Analysis of weighted networks

1. Introduction

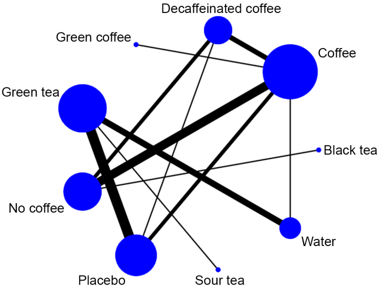
Vladimir Batagelj

IMFM Ljubljana, IAM & FAMNIT UP Koper, FMF UL Ljubljana

INSNA Sunbelt 2025 workshop

June 23-29, 2025, Paris, France

Outline



Vladimir Batagelj: vladimir.batagelj@fmf.uni-lj.si
Current version of slides (June 21, 2025 at 03 : 32): slides PDF
https://github.com/bavla/Nets

Introduction

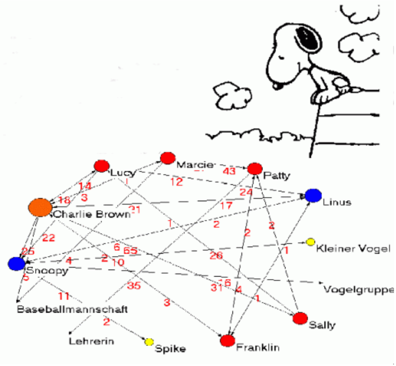
Workshop Sunbelt 2025

Workshop page on GitHub/bavla

igraph/R

Numerical, ordinal, nominal

Networks



Alexandra Schuler/ Marion Laging-Glaser: Analyse von Snoopy Comics

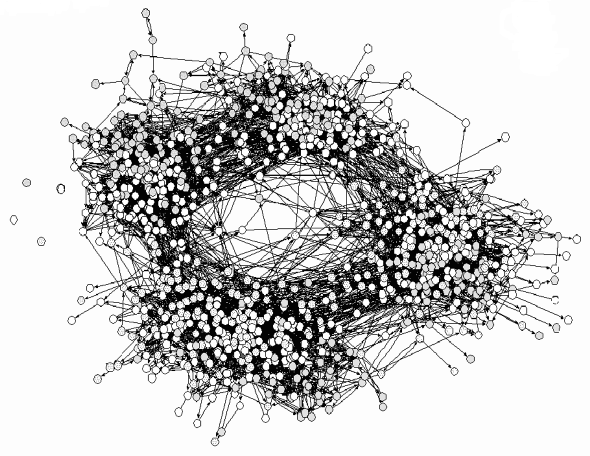
A *network* is based on two sets – set of *nodes* (vertices), that represent the selected *units*, and set of *links* (lines), that represent *ties* between units. They determine a *graph*. A link can be *directed* – an *arc*, or *undirected* – an *edge*.

Additional data about nodes or links can be known – their *properties* (attributes). For example: name/label, type, value, . . .

Network = Graph + Data

The data can be measured or computed.

”Countryside” school district



James Moody (2001) AJS Vol 107, 3,679–716, friendship relation

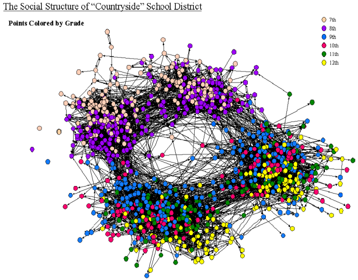
Only small or sparse networks can be displayed readably.

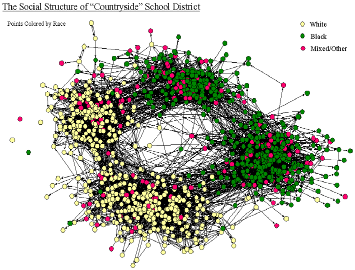
On large networks graph drawing algorithms can reveal their overall structure.

Can we explain the obtained structure?

Visualization: initial network exploration, reporting results, storytelling.

Display of properties – school (Moody)





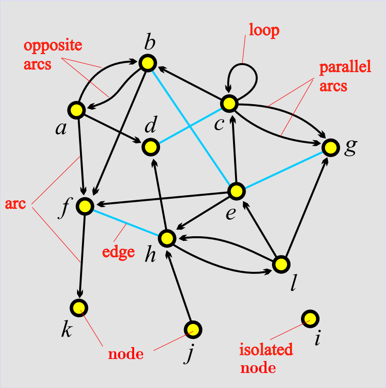
Besides the graph, we need data to understand the network!

Networks / Formally

A *network* $\mathcal{N} = (\mathcal{V}, \mathcal{L}, \mathcal{P}, \mathcal{W})$ consists of:

- a *graph* $\mathcal{G} = (\mathcal{V}, \mathcal{L})$, where $\mathcal{V}$ is the set of nodes, $\mathcal{A}$ is the set of arcs, $\mathcal{E}$ is the set of edges, and $\mathcal{L} = \mathcal{E} \cup \mathcal{A}$ is the set of links.
  $n = |\mathcal{V}|$, $m = |\mathcal{L}|$
- $\mathcal{P}$ *node value functions* / properties: $p : \mathcal{V} \to A$
- $\mathcal{W}$ *link value functions* / weights: $w : \mathcal{L} \to B$

Graph

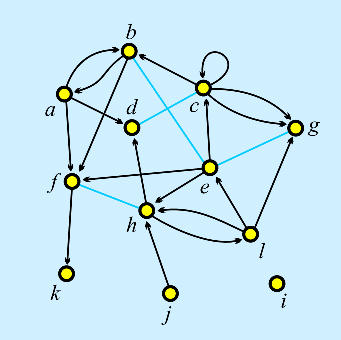


unit, actor – node, vertex
tie, line – link, edge, arc

arc $=$ directed link, (a, d)
a is the *initial* node,
d is the *terminal* node.

edge $=$ undirected link, $(c : d)$
c and d are *end* nodes.

Graph / Sets



$$\mathcal{V} = \{a, b, c, d, e, f, g, h, i, j, k, l\}$$

$$\mathcal{A} = \{(a, b), (a, d), (a, f), (b, a), (b, f), (c, b), (c, c), (c, g)_1, (c, g)_2, (e, c), (e, f), (e, h), (f, k), (h, d), (h, l), (j, h), (l, e), (l, g), (l, h)\}$$

$$\mathcal{E} = \{(b: e), (c: d), (e: g), (f: h)\}$$

$$\mathcal{G} = (\mathcal{V}, \mathcal{A}, \mathcal{E})$$

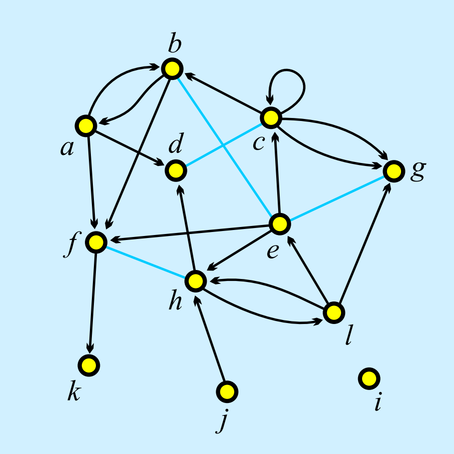
$$\mathcal{L} = \mathcal{A} \cup \mathcal{E}$$

$\mathcal{A} = \emptyset$ – *undirected* graph; $\mathcal{E} = \emptyset$ – *directed* graph.

The R package igraph doesn't support the mixing of arcs and edges. A graph is either directed or undirected.

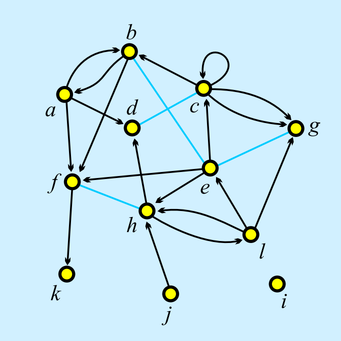
In most applications, an edge $(u: v)$ can be replaced by a pair of opposite arcs (u, v) and (v, u).

Graph / Sets – NET




```
*Vertices   12
   1 "a"      0.1020   0.3226
   2 "b"      0.2860   0.0876
   3 "c"      0.5322   0.2304
   4 "d"      0.3259   0.3917
   5 "e"      0.5543   0.4770
   6 "f"      0.1552   0.6406
   7 "g"      0.8293   0.3249
   8 "h"      0.4479   0.6866
   9 "i"      0.8204   0.8203
  10 "j"      0.4789   0.9055
  11 "k"      0.1175   0.9032
  12 "l"      0.7095   0.6475
*Arcs
   1   2
   2   1
   1   4
   1   6
   2   6
   3   2
   3   3
   3   7
   3   7
   5   3
   5   6
   5   8
   6  11
   8   4
  10   8
  12   5
  12   7
   8  12
  12   8
*Edges
   2   5
   3   4
   5   7
   6   8
```

Graph / Matrix



	a	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*	*k*	*l*
a	0	1	0	1	0	1	0	0	0	0	0	0
b	1	0	0	0	1	1	0	0	0	0	0	0
c	0	1	1	1	0	0	2	0	0	0	0	0
d	0	0	1	0	0	0	0	0	0	0	0	0
e	0	1	1	0	0	1	1	1	0	0	0	0
f	0	0	0	0	0	0	0	1	0	0	1	0
g	0	0	0	0	1	0	0	0	0	0	0	0
h	0	0	0	1	0	1	0	0	0	0	0	1
i	0	0	0	0	0	0	0	0	0	0	0	0
j	0	0	0	0	0	0	0	1	0	0	0	0
k	0	0	0	0	0	0	0	0	0	0	0	0
l	0	0	0	0	1	0	1	1	0	0	0	0

The graph G is *simple* if in the corresponding matrix all entries are 0 or 1.

Matrix representation provides a link to the tools of linear algebra.

Networks in igraph

The network $\mathcal{N} = (\mathcal{V}, \mathcal{L}, \mathcal{P}, \mathcal{W})$ can be described by two tables $(\mathcal{V}, \mathcal{P})$ and $(\mathcal{L}, \mathcal{W})$.

GraphSet.net, nodes.csv, links.csv

```
> nWdir <- paste0("https://raw.githubusercontent.com/",
+  "bavla/Nets/refs/heads/master/netsWeight/")
> V <- read.csv(paste0(nWdir,"nodes.csv"),sep="")
> L <- read.csv(paste0(nWdir,"links.csv"),sep="")
> V
> L
> N <- graph_from_data_frame(L,directed=TRUE,vertices=V)
> N
> V(N)$name <- V(N)$label
> N <- delete_vertex_attr(N,"label")
> E(N)$weight <- sample(1:7,ecount(N),replace=TRUE)
> N
> plot(N,edge.width=E(N)$weight)
> N$name <- "Network from data frame example"
> N$by <- "Vladimir Batagelj"
> N$cdate <- date()
> saveRDS(N,file="igraphDF.rds")

> as_data_frame(N,what="vertices")
> as_data_frame(N,what="edges")
```

Properties (attributes)

Properties of nodes $\mathcal{P}$ and links $\mathcal{W}$ can be measured on different scales: *numerical* (age, weight, number of contacts), *ordinal* (level of education), and *nominal/categorical* (citizenship, sex, type of contact). They can be *input* as data or *computed* from the network.

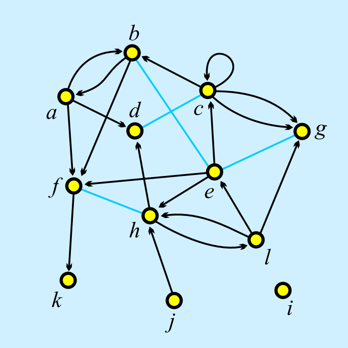
They can be represented (using factorization) as a numerical vector $\mathbf{v} = [v_i]$.

`Clustering` – partition of elements (nodes/links) – *nominal* or *ordinal* data about elements
$v_i \in \mathbb{N}$: element i belongs to the cluster/group v_i;

`Vector` – *numeric* data about elements
$v_i \in \mathbb{R}$: the property has value v_i on element i;

`Permutation` – *ordering* of elements
$v_i \in \mathbb{N}$: element i is at the v_i-th position.

Degrees



degree of node v, $\deg(v)$ = number of links with v as an endnode;
indegree of node v, $\text{indeg}(v)$ = number of links with v as a terminal node (endnode is both initial and terminal);
outdegree of node v, $\text{outdeg}(v)$ = number of links with v as an initial node.
initial node $v \Leftrightarrow \text{indeg}(v) = 0$
terminal node $v \Leftrightarrow \text{outdeg}(v) = 0$

$$n = 12,\ m = 23,\ \text{indeg}(e) = 3,\ \text{outdeg}(e) = 5,\ \deg(e) = 6$$

$$\sum_{v \in \mathcal{V}} \text{indeg}(v) = \sum_{v \in \mathcal{V}} \text{outdeg}(v) = |\mathcal{A}| + 2|\mathcal{E}| - |\mathcal{E}_0|,\ \sum_{v \in \mathcal{V}} \deg(v) = 2|\mathcal{L}| - |\mathcal{L}_0|$$

Properties (attributes)

*** structured data

When collecting the network data, consider providing as many properties as possible.

Properties (attributes)

The network N can be described by two tables (V, P) and (L,W).

```
> library(jsonlite)
> write_graph_netsJSON(T,file="test1.json")
> TT <- netsJSON_to_graph(fromJSON("test1.json"),directe
> TT
IGRAPH 0e9ae04 DNW- 9 13 --
+ attr: by (g/c), cdate (g/c), title (g/c), network (g/c)
| nArcs (g/n), nEdges (g/n), meta (g/x), name (v/c), age
| (v/n), fact (v/n), deg (v/n), type (e/c), weight (e/n)
+ edges from 0e9ae04 (vertex names):
 [1] Ana  ->Bor   Ana  ->Cene  Eva  ->Bor   Cene ->Ana
 [8] Cene ->Gaj   Gaj  ->Ana   Bor  ->Franc Franc->Dana
> V(TT)[[]]
+ 9/9 vertices, named, from 0e9ae04:
   name age   sex      x      y   fact deg
1   Ana  20  TRUE 0.1429 0.4882 0.2500   4
...
9   Jan  19 FALSE 0.7615 0.7889 0.5000   2
> E(TT)[[]]
+ 13/13 edges from 0e9ae04 (vertex names):
    tail  head tid hid type weight
1    Ana   Bor   1   2  arc      3
...
13   Iva   Jan   8   9  arc      5
>
```

Interactive drawing

https://igraph.org/r/doc/tkplot.html

```
> source("https://raw.githubusercontent.com/bavla/Rnet/ma
> Pt <- tkplot(CoreG,800,800,edge.curved=0,edge.width=E(C
# tkplot window is still active
> coor <- tk_coords(Pt,norm=F)
```

```
> write_graph_paj(T,file="test1.paj")
> write_graph_paj(T,file="test1.paj",coor=cbind(V(T)$x,V
```

Sources of networks

Indexes and repositories

- ICON - The Colorado Index of Complex Networks
- Netzschleuder
- Network Data Repository
- Kaggle
- UCI Network Data Repository
- KONECT - Koblenz Network Collection
- Bayesys
- Stanford Large Network Dataset Collection
- Pajek datasets
-

Named nodes!!!

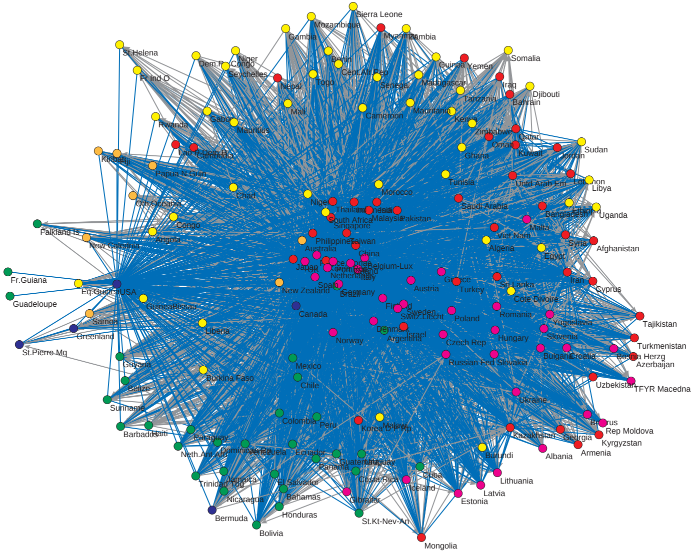
Representations of properties

In **`Pajek`** numerical properties of nodes are represented by *vector*s, nominal properties by *partitions* or as *label*s of nodes. Numerical property can be displayed as *size* (width and height) of node (figure), as its *coordinate*; and a nominal property as *color* or *shape* of the figure, or as a node's *label* (content, size and color).

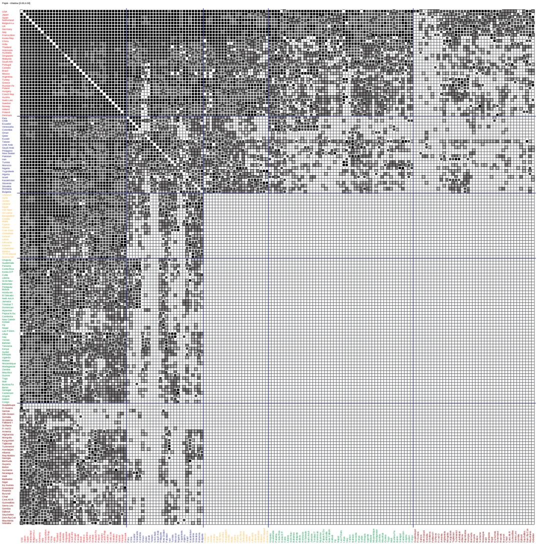
We can assign in **`Pajek`** numerical values to links. They can be displayed as *value*, *thickness* or *grey level*. Nominal vales can be assigned as *label*, *color* or *line pattern* (see Pajek manual, section ***4.3***).

A display of World Trade 1999 network

World Trade Flows: 1962-2000; info → weights in 1000 USD



Matrix display of World Trade 1999 network



Types of networks

Besides ordinary (directed, undirected, mixed) networks some extended types of networks are also used:

- *2-mode networks*, bipartite (valued) graphs – networks between two disjoint sets of nodes $\mathcal{N} = ((\mathcal{U}, \mathcal{V}), \mathcal{L}, \mathcal{P}, \mathcal{W})$
- *multi-relational networks* $\mathcal{N} = (\mathcal{V}, (\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_k), \mathcal{P}, \mathcal{W})$
- *temporal networks*, dynamic graphs – networks changing over time $\mathcal{N}_T = (\mathcal{V}, \mathcal{L}, \mathcal{P}, \mathcal{W}, T)$
- specialized networks: representation of genealogies as *p-graphs*; *Petri's nets*, . . .

The network (input) file formats should provide means to express all these types of networks. All interesting data should be recorded (respecting privacy).

Pictures in SVG: ***66 days***. Kansas Event Data System *KEDS*

Two-mode networks

In a *two-mode* (affiliation or bipartite) network $\mathcal{UV} = ((U, V), L, uv)$ the set of nodes is split into two disjoint sets (*modes*) U and V. In this paper, we will assume that the network is simple (no parallel links) and directed. Each arc $e \in L$ has its initial node in the set U and its terminal node in the set V. The function $uv\colon L \to \mathbb{R}^+$ assigns to each arc its weight. In general, the weight can be measured on different measurement scales (counts, ratio, interval, ordinal, nominal, binary,TQ, etc.).

The network $\mathcal{UV}$ can be represented with the corresponding *matrix* $\mathbf{UV} = [uv[u, v]]_{u\in U, v\in V}$ defined as

$$uv[u, v] = \begin{cases} w(u, v) & (u, v) \in L \\ \square & \text{otherwise} \end{cases}$$

We represent the number 0 with two symbols, 0 (weight 0) and $\square$ (no link) where $\square = 0$ with rules $\square + a = a$ and $\square \cdot a = \square$.

Two-mode networks

The set $\mathrm{N}^o_{UV}(u)$ of *out-neighbors* (successors) of the node $u \in U$

$$\mathrm{N}^o_{UV}(u) = \{v \in V : (u, v) \in L\}$$

and the set $\mathrm{N}^i_{UV}(u)$ of *in-neighbors* (predecessors) of the node $v \in V$

$$\mathrm{N}^i_{UV}(v) = \{u \in U : (u, v) \in L\}$$

In traditional two-mode networks, we usually assume that $U \cap V = \emptyset$. In the case $U = V$ we get an ordinary one-mode simple directed network.

In the following we will often, when it is obvious from the context, omit the subscript. For example, $\mathrm{N}^o_{UV}(u) = \mathrm{N}^o(u)$. The function $\delta : \{\mathbf{false}, \mathbf{true}\} \to \{0, 1\}$ is determined by

$$\delta(\mathbf{false}) = 0 \qquad \text{and} \qquad \delta(\mathbf{true}) = 1$$

Two-mode networks

We will also use some additional functions:

- *out/in-degree*

$$\text{od}_{UV}(u) = \sum_{v \in V} \delta((u,v) \in L) = |\,\mathrm{N}^o(u)| \quad \text{and} \quad \text{id}_{UV}(v) = \sum_{u \in U} \delta((u$$

- *weighted out/in-degree* (row/column sums)

$$\text{wod}_{UV}(u) = \sum_{v \in V} uv[u,v], \qquad \text{wid}_{UV}(v) = \sum_{u \in U} uv[u,v]$$

and

$$\text{wod}_{UV}(u/t) = \sum_{v \in \mathrm{N}^o(u) \cap \mathrm{N}^o(t)} uv[u,v]$$

It holds $\mathrm{N}^o(u) \cap \mathrm{N}^o(t) \neq \emptyset \Rightarrow \text{wod}_{UV}(u/t) \neq \square$.

- We denote $U_{UV}^{[d]} = \{u \in U : \text{od}_{UV}(u) \geq d\}$ and
$\mathcal{UV}^{[d]} = ((U_{UV}^{[d]}, V), L(U_{UV}^{[d]}), w|U_{UV}^{[d]})$.
$\hat{U} = \{u \in U : \text{wod}_{UV}(u) \neq 0\}$

Two-mode networks

The *transpose* $\mathbf{UV}^T$ of matrix $\mathbf{UV}$ is a matrix on $V \times U$ with entries $uv^T[v, u] = uv[u, v]$.

Reversed network $\widetilde{N} = ((V, U), \widetilde{L}, vu)$, $\widetilde{L} = \{(v, u) : (u, v) \in L\}$, $vu(v, u) = uv(u, v)$ for $(u, v) \in L$.

$$\mathrm{wod}_{VU}(v) = \mathrm{wid}_{UV}(v), \ldots$$

In the following, we will often identify networks by their matrices.

$\mathbf{VU} = \mathbf{UV}^T$

Size of network

The size of a network/graph is expressed by two numbers: number of nodes $n = |\mathcal{V}|$ and number of links $m = |\mathcal{L}|$.

In a *simple undirected* graph (no parallel edges, no loops) $m \leq \frac{1}{2}n(n-1)$; and in a *simple directed* graph (no parallel arcs) $m \leq n^2$.

Small networks (some tens of nodes) – can be represented by a picture and analyzed by many algorithms (***UCINET***, ***NetMiner***).

Also *middle size* networks (some hundreds of nodes) can still be represented by a picture (!?), but some analytical procedures can't be used.

Till 1990 most networks were small – they were collected by researchers using surveys, observations, archival records, . . . The advances in IT allowed to create networks from the data already available in the computer(s). *Large* networks became reality. Large networks are too big to be displayed in details; special

Large networks

Large network – several thousands or millions of nodes. Can be stored in computer's memory – otherwise *huge* network. 64-bit computers!

Jure Leskovec: SNAP – Stanford Large Network Dataset Collection

Social networks

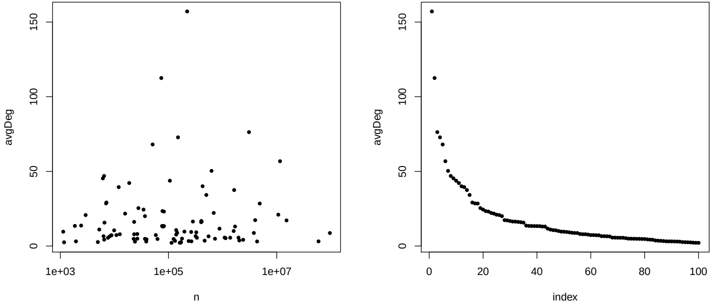
Name	Type	Nodes	Edges	Description
ego-Facebook	Undirected	4,039	88,234	Social circles from Facebook (anonymized)
ego-Gplus	Directed	107,614	13,673,453	Social circles from Google+
ego-Twitter	Directed	81,306	1,768,149	Social circles from Twitter
soc-Epinions1	Directed	75,879	508,837	Who-trusts-whom network of Epinions.com
soc-LiveJournal1	Directed	4,847,571	68,993,773	LiveJournal online social network
soc-Pokec	Directed	1,632,803	30,622,564	Pokec online social network
soc-Slashdot0811	Directed	77,360	905,468	Slashdot social network from November 2008
soc-Slashdot0922	Directed	82,168	948,464	Slashdot social network from February 2009
wiki-Vote	Directed	7,115	103,689	Wikipedia who-votes-on-whom network

Networks with ground-truth communities

Name	Type	Nodes	Edges	Communities	Description
com-LiveJournal	Undirected, Communities	3,997,962	34,681,189	287,512	LiveJournal online social network
com-Friendster	Undirected, Communities	65,608,366	1,806,067,135	957,154	Friendster online social network
com-Orkut	Undirected, Communities	3,072,441	117,185,083	6,288,363	Orkut online social network

Dunbar's number

Average degrees of the SNAP and Konect networks



Average degree $\bar{d} = \frac{1}{n}\sum_{v \in V} \deg(v) = \frac{2m}{n}$. Most real-life large networks are *sparse* – the number of nodes and links are of the same order. This property is also known as a Dunbar's number.

The basic idea is that if each node has to spend for each link certain amount of "energy" to maintain the links to selected other nodes then, since it has a limited "energy" at its disposal, the number of links should be limited. In human networks the Dunbar's number is between 100 and 150.

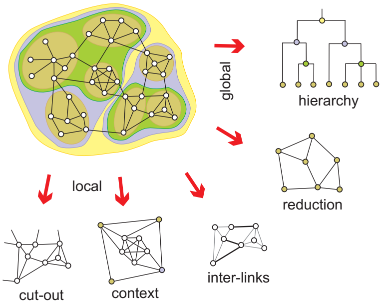
Complexity of algorithms

Let us look to time complexities of some typical algorithms:

	T(n)	1.000	10.000	100.000	1.000.000	10.000.000
LinAlg	O(n)	0.00 s	0.015 s	0.17 s	2.22 s	22.2 s
LogAlg	O($n \log n$)	0.00 s	0.06 s	0.98 s	14.4 s	2.8 m
SqrtAlg	O($n\sqrt{n}$)	0.01 s	0.32 s	10.0 s	5.27 m	2.78 h
SqrAlg	O(n^2)	0.07 s	7.50 s	12.5 m	20.8 h	86.8 d
CubAlg	O(n^3)	0.10 s	1.67 m	1.16 d	3.17 y	3.17 ky

For the interactive use on large graphs already quadratic algorithms, O(n^2), are too slow.

Approaches to large networks



In analysis of a *large* network (several thousands or millions of nodes, the network can be stored in computer memory) we can't display it in its totality; also there are only few algorithms available.

To analyze a large network we can use statistical approach or we can identify smaller (sub) networks that can be analyzed further using more sophisticated methods.

igraph

Examples and sources of weighted networks

Analysis of weighted networks

Assume that the weight w in the network $\mathcal{N} = (V, L, w)$, $w : L \to \mathbb{R}$ is "compatible" with our research question:

> *larger is the value of the weight $w(e) \Rightarrow$ more important is the link $e \in L$ with respect to our question*

To identify important nodes or subnetworks we can use: cuts, islands, (valued) cores, clustering (corrected dissimilarities), etc.

Using link cuts, islands, cores, skeletons (spanning tree, Pathfinder, k-neighbors), community detection, hubs and authorities, etc. we can identify the most active subnetworks. We can also apply clustering and blockmodeling methods. [**?**]

Nature of weights

1. (directly) measured: road traffic, baboons, ...
2. derived – computed from existing data
   1. projections of two-mode networks
      1. binary
      2. binary with NA
      3. nonnegative reals
      4. general
   2. network weight indexes: SPC weights, preferential attachment $w(u:v) = \deg(u) \cdot \deg(v)$, link betweenness, short cycles counts; (dis)similarity between end-nodes
   3. ... football
3. signed networks

many zeros, threshold, only for links of a given network

Normalizations

Often some nodes/links prevail. How to make weights comparable? [?, ?, p. 94] We assume $w : \mathcal{L} \to \mathbb{R}_0^+$

$$\mathrm{Geo}_{uv} = \frac{w_{uv}}{\sqrt{w_{uu} w_{vv}}} \qquad \mathrm{GeoDeg}_{uv} = \frac{w_{uv}}{\sqrt{\deg_u \deg_v}}$$

$$\mathrm{Input}_{uv} = \frac{w_{uv}}{w_{vv}} \qquad \mathrm{Output}_{uv} = \frac{w_{uv}}{w_{uu}}$$

$$\mathrm{Min}_{uv} = \frac{w_{uv}}{\min(w_{uu}, w_{vv})} \qquad \mathrm{Max}_{uv} = \frac{w_{uv}}{\max(w_{uu}, w_{vv})}$$

$$\mathrm{MinDir}_{uv} = \begin{cases} \frac{w_{uv}}{w_{uu}} & w_{uu} \leq w_{vv} \\ 0 & otherwise \end{cases} \qquad \mathrm{MaxDir}_{uv} = \begin{cases} \frac{w_{uv}}{w_{vv}} & w_{uu} \leq w_{vv} \\ 0 & otherwise \end{cases}$$

Missing diagonal weights

In the case of networks without loops we define the diagonal weights for undirected networks as the sum of out-diagonal elements in the row (or column)

$$w_{vv} = \sum_u w_{vu}$$

and for directed networks (for example, trade among world countries) as some mean value of the row and column sum, for example

$$w_{vv} = \frac{1}{2}(\sum_u w_{vu} + \sum_u w_{uv})$$

or

$$w_{vv} = \sqrt{\sum_u w_{vu} \cdot \sum_u w_{uv}}$$

Usually we assume that the network does not contain any isolated node.

Important nodes and links in network

To identify important / interesting elements (nodes, links) in a network we often try to express our intuition about important / interesting elements using an appropriate measure (index, weight) following the scheme

> *larger is the measured value of an element,*
> *more important / interesting is this element*

Too often, in the analysis of networks, researchers uncritically pick some measures from the literature. For a formal approach see Roberts.

The (importance) measure can be obtained as input data – an *observed* property, or computed from the network description – a *structural* property.

An interesting question is studying associations among structural and observed properties. Can an observed property be explained with some structural property/ies?

Node indices

The most important distinction between different node *indices* is based on the view/decision of whether the network is considered directed or undirected.

Closeness, betweenness, clustering, eigen-vector, ...

Weighted degree, indegree and outdegree

$$\mathrm{wdeg}(v) = \sum_{e \in \mathrm{star}(v)} w(e),$$

$$\mathrm{windeg}(v) = \sum_{e \in \mathrm{instar}(v)} w(e) \quad \text{and} \quad \mathrm{woutdeg}(v) = \sum_{e \in \mathrm{outstar}(v)} w(e)$$

Hubs and authorities

To each node v of a network $\mathcal{N} = (\mathcal{V}, \mathcal{L})$ with matrix $\mathbf{W}$ we assign two values: quality of its content (*authority*) x_v and quality of its references (*hub*) y_v.

A good authority is selected by good hubs; and good hub points to good authorities (see Kleinberg).

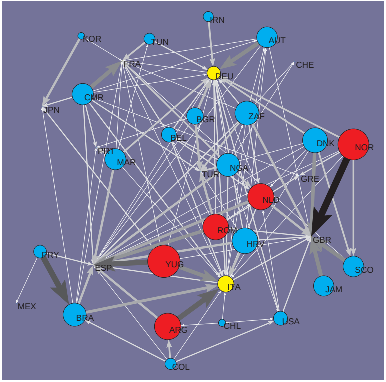
$$x_v = \sum_{u:(u,v)\in\mathcal{L}} w[u,v] \cdot y_u \qquad \text{and} \qquad y_v = \sum_{u:(v,u)\in\mathcal{L}} w[v,u] \cdot x_u$$

Let $\mathbf{x}$ and $\mathbf{y}$ be the authority and hub vectors. Then we can write these two relations as $\mathbf{x} = \mathbf{W}^T\mathbf{y}$ and $\mathbf{y} = \mathbf{W}\mathbf{x}$.

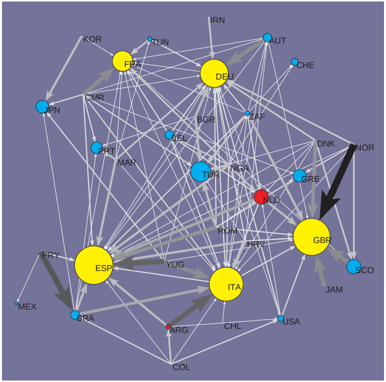
We start with $\mathbf{y} = [1, 1, \ldots, 1]$ and then compute new vectors $\mathbf{x}$ and $\mathbf{y}$. After each step we normalize both vectors. We repeat this until they stabilize.

We can show that this procedure converges. The limit vector $\mathbf{x}^*$ is the principal eigen vector of matrix $\mathbf{W}^T\mathbf{W}$; and $\mathbf{y}^*$ of matrix $\mathbf{W}\mathbf{W}^T$.

... Hubs and authorities: football



Exporters (hubs)



Importers (authorities)

Example: Krebs, Krempl. World Cup 1998 in Paris, 22 national teams. A player from first country is playing in the second country.

Important links – Weighted networks

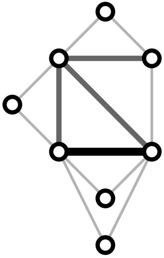
A very important characteristic of weight is its nature – should it be considered a *similarity* (larger is the link weight more similar are its end-nodes) or a *dissimilarity* (larger is the link weight more different are its end-nodes; weight for links with equal end-nodes is usually 0)?

Structural weights are computed based on network structure. For example

$$w(e(u, v)) = \deg(u) \cdot \deg(v)$$

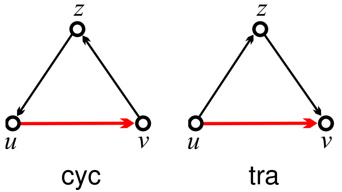
More substantial weights exist.

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 $w(e) \geq k - 2$.

If a graph $\mathcal{G}$ is mixed we replace edges with pairs of opposite arcs. 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.



Cuts

The standard approach to find interesting groups inside a network is based on properties/weights – they can be *measured* or *computed* from network structure.

The *node-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 links from $\mathcal{L}$ that have both endnodes in $\mathcal{V}'$.

The *link-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 endnodes of the links from $\mathcal{L}'$.

Normalizations

In networks obtained from large two-mode networks, there are often huge differences in weights. Therefore it is not interesting to compare the nodes according to the raw data – the nodes with large weights will prevail. First, we have to *normalize* the network to make the weights comparable.

There exist several ways how to do this. Some of them are presented in the following. They can be used also on other weighted networks. Often a given weighted network is essentially a projection of a two-mode network; sometimes without loops. Assume that the network is described with a square matrix **C**.

Let us denote the row sum $R(u) = \sum_{v \in \mathcal{V}} c[u, v]$, which is the total of weights from node u to other nodes. Let $C(v) = \sum_{u \in \mathcal{V}} c[u, v]$ denote the column sum for the node v. If the network is a projection of a two-mode network we set $R(u) = C(u) = c[u, u]$.

. . . Normalizations

Markov (also known as stochastic, affinity, output, row) normalization: For $R(u) > 0$

$$c_M[u, v] = \frac{c[u, v]}{R(u)}$$

If $R(u) = 0$ also $c_M[u, v] = 0$.

Jaccard-like normalization:

$$c_J[u, v] = \frac{c[u, v]}{R(u) + R(v) - c[u, v]}$$

Salton-like normalization:

$$c_S[u, v] = \frac{c[u, v]}{\sqrt{R(u) \cdot R(v)}}$$

. . . Normalizations

Balassa or activity normalization: Let $T = \sum_{u,v} c[u, v]$ be the total sum of weights in the network.

$$A[u, v] = \frac{c[u, v] \cdot T}{R(u) \cdot C(v)}$$

If $A[u, v] > 1$ the measured weight is larger than expected.

$$c_B[u, v] = \log_2 A[u, v] \quad \text{for } A[u, v] > 0$$

...Normalizations

$$c_{\min}[u,v] = \frac{c[u,v]}{\min(R(u),R(v))} \qquad c_{\max}[u,v] = \frac{c_{uv}}{\max(c_{uu},c_{vv})}$$

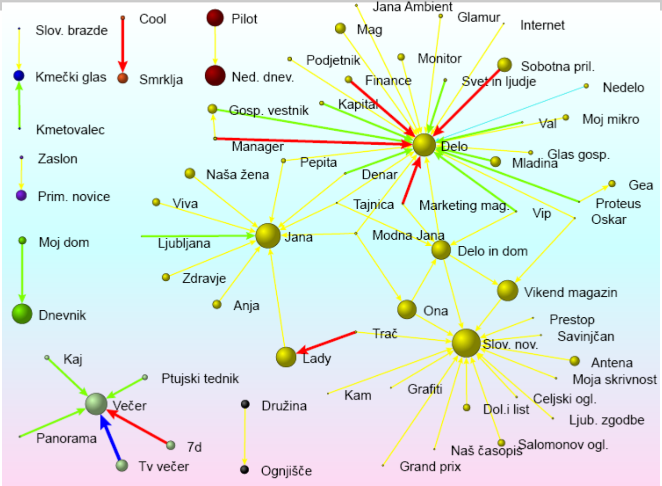
$$c_{\mathrm{minDir}}[u,v] = \begin{cases} \dfrac{c[u,v]}{R(u)} & R(u) \leq R(v) \\ 0 & \textit{otherwise} \end{cases}$$

$$c_{\mathrm{maxDir}}[u,v] = \begin{cases} \dfrac{c[u,v]}{R(v)} & R(u) \leq R(v) \\ 0 & \textit{otherwise} \end{cases}$$

After a selected normalization the important parts of the network are obtained by link-cuts or islands approaches.

Reuters Terror News: GeoDeg, MaxDir, MinDir. Slovenian journals.

minDir of Slovenian journals 2000

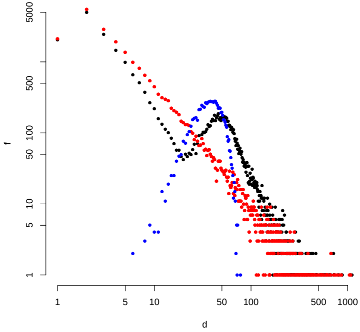


Over 100000 people were asked in the years 1999 and 2000 about the journals they read. They mentioned 124 different journals. (source Cati)

EAT – The Edinburgh Associative Thesaurus

```
> setwd("C:/Users/vlado/docs/papers/2025/Sunbelt/ws/R")
> EAT <- read_graph("EATnew.net",format="pajek")
> EAT
> ad <- degree(EAT,mode="all",loops=TRUE)
> id <- degree(EAT,mode="in",loops=TRUE)
> od <- degree(EAT,mode="out",loops=TRUE)
> top(ad,10)
    name value
1     ME  1108
2    MAN  1074
> wad <- strength(EAT,mode="all",weights=E(EAT)$weight,loops=TRUE)
> wid <- strength(EAT,mode="in",weights=E(EAT)$weight,loops=TRUE)
> wod <- strength(EAT,mode="out",weights=E(EAT)$weight,loops=TRUE)
> top(wad,10)
  name value
1  MONEY  4484
2  WATER  3396
...
> top(wid,10)
> top(wod,10)
> w <- E(EAT)$weight
> E(EAT)[1:20]
> names(w) <- attr(E(EAT),"vnames")
> top(w,30)
                name value
1          LOBE|EAR    91
2    CHEDDAR|CHEESE    90
3         HONG|KONG    89
...
```

EAT degree distributions




```
> Ta <- table(ad); Ti <- table(id); To <- table(od)
> d <- as.integer(names(Ta)); f <- unname(Ta)
> di <- as.integer(names(Ti[Ti>0])); fi <- unname(Ti[Ti>0])
> do <- as.integer(names(To[To>0])); fo <- unname(To[To>0])
> plot(d,f,log="xy",axes=FALSE,pch=16,cex=0.8)
> axis(1); axis(2)
> points(di,fi,pch=16,cex=0.8,col="red")
> points(do,fo,pch=16,cex=0.8,col="blue")
```

Krebs

Transformations of weighted networks

(dealing with large ranges of values of weight, making nodes comparable, Balassa index),

Visualization of weighted networks

(graph drawing, monotonic recoding, matrix representation, ordering of nodes),

Important nodes

- hubs and authorities,

Skeletons

important parts of the network:

cuts, k-neighbors, Pathfinder, cores, trusses, backbone, islands,

Cuts in R

The threshold value t is determined on the basis of distribution of values of weight w or property p. Usually we are interested in cuts that are not too large, but also not trivial.

Node-cut: p stored in a vector

Link-cut: weighted network

Temporal weighted networks

Conclusions

Analysis of weighted networks

The structure of the network $N = (V, L, W, P)$ is determined by the graph $G = (V, L)$, where V is the set of nodes and L is the set of links. In addition, data about links (weights from W) and nodes (properties from P) is often available. The network N is weighted if its set of weights is nonempty. The weights can be either measured (such as trade networks - BACI/CEPII) or computed (for example, a projection of a two-mode network).

Analysis of weighted networks

The workshop will cover the following topics:

- examples and sources of weighted networks,
- transformations of weighted networks (dealing with large ranges of values of weight, making nodes comparable, Balassa index),
- visualization of weighted networks (graph drawing, monotonic recoding, matrix representation, ordering of nodes),
- clustering and blockmodeling.
- important nodes - hubs and authorities,
- skeletons - important parts of the network: cuts, k-neighbors, Pathfinder, cores, trusses, backbone, islands,
- temporal weighted networks.

Analysis of weighted networks

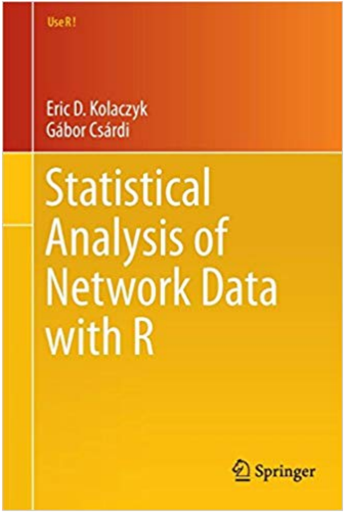
Most of the topics are discussed in the book Batagelj, Doreian, Ferligoj, Kejžar (2014) Understanding Large Temporal Networks and Spatial Networks.

The workshop is based on the programming system R. The network data and additional R code will be available on GitHub/bavla.

Resources

- Weighted network data sets
- Slides and papers on weighted networks
- netsWeight package

igraph



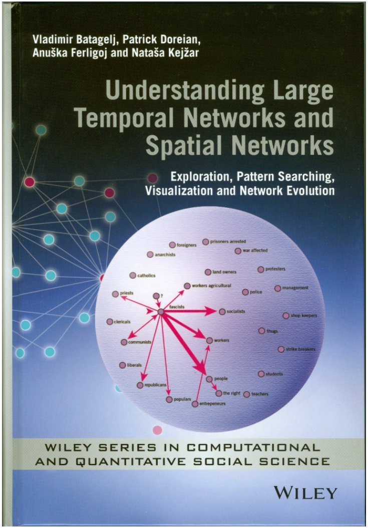
A book on Statistical Analysis of Network Data with R using the package igraph was written by Kolaczyk, Eric D. and Csárdi, Gábor (Springer 2014).

Another book on igraph is prepared by Gábor Csárdi, Tamás Nepusz and Edoardo M. Airoldi draft.

igraph can be installed from CRAN

`https://cran.r-project.org/web/packages/igraph/index.html`

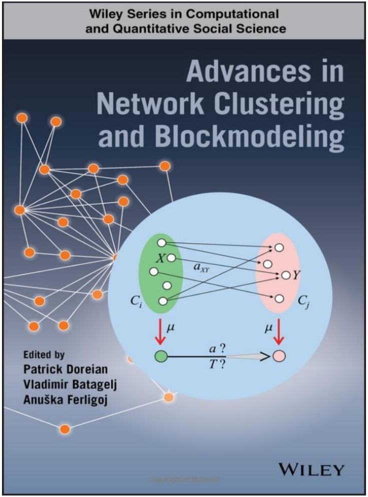
Understanding large networks



This course is closely related to chapters 2 and 3 in the book:

Vladimir Batagelj, Patrick Doreian, Anuška Ferligoj and Nataša Kejžar: Understanding Large Temporal Networks and Spatial Networks: Exploration, Pattern Searching, Visualization and Network Evolution. Wiley Series in Computational and Quantitative Social Science. Wiley, October 2014.

Advances in Network Clustering and Blockmodeling



An overview of the network clustering and blockmodeling is available in the book:

Patrick Doreian, Vladimir Batagelj, Anuška Ferligoj (Eds.): Advances in Network Clustering and Blockmodeling. Wiley Series in Computational and Quantitative Social Science. Wiley, 2020.

References I

- De Nooy, W, Mrvar, A, Batagelj, V: Exploratory Social Network Analysis with Pajek; Revised and Expanded Edition for Updated Software. Structural Analysis in the Social Sciences, CUP, July 2018.
- Batagelj, V, Doreian, P, Ferligoj, A, Kejžar, N: Understanding Large Temporal Networks and Spatial Networks: Exploration, Pattern Searching, Visualization and Network Evolution. Wiley Series in Computational and Quantitative Social Science. Wiley, October 2014
- Batagelj, V, Zaveršnik, M: Fast algorithms for determining (generalized) core groups in social networks. Advances in Data Analysis and Classification, 2011. Volume 5, Number 2, 129-145.
- Batagelj, V, Cerinšek, M: On bibliographic networks. Scientometrics 96 (2013) 3, 845-864.

References II

- Cerinšek, M, Batagelj, V: Sources of Network Data. Encyclopedia of Social Network Analysis and Mining. Reda Alhajj, Jon Rokne (Eds.), Springer, 2017
- Batagelj, V: On fractional approach to analysis of linked networks. Scientometrics 123 (2020) 2: 621-633
- Batagelj, V, Maltseva, D: Temporal bibliographic networks. Journal of Informetrics, Volume 14, Issue 1, February 2020, 101006.
- Maltseva, D, Batagelj, V: Social network analysis as a field of invasions: bibliographic approach to study SNA development. Scientometrics, 121(2019)2, 1085-1128
- Schvaneveldt, RW (Ed.) (1990) Pathfinder Associative Networks: Studies in Knowledge Organization. Norwood, NJ: Ablex.
- Wikipedia: Regular language.