A comparative study of social network analysis tools

David Combe[1] *, Christine Largeron[1], Előd Egyed-Zsigmond[2] and Mathias Géry[1]
Université de Lyon
[1] *CNRS, UMR 5516, Laboratoire Hubert Curien, F-42000, Saint-Étienne, France*
Université de Saint-Étienne, Jean-Monnet, F-42000, Saint-Étienne, France
Email: {david.combe, christine.largeron, mathias.gery}@univ-st-etienne.fr
[2] *UMR 5205 CNRS, LIRIS*
7 av J. Capelle, F 69100 Villeurbanne, France
Email: elod.egyed-zsigmond@insa-lyon.fr

Abstract. Social networks have known an important development since the appearance of web 2.0 platforms. This leads to a growing need for social network mining and social network analysis (SNA) methods and tools in order to provide deeper analysis of the network but also to detect communities in view of various applications. For this reason, a lot of works have focused on graph characterization or clustering and several new SNA tools have been developed over these last years. The purpose of this article is to compare some of these tools which implement algorithms dedicated to social network analysis.

Keywords: Social Network Analysis, Tools, Benchmark, Community detection

1. Introduction

The explosion of Web 2.0 (blogs, wikis, content sharing sites, social networks, etc.) opens up new perspectives for sharing and managing information. In this context, among several emerging research fields concerning "Web Intelligence", one of the most exciting is the development of applications specialized in the handling of the social dimension of the Web. Particularly, building and managing virtual communities for Virtual Enterprises require the development of a new generation of tools integrating social network modeling and analysis.

Several decades ago, the first works on Social Networks Analysis (SNA) was carried out by researchers in Social Sciences who wanted to understand the behaviour and evolution of human networks [34,33]. Several indicators were proposed to characterize the actors as well as the network itself. One of these indicators, for example, was the centrality that can be used in marketing to discover the early adopters or the people whose activity is likely to spread information to many people in a shortest way.

Nowadays, the wide use of Internet around the world allows to connect a lot of people. According to the Facebook Factsheet page, there are currently over 500 millions active users and according to Datamonitor they will be around one billion in 2012. As pointed in the Gartner study [17], this very important development of the networks gives rise to a growing need for social network mining and social network analysis methods in order to provide deeper comprehension of the network and to detect communities and study their evolution for applications in areas such as community marketing, social shopping, recommendation mechanisms and personalization filtering or alumni management. For this reason, while many new technologies

*Corresponding author.

This work has been partly funded by the Web Intelligence project (région Rhône-Alpes)
http://www.web-intelligence-rhone-alpes.org

(wikis, social bookmarks and social tagging, etc) and services (GData, Google Friend Connect, OpenSocial, Facebook Beacon,...) were proposed on internet, several new SNA tools have been developed. These tools are very useful to analyze theoretically a social network but also to represent it graphically. They compute different indicators which characterize the network's structure, the relationships between the actors as well as the position of a particular actor. They also allow the comparison of several networks. The purpose of this article is to present some actual major tools and to describe some of their functionalities. A similar comparison has already been done in [22], but with a more statistical vision. Our comparative survey on the state-of-the-art tools for network visualization and analysis, is focused on three main points:

- Graph visualization;
- Computation of various indicators providing a local (i.e. at the node level) or a global description (i.e. on the whole graph);
- Community detection (i.e. clustering);

In order to present the characteristics of the different tools, the main concepts used to represent social networks are defined in the next section. The different measures we want to find in a SNA tool are presented in section 3. We will describe the benchmarking approach and the results of this comparative study in section 4 and then we will conclude.

2. Notations

The theoretical framework for social network analysis was introduced in the 1960s. Following the basic idea of Moreno [26] who suggested to represent agents by points connected by lines, Cartwright and Harary have proposed to analyze this sociogram using the graph theory. For this reason, they are considered as the founders of the modern graph theory for social network analysis [8].

Two types of graphs can be defined to represent a social network: one-mode and two-mode graphs.

2.1. One-mode Graph

When the relationships between actors are considered, the social network can be represented by a graph $G = (V, E)$ where V is the set of nodes (or vertices) associated to the actors , and $E \in V \times V$ is the set of edges which correspond to their relationships. This is the case, for instance in a classical dataset [35] related to a karate club where the nodes correspond to the members of the club and where the edges are used to describe their friendships. When the relationships are directed, edges are replaced by arcs. Nodes as well as edges can have attributes. In that case, we can talk then about labeled graphs.

2.2. Two-mode graph

When the relationships between two types of elements are considered, for example the members and the competitions in the karate club, a two-mode graph is most suited to represent the social network. A two-mode graph, also known as bipartite graph, is a graph with two types of vertices. The edges are allowed only between nodes of different types.

The most common way to store two-mode data is a rectangular data matrix with the two node types respectively in rows and columns. For example, a 2 dimensional matrix with the actors in rows and the events in columns can represent a two-mode graph for the karate club. This representation is very common in SNA [19]. Two-modes graphs can be transformed in one-mode graphs using a projection on one node type and creating edges between these nodes using different aggregation functions.

The concept of graph can be generalized by a hypergraph, in which two sets of vertices can be connected by an edge. A multigraph is a graph which is permitted to have edges that have the same end nodes.

In the next sections, we note $|V|$ and $|E|$ the number of vertices and edges in G and $deg(v)$, the degree of the node v giving the number of adjacent edges to v.

3. Expected functionalities of network analysis tools

This work focuses on different functionalities provided by network analysis tools. These functionalities are firstly the visualization of the network, secondly the computation of statistics based on nodes and on edges, and finally, community detection (or clustering).

3.1. Visualization

Visualization is one of the most wanted functionalities in graph handling programs, and this stays true for network analysis software.

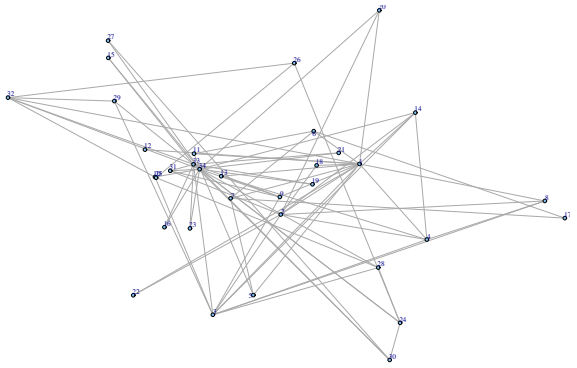

Fig. 1. Visualization of Zachary's Karate club using the igraph library and spring layout

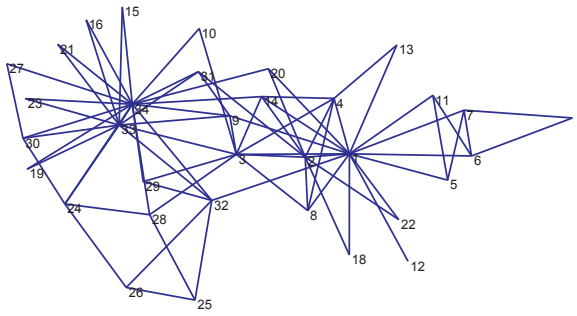

Fig. 2. Visualization of Zachary's Karate club using the Pajek application and Kamada-Kawai layout

Many algorithms consist in pushing isolated vertices toward empty spaces and in grouping adjacent nodes. These algorithms are directly inspired by physical phenomena. For example, edges can be seen as springs and nodes can be handled as electrically charged particles. The location of each element is recalculated step by step. These methods require several iterations in order to provide a good result on large graphs. Force-based layouts are simple to develop but are subject to poor local minimum results (see Fig. 1).

Among these algorithms, we can mention, Fruchterman Reingold, which is a well-used force-based algorithm for graph visualization [15]. An example is provided on Fig. 3. An alternative is the algorithm of Kamada-Kawai [24] (see Fig. 2), which has a faster convergence than Fruchterman Reingold, but which often does not give so good results than this last one. It can be envisaged to use Kamada-Kawai in order to calculate a first placement of the vertices. These two methods are among those called "spring algorithms".

Some other layouts are different in the way they provide a view of the neighborhood for a node (i.e. radial layout, hyperbolic layout). 3D graph visualization is the logical extension of planar representations. Most of the methods proposed are adaptable to 3D.

Local zoom based, so called fish-eye functionality can be also interesting to visually explore large graphs [16].

3.2. Indicator based network description

Many quantitative indicators have been defined on networks [34,33].

The descriptors at the network level are used to compare the proportion of nodes versus edges, or to evaluate properties of the graph like the randomness or small world distributions.

On the other hand, the descriptors at the node level are useful for detecting the nodes strategically placed in the network or highlighting those that take an important part in communication such as bridges or hubs.

3.2.1. Vertex and edge scoring

The place of a given actor in the network can be described using measures based on vertex scoring. Common types of vertex scoring are the centrality measures. Within graph theory and network analysis, there are various measures of the centrality of a vertex to determine the relative importance of this vertex within the graph. For example, to measure how important a person is within a social network, Freeman [14] has distinguished three main centralities:

a) Degree centrality: The first and simplest measure is the degree centrality. It emphasizes nodes with the high degrees [28].
In oriented graphs, we can distinguish:

   – incoming degree of a vertex v:

$$N^{+}(v) = |\{i \in V : (i, v) \in E(G)\}| \quad (1)$$

   – outgoing degree:

$$N^{-}(v) = |\{i \in V : (v, i) \in E(G)\}| \quad (2)$$

   – degree centrality [14]:

$$C_D(v) = \frac{deg(v)}{|V| - 1} \quad (3)$$

b) Closeness centrality: For connected graphs, closeness centrality is the inverse of the average distance

to all other nodes. This indicator can be useful for many applications in the real world. For instance, if edges were streets, the crossroad (vertex) with the highest closeness centrality would be the best place for emergency services.
Closeness centrality is defined by:

$$C_C(v) = \frac{|V| - 1}{\sum_{u \in V, u \neq v} d(v, u)} \quad (4)$$

where $d(v, u)$ is a distance, like for example the number of edges in the shortest path between two nodes or the sum of the weight of these edges, in weighted graphs.

c) Betweenness centrality: Betweenness centrality is another centrality measure of a vertex within a graph. Vertices that occur on many shortest paths between other vertices have higher betweenness than those that do not [14]. An improved implementation of this indicator has been proposed by Ulrik Brandes with a running time of $O(|V|.|E|)$ [6].
The betweenness of vertex u is defined by:

$$B_C(v) = \sum_{(u,w) \in V \times V, u \neq w, u \neq v, w \neq v} \frac{\sigma_{uw}(v)}{\sigma_{uw}} \quad (5)$$

where σ_{uw} is the number of shortest paths from nodes u to w and $\sigma_{uw}(v)$ is the number of shortest paths from u to w that pass through v. Redefining the graph, betweenness can also be defined for an edge e:

$$B_{EC}(e) = \sum_{(v,w) \in E, v \neq w} \frac{\sigma_{vw}(e)}{\sigma_{vw}} \quad (6)$$

where σ_{vw} is the number of shortest paths from nodes v to w and $\sigma_{vw}(e)$ is the number of shortest paths from v to w that pass through e.
There is also another type of centrality measure: the eigenvector centrality that measures the importance of a node in a network. It is based on the principle that connections to nodes having a high degree contribute more to the score of the node in question than connections to nodes having a low score.
These different measures can also be calculated on oriented graphs. For them, other measures can be defined, like for instance PageRank or HITS.

d) PageRank: The score computed by Page Rank [7] is higher for nodes that are highly connected and connected with nodes that are highly connected themselves. If $L(v, u)$ is the number of links from page v to u, then the Page Rank $PR(u)$ of the vertex u can be defined as:

$$PR(u) = (1 - d) + d * \sum_{v|(v,u) \in E} \frac{PR(v)}{L(v, u)} \quad (7)$$

The parameter d is a damping factor. PageRank score is iterated until convergence.
PageRank is a variant of the Eigenvector centrality measure.

e) HITS algorithm: Hyperlink-Induced Topic Search (HITS, also known as hubs and authorities) calculates two scores: hub and authority score [25]. The more a vertex has outgoing arcs, the higher is its hub score. The more a vertex has incoming links, the higher is its authority score. At the beginning every node are considered as hub and authority scores are fixed to a constant. Then the scores are updated and they converge after few iterations.
If u is one of the m vertex connected to v, the scores $auth(v)$ and $hub(v)$ are computed for v at the new iteration as follows:

$$\forall v, auth(v) = \sum_{u|(u,v) \in E} hub(u) \quad (8)$$

$$\forall v, hub(v) = \sum_{u|(v,u) \in E} auth(u) \quad (9)$$

These different measures can also be calculated on oriented graphs.

3.2.2. Network scoring

Network density is the rate of edges in the network over the number of edges that could exist in the network. This measure shows if the underlying graph is sparse or dense.

These indicators have since been translated in versions applicable to directed graphs, useful in information dissemination theory. This asymmetry leads to the concept of prestige.

a) Dyad Census: A dyad is a term borrowed from sociology used to describe a group of two people, i.e. the smallest possible social group. By extension, it

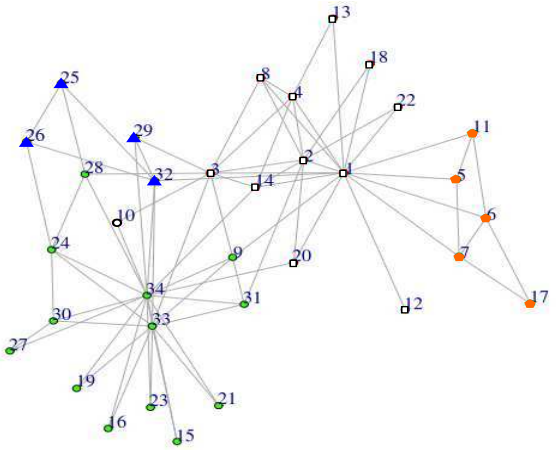

Fig. 3. Community detection with igraph and the spinglass algorithm

is used in social network analysis for designing two interacting nodes.
Four states are observable between two nodes (a and b) for directed graphs:

- no arc
- two mutual arcs
- a to b
- b to a

Each dyad is classified into one of the mutual, asymmetric or null categories and the proportion of each of these cases is provided. These counts help to know if the links follow a random or a small-world distribution [21].

b) Triad Census: In order to extend the dyad count, Davis and Leinhardt [11] have proposed the triad count, with 16 distinct cases (directed graphs). Triadic analysis performs the count of the triads in each configuration. Information provided is again useful for comparing a network with the random model.

3.2.3. Graph and vertice similarity

In social network analysis tools, one can expect to find functions expressing similarity of nodes in a graph and also functions to measure the similarity between graphs themselves. Some examples of similarity measures available in softwares are the Jaccard, Dice or Tanimoto similarity.

3.3. Clustering or community detection

The aim of clustering is to detect groups of nodes with dense connections within the groups and sparser connections between the groups. These groups are called *clusters* by statisticians and data mining professionals while sociologists prefer to use the word *communities*.

A very complete survey on graph clustering can be found in [13].

3.3.1. Main approaches of community detection

Among the different methods proposed to detect communities, two main approaches can be distinguished: on the one hand there is the hierarchical approach in which the nodes are aggregated in a hierarchy of clusters from the discrete partition to the whole network [23]. This approach evaluates the proximity between two nodes through a similarity measure and builds the groups using an agglomerative strategy, like the single linkage algorithm or the complete linkage algorithm. On the other hand, there is the partitional clustering which consists in directly dividing the network into a predefined number of groups. The minimum cut method is an example of this approach in which the groups are defined so as the number of edges between them is minimized.
The softwares considered in this benchmarking include three clustering methods. The first one is the Newman and Givan [27] method. This is a hierarchical method, based on the betweenness of the edges, which consists in removing the edge with highest betweenness, and repeating this process until no edge remains.

The second method, called Walktrap [31], is a partitional algorithm that uses a random walk in the graph in order to detect the components in which the walker tends to stay. A calculated distance between two vertices is calculated as the probability for a walker to go from a vertex to another. A hierarchical clustering is then performed in order to obtain the clusters.

The last algorithm is called Spinglass [32]. Fig.3 shows an example of community detection done with the spinglass algorithm of igraph. In this figure, different vertex shapes indicate different communities.

With hierarchical methods, a dendogram (Fig. 9) is the best representation for choosing the number of clusters to retain. Another way to determine the number of groups that must be retained consists in maximizing a particular criteria such as modularity.

3.3.2. Clustering validation

Modularity is a quality function useful to evaluate clustering. It has been proposed by Newman and

Girvan[27]. Modularity is defined by:

$$Q = \frac{1}{2.|E|} \sum_{(u,v)\in V\times V} (A_{uv} - P_{uv})\delta(C_u, C_v) \quad (10)$$

where the couple (u, v) runs over all pairs of vertices, A is the adjacency matrix where A_{uv} contains 1 if u and v are linked by an edge and 0 otherwise, P_{uv} is the expected number of edges between u and v, C_v is the group to which vertex v belongs and δ is the Kronecker delta, which is 1 if its two arguments are equal, and 0 otherwise. The clustering corresponding to a unique partition containing the whole graph has a modularity value of zero.

4. Benchmarking

Many tools have been created for network analysis and visualization purposes. A long list of tools is available on Wikipedia[1], with very different approaches. Many are purely academic software. Some are oriented toward visualization, other consist in APIs allowing graph and hypergraph modeling with sometimes the possibility of animation on vertices such as JUNG. Some tools are optimized for large data manipulation. Others propose low level implementations of specific algorithms.

In this survey, the official documentation has been inspected for libraries. We consider 4 tools: Pajek, Gephi, igraph and NetworkX that will be presented further in this section. The choice of them is based on:

- a balance between well established tools and newer ones, based on recent development standards (in terms of ergonomics, modularity and data portability),
- a SNA point of view. The tools must provide basic metrics for networks,
- the networks size can reach tens of thousands of nodes.

Pajek is a *legacy* software, with its own graph-oriented approach. Gephi represents a modern answer for graph study with GUI (graphical user interface), open source philosophy and plugin orientation. Networkx and igraph are two essential libraries for efficient large graph handling. The first one can be integrated easily in any problematic and the second one can be recommended for a *Mathlab-like* (console-based) approach.

The following sections describe the dataset and the criteria used in the benchmark.

4.1. Dataset

The dataset considered in this survey is a widely used data set in SNA literature. This dataset presents the affiliation graph between 34 members of the karate club of a US university in 1970.

Zachary's Karate Club[2] has 34 vertex and 78 edges. Each vertex is numbered. An edge is present between two nodes when the two corresponding individuals "consistently interacted in contexts outside those of karate class, workouts and club meetings" [35].

4.2. Evaluated criteria

In our benchmark, we have selected a set of evaluation criteria. These criteria are the license of the tool, the data format handled, the graph types supported, the amount of nodes that can be loaded in a reasonable time, the available indicators, the clustering algorithms included and the visualization layouts available. Each criterion is detailed in the following sections.

4.2.1. File formats

There are mainly three ways to express in a serial manner the structure of a network:

- adjacency matrix (square for directed graphs, triangular for undirected ones)
- adjacency lists (for directed graphs), where the source node is followed by the list of the nodes that are the targets of every arcs starting from the node
- vertices pairs.

Several file formats have been created in order to provide graph representations. Here are the main ones:

a) Pajek graph file format (.net extension), while not very well documented, is very popular among social network analysis tools (Fig. 4). It represents in a text file, first the vertices (one per line) and then the edges. This format is not often handled in the other implementations except the Pajek pro-

[1] `http://en.wikipedia.org/wiki/Social_network_analysis_software`

[2] The dataset is available at `http://vlado.fmf.uni-lj.si/pub/networks/data/ucinet/ucidata.htm`, at the date of April 2010, 5th.

```
*Vertices      34
      1 "1"
      2 "2"
                    ⋮
     34 "34"
*Arcs
      1      2       1
      1      3       1
      1      4       1
                    ⋮
     34     31       1
     34     32       1
     34     33       1
```

Fig. 4. Zachary dataset extract in Pajek .net format

gram, which allows edge representation with a matrix or an edge list or arc list (for directed graphs). Weighted networks are allowed. Weights in the optional third column are for the arcs.

b) GML (Graph Modelling Language) is also a structures text file, where nodes and edges begin with "node" and "edge" keywords and their content is between "[" and "]". It allows annotations as content, such as coordinates for vertices (see Fig. 5).
GML supports:

- directed and undirected graphs
- node and edge labels
- graphical placement of nodes (coordinates)
- other annotations

c) GraphML is an XML-based graph description language (see Fig. 6). As described in its documentation [3], it supports:

- directed, undirected, and mixed graphs,
- hypergraphs,
- hierarchical graphs,
- graphical representations, and
- application-specific attribute data.

As all XML-based representation, it is quite a verbose one.

d) DL (Data Language) format comes from the Ucinet program [5]. The common extension for this format is *.dat*. An example is given Fig. 7.
DL format supports:

```
Creator "Mark Newman on Fri Jul...2006"
graph
[
  node
  [
    id 1
  ]
  node
  [
    id 2
  ]
                    ⋮
  node
  [
    id 34
  ]
  edge
  [
    source 2
    target 1
  ]
                    ⋮
  edge
  [
    source 34
    target 33
  ]
]
```

Fig. 5. Zachary dataset extract in GML format

- edge representation with a full matrix, a half-matrix, an arcs list or an edges list,
- index labels,
- rectangular matrices for two-mode networks.

e) DOT is an other popular graph description language, handled mainly by Graphviz [12].

f) GEXF[4] is an XML-based format, from the GEXF Working Group. It supports

- dynamic graphs,
- application-specific attribute data, through the use of users XML namespaces,
- hierarchical structure (nodes can contain nodes)
- visualization and positioning information such as 3D coordinates, colours, shapes.

[3] http://graphml.graphdrawing.org

[4] http://gexf.net

```
<?xml version="1.0" encoding="UTF-8"?>
<graphml xmlns="http://graphml.
               graphdrawing.org/xmlns"
                    ⋮
  <graph id="G"
         edgedefault="undirected">
    <node id="1"/>
    <node id="2"/>
    <edge id="e1" source="1"
                   target="2"/>
                    ⋮
  </graph>
</graphml>
```

Fig. 6. Zachary dataset extract in GraphML format

```
DL
N=34 NM=2
FORMAT = FULLMATRIX DIAGONAL PRESENT
LEVEL LABELS:
ZACHE
ZACHC
DATA:
 0 1 1 1 1 1 1 1 ... 0 0 0 0 0 0 1 0 0
 1 0 1 1 0 0 0 1 ... 0 0 0 0 0 1 0 0 0
 1 1 0 1 0 0 0 1 ... 0 0 1 1 0 0 0 1 0
 1 1 1 0 0 0 0 1 ... 0 0 0 0 0 0 0 0 0
 1 0 0 0 0 0 1 0 ... 0 0 0 0 0 0 0 0 0

                    ⋮
```

Fig. 7. Zachary dataset in DAT format

4.3. Evaluated tools

Two libraries and 2 stand-alone programs have been compared here:

- Pajek [5]
- Gephi [6]
- igraph [7]
- NetworkX [8]

The two libraries, igraph and NetworkX, use a general purpose environment called *R*.

The R environment It is dedicated to statistics. It is organized into many packages amongst which, some are dedicated to social network analysis. The covered functionalities are:

- *tnet* [30] for weighted, two-mode, and longitudinal networks (networks study over the time) analysis,
- *statnet* [18] for statistical analysis of social networks,
- *sna* includes node and graph-level indices, structural distance and covariance methods, structural equivalence detection, theoretic models fitting, random graph generation, and 2D/3D network visualization

These packages are available on the Comprehensive R Archive Network[9].

igraph and *NetworkX* are two libraries suitable for social network analysis within the R environment. NetworkX can also be called in Python programs.

4.4. Benchmarking results

The benchmarking results are summarized in Table 1.

They are detailed in this section, following the evaluation criteria introduced previously (see 4.2): the license of the tool, the data format handled, the graph types supported, the available indicators, the clustering algorithms included and the visualization layouts available.

The first point is licensing. It appears that NetworkX has the most permissive license, allowing integration in proprietary software. Both igraph and Gephi have chosen GNU GPL which does not allow the integration in proprietary software. Pajek source code is undisclosed and the use of the software for commercial use is not free. In matter of data format, Gephi handles all the formats mentioned here. GEXF is not available elsewhere mainly because this format started in the Gephi project. DL comes with UCINET ; this last one being a project linked to Pajek, it is one of the preferred formats for this tool. GML and GraphML are not supported in Pajek, so you can prefer the .net format, which is universal in our panel.

Concerning the bipartite graphs study and their manipulation, most tools propose a few primitives, such as projection (conversion of a bipartite graph into a one-mode graph), but we would not recommend Gephi for that as two-modes graphs is not strictly two-mode

[5] http://pajek.imfm.si/doku.php
[6] http://gephi.org
[7] http://igraph.sourceforge.net
[8] http://networkx.lanl.gov/
[9] http://cran.r-project.org/

Software	Pajek [4]	Gephi [3]	NetworkX [20]	igraph [9]
Version	1.26	0.7 alpha	0.6	0.5.3
Type	Stand-alone software	Stand-alone software	Library	Library
Platform	Windows	Java	Python	R / Python / C libraries
License	Free for non-commercial use	GNU GPL	BSD License	GNU GPL
Expectable computing time	Fast (C)	Medium (Java)	Fast (C, Python)	Fast (C)
Tractable number of nodes	500,000 nodes	150,000 nodes	1,000,000 nodes	> 1.9 million relations (without attributes)
Time to load 10^5 nodes and 10^6 edges	24 seconds	40 seconds	137 seconds	11 seconds
File formats				
GML	No	Yes	Yes	Yes
Pajek (.net)	Yes	Import only	Yes	Yes
GraphML	Export only	Yes	Yes	Yes
DL	Yes	Yes	No	No
GEXF	No	Yes	No	No
Graph types				
Two-mode graphs	Yes	No	Yes	Yes
Multi-relational graphs	Yes	No	No	No
Temporality	Yes	No	Yes	No
Visualization layouts				
Fruchterman Reingold	Yes	Yes	No	Yes
Kamada Kawai	Yes	Yes	No	Yes
Other spring layouts	No	Yes	Yes	Yes
Indicators				
Degree centrality	Yes	Yes	Yes	Yes
Betweenness centrality	Yes	Yes	Yes	Yes
Closeness centrality	Yes	Yes	Yes	Yes
Dyad census	No	No	No	Yes
Triad census	Yes	No	No	Yes
HITS	No	Yes	Yes	Yes
Page Rank	No	Yes	Yes	Yes
Clustering algorithms				
Edge betweenness	No	No	No	Yes
Walktrap	No	No	No	Yes
Spinglass	No	No	No	Yes
Dendogram display	Yes	Yes	No	Yes

Table 1. Features and availability of the main algorithms in the retained software

graph enabled. Pajek can handle links from different kinds. The temporality starts being taken into account in different projects. For now, the data can be filtered in function of a year associated to the nodes for example, if the data format is adapted.

The tool appearing as the less efficient in matter of allowed vertices in memory is Gephi. After 200,000 nodes on our reference computer (Intel Core 2 Duo 2.5 GHz, 2 Go RAM, Windows), some errors or messages invite to increase the dedicated memory for the virtual machine pop up. The visualization pane is an important part of Gephi, while the other tools can process indicators independently of drawing the Graph. Such an architecture could penalize the application for this criterion. Pajek does not suffer for this point and can load 500,000 in 52 minutes. igraph is very fast for data loading (22 seconds for 2.9 millions of nodes, but the dataset was attribute-free (no name for nodes provided, as *.net* import is quite restricted for this tool). Gephi and NetworkX appears to be limited in their ca-

pacity by the RAM consumption. NetworkX is quite slow for loading 100,000 nodes, but the loading is reasonable beyond. Some features such as management of multi-graphs can be the cause of degraded performance.

The four softwares are suitable for computing common indicators, such as graph statistics, degree centrality, closeness centrality and betweenness centrality (igraph and NetworkX implementations of betweenness centrality are based on the algorithm from Brandes [6].) Dyad and triad census are available in igraph and Pajek (for triad census). For HITS and PageRank indexes you can not rely on Pajek which is not up to date. If you need to create your own indicators, the two libraries and Gephi are useful.

Community detection is experimental in Gephi with a beta version of Markov cluster algorithm (MCL) while few algorithms are available in igraph. Pajek offers hierarchical clustering capabilities. It can provide a dendogram representation of a hierarchical clustering, as an EPS (PostScript) image. igraph offers the dendogram plotting capabilities of R. As demonstrated on the igraph website[10], the few lines provided in Fig. 8 gives the Fig. 9. Gephi, Pajek and igraph gives a dendogram representation for the communities obtained. Any connection between the visualization Fig. 2 and the dendogram Fig. 9 must be done respecting the fact igraph enumerates nodes from 0.

Concerning visualization layouts, NetworkX lacks of basic algorithms. If you need advanced visualization, you have to switch your data to an other platform. The three other tools perform the Fruchterman Reingold and Kamada Kawai popular force-based algorithms.

```
library(igraph)
g <- read.graph("karate.net",
format="pajek")
wt <- walktrap.community(g,
modularity=TRUE)
dend <- as.dendrogram(wt,
use.modularity=TRUE)
plot(dend, nodePar=list(pch=c(NA, 20)))
```

Fig. 8. Plot a dendogram with Walktrap and igraph

NetworkX is well documented, it is interesting but the clustering algorithms are missing. Nodes and edges can be any kind of objects (the only condition is to provide a hash function for it). Using programming languages it makes easy to redefine objects such as nodes in order to handle them as arbitrary objects. It has also some interesting functions if you use bipartite graphs.

[10] http://igraph.sourceforge.net

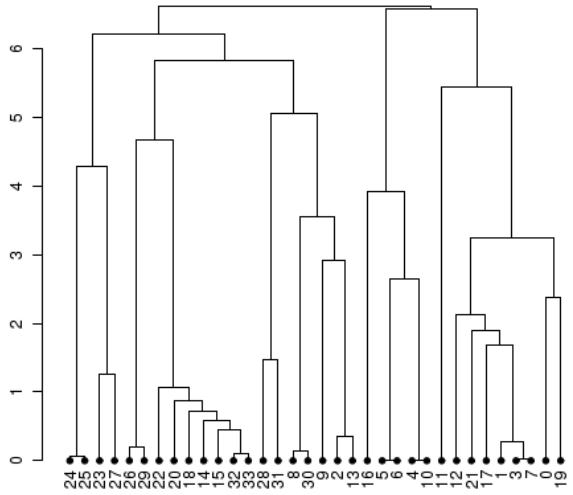

Fig. 9. Dendogram of the Walktrap algorithm results on the Zachary dataset (igraph website example)

igraph offers many algorithms among which some clustering oriented ones. It is available for both Python and R environments, and C libraries are available as well. With R, it is easy to integrate igraph routines in a statistical process. A graphical user interface exists which offers easy visualization and some basic analysis functions. igraph is performance-oriented and majority of its functionalities are implemented in C. 3D visualization layouts are available. It offers some node-related neighborhood similarity indexes such as Jaccard, Dice and the inverse log-weighted similarities [1].

Pajek is a closed-source software. It is fast tool and comfortable for some visualization purposes. It is not as extensible as the three other studied software. Nevertheless, Pajek is useful in hierarchical data manipulation and provides powerful and accessible data manipulation functions. 3D visualization and its export in VRML are also available.

Gephi is a quite new tool and it is updated frequently. Many functionalities are already supported, but several algorithms are missing. Its ergonomics makes Gephi easy to use. The rendering is highly customizable and quite fast. It is possible to move vertices while layout algorithms are performing.

4.5. Other interesting software for social network analysis

There are many other SNA tools available, we tested some of them such as:

- GraphViz [12] is dedicated to graph visualization.
- Tulip [10] can handle over 1 million vertices and 4 millions edges. It has visualization, clustering and extension by plug-ins capabilities.
- UCInet [5] is not free. It uses Pajek and Netdraw for visualization. It is specialized in statistical and matricial analysis. It calculates indicators (such as triad census, Freeman betweenness) and performs hierarchical clustering.
- JUNG [29], for Java Universal Network/Graph Framework, is mainly developed for creating interactive graphs in Java user interfaces, JUNG has been extended with some SNA metrics.
- GUESS [2] is dedicated to visualization purposes. It is published under the GPL license.

The reasons why other tools haven't been detailed above are:

- their narrow and specialized functionalities focalized on a single aspect, i.e. GUESS on visualization,
- factually replaced by other tools with the same target features and audience (Tulip with Gephi),
- are not focused on a computer science vision,
- are not freely available.

4.6. How to choose the right software for you?

The first question you have to ask is: "How to choose the right software?". If you need standard graph visualization, it is likely that you can find a software that suits you. If your data is not in a standardized format given in the list above, the best way is to generate a suited representation from your memory-loaded graph or to convert it into GML for example. You can also look on this Wikipedia page[11] for a list of input/output formats allowed by a large panel of programs. If your need is specific or your graph needs to be handled with specific attributes for vertices and edges, you should take a look to the libraries. In order to choose a program for visualizing or manipulating graphs, it is advisable to try a few of them to check if the approach fits your problematic. For libraries, the choice depends on your favorite language. The computation time can also be an important criterion in your choice ; if it is the case, prefer a Python or a C-based software. If you are interested in an interactive console (as the MATLAB experience), you should definitely try igraph on R.

[11] http://en.wikipedia.org/wiki/Social_network_analysis_software

5. Conclusion

The fact that Social Network Analysis is situated between several domains (sociology, computer science, mathematics and physics) has led to many different methodological approaches and to a lot of tools. That is why so many programs have been created in order to manipulate and study them[11]. While a standalone software is very useful for graph visualization (up to a maximum of few thousands of nodes), data format conversion or indicators computation, libraries are more adapted for tasks involving tens of thousands of nodes and for operations such as the union and the difference between sets of nodes or for the clustering. A fair separation of the algorithms, the user interface and the visualization pane is important. Gephi adopted this approach with the recent release of the *Gephi toolkit*, a library created from the Gephi logic and algorithms.

We can also say that today the freely available tools are able to provide a very rich set of functionalities, but if one wants specific analysis, a commercial software or complementary code developments may be needed.

Finally at this moment, the main challenges concerning the graph exploration are oriented toward high-level visualization (i.e. hierarchical graphs), while amongst the possible enhancements of social network analysis tools, we can mention firstly the temporal analysis which should allow to study the evolution of networks over time, and secondly social mining which simultaneously exploits the attributes of nodes and the graph structure.

References

[1] L.A. Adamic and E. Adar. Friends and neighbors on the web. *Social Networks*, 25(3):211–230, 2003.

[2] E. Adar. Guess: a language and interface for graph exploration. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, page 800. ACM, 2006.

[3] M. Bastian, S. Heymann, and M. Jacomy. Gephi: An Open Source Software for Exploring and Manipulating Networks. In *International AAAI Conference on Weblogs and Social Media*, pages 361–362, 2009.

[4] V. Batagelj and A. Mrvar. Pajek-program for large network analysis. *Connections*, 21(2):47–57, 1998.

[5] S.P. Borgatti, M.G. Everett, and L.C. Freeman. Ucinet for Windows: Software for social network analysis. *Harvard, MA: Analytic Technologies*, 2002.

[6] U. Brandes. A Faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25:163–177, 2001.

[7] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer networks and ISDN systems*, 30(1-7):107–117, 1998.

[8] D. Cartwright and F. Harary. A graph theoretic approach to the investigation of system-environment relationships. *Journal of Mathematical Sociology*, 5:87–111, 1977.

[9] G. Csárdi and T. Nepusz. The igraph software package for complex network research. *InterJournal Complex Systems*, 1695, 2006.

[10] A. David. Tulip. *Lecture notes in computer science*, pages 435–437, 2002.

[11] J. A. Davis and S. Leinhardt. The Structure of Positive Interpersonal Relations in Small Groups. *Sociological Theories in Progress*, 2:218–251, 1967.

[12] J. Ellson, E. Gansner, L. Koutsofios, S. North, and G. Woodhull. Graphviz - open source graph drawing tools. In *Graph Drawing*, pages 594–597. Springer, 2001.

[13] Santo Fortunato. Community detection in graphs. *Physics Reports*, page 103, juin 2009.

[14] L.C. Freeman. Centrality in social networks conceptual clarification. *Social networks*, 1(3):215–239, 1979.

[15] T.M.J. Fruchterman and E.M. Reingold. Graph Drawing by Force-directed Placement. *Software: Practice and Experience*, 21(11):1129–1164, 1991.

[16] E.R. Gansner, Y. Koren, and S. North. Topological fisheye views for visualizing large graphs. *IEEE Transactions on Visualization and Computer Graphics*, pages 457–468, 2005.

[17] Gartner. Hype Cycle for social software, 2008. G00158239, 2008.

[18] S.M. Goodreau, M.S. Handcock, D.R. Hunter, and C.T. Butts. A statnet Tutorial. *Journal of statistical software*, 24(9):1, 2008.

[19] J.-L. Guillaume and M. Latapy. Bipartite structure of all complex networks. *Information Processing Letters*, 90(5):215–221, 2004.

[20] A.A. Hagberg, D.A. Schult, and P.J. Swart. Exploring network structure, dynamics, and function using NetworkX. In *Proc. 7th SciPy Conf., Varoquaux G, Vaught T, and Millman J (Eds)*, pages 11–15, 2008.

[21] PW. Holland and S. Leinhardt. A method for detecting structure in sociometric data. *American Journal of Sociology*, 76(3):492–513, 1970.

[22] M. Huisman and M.A.J. Van Duijn. *Software for social network analysis*, pages 270–316. 2004.

[23] S.C. Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, September 1967.

[24] T. Kamada and S. Kawai. An algorithm for drawing general undirected graphs. *Information processing letters*, 31(12):7–15, 1989.

[25] J.M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.

[26] J.L. Moreno. *Who shall survive?* New York: Beacon Press, 1934.

[27] M.E.J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):26113, 2004.

[28] J. Nieminen. On the centrality in a graph. *Scandinavian Journal of Psychology*, 15(1):332–336, 1974.

[29] J. O'Madadhain, D. Fisher, S. White, and Y. Boey. The jung (java universal network/graph) framework. *University of California, Irvine, California*, 2003.

[30] T. Opsahl. Structure and Evolution of Weighted Networks. pages 104–122, 2009.

[31] P. Pons and M. Latapy. Computing communities in large networks using random walks. *Computer and Information Sciences-ISCIS 2005*, pages 284–293, 2005.

[32] J. Reichardt and S. Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74(1):16110, 2006.

[33] J. Scott. *Social Network Analysis : A handbook*. Newbury Park, CA, Sage Publications, 1994.

[34] S. Wasserman and K. Faust. *Social Network Analysis*. Cambridge University Press, 1994.

[35] WW Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33(4):452–473, 1977.