Network

Introduction

Let $\mathbf{U} = \{X_1, X_2, \ldots, X_n\}$ be a finite set of units. Connections among units are described using one or more *binary relations* $R_t \subseteq \mathbf{U} \times \mathbf{U}, \quad t = 1, \ldots, r$, which determine *a network* $\mathcal{N} = (\mathbf{U}, R_1, R_2, \ldots, R_r)$.

Example: A relation can represent friendship, negative relation, kinship relation (…is a child of…, …is a daughter of …, …is married to…), citations…

In the following we will use mostly one relation R.

$X_i R X_j$ is read as:
unit X_i is in relation R with unit X_j.

Example: if R corresponds to relation 'liking', then $X_i R X_j$, means that person X_i likes person X_j.

Relations can have additional properties:

- *reflexive relation*: $\forall x \in \mathbf{U} : xRx$
- *irreflexive relation*: $\forall x \in \mathbf{U} : \neg xRx$
- *symmetric relation*: $\forall x, y \in \mathbf{U} : (xRy \Rightarrow yRx)$ (marriage)
- *asymmetric relation*: $\forall x, y \in \mathbf{U} : \neg(xRy \wedge yRx)$ (is a son of)
- *antisymmetric relation*: $\forall x, y \in \mathbf{U} : (xRy \wedge yRx \Rightarrow x = y)$
- *transitive relation*: $\forall x, y, z \in \mathbf{U} : (xRy \wedge yRz \Rightarrow xRz)$
- *intransitive relation*: $\forall x, y, z \in \mathbf{U} : (xRy \wedge yRz \Rightarrow \neg xRz)$
- *comparable relation*: $\forall x, y \in \mathbf{U} : (x \neq y \Rightarrow xRy \vee yRx)$
- *strictly comparable relation*: $\forall x, y \in \mathbf{U} : (xRy \vee yRx)$
- *partial ordering*: a reflexive, antisymmetric and transitive relation;
- *equivalence relation*: a reflexive, symmetric and transitive relation.

A network defined using relation R can be represented in different ways:

- Representation using corresponding **binary matrix** $\mathbf{R} = [r_{ij}]_{n \times n}$, where

  $$r_{ij} = \begin{cases} 1 & X_i R X_j \\ 0 & \text{otherwise} \end{cases}$$

  Sometimes r_{ij} is a real number, expressing the strength of relation R between units X_i in X_j.

- **list of neighbours**
  A network can be described by specifying the list of all other units with which the unit is in relation.

- description by a **graph** $G = (V, L)$ where V is set of vertices and L set of (directed or undirected) lines. Vertices represent units of a network, lines represent the relation.

  A graph is usually represented by a picture: vertices are drawn as small circles, directed lines are drawn as arcs and undirected lines as edges connecting the corresponding two vertices.

$X_i R X_j \Rightarrow$ there exists directed line from unit X_i to X_j in corresponding graph. Vertex X_i is called *initial*, vertex X_j is called *terminal* vertex.

A line whose initial and terminal vertices are the same is called *a loop*.

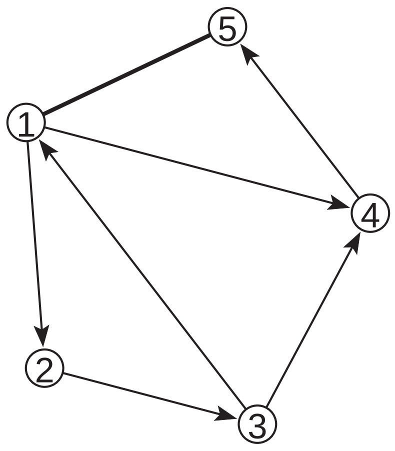
If directed lines between two vertices exist in both directions, they are sometimes replaced by a single undirected line.

Sometimes we do the opposite: an undirected line is replaced by two directed lines in opposite directions.

We will denote the number of vertices in a graph with n, and number of directed lines with m.

Matrix, graph and list of neighbours

	1	2	3	4	5
1	0	1	0	1	1
2	0	0	1	0	0
3	1	0	0	1	0
4	0	0	0	0	1
5	1	0	0	0	0



Arcslist

1 2 4

2 3

3 1 4

4 5

Edgeslist

1 5

Types of networks

- *undirected network* – the relation is symmetric – all lines are undirected – edges, $L = E$.
- *directed network* – the relation is not symmetric – all lines are directed – arcs, $L = A$.
- *mixed network* – both arcs and edges exist in a corresponding graph – $L = A \cup E$.
- *two-mode network*
  A two-mode network consists of two sets of units (e. g. people and events), relation connects the two sets, e. g. participation of people in social events.

Small and large networks

Networks with some 10 units and lines are called *small networks*, while networks with some 1000 units and lines are called *large networks*.

Dense and sparse networks

A network is called *sparse* if the number of lines in the corresponding graph is of the same order as the number of vertices ($n \approx km$). Large networks that are sparse can still be efficiently analysed with some algorithms. In real life we often find very large but sparse networks.

In general, the number of lines can be much higher than the number of vertices. Such networks are called *dense*.

If every unit is connected to every other unit the number of lines is n^2 (number of elements in matrix).

If every unit is connected to every other unit except to itself (graph without loops), the number of lines is $n(n-1)$ (number of elements in matrix without diagonal).

According to that the density of a network can be defined:

For networks with loops:

$$Density1 = \frac{m}{n^2}$$

For networks without loops:

$$Density2 = \frac{m}{n(n-1)}$$

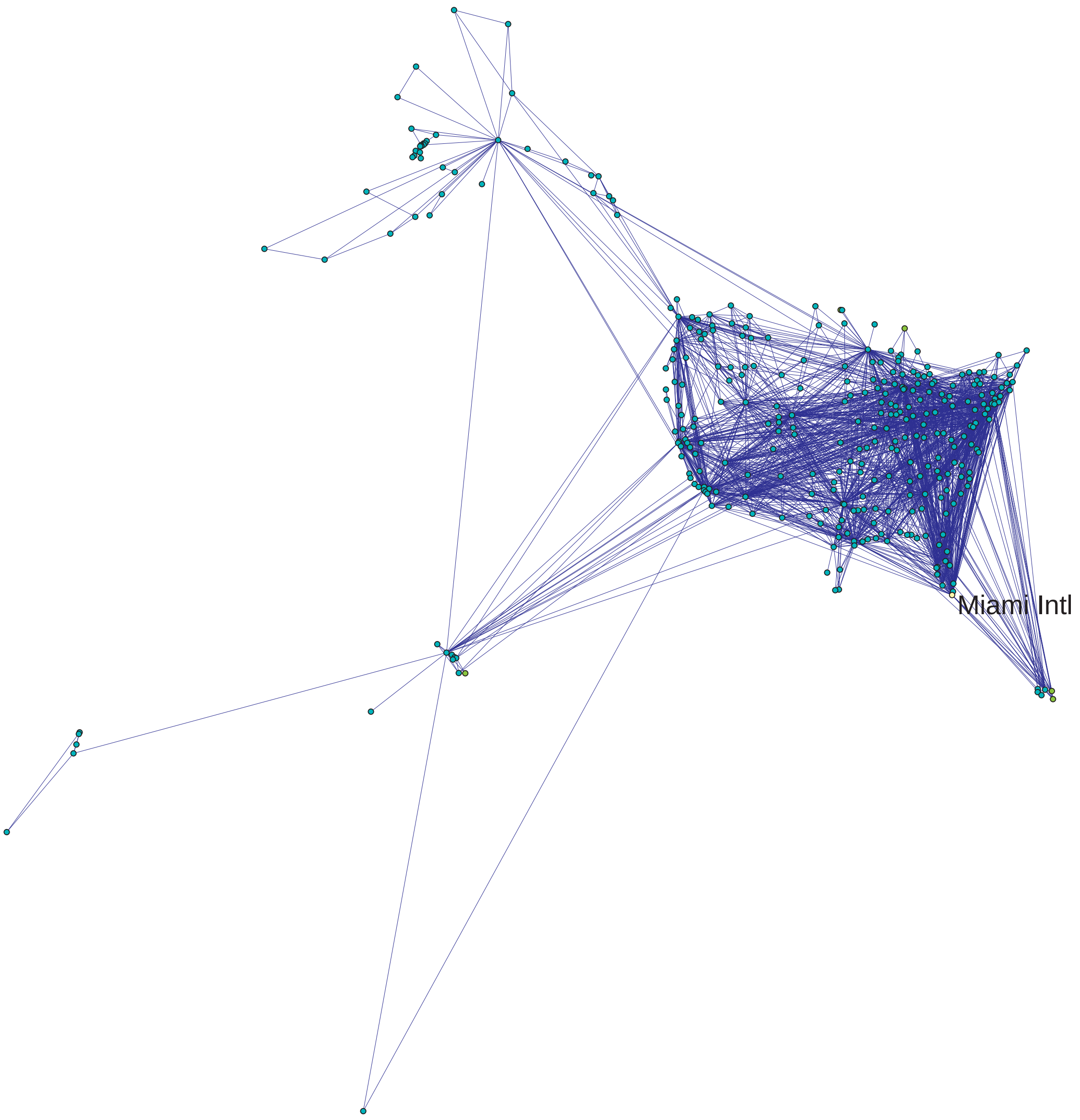
If at most one line can exist among any two vertices the density is a real number between 0 and 1.

Density of a network is one of the measures by which we can compare different networks.

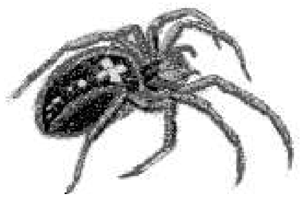
Example of large networks

- social networks
  - connections among people (friendship);
  - relation among political parties;
  - trade among organizations, countries;
  - genealogies;
  - citation networks;
  - computer networks (local networks, Internet, links among home pages);
  - telephone calls;
- flow charts in computer science;
- Petri nets;
- organic molecule in chemistry;
- connections among words in text;
- transportation networks (airlines, streets, electric networks. . .).

Airlines among 332 American airports (332 vertices, 2116 lines)



Pajek



`Pajek` is a program package for Windows 32 and 64, which enables analyses of *large networks*. Program is freely available at:

http://mrvar.fdv.uni-lj.si/pajek/

Analyses in Pajek are performed using six data structures:

1. network,
2. partition,
3. cluster,
4. permutation,
5. vector,
6. hierarchy.

First look at networks only.

Network

Default extension is .NET

Network can be defined in different ways on input file. Look at three of them:

1. List of neighbours (Arcslist / Edgeslist)

```
*Vertices 5
1 ”a”
2 ”b”
3 ”c”
4 ”d”
5 ”e”
*Arcslist
1 2 4
2 3
3 1 4
4 5
*Edgeslist
1 5
```

Explanation:

Data must be prepared in an input (ASCII) file. Program *NotePad* can be used for editing. In DOS environments the editor is called *Edit*. Much better than either is a shareware editor, *TextPad* (http://www.textpad.com/).

Words, starting with *, must always be written in first column of the line. They indicate the start of a definition of vertices or lines.

Using *Vertices 5 we define a network with 5 vertices. This must always be the first statement in definition of a network.

Definition of vertices follows after that – to each vertex we give a label, which is displayed between ”and ”. If labels of vertices are equal to their sequential numbers this part can be omitted.

Using *Arcslist, a list of directed lines from selected vertices are declared (1 2 4 means, that there exist two lines from vertex 1, one to vertex 2 and another to vertex 4).

Similarly *Edgeslist, declares list of undirected lines from selected vertex.

In the file and at the end of the file no additional empty lines are allowed (empty line means end of network definition).

2. *Pairs of lines (Arcs / Edges)*

```
*Vertices 5
1 ”a”
2 ”b”
3 ”c”
4 ”d”
5 ”e”
*Arcs
1 2 1
1 4 1
2 3 2
3 1 1
3 4 2
4 5 1
*Edges
1 5 1
```

Explanation:

This is the most general format. In this case every line is defined separately in new line - initial and terminal vertex of every line are given. Directed lines are defined using *Arcs, undirected lines are defined using *Edges. The third number in rows defining lines gives the value of the line. Lines from 2 to 3 and from 3 to 4 have value 2, all others have value 1.

In the previous format (Arcslist / Edgeslist) values of lines cannot be defined – the format is suitable only if all values of lines are 1.

If values of lines are not important the third number can be omitted (all lines get value 1).

3. Matrix

```
*Vertices 5
1 ”a”
2 ”b”
3 ”c”
4 ”d”
5 ”e”
*Matrix
0 1 0 1 1
0 0 2 0 0
1 0 0 2 0
0 0 0 0 1
1 0 0 0 0
```

Explanation:

In this format directed lines (arcs) are given in the matrix form (*Matrix). If we want to transform bidirected arcs to edges we can use
Network/Create New Network/
Transform/Arcs to Edges/Bidirected only

Only those elements necessary to define network were described so far. Additionally, Pajek enables precise definition of elements used for drawing networks (coordinates of vertices in space, shapes and colors of vertices and lines, ...).

Among internal formats (.NET), Pajek recognizes several other formats: UCINET DL; Vega; GEDCOM and some chemical formats: BS (Ball and Stick), MAC (Mac Molecule) and MOL (MDL MOLfile).

Interactive definition of networks

Simple networks can be defined inside program Pajek as well without definition in an input file:

First the empty network (network without lines) is built, later lines are added.

Select: Network/Create New Network/Empty Network

In our example with 5 vertices we input:
Enter number of vertices: 5

Then we select Draw/Network (or press Ctrl+G). The network is represented by layout in a new window. Vertices can be moved to other positions by clicking with left mouse button on selected vertex, holding the mouse button down and moving the mouse.

After that we define for every vertex all lines connected to the vertex: We click with right mouse button on selected vertex. A new window is opened, in which a double click on Newline vertices neighbouring a selected vertex are entered.

If an arc is going from a selected vertex to vertex 2, we input -2, if an arc is coming from vertex 2, we enter $+2$, if the two vertices are connected by an edge we enter only 2.

The value of a line is entered by clicking on the highlighted selection with the right mouse button in the same window and entering the desired value.

A line can be deleted by double clicking the highlighted selection in the same window with the left mouse button.

The Draw window can be refreshed using the command Redraw.

Some commands in Draw window

All commands are described in the html file (Manual) that is available on the Pajek homepage.

- **Options/Mark vertices using** – selects the way vertices are marked in the picture
- **Options/Lines** – visibility or nonvisibiliy of arcs and edges, selects the way lines are marked in the picture
- **Options/Size** – selects the size of vertices, size of font, size of arrows and width of lines
- **Options/Colors** – selects background color, color of vertices, lines, font…

Determining layouts of networks

The imagination of a network is often obtained only by a picture of it. Several automatic and manual drawing routines are included in program Pajek.

Automatic layout generation

- **Energy** – Idea: the network is represented like a physical system, and we are searching for the state with minimal energy. Two algorithms are included:
  - **Layout/Energy/Kamada-Kawai** – slower
  - **Layout/Energy/Fruchterman-Reingold** – faster, drawing in a plane or space (2D or 3D), and selecting the repulsion factor

  Using **Options/Interrupt** – we select in how much time (in seconds) the picture si optimized.
- **EigenValues** – A selection of 2 or 3 eigenvectors are computed and those eigenvectors become the coordinates of vertices. Nice pictures are usually obtained if there are symmetries in the network (social networks are not of that type, usually).

Manual drawing

We can move the vertex by clicking with left mouse button on a selected vertex and moving the mouse.

Pictures in space: a picture can be rotated by pressing (holding down) keys x, X, y, Y, z, Z (the key stands for axis of rotation, small/capital letters mean positive or negative orientation).

Any axis of rotation can be selected using **Spin/Normal**. After that, rotation around the selected axis is executed by holding down s or S.

A part of the network can be zoomed by selecting a window with the picture using the right mouse button. The complete picture is obtained using Redraw.

If we want to save changes for a network (in our case the picture), we select the icon for saving the network or **File/Network/Save**, and then choose the type of presentation and the name of the file. Types of presentation were defined in the beginning (input formats). If the name is equal to the name of the network that already exists, the old network is lost.

If we select Arc/Edges presentation, the new file will look like (the difference is only that coordinates of vertices are added):

```
*Vertices 5
1 ”a”      0.1672    0.3272    0.5000
2 ”b”      0.2029    0.7394    0.5000
3 ”c”      0.6144    0.8334    0.5000
4 ”d”      0.8328    0.4794    0.5000
5 ”e”      0.5565    0.1666    0.5000
*Arcs
1 2 1
1 4 1
2 3 2
3 1 1
3 4 2
4 5 1
*Edges
1 5 1
```

Output formats

The obtained pictures in a plane or space can be stored in output formats that are suitable for including pictures in text processors (e. g. Word) or examining using special viewers. Saving in output formats is available in option **Export**. The possible formats are:

- EPS – The picture can be shown using program GSView or included in text editors (e. g. LaTeX).
- SVG – *Scalable Vector Graphics* – it is vector graphics format (like EPS) – the picture can be resized without loosing the quality. SVG is recognized by Internet browsers like Firefox and Explorer. The picture in SVG can be further edited using InkScape (http://inkscape.org/)
- JPEG – It is just a snapshot of the picture. Since it is not a vector graphics the quality is not very good.
- Bitmap (BMP) – Again - snapshot of the Draw window – the obtained file can be quite large in size.
- Vosviewer – Network layout is exported to VOSviewer: http://www.vosviewer.com/
  We must first install the program, then tell Pajek where it is installed (*Locate VOSviewer*) and finally layouts can

be sent to VOSviewer (*Send to VOSviewer*).

- X3D – Extensible 3D Graphics – for pictures in space. X3D is recognized by Internet browsers using special plug-ins like Instant Reality: http://www.instantreality.org/

  The main advantage of the representation is that the layout is not statical but dynamical – we can travel in the picture, rotate the picture, select different views…

- Kinemages (*KINEmatic iMAGES*)
  The programs Mage and King are freely available at:
  http://kinemage.biochem.duke.edu/software/
  simple animation can be done with this programs too.

- VRML - Virtual Reality Modeling Language – for pictures in space. VRML is not supported anymore, X3D is a proper succesor of VRML.

Nice pictures of networks

Some properties of nice pictures of networks:

- not too many crossings of lines,
  a graph that can be drawn without crossing of lines is called *a planar* graph)
- not too many small angles among lines that have one vertex in common
- not too long or too short lines
  (all lines approximately of the same length)
- vertices should not be too close to lines

In Pajek coordinates of vertices are numbers between 0 and 1. The decimal delimiter is a period. In the case that coordinates are from some other interval, the picture can be extended/shrunk to the size of the window using Options/Transform/Fit Area.

Other commands in Options/Transform are: (commands can be applied to the whole picture or just selected part of it):

Resize – extend/shrink the picture for some factor in selected directions

Translate – translate the picture in a selected direction

Rotate 2D – rotate the picture in xy plane for selected angle

The properties of a nice picture can be checked in Pajek by selecting the desired property using Info in the Draw window:

- *Closest Vertices* – find the closest vertices, and color them yellow. In another window the distance among vertices is written.
- *Smallest Angle* – select the smallest angle among lines having one vertex in common. The corresponding three vertices are colored green. The angle is written in another window.
- *Shortest/Longest Line* – find the shortest line (color the corresponding two vertices red) and longest line (color the corresponding two vertices blue). The length of the two lines is displayed.
- *No. of Crossings* – count the number of crossings of lines (corresponding vertices are colored pink).
- *Vertex Closest to Line* – find the vertex that is the closest to any line. The corresponding three vertices are colored white. The distance of vertex from line is also displayed.

Basic information about a network

Basic information about a network can be obtained by
<u>Network/Info/General</u>
which is available in the main window of the program. We get

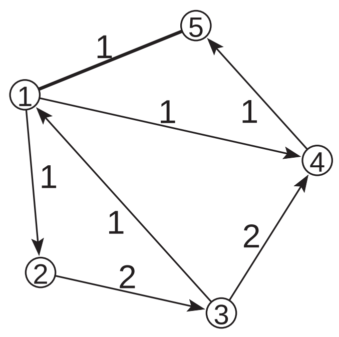
- number of vertices
- number of arcs
- number of directed loops
- number of edges
- number of undirected loops
- both densities of lines ($Density1$ in $Density2$)

Additionally we must answer the question:
Input 1 or 2 numbers: +/highest, -lowest
where we enter the number of lines with the highest/lowest value or interval of values that we want to output.

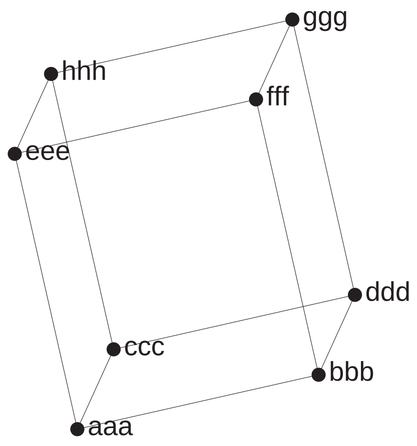
If we enter $\boxed{10}$, 10 lines with the highest value will be displayed. If we enter $\boxed{-10}$, 10 lines with the lowest value will be displayed. If we enter $\boxed{3\ \ 10}$, lines with the highest values from rank 3 to rank 10 will be displayed.

Examples

1. Build the following network interactively in Pajek:



2. Prepare the input file ex1.net with the description of network in the shape of cube, with the labels of vertices: aaa, bbb, ccc, ddd, eee, fff, ggg and hhh: Read the network, draw it in plane and space, and save it again.



3. Read the network that is stored in the file ex2.net and try all algorithms for automatic layout generation.