Visualisation and exploration of scientific data using graphs

Ben Raymond and Lee Belbin

Australian Government, Department of the Environment and Heritage
Australian Antarctic Division
Channel Highway, Kingston 7050 Australia
ben.raymond@aad.gov.au

Abstract. We present a prototype application for graph-based data exploration and mining, with particular emphasis on scientific data. The application has a Flash-based graphical interface and uses semantic information from the data sources to keep this interface as intuitive as possible. Data can be accessed from local and remote databases and files. The user can generate a number of graphs that represent different views of the data. Graphs can be explored using an interactive visual browser, or graph-analytic algorithms. We demonstrate the approach using marine sediment data, and show that differences in benthic species compositions in two Antarctic bays are related to heavy metal contamination.

1 Introduction

Structured graphs have been recognised as an effective framework for scientific data mining — e.g. [1, 2]. A graph consists of a set of nodes connected by edges. In the simplest case, each node represents an entity of interest, and edges between nodes represent relationships between entities. Graphs thus provide a natural framework for investigating relational, spatial, temporal, and geometric data [2]. Graphs have also seen a recent explosion in popularity in science, as network structures have been found in a variety of fields, including social networks [3, 4], trophic webs [5], and the structures of chemical compounds [6–8]. Networks in these fields provide both a natural representation of data, as well as analytical tools that give insights not easily gained from other perspectives.

The Australian Antarctic Data Centre (AADC) sought a graph-based visualisation and exploration tool that could be used both as a component of in-house mining activities, as well as by clients undertaking scientific analyses.

The broad requirements of this tool were:

1. *Provide functionality to construct, view, and explore graph structures, and apply graph-theoretic algorithms.*
2. *Able to access and integrate data from a number of sources.* Data of interest typically fall into one of three categories:

- databases within the AADC (e.g. biodiversity, automatic weather stations, and state of the environment reporting databases). These databases are developed and maintained by the AADC, and so have a consistent structure and are directly accessible.
- flat data files (including external remote sensed environmental data such as sea ice concentration [9], data collected and held by individual scientists, and data files held in the AADC that have not yet been migrated into actively-maintained databases).
- web-accessible (external) databases. Several initiatives are under way that will enable scientists to share data across the web (e.g. GBIF [10]).

3. *Be web browser-based.* A browser-based solution would allow the tool to be integrated with the AADC's existing web pages, and thus allow clients to explore the data sets before downloading. It would also allow any bandwidth-intensive activities to be carried out at the server end, an important consideration for scientists on Antarctic bases wishing to use the tool.
4. *Have an intuitive graphical interface* (suitable for a general audience) that would also provide sufficient flexibility for more advanced users (expected to be mostly internal scientists).
5. *Integrated with the existing AADC database structure.* To allow the interface to be as simple as possible, we needed to make use of the existing data structures and environments in the AADC. For example, the AADC keeps a data dictionary, which provides limited semantic information about AADC data, including the measurement scale type (nominal, ordinal, interval, or ratio) of a variable. This information would allow the application to make informed processing decisions (such as which dissimilarity metric or measure of central tendency to use for a particular variable) and thus minimise the complexity of the interface.

Existing software that we were aware of met some but not all of these requirements. A summary of a selection of graph software is presented in Table 1 (an exhaustive review of all available graph software is beyond the scope of this paper). This paper describes a prototype tool that can be used to create and explore graph structures from a variety of data sources. The graphical interface has been written as a Flash application; the server-side code is written in ColdFusion (our primary application development environment). The interface can also accept text-based commands for users wishing additional flexibility.

2 Methods

The exploratory analysis process can be divided into three main stages — graph construction; visual, interactive exploration; and the application of specific analytical algorithms. In practice, these components would be used in an interactive, cyclical exploratory process. We discuss each of these aspects in turn.

Table 1. A functional summary of a selection of graph software. BG: the package provides functionality for constructing graphs from tabular or other data (manual graph construction excluded); DB,WS: direct access to data from databases/web services; L&D: provides tools for the layout and display of graphs; A: provides algorithms for the statistical analysis of graphs; Int.: interface type; BB: is web browser-based. †Small graphs only. ‡Designed for large graphs. *Limited functionality when run as an applet

Package	BG	DB	WS	L&D	A	Int.	BB	Summary
GGobi[28]	✓	✓	✗	✓†	✗	GUI	✗	General data visualisation system with some graph capabilities
Zoomgraph[29]	✓	✓	✗	✓‡	✓	Text	✓*	Zoomable viewer with database-driven back end
UCINET[31]	✓			✓	✓	GUI	✗	Popular social network analysis package
Pajek[30]	✗			✓‡	✓	GUI	✗	Analysis and visualization of large networks
Tulip[34]	✗			✓‡	✓	GUI	✗	Large graph layout and visualisation
LGL[35]	✗			✓‡	✗	GUI	✓	Large graph layout
GraphViz [36]	✗			✓	✗	Text	✗	Popular layout package
SUBDUE[13]	✗			✗	✓	Text	✗	Subgraph analysis package

2.1 Graph construction

Currently, data can be accessed from one or more local or remote databases (local in this context means "within the AADC") or user files. Accessing multiple data sources allows a user to integrate their data with other databases, but is predictably made difficult by heterogeneity across sources. We extract data from local databases using SQL statements; either directly or mediated by graphical widgets. Local files can be uploaded using http/get and are expected to be in comma-separated text format. Users are encouraged to use standardised column names (as defined by the AADC data dictionary), allowing the semantic advantages of the data dictionary to be realised for file data. Remote databases can be accessed using web services. Initially we have provided access only to GBIF data [10] through the DiGIR protocol. Data from web service sources are described by XML schema, which can be used in a similar manner to the data dictionary to provide limited semantic information.

To construct a graph representation of these data, the user must specify which variables are to be used to form the nodes, and a means of forming edges between nodes. Nodes are formed from the discrete values (or n-tuples) of one or more variables in the database. The graphical interface provides a list of available data sources, and once a data source is selected, a list of all variables provided by that data source. This information comes from the column names in a user file or database table, or from the "concepts" list of a DiGIR XML resource file. Available semantic information is used to decide how to discretise the node variables. Continuous variables need to be discretised to form individual nodes.

A simple equal-interval binning option is provided for this purpose. Categorical or ordinal (i.e. discrete) variables need no discretisation, and so this dialogue is not shown unless necessary.

Once defined, each node is assigned a set of attribute data. These data are potentially drawn from all other columns in the database. The graphical interface allows attribute data to be drawn from a different data source provided that the sources can be joined using a single variable. More complex joins can be achieved using text commands. Attribute data are used to create the connectivity of the graph. Nodes that share attribute values are connected by edges, which are optionally weighted to reflect the strength of the linkage between the nodes. The application automatically chooses a weighting scheme that is appropriate to the attribute data type; this choice can be overridden by the user if desired.

Once data sources and variables have been defined, the application parses the node attributes to create edges, and builds an XML (in fact GXL, [11]) document that describes the graph. The graph can be either visually explored, or processed with one of many graph-based analytic algorithms.

2.2 Graph visualisation

Graph structures are displayed to the user in an interactive graph browser. The browser is a modified version of the Touchgraph LinkBrowser [12], which is an open-source Java tool for graph layout and interaction. Layout is accomplished using a spring-model method, in which each edge is considered to be a spring, and the node positions are chosen to minimise the global energy of the spring system. Nodes also have mutual repulsion in order to avoid overlap in the layout.

While small graphs can reasonably be displayed in their entirety, large graphs often cannot be displayed in a comprehensible form on limited screen real estate. We solve this problem by allowing large graphs to be explored as a dynamic series of smaller graphs (see below). We discuss alternative approaches, such as hierarchical views with varying level of detail, in the discussion.

Interaction with the user is achieved through three main processes: node selection, neighbourhood adjustment, and edge manipulation. The displayed graph is focused on a selected node. The neighbourhood setting determines how much of the surrounding graph is displayed at any one time. This mechanism allows local regions of a graph to be displayed. Edge manipulation can be done using a slider that sets the weight threshold below which edges are not displayed. It is difficult to judge *a priori* which edges to filter out, as weak edges can obscure the graph structure in some cases but may be crucial in others. A practical solution is to create a graph with relatively high connectivity (many weak links), and then allow the user to remove links in an interactive manner.

The graph layout is done dynamically, and changes smoothly as the user varies the interactive settings. The graph layout uses various visual properties of the nodes and edges to convey information, including colour, shape, label, and mouse-over popup windows. We also allow attributes of the nodes to set the graph layout. This is particularly useful with spatial and temporal data.

An alternative visualisation option is to save the XML document and import it into the user's preferred graph software. This might be appropriate with extremely large graphs, since this visualisation tool does not work well with such graphs.

2.3 Analytical tools

The fields of graph theory and data mining have developed a range of algorithms that assess specific properties of graph structures, including subgraph analyses (e.g. [13–17]), connectivity and flow [8], graph simplification [5, 18], clustering, and outlier detection [19, 20]. Many of the properties assessed by these tools have interpretations in terms of real-world phenomena (e.g. [21–23]) that are not easily assessed from non-graph representations of the data. These provide useful analytical information to complement existing scientific analyses, and also the possibility of building graphs based on analyses of other graphs.

A simple but very useful example is an operator that allows the similarity between two graphs to be calculated. We use an edge-matching metric, equal to the number of edges that appear in both graphs, as a fraction of the total number of unique edges in the two graphs (an edge is considered to appear in both graphs if the same two nodes appear in both graphs, and they are joined by an edge in both graphs). This provides a simple method for exploring the relationships between graphs, and also a mechanism for creating graphs of graphs: given a set of graphs, one can construct another graph $\mathcal{G}$ in which each graph in the set is represented by a node. Using a graph similarity operator, one can calculate the similarity between each pair of graphs in the set, and use this similarity information to create weighted edges between the nodes in $\mathcal{G}$. The visualisation tool allows a node in a graph to be hyperlinked to another graph, so that each node in a graph of graphs can be explored in its own right. We demonstrate these ideas in the Results section, below.

We have chosen not to implement other algorithms at this stage, concentrating instead on the graph construction and visual exploratory aspects. We raise future algorithm development options in the Discussion section, below.

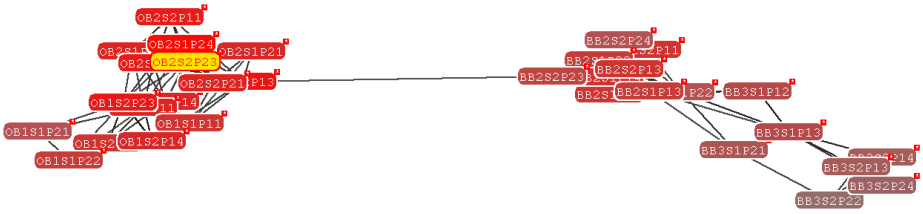

Fig. 1. A graph of Antarctic marine sample sites, linked by their species attribute data. Sites are clearly separated into two clusters on the basis of their species, indicating two distinct types of species assemblage. Node labels are of the form $X\mathrm{B}y\mathrm{S}s\mathrm{P}pr$ and denote the position of the sample in the nested experimental hierarchy. BBy denotes samples from one of two locations in contaminated Brown Bay and OBy denotes uncontaminated O'Brien Bay; s denotes the site number within location; p denotes the plot number within site; and r denotes the core replicate number within plot

3 Results

We use a small Antarctic data set to demonstrate the graph construction and visualisation tools in the context of an exploratory scientific investigation.

Australia has an on-going research programme into the environmental impacts of human occupation in Antarctica (see `http://www.aad.gov.au/default.asp?casid=13955`). A recent component of this programme was an investigation into the relationships between benthic species assemblages and pollution near Australia's Casey station [24]. Marine sediment samples were collected from two sites in Brown Bay, which is adjacent to a disused rubbish tip and is known to have high levels of many contaminants. Samples were collected at approximately 30 m and 150 m from the tip. Control samples were collected from two sites in nearby, uncontaminated O'Brien Bay. Four replicate samples were collected from two plots at each site, giving a total of 32 samples. Sediment samples were collected by divers using plastic corers and analysed for fauna (generally identified to species or genus level) and heavy metal concentrations (Pb, Cd, Zn, As, Cr, Cu, Fe, Ni, Ag, Sn, Sb). These metals are found in man-made products (e.g. batteries and steel alloys) and can be used as indicators of anthropogenic contamination. Details of the experimental methods are given in [24].

This data set has a very simple structure, comprising a total of 14 variables: `site_name`, `species_id`, `species_abundance`, and measured concentrations of the 11 metals listed above. Site latitude and longitude were not recorded but the `site_name` string provides information to the site/plot/replicate level (see Fig. 1 caption). All of the above information appears in one database table. The `species_id` identifier links to the AADC's central biodiversity database, which provides additional information about each species (although we do not use this additional information in the example presented here). Standard practice would

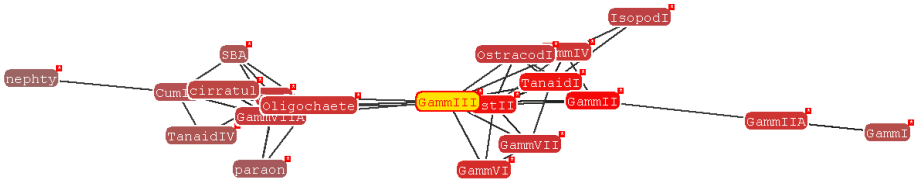

Fig. 2. A graph of Antarctic marine species, linked by their site attribute data. This graph provides complementary information to that shown in Fig. 1, and confirms that two distinct species assemblages exist, with possible outliers *GammIIA*, *GammI*, and *nephty*. Eight other disconnected outlier species are not shown

normally also see a separate table for the sample site details, but in this case there are only a small number of sample sites that are specific to this data set.

Despite the simplicity of the data set, there are a large number of graphs that can be generated. The key questions to be answered during the original investigation related to spatial patterns in species assemblages, and the relationships of any such patterns to contamination (heavy metal concentrations).

Spatial patterns in species assemblages can be explored using sites as nodes, and edges generated on the basis of species attribute data. To create this graph, we needed only to select `site_name` as entities, and `species_id` as attributes in the graphical interface. Both of these variables were recognised by the data dictionary as categorical, and so no discretisation was needed. Furthermore, an edge weighting function suitable for species data was automatically selected. This function is based on the Bray-Curtis dissimilarity, which is commonly used with ecological data:

$$w_{ij} = 1 - \sum_k \frac{|x_{ik} - x_{jk}|}{(x_{ik} + x_{jk})} \ , \tag{1}$$

where w_{ij} denotes the weight of the edge from node i to node j, and x_{ik} denotes the kth attribute of node i.

The resultant graph is shown in Fig. 1. Weak edges have been pruned, leaving a core structure of two distinct clusters of sites: the left-hand cluster corresponds to sites from O'Brien Bay; the right-hand cluster Brown Bay. This strong clustering suggests that the species assemblages of the two bays are distinct. Furthermore, each cluster shows spatial autocorrelation — that is, samples from a given site in a given bay are most similar to other samples from the same site (e.g. BB3 nodes are generally linked to other BB3 nodes). We generated an alternative view of the data by swapping the definitions for entity and attribute, giving a graph of `species_id` nodes with edges calculated on the basis of `site_id` attribute data (Fig. 2). This graph confirms the presence of two broadly distinct species assemblages, but suggests that there are several outlier species that may

Fig. 3. The same graph as Fig. 1, but with edge colouring changes to indicate similarity of chromium between sites. Darker edges are those that are better "explained" by chromium patterns (see text for details). The O'Brien Bay cluster (left) has a strong similarity of chromium values within it, whereas the Brown Bay cluster has dissimilar values both within itself and to the O'Brien Bay cluster. These results, and similar results with other metal variables, suggest that species differences between the two bays may be related to heavy metal concentrations

diverge from this bimodal pattern of spatial distribution.

Having established some patterns in species assemblages, we wish to explore the relationships between these patterns and measured metal contamination. A convenient method for this is through the graph similarity operator. We generated a second graph of sites, using chromium as attribute data (graph not shown), and made an edge-wise weight comparison between the site-species graph and the site-chromium graph. The result is shown in Fig. 3. The structure of this graph is identical to that in Fig. 1, but the colouring of the edges indicates the weight similarity. Darker grey indicates edges that have similar weights in both the site-species and site-chromium graphs. Samples from the uncontaminated O'Brien Bay have similar chromium values (in fact, mostly near zero). More notably, the single edge linking the O'Brien Bay cluster to the Brown Bay cluster is not well explained in terms of chromium, suggesting that the clustering might be related to differences in chromium values. (The general dissimilarity between chromium values in the Brown Bay cluster is an artefact of their high but variable values, which have fallen into different bins during the discretisation process. We discuss the problem of discretising attributes below). Similar results were obtained using the other metal variables, supporting the notion that the benthic species assemblages of these bays is related to heavy metal contamination.

Finally, we use a graph of graphs to explore the similarities between the spatial patterns of the various heavy metals. We generated 11 graphs, one for each metal, using sites as entities and the metal as attribute data. The pairwise similarities between each of these graphs were calculated. Fig. 4 shows the resultant graph, in which each node represents an entire site-metal graph, and the edges indicate the similarities between those graphs. The graph suggests that copper, lead, iron, and tin are distributed similarly, and that their distribution is different to that of nickel, chromium, and the other metals. This was confirmed by inspecting histograms of metal values at each location: values of copper, lead, iron, and tin were higher at one of the Brown Bay locations (the one closest to

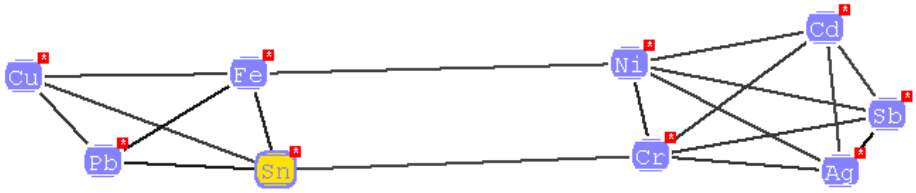

Fig. 4. A graph of graphs. Each node represents an entire subgraph — in this case, a graph of sites linked by a metal attribute. This graph of graphs indicates that the spatial distributions of copper, lead, iron, and tin are similar, and different to those of nickel, chromium, and the other metals

the tip) than the other, whereas the remaining metals showed similar levels at each of the two Brown Bay locations.

4 Discussion

Graphs have been previously been recognised for their value in data mining and exploratory analyses. However, existing software tools for such analyses (that we were aware of) did not meet our requirements. We have outlined a prototype web-based tool that builds graph structures from data contained in databases or files, and presents the graphs for visual exploration or algorithmic analysis.

The construction phase requires the user to define the variables that will be used to form the graph nodes. While there may be certain definitions that are logical or intuitive in the context of a particular database (for example, it is probably intuitive to think of species as nodes when exploring a database of wildlife observations), the nodes can in fact be an arbitrary combination of any of the available variables. This is a powerful avenue for interaction and flexibility, as allows the user to interpret the data from a variety of viewpoints, a key to successful data mining [25].

One of the notable limitations of our current implementation is the requirement that attribute data be discrete. (Edges are only formed between nodes that have an exact match in one or more attributes). Continuous attributes must be discretised, which is both wasteful of information and can lead to different graph structures with different choices of discretisation method. Discretisation is potentially particularly problematic for Antarctic scientific data sets, which tend not only to be relatively small but also sparse. Sparsity will lead to few exact matches in discretised data, and to graphs that may have too few edges to convey useful information. Future development will therefore focus on continuous attribute data.

The visualisation tool that we have discussed is best suited to relatively small graphs. This is generally not an issue with Antarctic scientific data sets, which tend to be of manageable size, but conventional data mining of very large data sets would be problematic. Other visualisation tools, specifically designed for large graphs (e.g. [18, 26, 27]) might be useful for visualising such graphs. FADE [18] and MGV [26] use hierarchical views that can range from global structure of a graph with little local detail, through to local views with full detail.

Many other packages for graph-based data exploration exist, and we have incorporated the features of some of these into our design. The GGobi package [28] has a plugin that allows users to work directly with databases. GGobi also ties into the open-source statistical package R to provide graph algorithms. Zoomgraph [29] takes the same approach. This is one method of providing graph algorithms without the cost of re-implementation. Another is simply to pass the graph to the user, who can then use one of the many freely-available graph software packages (e.g. [30–33]). Yet another approach, which we are currently investigating, is the use of analytical web services. Our development has been done in Coldfusion, which can make use of Java and can also expose any function as a web service. This may allow us to deploy an existing Java graph library such as Jung [33] as a set of web services. This approach would have the advantage that external users could also make use of the algorithms, by passing their GXL files via web service calls.

References

1. T. Washio and H. Motoda, *State of the art graph-based data mining*, SIGKDD Explorations: Newsletter of the ACM Special Interest Group on Knowledge Discovery & Data Mining, 5(1) (2003), pp. 59–68
2. M. Kuramochi, M. Desphande, and G. Karypis, *Mining Scientific Datasets Using Graphs*, in Next Generation Data Mining, H. Kargupta, A. Joshi, K. Sivakumar, and Y. Yesha (eds), MIT Press (2003)
3. R.L. Brieger, *The analysis of social networks*, in Handbook of Data Analysis, M. Hardy and A. Bryman (eds), London, SAGE Publications (2004), pp. 505–526
4. D. Lusseau and M.E.J. Newman, *Identifying the role that individual animals play in their social network*, Biology Letters, in press
5. J.J. Luczkovich, S.P. Borgatti, J.C. Johnson, and M.G. Everett, *Defining and measuring trophic role similarity in food webs using regular equivalence*, Journal of Theoretical Biology, 220(3) (2003), pp. 303–321
6. S.-H. Yook, Z.N. Oltavai, and A.-L. Barabási, *Functional and topological characterization of protein interaction networks*, Proteomics, 4 (2004), pp. 928–942
7. J. Gonzalez, L. B. Holder, and D. J. Cook, *Application of graph-based concept learning to the predictive toxicology domain*, in Proceedings of the Predictive Toxicology Challenge Workshop (2001)
8. L. De Raedt and S. Kramer, *The level wise version space algorithm and its application to molecular fragment finding*, in Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (2001)

9. J. Comiso, *Bootstrap sea ice concentrations for NIMBUS-7 SMMR and DMSP SSM/I*, Boulder, CO, USA: National Snow and Ice Data Center (1999, updated 2002)
10. Global Biodiversity Information Facility, `http://www.gbif.net`
11. A. Winter, B. Kullbach, and V. Riediger, *An overview of the GXL graph exchange language*, Software Visualization, S. Diehl (ed.), Springer-Verlag (2001)
12. A. Shapiro, *Touchgraph*, `http://www.touchgraph.com`.
13. D.J. Cook and L.B. Holder, *Graph-based data mining*, IEEE Intelligent Systems, 15(2) (2000), pp. 32-41
14. M. Kuramochi and G. Karypis, *Finding frequent patterns in a large sparse graph*, in Proceedings of the SIAM International Conference on Data Mining, Florida (2004)
15. C. Cortes, D. Pregibon, and C. Volinsky, *Computational methods for dynamic graphs*, J. Computational and Graphical Statistics, 12 (2003), pp. 950–970
16. A. Inokuchi, T. Washio, and H. Motoda, *Complete mining of frequent patterns from graphs: mining graph data*, Machine Learning, 50 (2003), pp. 321–354
17. X. Yan and J. Han, *CloseGraph: Mining closed frequent graph patterns*, in Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2003)
18. A. Quigley and P. Eades, *FADE: graph drawing, clustering, and visual sbstraction*, Proceedings of the 8th International Symposium on Graph Drawing (2000), pp. 197–210
19. S. Shekhar, C.-T. Lu, p. Zhang, *Detecting graph-based spatial outliers: algorithms and applications (a summary of results)*, in Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2001), pp. 371–376
20. C.C. Noble and D.J. Cook, *Graph-based anomaly detection*, in Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2003), pp. 631–636
21. M. Girvan and M.E.J. Newman, *Community structure in social and biological networks*, Proc. Natl. Acad. Sci. USA 99 (2002), pp. 7821–7826
22. B. Drossel, A.J. McKane, *Modelling food webs*, in Handbook of Graphs and Networks: From the Genome to the Internet, S. Bornholdt and H.G. Schuster (eds), Wiley-VCH, Berlin (2003)
23. J. Moody, *Peer influence groups: identifying dense clusters in large networks*, Social Networks, 23 (2001), pp. 216–283
24. J.S. Stark, M.J. Riddle, I. Snape, R.C. Scouller, *Human impacts in Antarctic marine soft-sediment assemblages: correlations between multivariate biological patterns and environmental variables at Casey Station*, Estuarine, Coastal and Shelf Science, 56 (2003), pp. 717–734
25. J. Neville and D. Jensen, *Supporting relational knowledge discovery: lessons in architecture and algorithm design*, Proceedings of the International Conference on Machine Learning Workshop on Data Mining Lessons Learned (2002)
26. J. Abello and J. Korn, *MGV: a system for visualizing massive multi-digraphs*, IEEE Transactions on Visualization and Computer Graphics, 8 (2002), pp. 21–38
27. G.J. Wills, *NicheWorks — interactive visualization of very large graphs*, J. Computational and Graphical Statistics, 8(2) (1999), pp. 190–212
28. D.F. Swayne, A. Buja, and D. Temple Lang, *Exploratory visual analysis of graphs in GGobi*, in Proceedings of the 3rd International Workshop on Distributed Statistical Computing, Vienna, 2003
29. E. Adar and J.R. Tyler, *Zoomgraph*, `http://www.hpl.hp.com/research/idl/projects/graphs/`

30. V. Batagelj and A. Mrvar, *Pajek - Program for Large Network Analysis*, http://vlado.fmf.uni-lj.si/pub/networks/pajek/
31. S. Borgatti and R. Chase, *UCINET: social network analysis software*, http://www.analytictech.com/ucinet.htm
32. B. Bongiovanni, S. Choplin, J.F. Lalande, M. Syska, and Y. Verhoeven, *Mascotte Optimization project*, http://www-sop.inria.fr/mascotte/mascopt/index.html
33. S. White, J. O'Madadhain, D. Fisher, Y.-B. Boey, *Java Universal Network/Graph Framework*, http://jung.sourceforge.net
34. D. Auber, *Tulip — A Huge Graph Visualization Framework*, http://www.tulip-software.org/
35. A.T. Adai, S.V. Date, S. Wieland, and E.M. Marcotte, *LGL: creating a map of protein function with an algorithm for visualizing very large biological networks*, Journal of Molecular Biology, 340 (1) (2004), pp. 179–190
36. J. Ellson and S. North, *Graphviz - Graph Visualization Software*, http://www.graphviz.org/