Vladimir Batagelj / Andrej Mrvar

Pajek—Program for Analysis and Visualization of Large Networks

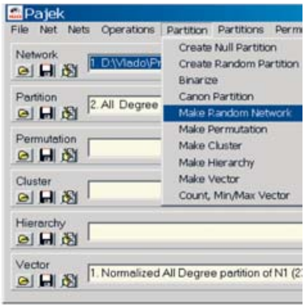
Pajek is a program for Windows for analysing and visualizing large networks with some thousands or even millions of vertices. In the Slovene language the word “pajek” means “spider.” The latest version of Pajek is freely available for non-commercial use at its home page *http://vlado.fmf.uni-lj.si/pub/networks/pajek/*

We started the development of Pajek in November 1996. Pajek is implemented in Delphi (Pascal). Some procedures were contributed by Matjaz Zaversnik. The main motivation for the development of Pajek was the observation that several sources of large networks exist that are already in machine-readable form. Pajek is intended to provide tools for analysing and visualizing such networks: collaboration networks, organic molecules in chemistry, protein-receptor interaction networks, genealogies, Internet networks, citation networks, diffusion (AIDS, news, innovations) networks, data-mining (2-mode networks), etc. See also the collection of large networks at: *http://vlado.fmf.uni-lj.si/pub/networks/data/*

The design of Pajek is based on our previous experiences gained in the development of graph data structure and the algorithm libraries Graph and X-graph, the collection of network analysis and the visualization programs STRAN, RelCalc, Draw, Energ, and the SGML-based graph description markup language NetML. The main goals in the design of Pajek are:

- to support abstraction by (recursive) decomposition of a large network into several smaller networks that can be treated further using more sophisticated methods;
- to provide the user with some powerful visualization tools;
- to implement a selection of efficient (subquadratic) algorithms for the analysis of large networks.

With Pajek we can: find clusters (components, neighbourhoods of important vertices, cores, etc.) in a network, extract vertices that belong to the same clusters and show them separately, possibly with parts of the context (detailed local view), shrink vertices in clusters and show relations among clusters (global view). Besides ordinary (directed, undirected, mixed) networks Pajek also supports 2-mode networks (bipartite valued) graphs—networks between two disjointed sets of vertices, and temporal networks (dynamic graphs—networks changing over the course of time).



Network: 1. D:\Vlado\Pr

Partition: 2. All Degree

Permutation:

Cluster:

Hierarchy:

Vector: 1. Normalized All Degree partition of N1 (2

Data structures

In Pajek, analysis and visualization are performed using 6 data types:

- network (graph),
- partition (nominal or ordinal properties of vertices),
- vector (numerical properties of vertices),
- cluster (subset of vertices),
- permutation (reordering of vertices, ordinal properties), and
- hierarchy (general tree structure on vertices).

We intend to extend this list with a support of multiple networks and partitions of lines. The power of Pajek is based on several transformations that support different transitions among these data structures. The menu structure of Pajek's main window is also based on these. Pajek's main window uses a "calculator" paradigm with a list-accumulator for each data type. The operations are performed on the currently active (selected) data and return the results through accumulators. The procedures are available through the main window menus. Frequently used sequences of operations can be defined as macros. This also allows groups of users from different fields (social networks, chemistry, genealogy, computer science, mathematics ...) access to adaptations of Pajek for specific tasks. Pajek also supports repetitive operations on a series of networks.

Algorithms

To support the design goals we implemented several algorithms known from the literature on the subject, but for some tasks, new and efficient algorithms suitable for dealing with large networks had to be developed. They mainly provide different ways of identifying interesting substructures in a given network. To extend the range of Pajek, on very large networks most basic operations work in-place (destroying the input network). In Pajek, several known efficient algorithms are implemented, such as:

- simplifications and transformations: deleting loops, multiple edges, transforming arcs to edges etc.;
- components: strong, weak, biconnected, symmetric;
- decompositions: symmetric-acyclic, hierarchical clustering;
- paths: shortest path(s), all paths between two vertices;
- flows: maximum flow between two selected vertices;
- neighborhood: k-neighbours;
- CPM—critical paths;
- social networks algorithms: centrality measures (see Figure 1), hubs and authorities, measures of prestige, brokerage roles, structural holes, diffusion partitions;
- measures of dependencies among partitions/vectors: Cramer's V, Spearman rank correlation coefficient, Pearson correlation coefficient, Rajski co-efficient;
- extracting subnetwork;
- shrinking clusters in network (generalized blockmodeling);
- reordering: topological ordering, Richards' numbering, Murtagh's seriation and clumping algorithms, depth/breadth first search.

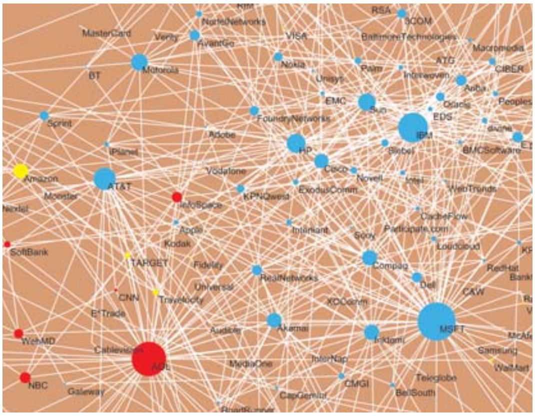


Figure 1: A zoom view of the main part of Internet industries (collected by Valdis Krebs) 219 vertices, 631 edges. Each node in the network represents a company that competes in the Internet industry, 1998 to 2001; red—content, blue—infrastructure, yellow—commerce. Two companies are connected with an edge if they have announced a joint venture, strategic alliance or other partnership. The vertex size is proportional to its betweenness.

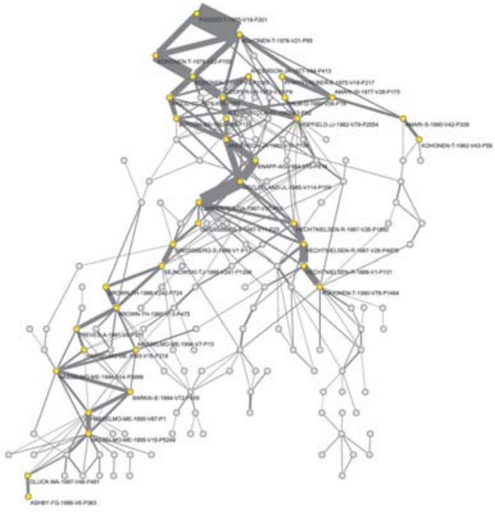


Figure 2: Main subnetwork at level 0.007 of the SOM (self organizing maps) citation network (4470 vertices, 12731 arcs). The arc weights are proportional to the number of different source-sink paths passing through the arc.

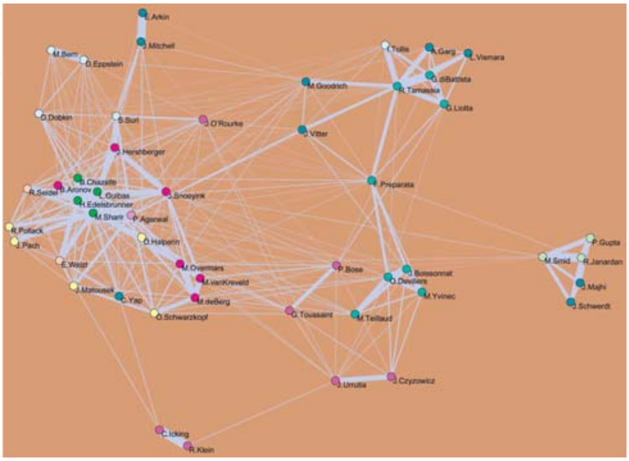


Figure 3: pS-core at level 46 of the collaboration network (7343 vertices, 11898 edges, edge weight counts the number of common works) in the field of computational geometry.

Special algorithms

We also included in Pajek several algorithms resulting from our own research in analysis of large networks.

- islands: If we represent a given or computed value of vertices/lines as a height of vertices/lines and we immerse the network in water up to a selected level, we get islands. Varying the level, we get different islands. Islands are a very general and efficient approach to determine the "important" sub-networks in a given network.
- citation weights: Citation network analysis started in 1964 with the paper by Garfield et al. In 1989 Hummon and Doreian proposed three indices.
- weights of arcs that provide us with an automatic way to identify the (most) important part of the citation network. We developed algorithms to compute two of these indices efficiently. See Figure 2.
- cores and generalized cores: The notion of core was introduced by Seidman in 1983. Vertices belonging to a k-core have to be linked to at least k other vertices of the core. A very efficient algorithm exists for determining cores. The notion of core can be extended to other vertex functions and for several of them the corresponding cores can be efficiently determined. See Figure 3.
- pattern searching: If a selected pattern determined by a given graph does not occur frequently in a sparse network, the straightforward backtracking algorithm applied for pattern searching quickly finds all appearances of the pattern even in the case of very large networks. Pattern searching was successfully applied to searching for patterns of atoms in molecula (carbon rings) and searching for re-linking marriages in genealogies.
- triads: A triad is a subgraph on three given vertices. There are 16 types of triads. Several network properties can be expressed in terms of their triadic spectrum—the distribution of all their triads.

- triangular networks: We can assign to a given graph a triangular network in which every line of the original graph receives as its weight the number of triangles that contain it. Triangular weights, combined with islands, provide us with a very efficient way of identifying dense parts of a graph.
- generating large random networks: Pajek contains very efficient algorithms for generating random networks of the Erdös-Renyi type (undirected, directed, acyclic, undirected bipartite, directed bipartite, acyclic bipartite, 2-mode, and others). It also provides some procedures for generating random scale free networks.
- normalizations: The normalization approach was developed for quick inspection of (1-mode) networks obtained from 2-mode networks—a kind of network-based data-mining. In networks obtained from large 2-mode networks there are often huge differences in weight. Thus it is not possible to compare the vertices according to the raw data. Beforehand, we have to normalize the network to make the weights comparable. There are several ways of doing this. For example: $$\mathrm{Geo}_{uv} = \frac{w_{uv}}{\sqrt{w_{uu}w_{vv}}}$$

After a selected normalization, the important parts of a network are obtained by line-cutting the normalized network at selected level t and preserving components with at least k vertices.

Algorithms for small networks

Although it was developed primarily for analysis of large networks, Pajek is also often used especially for visualizing small networks. It also contains some data analysis procedures with higher order time complexities which can be therefore be used only on smaller networks, or selected parts of large networks: hierarchical clustering, generalized block modelling, partitioning signed graphs, TSP (Traveling Salesman Problem), computing geodesics matrices, etc.

Layout Algorithms and Layout Features

Since large networks cannot be visualized in detail in a single view, we have first to identify interesting substructures in such networks and then visualize them as separate views. Special emphasis is laid in Pajek on automatic generation of network layouts. Several standard algorithms for automatic graph drawing are implemented: spring embedders (Kamada-Kawai and Fruchterman-Reingold), layouts determined by eigenvectors (the Lanczos algorithm), drawing in layers (genealogies and other acyclic structures), fish-eye views and block (matrix) representation. See Figure 4.

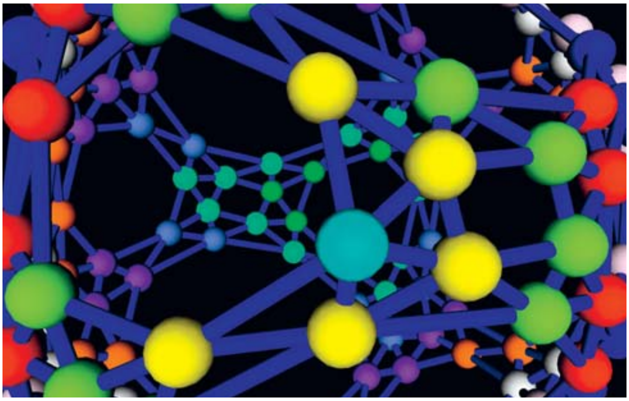


*Figure 4:
An eigenvector-based
3D layout of a
5-regular graph.*

These algorithms were modified and extended to enable additional options: drawing with constraints (optimizing the selected part of the network, fixing some vertices to predefined positions, using values of edges as similarities or dissimilarities), drawing in 3D space. Pajek also provides tools for manual editing of graph layouts. The values of vectors can be used to determine several elements of network display such as X, Y, Z coordinates and the size of the vertex shape. The partition can be represented graphically by the color and shape of vertices. The values of edges can also be represented by thickness and/or color. Pajek also supports drawing sequences of networks in its Draw window, and exports sequences of networks in suitable formats that can be examined with special 2D or 3D viewers (such as SVG and Mage). Pictures in SVG can be further controlled using support written in Javascript.

Interfaces

Pajek also supports some non-native input formats: UCINET DL files; chemical MDLMOL and BS; and genealogical GEDCOM. The layouts can be exported in the following output graphic formats that can be examined by special 2D and 3D viewers: Encapsulated PostScript (EPS), Scalable Vector Graphics (SVG), VRML, MDLMOL/chime, and Kinemages (Mage). The main window menu Tools enables export of Pajek's data to statistical programs R and SPSS. In the Tools menu, the user can prepare calls to her/his favorite viewers and other tools. It is also possible to run Pajek (+macros) from other programs (R, Ucinet, and others).

This presentation of Pajek is a shortened and updated version of the chapter V. Batagelj, A. Mrvar. *Pajek—Analysis and Visualization of Large Networks,* in Jünger, M., Mutzel, P. (Eds.) *Graph Drawing Software,* pp 77-103. Springer, Berlin, 2003

This work was partially supported by the Ministry of Education, Science and Sport of Slovenia, Projects J1-8532 and Z5-3350.

Vladimir Batagelj / Andrej Mrvar

Pajek – Ein Programm zur Analyse und Visualisierung großer Netzwerke

Pajek ist ein unter Windows laufendes Programm zur Analyse und Visualisierung von großen Netzwerken mit Tausenden, ja, Millionen von Knoten (Vertices). „Pajek“ ist das slowenische Wort für „Spinne“. Die neueste Version von Pajek ist für nicht-kommerzielle Zwecke frei unter *http://vlado.fmf.uni-lj.si/pub/networks/pajek/* erhältlich.

Die Entwicklung von Pajek begann im November 1996. Das Programm ist in Delphi (Pascal) geschrieben. Einige Prozeduren hat Matjaz Zaversnik beigetragen. Hauptmotivation für die Entwicklung von Pajek war die Beobachtung, dass zahlreiche Quellen großer Netzwerke bereits in maschinenlesbarer Form vorliegen. Pajek sollte Werkzeuge zur Analyse und Visualisierung von solchen Netzwerken zur Verfügung stellen: von Kooperationsnetzwerken, organischen Molekülen in der Chemie, Netzwerken von Protein-Rezeptor-Wechselwirkungen, Genealogien, Internet-Netzwerken, Zitiernetzwerken, Diffusionsnetzwerken (AIDS, Nachrichten,