LINK: A Software System for Experimentation with Graphs and Hypergraphs

Jonathan Berry*
Elon College, N.C.

March 3, 1997

In the recent past, there have been several efforts to construct software systems for discrete mathematics, and in particular, for the manipulation of graphs. None, however, has resulted in a product with influence comparable to the now-familiar symbolic mathematics packages.

Some notable existing systems for discrete mathematics (more are described in [6]) are *Combinatorica* [13], Steven Skiena's extension package for *Mathematica*, *Netpad* [10] due to Nathaniel Dean and others at Bellcore, *SetPlayer* [1], due to Mark Goldberg and his students at Rensselaer Polytechnic Institute, and Gregory Shannon et. al.'s *GraphLab* [12]. For various reasons, none of these systems has the potential to be a widely-useful environment for both graph manipulation and computation. The authors of these systems recognized the need for a more comprehensive effort, and along with Daniel Gorenstein, the founding director of DIMACS, proposed and obtained funding for *LINK*, which was to be a freely-available and portable software system for discrete mathematics research and education. The first public release was made in July, 1996, and the system is now freely available from the *LINK* web site. [1]:

`http://dimacs.rutgers.edu/Projects/LINK.html`

---

*supported by DIMACS and the *LINK* grant: CCR-9214487. DIMACS is a cooperative project of Rutgers University, Princeton University, AT&T Laboratories, Lucent Technologies/Bell Laboratories Innovations, and Bellcore. DIMACS is an NSF Science and Technology Center, funded under contract STC-91-19999; and also receives support from the New Jersey Commission on Science and Technology.

[1] I took over the direction of the project in June, 1995, and spent a year at DIMACS preparing the public release.

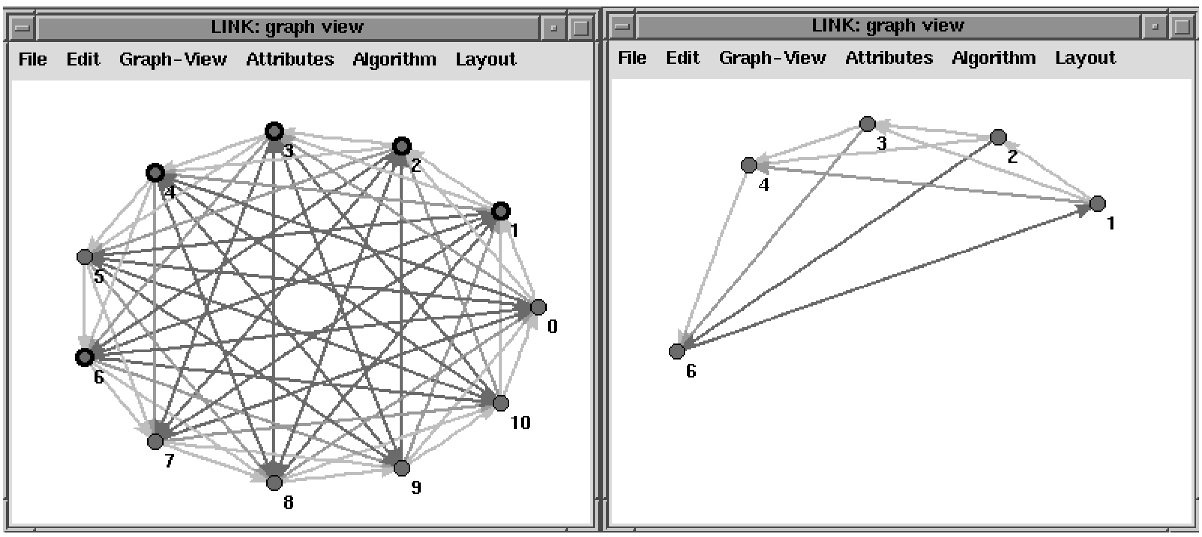

Figure 1: A Latka tournament and an induced subgraph extracted with the *LINK* GUI

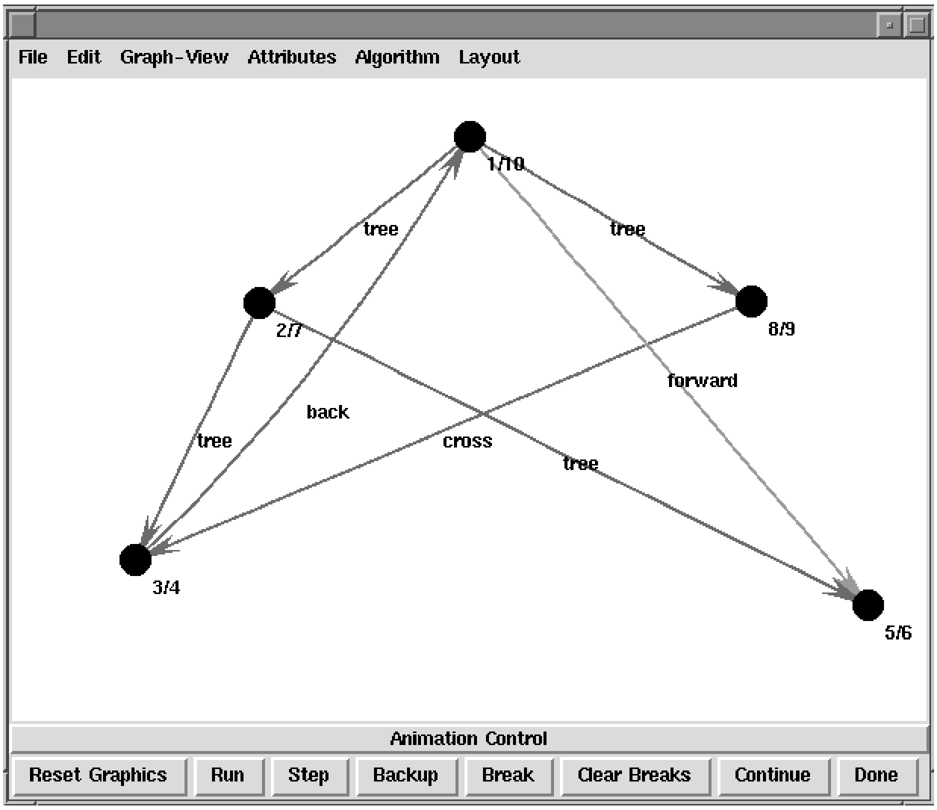

Figure 2: The animation controller processing a depth-first search of a digraph. The vertex labels are modified during the animation to show the search discovery and finishing times, while the edge colors and labels are modified to classify the edges with respect to the search.

LINK 's two most distinctive features are its layered interface and its extendibility. A graphical user interface (GUI) allows users to create and manipulate graphs and hypergraphs, a command interpreter makes small computations straightforward [2], and an extendible C++ library of graph and hypergraph objects and primitive algorithms is provided for the implementation of algorithms and animations. Vertex and edge attributes such as color, size, width, weight, and label can be manipulated, and induced subgraphs can be extracted or collapsed using the GUI. Edges are stored as cubic splines, and can be dragged around using the mouse (as can vertices) to make more attractive layouts.

The system features a 170 page manual which includes dozens of examples of coding both in the command language and with the C++ libraries, and I am committed to improving the system and its documentation and filling out the algorithms library.

1 Example Uses in Research

LINK has already demonstrated its usefulness in research, and its role in two recent projects will be summarized below.

1.1 Latka Tournaments

In the first project, Brenda Latka, while visiting DIMACS from Lafayette College, used *LINK* to assist in her study of infinite antichains of tournaments (complete directed graphs). An antichain of tournaments

[2] Link's graphical and command-line interfaces are written in STk[7], a free software package combining Scheme, a small LISP dialect which is ideal for writing very short programs to process lists of objects, with Tk[11], an easily-extendible graphics system.

is a set for which it is impossible to embed any tournament in the set within any other. Latka is interested in the construction of infinite antichains of tournaments. [4, 8]. Arguments to prove that a given set of tournaments is an antichain typically show that a specific sub-tournament of any given tournament cannot be mapped to any sub-tournament of any other tournament in that class. A crucial element of these arguments is the use of a non-trivial edge attribute: the number of directed 3-cycles in which an edge participates. Sub-tournaments and these edge attributes are easily visualized using *LINK*'s features: the former can be extracted by clicking on vertices and selecting a menu option, while the latter can be computed (and the edges colored) by small programs written in *LINK*'s command language. Figure 1 shows a Latka tournament and an induced sub-tournament extracted by pointing and clicking [3].

Once *LINK* functionality had been used to generate a special class of tournaments defined by Latka, compute the edge attributes, and color their edges accordingly, Latka and I used *LINK* to visualize dozens of these tournaments and variations upon them. Soon patterns began to emerge, leading Latka to conjecture that adding an extra parameter to her initial tournament construction would reveal the first known *infinite* set of infinite antichains of tournaments. The conjecture, still unproven, will be detailed in a future paper.

During this time, I was also serving as a mentor for Chris Burrows, a participant in the the *NSF Research Experience for Undergraduates (REU)* at DIMACS. He used *LINK* to study isomorphisms of certain sub-tournaments by implementing some special invariants described by Latka and using *LINK*'s point and click access to *nauty* [4] [9]. During this process, we observed that one of the Latka tournaments also happened to be a Paley tournament. [5] Burrows then used *LINK* to find two additional Latka-Paley tournaments, then develop the conjecture that no other Latka tournaments are Paley tournaments.

1.2 Market Basket Analysis

In the second project, a set of supermarket data compiled and studied previously at Bell Laboratories was analyzed in a more meaningful way with *LINK*. Considering each type of item in a shopper's "basket" (e.g., bananas, 2% milk, skim milk) to be a vertex, "market-basket analysis" attempts to identify customer buying patterns by examining receipts. Correlations between purchases identified by the analysis can be used, for example, to arrange products more advantageously on the shelves or manipulate prices.

When a shopper purchases a set of items at once, we must represent this grouping somehow. An obvious approach to the problem is to place an edge between each pair of vertices in a basket, thus producing a graph where each purchase is represented by a clique. Given such a graph, however, the original "baskets" cannot by reconstructed. Nathaniel Dean and I used *LINK* to re-model the problem using hypergraphs (graphs where each edge might contain more or fewer than two vertices) where each hyperedge represents a single shopper's basket. In addition to the considerable space savings inherent in this solution, more real-world information is preserved. Furthermore, the STk command language used by *LINK* makes it possible to pose interactive queries such as: "find all purchases in which both a snack food item and a beverage were purchased." A paper describing this work in more detail is available at the *LINK* web site [2].

2 Example Uses in Education

LINK has been designed to be extendible enough to be incorporated into discrete mathematics and computer science curricula at various levels. Its GUI is flexible enough to be used in presentation, and has been used in that capacity by the Leadership Program at DIMACS, which prepares elementary and secondary school instructors to teach discrete mathematics. In addition to aiding in presentations, *LINK* is a tool which computer science students can use to develop their own algorithm animations. Figure 2

[3] See the *LINK* web page for a color image

[4] This is a well-known and practical isomorphism testing tool.

[5] The well-studied Paley tournaments consist of p vertices, where p is prime and congruent to 3 mod 4. Arc (i, j) exists iff $j - i$ is a quadratic residue mod p.

is a snapshot of an animation of depth-first search. The *LINK* programmer's manual contains dozens of basic examples along with coverage of the C++ library routines.

During his eight week REU experience, Chris Burrows was able to learn *LINK* well enough to make fundamental contributions. Specifically, he generalized the depth-first and breadth-first search library routines sufficiently to give the more involved graph algorithms easy access to a rich variety of search-related attributes. He then used the latter to program elegant implementations of topological sort and strongly-connected components. He also developed an animation for breadth-first search and begin work on the like for other fundamental algorithms before he began working on the Latka tournament problems mentioned above.

In addition to its presentation and development features, *LINK*'s computing environment offers the ability to experiment with data sets too complex and/or numerous to draw by hand. The language used is Scheme, a small standardized language which follows a programming paradigm similar to the languages of Mathematica and Maple. Using this language, it is possible with fewer than ten lines of code, for example, to create and display a series of graphs, assign random weights their edges, then find, color, and thicken the edges of a minimum spanning tree in each. Of course, once the graph windows have appeared, the student can manipulate them using both the GUI and the command language.

An on-line tutorial explains simple tasks such as creating and displaying various types of graphs and manipulating vertex and edge weights, the user's manual gives many additional examples, and the *LINK* web site contains an image illustrating the minimum spanning tree example cited above.

3 Design Philosophy

LINK was designed for discrete mathematics research and education, not for processing massive data sets. Issues such as flexibility, extensibility, and portability took clear precedence over speed, and decisions were made accordingly. The interpretive STk graphics system is much slower than a pure Windows or X/Motif application, but for *LINK*'s anticipated users, the benefits of extendible graphics, Scheme command language, and portability far outweigh the cost in speed. A study of the experiences of secondary school teachers using Netpad and Combinatorica especially pointed out the need for flexibility and extensibility in educational applications [5].

Currently, *LINK* runs only on Unix systems (including Linux), but there is no major obstacle preventing a port to Windows, since the STk graphics system has already been ported. *LINK* for Windows might become a reality before the end of 1997. Although a port to the Macintosh is more problematic (since STk has not yet been ported there), portability remains an important priority and concern, and we are committed to making *LINK* more accessible in the future.

The current system is described in the manual available from the web site. The latter also contains links to a preview paper, a paper concerning *LINK* and market basket analysis, and a paper to appear in the 1997 ACM SIGCSE/SIGCUE *Conference on Integrating Technology in Computer Science Education* which provides an extended example: using *LINK* to present a strongly connected components algorithm as a puzzle, then explain it in more detail as students gain intuition. The early development of *LINK* is detailed in [3].

There is still much more work to be done to make *LINK* a widely useful tool. The algorithms library still lacks numerous fundamental components, and there are no course materials designed to make use of *LINK*. DIMACS, Elon College, and RPI are actively seeking funding to prepare an extensive yet modular set of packages of web-based course materials intended to help teach various topics in discrete mathematics and algorithms using *LINK* as a primary tool.

I plan to organize a DIMACS workshop on using *LINK* once it is available for Windows.

References

[1] D. Berque, R. Cecchini, M. Goldberg, and R. Rivenburgh. The setplayer system for symbolic computation on power sets. *Journal of Symbolic Computation*, 14:645–662, 1992.

[2] J. Berry and N. Dean. Market basket analysis using LINK. http://dimacs.rutgers.edu/Projects/LINK.html, 1996.

[3] J. Berry, N. Dean, P. Fasel, M. Goldberg, E. Johnson, J. MacCuish, G. Shannon, and S. Skiena. LINK: A combinatorics and graph theory workbench for applications and research. Technical Report 95-15, Center for Discrete Mathematics and Theoretical Computer Science (see also: http://dimacs.rutgers.edu), Piscataway, NJ, 1995.

[4] G. Cherlin and B. Latka. A decision problem involving tournaments. Technical Report 96-11, Center for Discrete Mathematics and Theoretical Computer Science, 1996.

[5] N. Dean and Y. Liu. Discrete mathematics software for k–12 education. In *Discrete Mathematics in the Schools: Making an Impact, D. Franzblau, F. Roberts, and J. Rosenstein, eds.* American Mathematical Society, DIMACS series, To appear.

[6] N. Dean and G. E. Shannon, editors. *Computational Support for Discrete Mathematics.* American Mathematical Society, 1993. DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Volume 15.

[7] E. Gallesio. The stk reference manual. Technical Report RT 95-31a, I3S CNRS, Université de Nice - Sophia Antipolis, France, 1995.

[8] B. Latka. Finitely constrained classes of homogeneous directed graphs. *The Journal of Symbolic Logic*, 59(1):124–139, March 1994.

[9] B. McKay. Nauty user's guide. Technical Report TR-CS-90-02, Australian National University, 1990.

[10] M. Mevenkamp, N. Dean, and C. Monma. NETPAD user's guide and reference guide, 1990.

[11] J. Ousterhout. *Tcl and the Tk Toolkit.* Addison-Wesley, 1994.

[12] G. Shannon, L. Meeden, and D. Friedman. SchemeGraphs: An object-oriented environment for manipulating graphs, 1990. Software and documentation.

[13] S. Skiena. *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica.* Addison-Wesley, 1990.