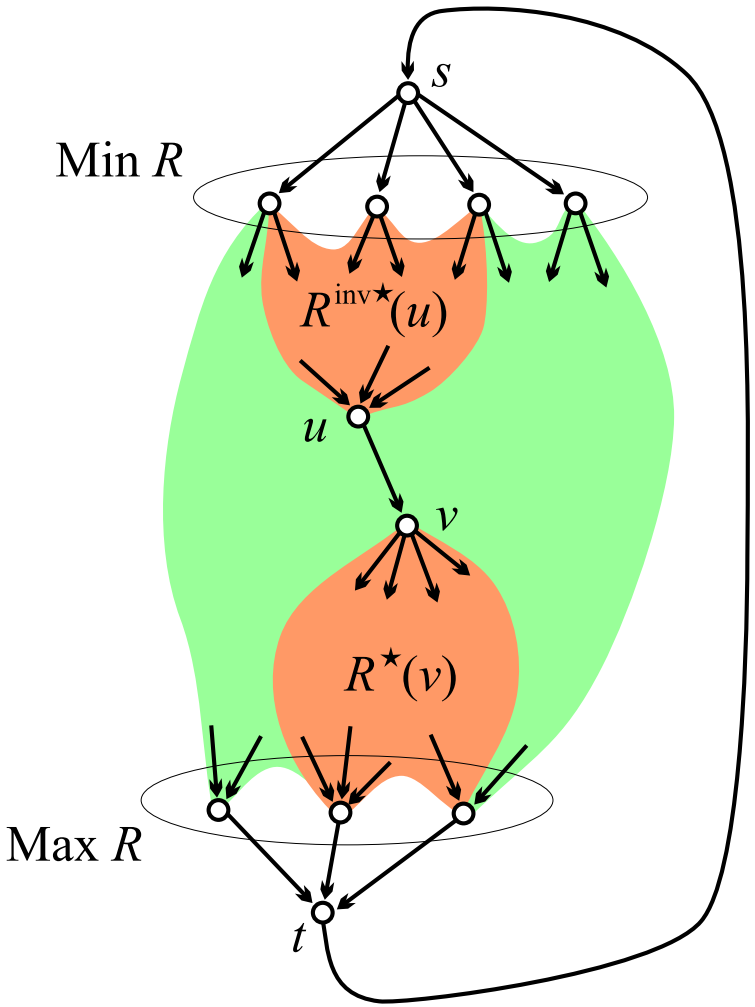
$$uRv \equiv u \text{ cites } v$$

which determines a citation network $\mathcal{N} = (\mathcal{U}, R)$.

A citing relation is usually irreflexive (no loops) and (almost) acyclic. We shall assume that it has these two properties. Since in real-life citation networks the strong components are small (usually 2 or 3 nodes) we can transform such network into an acyclic network by shrinking strong components and deleting loops. Other approaches exist. It is also useful to transform a citation network to its standardized form by adding a common source node $s \notin \mathcal{U}$ and a common sink node $t \notin \mathcal{U}$. The source s is linked by an arc to all minimal elements of R; and all maximal elements of R are linked to the sink t. We add also the 'feedback' arc (t, s).



Search path count method



The search path count (SPC) method is based on counters $n(u, v)$ that count the number of different paths from s to t through the arc (u, v). To compute $n(u, v)$ we introduce two auxiliary quantities: $n^{-}(v)$ counts the number of different paths from s to v, and $n^{+}(v)$ counts the number of different paths from v to t.

Fast algorithm for SPC

It follows by basic principles of combinatorics that

$$n(u,v) = n^-(u) \cdot n^+(v), \qquad (u,v) \in R$$

where

$$n^-(u) = \begin{cases} 1 & u = s \\ \sum_{v:v\mathbf{R}u} n^-(v) & otherwise \end{cases}$$

and

$$n^+(u) = \begin{cases} 1 & u = t \\ \sum_{v:u\mathbf{R}v} n^+(v) & otherwise \end{cases}$$

This is the basis of an efficient algorithm for computing $n(u,v)$ – after the topological sort of the graph we can compute, using the above relations in topological order, the weights in time of order $O(m)$, $m = |R|$. The topological order ensures that all the quantities in the right sides of the above equalities are already computed when needed.

Hummon and Doreian indices and SPC

The Hummon and Doreian indices are defined as follows:

- search path link count (SPLC) method: $w_l(u, v)$ equals the number of "*all possible search paths through the network emanating from an origin node*" through the arc $(u, v) \in R$.
- search path node pair (SPNP) method: $w_p(u, v)$ "*accounts for all connected node pairs along the paths through the arc* $(u, v) \in R$".

We get the SPLC weights by applying the SPC method on the network obtained from a given standardized network by linking the source s by an arc to each nonminimal vertex from $\mathcal{U}$; and the SPNP weights by applying the SPC method on the network obtained from the SPLC network by additionally linking by an arc each nonmaximal vertex from $\mathcal{U}$ to the sink t.

Node weights

The quantities used to compute the arc weights w can be used also to define the corresponding node weights t

$$\begin{aligned} t_c(u) &= n^-(u) \cdot n^+(u) \\ t_l(u) &= n_l^-(u) \cdot n_l^+(u) \\ t_p(u) &= n_p^-(u) \cdot n_p^+(u) \end{aligned}$$

They are counting the number of paths of selected type through the node u.

`Network/Acyclic Network/Citation Weights`

Properties of SPC weights

The values of counters $n(u, v)$ form a flow in the citation network – the Kirchoff's node law holds: For every node u in a standardized citation network *incoming flow* = *outgoing flow*:

$$\sum_{v:vRu} n(v, u) = \sum_{v:uRv} n(u, v) = n^-(u) \cdot n^+(u)$$

The weight $n(t, s)$ equals to the total flow through network and provides a natural normalization of weights

$$w(u, v) = \frac{n(u, v)}{n(t, s)} \quad \Rightarrow \quad 0 \leq w(u, v) \leq 1$$

and if C is a minimal arc-cut-set $\sum_{(u,v)\in C} w(u, v) = 1$.

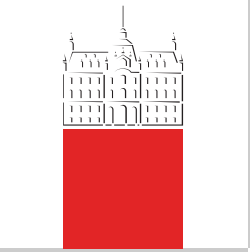
In large networks the values of weights can grow very large. This should be considered in the implementation of the algorithms.

Nonacyclic citation networks

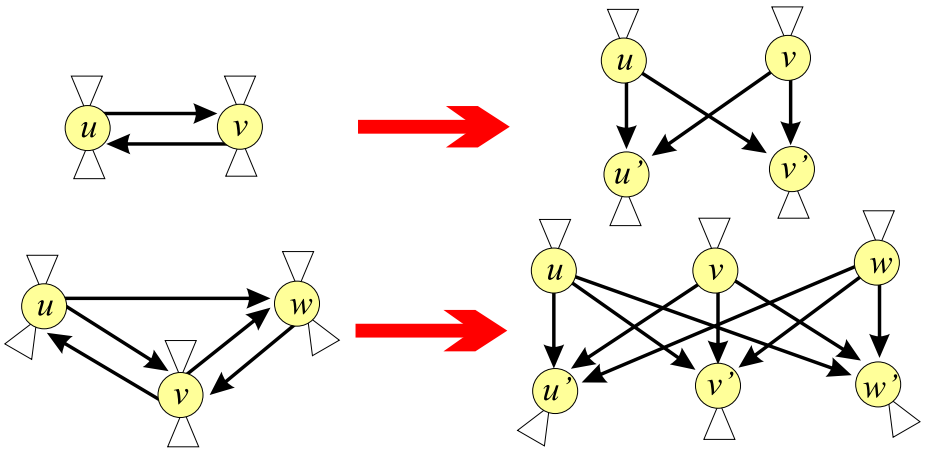
If there is a cycle in a network then there is also an infinite number of trails between some units. There are some standard approaches to overcome the problem: to introduce some 'aging' factor which makes the total weight of all trails converge to some finite value; or to restrict the definition of a weight to some finite subset of trails – for example paths or geodesics. But, new problems arise: What is the right value of the 'aging' factor? Is there an efficient algorithm to count the restricted trails?

The other possibility, since a citation network is usually almost acyclic, is to transform it into an acyclic network

- by identification (shrinking) of cyclic groups (nontrivial strong components), or
- by deleting some arcs, or
- by transformations such as the 'preprint' transformation.



Preprint transformation



The preprint transformation is based on the following idea: Each paper from a strong component is duplicated with its 'preprint' version. The papers inside strong component cite preprints.

Large strong components in citation network are unlikely – their presence usually indicates an error in the data.

An exception from this rule is the HEP citation network of High Energy Particle Physics literature from **arXiv**. In it different versions of the same paper are treated as a unit. This leads to large strongly connected components. The idea of preprint transformation could be used also in this case to eliminate cycles.

Probabilistic flow

Another way to measure the importance of nodes and arcs in acyclic networks is the following. Let $\mathcal{N} = (\mathcal{V}, \mathcal{A})$ be a standardized acyclic network with source $s \in \mathcal{V}$ and sink $t \in \mathcal{V}$. The node potential, $p(v)$, is defined by

$$p(s) = 1 \quad \text{and} \quad p(v) = \sum_{u:(u,v)\in\mathcal{A}} \frac{p(u)}{\text{outdeg}(u)}$$

The flow on the arc (u, v) is defined as $\varphi(u, v) = \frac{p(u)}{\text{outdeg}(u)}$. It follows immediately that

$$p(v) = \sum_{u:(u,v)\in\mathcal{A}} \varphi(u, v)$$

and also,

$$\sum_{u:(v,u)\in\mathcal{A}} \varphi(v, u) = \sum_{u:(v,u)\in\mathcal{A}} \frac{p(v)}{\text{outdeg}(v)} = \frac{p(v)}{\text{outdeg}(v)} \sum_{u:(v,u)\in\mathcal{A}} 1 = p(v)$$

. . . probabilistic flow

Therefore, for each $v \in \mathcal{V}$

$$\sum_{u:(u,v)\in\mathcal{A}} \varphi(u,v) = \sum_{u:(v,u)\in\mathcal{A}} \varphi(v,u) = p(v)$$

which states that Kirchoff's law holds for the flow φ.
The probabilistic interpretation of flows has two parts:

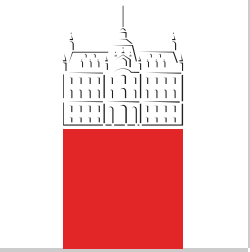
1. The node potential of v, $p(v)$, is equal to the probability that a random walk starting in the source s goes through the node v, and
2. The arc flow on (u,v), $\varphi(u,v)$, is equal to the probability that a random walk starting in the source, s, goes through the arc (u,v).

Note that the measures p and φ consider only "users" (future) and do not depend on the past.

. . . probabilistic flow

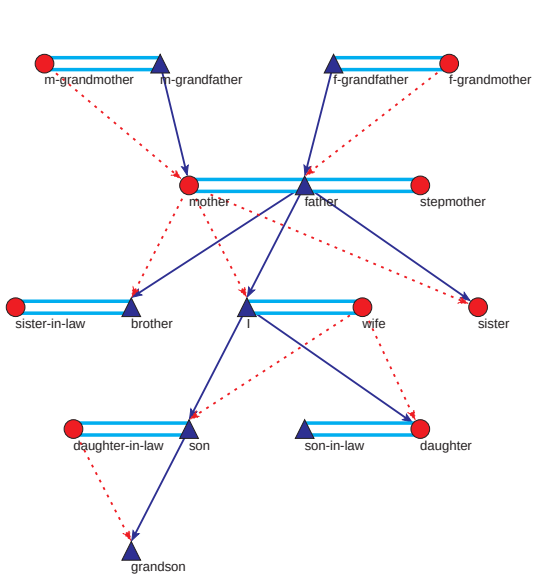
SN5 citation network, flows multiplied with 10^6

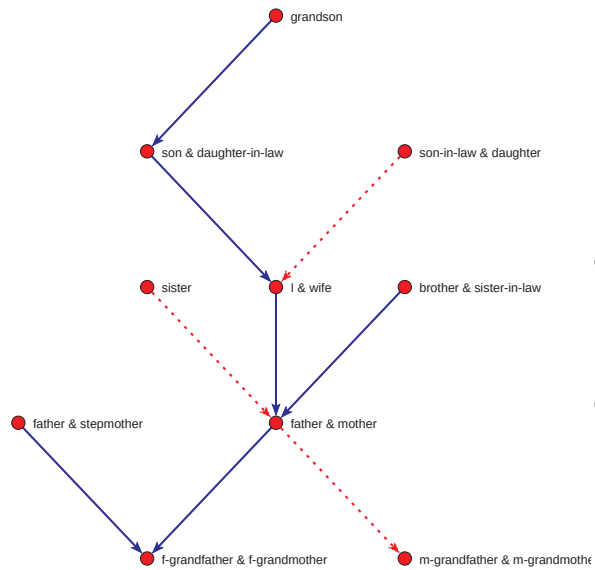
1	BARABASI_A(1999)286:509	2481.1796	26	SCHOUTEN_L(1993)22:369	701.4421
2	WATTS_D(1998)393:440	2413.1823	27	LANEMAN_J(2004)50:3062	697.3903
3	ALBERT_R(2002)74:47	2099.6951	28	BOYD_S(2004):	681.9335
4	WASSERMA_S(1994):	1807.7400	29	BARABASI_A(2004)5:101	662.5071
5	RONAYNE_J(1987):	1697.2066	30	AMARAL_L(2000)97:11149	659.7386
6	WANT_R(1992)10:91	1694.8577	31	CARZANIG_A(2001)19:332	658.6667
7	JEONG_H(2001)411:41	1656.5485	32	BURT_R(1992):	635.2949
8	FREEMAN_L(1979)1:215	1559.2715	33	[ANONYMO(2008):	621.7552
9	NEWMAN_M(2003)45:167	1521.8437	34	JADBABAI_A(2003)48:988	599.2536
10	HOLBEN_B(1998)66:1	1494.6278	35	BORGATTI_S(2002):	594.1600
11	ALBERT_R(2000)406:378	1171.6774	36	ALBERT_R(1999)401:130	589.2179
12	JEONG_H(2000)407:651	1142.8359	37	NEWMAN_M(2001)98:404	584.6247
13	FREEMAN_L(1977)40:35	1083.6487	38	[ANONYMO(2006):	570.9648
14	GIRVAN_M(2002)99:7821	1055.2631	39	SHANNON_P(2003)13:2498	566.9214
15	[ANONYMO(2011):	940.7750	40	KATZELA_I(1996)3:10	558.6965
16	SELBY_P(1996)348:313	884.8034	41	LYNN_D(2008)4:	512.6603
17	ZADEH_L(1997)90:111	873.1572	42	BLUMENTH_D(1994)82:1650	509.7068
18	[ANONYMO(2009):	808.5712	43	[ANONYMO(2007):	508.8429
19	[ANONYMO(2010):	789.0410	44	DOROGOVT_S(2002)51:1079	505.0996
20	STROGATZ_S(2001)410:268	785.9130	45	SCOTT_J(2000):	497.5110
21	BOCCALET_S(2006)424:175	782.6379	46	RAVASZ_E(2002)297:1551	496.6379
22	GRANOVET(1973)78:1360	777.3402	47	UETZ_P(2000)403:623	496.3690
23	PARTON_R(1994)127:1199	751.0084	48	ERDOS_P(1959)6:290	483.7304
24	GAREY_M(1979):	734.1673	49	DIAMOND_D(2008)75:606	466.9622
25	HAGMANN_P(2008)6:1479	718.6859	50	VANLANSC_J(2001)91:1574	465.9244

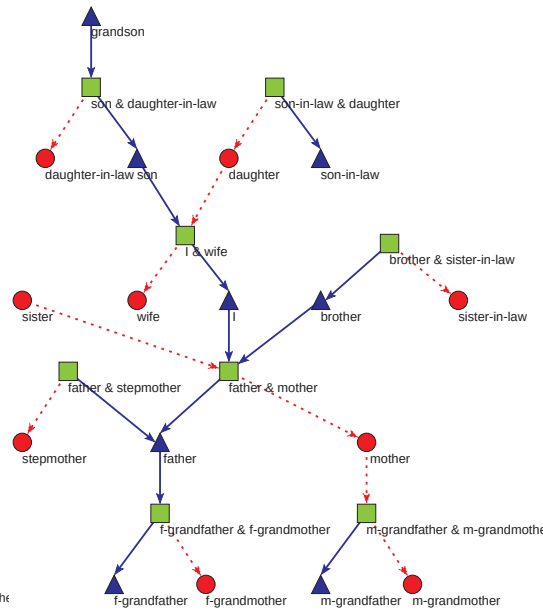


Genealogies

Another example of acyclic networks are genealogies. In 'Sources' we already described the following network representations of genealogies:







Ore graph, *p*-graph, and bipartite *p*-graph

paper

Properties of representations

p-graphs and bipartite *p*-graphs have many advantages:

- there are less nodes and links in *p*-graphs than in corresponding Ore graphs;
- *p*-graphs are directed, acyclic networks;
- every semi-cycle of the *p*-graph corresponds to a relinking marriage. There exist two types of relinking marriages: blood marriage: e.g., marriage among brother and sister, and non-blood marriage: e.g., two brothers marry two sisters from another family.
- *p*-graphs are more suitable for analyses.

Bipartite *p*-graphs have an additional advantage: we can distinguish between a married uncle and a remarriage of a father. This property enables us, for example, to find marriages between half-brothers and half-sisters.

Genealogies are sparse networks

A genealogy is regular if every person in it has at most two parents. Genealogies are sparse networks – number of links is of the same order as the number of nodes.

For a regular Ore genealogy $(\mathcal{V}, (\mathcal{A}, \mathcal{E}))$ we have:

$$|\mathcal{A}| \leq 2|\mathcal{V}|, \quad |\mathcal{E}| \leq \frac{1}{2}|\mathcal{V}|, \quad |\mathcal{L}| = |\mathcal{A}| + |\mathcal{E}| \leq \frac{5}{2}|\mathcal{V}|$$

p-graphs are almost trees – deviations from trees are caused by relinking marriages ($\mathcal{V}_p$, $\mathcal{A}_p$ – nodes and arcs of p-graph, n_{mult} – # of nodes with multiple marriages):

$$|\mathcal{V}_p| = |\mathcal{V}| - |\mathcal{E}| + n_{mult}, \quad |\mathcal{V}| \geq |\mathcal{V}_p| \geq \frac{1}{2}|\mathcal{V}|, \quad |\mathcal{A}_p| \leq 2|\mathcal{V}_p|$$

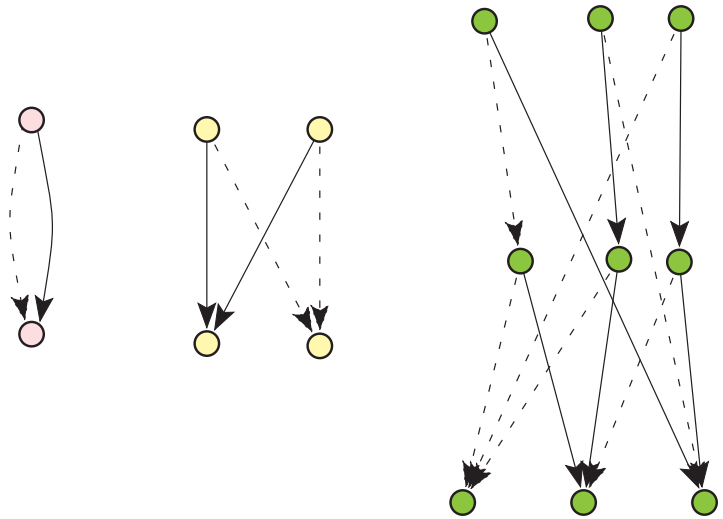
and for a bipartite p-graph, we have

$$|\mathcal{V}| \leq |\mathcal{V}_b| \leq \frac{3}{2}|\mathcal{V}|, \quad |\mathcal{A}_b| \leq 2|\mathcal{V}| + n_{mult}$$

Number of nodes and links in Ore and p-graphs

data	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{A}\rvert$	$\frac{\lvert\mathcal{L}\rvert}{\lvert\mathcal{V}\rvert}$	$\lvert\mathcal{V}_i\rvert$	n_{mult}	$\lvert\mathcal{V}_p\rvert$	$\lvert\mathcal{A}_p\rvert$	$\frac{\lvert\mathcal{A}_p\rvert}{\lvert\mathcal{V}_p\rvert}$
Drame	29606	8256	41814	1.69	13937	843	22193	21862	0.99
Hawlina	7405	2406	9908	1.66	2808	215	5214	5306	1.02
Marcus	702	215	919	1.62	292	20	507	496	0.98
Mazol	2532	856	3347	1.66	894	74	1750	1794	1.03
President	2145	978	2223	1.49	282	93	1260	1222	0.97
Royale	17774	7382	25822	1.87	4441	1431	11823	15063	1.27
Loka	47956	14154	68052	1.71	21074	1426	35228	36192	1.03
Silba	6427	2217	9627	1.84	2263	270	4480	5281	1.18
Ragusa	5999	2002	9315	1.89	2347	379	4376	5336	1.22
Tur	1269	407	1987	1.89	549	94	956	1114	1.17
Royal92	3010	1138	3724	1.62	1003	269	2141	2259	1.06
Little	25968	8778	34640	1.67	8412				1.01
Mumma	34224	11334	45565	1.66	11556				1.00
Tilltson	42559	12796	54043	1.57	16967				1.00

Relinking index



Let n denotes number of nodes in p-graph, m number of arcs, k number of weakly connected components, and M number of maximal nodes (nodes having output degree 0, $M \geq 1$).

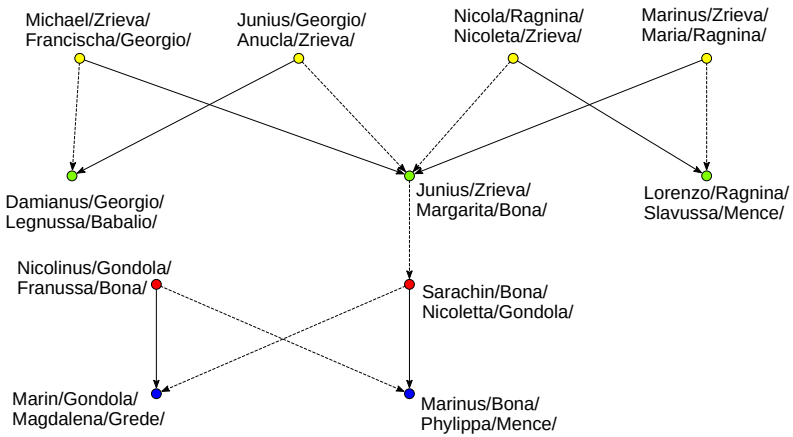
The relinking index is defined as:

$$RI = \frac{k + m - n}{k + n - 2M}$$

For a trivial graph (having only one node) we define $RI = 0$.
It holds $0 \leq RI \leq 1$. $RI = 0$ iff network is a forest.

Pattern searching

If a selected pattern determined by a given graph does not occur frequently in a sparse network the straightforward backtracking algorithm applied for pattern searching finds all appearences of the pattern very fast even in the case of very large networks. Pattern searching was successfully applied to searching for patterns of atoms in molecula (carbon rings) and searching for relinking marriages in genealogies.



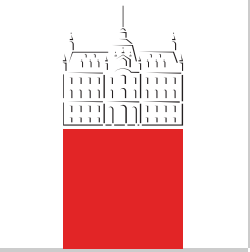
Three connected relinking marriages in the genealogy (represented as a *p*-graph) of ragusan noble families. A solid arc indicates the _ *is a son of* _ relation, and a dotted arc indicates the _ *is a daughter of* _ relation. In all three patterns a brother and a sister from one family found their partners in the same other family.

. . . Pattern searching

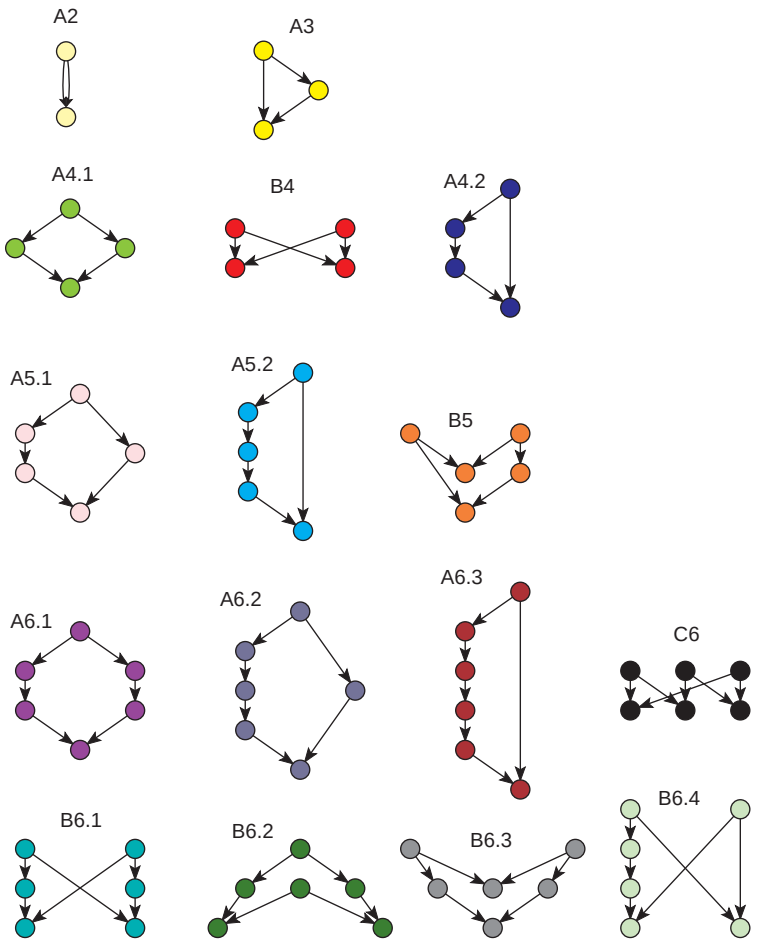
To speed up the search or to consider some additional properties of the pattern, a user can set some additional options:

- nodes in network should match with nodes in pattern in some nominal, ordinal or numerical property (for example, type of atom in molecula);
- values of edges must match (for example, edges representing male/female links in the case of *p-graphs*);
- the first node in the pattern can be selected only from a given subset of nodes in the network.

`Networks/Fragment (First in Second)`



Relinking patterns in *p*-graphs



frag16.paj

All possible relinking marriages in *p*-graphs with 2 to 6 nodes. Patterns are labeled as follows:

- first character – number of first nodes: A – single, B – two, C – three.
- second character: number of nodes in pattern (2, 3, 4, 5, or 6).
- last character: identifier (if the two first characters are identical).

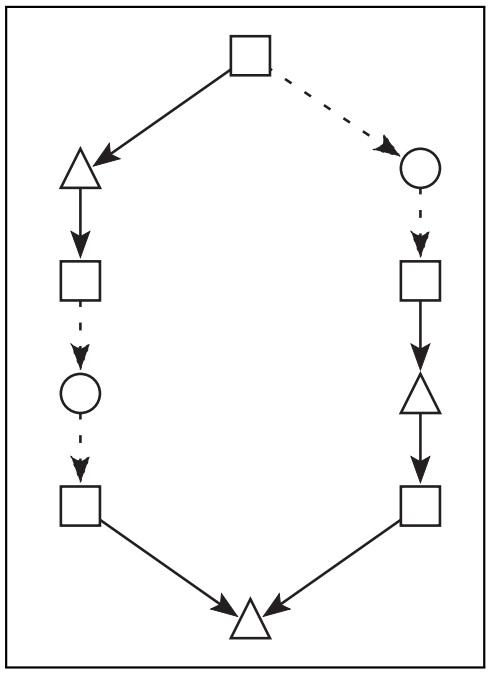
Patterns denoted by A are exactly the blood marriages. In every pattern the number of first nodes is equal to the number of last nodes.

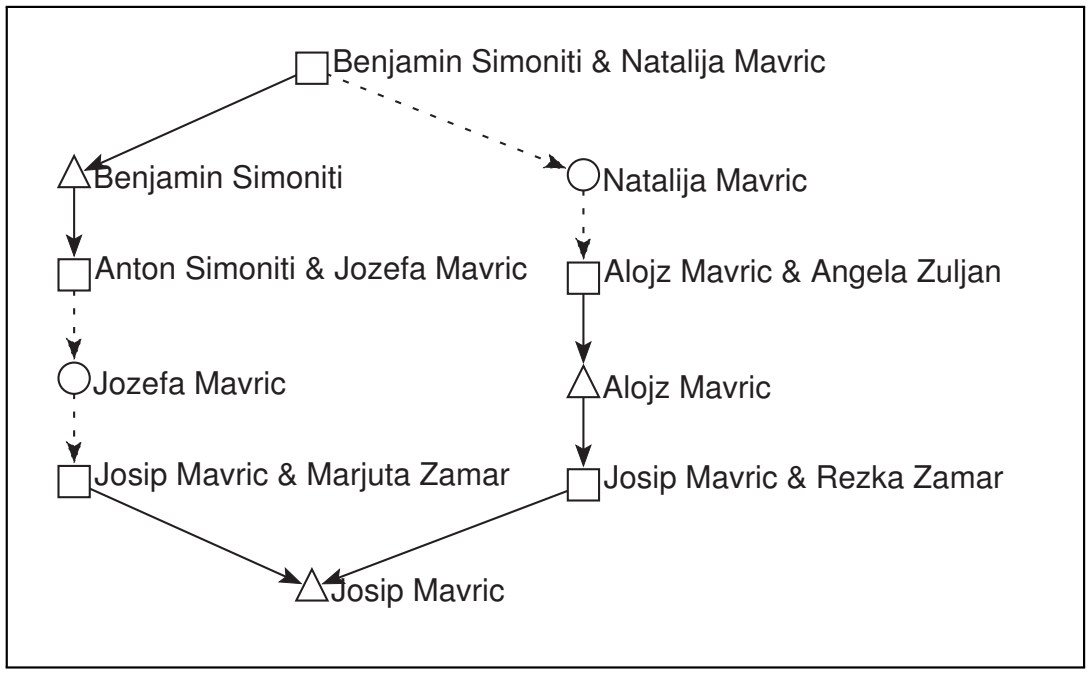
Frequencies normalized with number of couples in p-graph $\times$ 1000

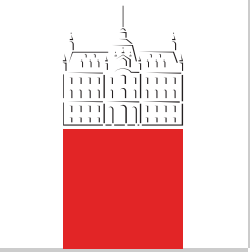
pattern	Loka	Silba	Ragusa	Turcs	Royal
A2	0.07	0.00	0.00	0.00	0.00
A3	0.07	0.00	0.00	0.00	2.64
A4.1	0.85	2.26	1.50	**159.71**	18.45
B4	3.82	11.28	10.49	**98.28**	6.15
A4.2	0.00	0.00	0.00	0.00	0.00
A5.1	0.64	3.16	2.00	36.86	11.42
A5.2	0.00	0.00	0.00	0.00	0.00
B5	1.34	4.96	23.48	46.68	7.03
A6.1	1.98	12.63	1.00	**169.53**	11.42
A6.2	0.00	0.90	0.00	0.00	0.88
A6.3	0.00	0.00	0.00	0.00	0.00
C6	0.71	5.41	9.49	36.86	4.39
B6.1	0.00	0.45	1.00	0.00	0.00
B6.2	1.91	17.59	31.47	**130.22**	10.54
B6.3	3.32	13.53	40.96	**113.02**	11.42
B6.4	0.00	0.00	2.50	7.37	0.00
Sum	14.70	72.17	123.88	798.53	84.36

Most of the relinking marriages happened in the genealogy of Turkish nomads; the second is Ragusa while in other genealogies they are much less frequent.

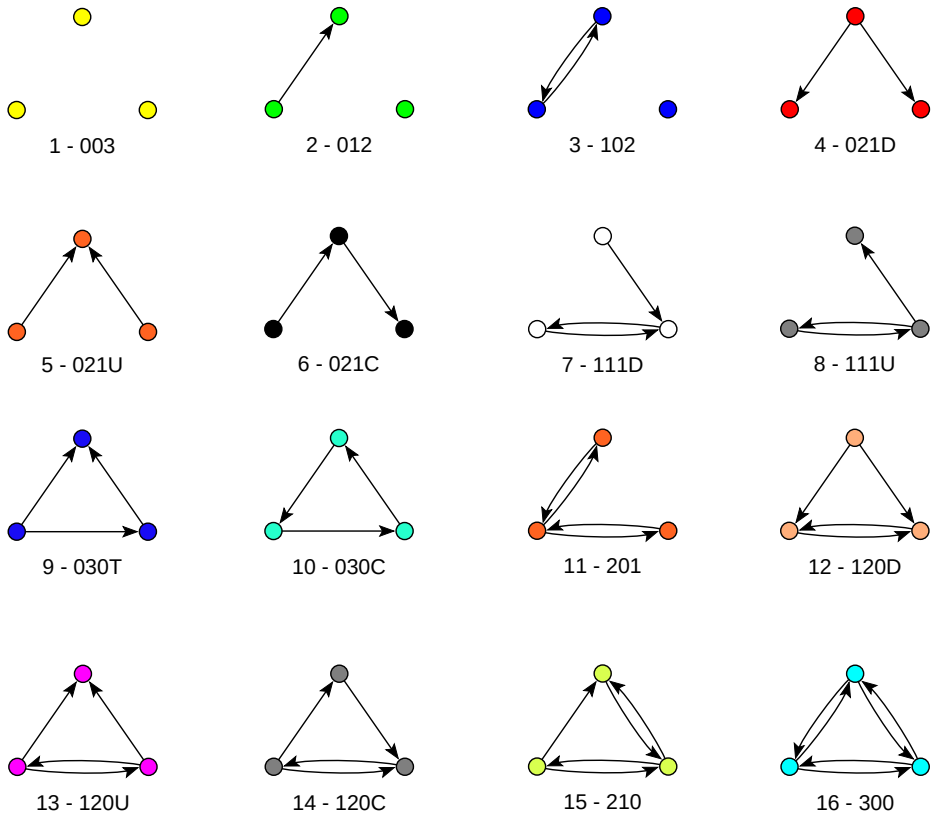
Bipartite p-graphs: Marriage among half-cousins







Triads



`Network/Info/Triadic Census`

Let $\mathcal{G} = (\mathcal{V}, R)$ be a simple directed graph without loops. A triad is a subgraph induced by a given set of three nodes. There are 16 nonisomorphic (types of) triads. They can be partitioned into three basic types:

- the null triad 003;
- dyadic triads 012 and 102; and
- connected triads: 111D, 201, 210, 300, 021D, 111U, 120D, 021U, 030T, 120U, 021C, 030C and 120C.

Triadic spectrum

Triad:	BA	CL	RC	R2C	TR	HC	39+	p1	p2	p3	p4
003		✓	✓		✓	✓				✓	✓
012					✓	✓	✓			✓	✓
102	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
021D			✓	✓	✓	✓	✓				✓
021U			✓	✓	✓	✓	✓			✓	✓
021C									✓		✓
111D											✓
111U								✓	✓		
030T			✓	✓	✓	✓	✓		✓		✓
030C								✓	✓		✓
201											
120D			✓	✓	✓	✓	✓			✓	✓
120U			✓	✓	✓	✓	✓	✓	✓		✓
120C							✓		✓		
210						✓	✓		✓		
300	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Triad Micro-Models:
BA: Ballance (Cartwright and Harary, '56)
CL: Clustering Model (Davis. '67)
RC: Ranked Cluster (Davis & Leinhardt, '72)
R2C: Ranked 2-Clusters (Johnsen, '85)
TR: Transitivity (Davis and Leinhardt, '71)
HC: Hierarchical Cliques (Johnsen, '85)
39+: Model that fits D&L's 742 mats N :39-72
p1-p4: Johnsen, 1986. Process Agreement Models.

Moody

Several properties of a graph can be expressed in terms of its triadic spectrum – distribution of all its triads. It also provides ingredients for p^* network models.

A direct approach to determine the triadic spectrum is of order $O(n^3)$; but in most large graphs it can be determined much faster.