Editorial for the Special Issue on Software Support for Teaching Discrete Mathematics

VALERIE J. HARVEY, ROBERT MORRIS UNIVERSITY
SUSAN H. RODGER, DUKE UNIVERSITY
Guest Editors

Discrete mathematics is routinely supported in certain programs beyond mathematics (as discrete mathematics or symbolic logic) and philosophy (as introductory logic or symbolic logic). Teaching discrete mathematics, which provides the basis for formal methods, is traditional in computer science (CS) and software engineering (SE), and is specified by the Accreditation Board for Engineering and Technology (ABET) computing accreditation criteria. In the September 2003 issue of *Communications of the ACM,* K. Devlin [Devlin 2003.], K. Bruce et al. [Bruce et al. 2003], and P. Henderson [Henderson 2003] all found that the form of mathematics of particular value to computer scientists and software engineers is discrete mathematics. The report of the ITICSE 2000 Working Group on Formal Methods in Education cites the role of discrete mathematics in formal methods and asserts that "without formal methods, the [software] engineering is non-existent or suspect" [Almstrum et al. 2000].

This issue of JERIC covers a range of software support for teaching discrete mathematics, from fundamental presentations of logic, specification, and proofs, as proposed by **John Cigas** and **Wen-Jun Hsin** in reporting on the use of puzzles, to the teaching of advanced computer science with the Java program-verification tool reported on by **Timothy Gegg-Harrison**, and the techniques and resources described by **Sutner** for CDM, involving Mathematica and other tools. The need for software support begins with individual tools that can be used to teach, among other topics, logic, sets, and graphs, and proceeds to complex systems. To be useful, the most ambitious projects such as CSDM and program verification require complex systems. And, as Sutner points out (in the section of his article entitled "Afterlife"), for complex environments, it is necessary to provide for subsequent access to the material, for the problems of replicating the libraries, and for the computational behavior of a given system's computational context after a period of time.

For the theory of computing, **Ross** describes a hypertextbook (in progress) which currently covers many topics in automata theory, including finite automata, regular expressions, regular grammars, context-free grammars, and parsing. The hypertextbook is built around Web technologies such as text, sound, pictures, slide shows, video, and active learning models. The idea is to allow students to experiment with concepts as they read the material. There have been various projects of this type, and they obviously

Authors' addresses: Valery J. Harvey, Department of Computer Science, Robert Morris University, Moon Township, PA 15108; email: harvey@rmu.edu. Susan H. Rodger, Department of Computer Science, Duke University, Durham, NC 27708; email: rodger@cs.duke.edu

require a great deal of theoretical and practical effort over a period of time to deliver an even and coherent learning environment [Nezami 1988].

An instructor of discrete mathematics often seeks software resources to teach a number of topics, such as logic, circuits, proofs, sets and set theory, number theory, encryption, graph theory (including trees and networks), automata, grammars and languages, and perhaps knot theory or group theory. This editorial provides some guidance regarding such resources.

Choice of notation and representation can play a role in selecting software, for example with Venn diagrams. In mathematics, shading in a Venn diagram signifies that the shaded space may be nonempty. A useful introduction to formal set notation and Venn diagrams using this approach is provided by a Utah State University project [Cannon et al. 2006]. While it is useful in identifying patterns and for practicing set theory expressions, it lacks the expressiveness, in terms of existential import, of the method for categorical logic formulated by Copi [Copi et al. 2005, Ch. 6 and 7] and others. In their approach, shading means the space is empty, and nonempty spaces in the diagram are indicated by an X. Shin [Shin et al. 2003] provides a comparative and historical treatment of diagrams with a focus on Euler, Venn, Pierce, and Shin diagrams. There are good tools for presenting and practicing with categorical logic in work by Kemerling [2002], DeHaven and Keegan [2004], and Kentler et al. [1998].

An instructor may be uncertain whether a given software product will directly support teaching and student learning. Does the software just present or simulate a topic, or does it give students the opportunity to shape the interactive presentation and supply input, and if so, how? If an instructor wants to cover the roles of graph coloring in information technology, there is Mawata's [2004] excellent graph coloring page, which explains the concepts and suggests good exercises. There are different kinds of graph examples in this work (Platonic graphs, the Petersen graph, and an exploration of the role of subgraphs isomorphic to K_3) and an application example dealing with scheduling. Students can easily experiment to determine what it takes to raise the chromatic number; experience with this module shows that students exploit the information about scheduling. Without the needs of wireless transmission systems being explicitly addressed on the page (they can be described by the instructor or be provided by the appropriate references [Malkevitch 2003; 2006]) the role of graph coloring in cell-phone technology can be presented with ease. Since students can explore graphs freely with this resource and design some nonplanar embeddings, they might question how to handle tunnels in an urban cell-phone transmission environments like San Francisco or New York. For students who want to edit the graphs outside of class or for the instructor giving a classroom presentation, Mawata's software is easy to use. However, as the student draws the graph, it automatically executes the coloring constraint. A different resource, with a much less technically oriented mode of presentation, LaFlamme [2006], covers the four-color map problem, the edge-coloring program, and the dominant set concept, as well as vertex coloring, and (with paint cans!) offers the students the opportunity to choose the colors to see if they applied the coloring constraint correctly. From a presentation standpoint, Mawata's module is directly accessible from the Internet, while the LAF software must be downloaded as a zip file and installed. Experience shows both are useful in teaching; some students relate best to one resource, and some to the other.

How accessible is the software? Is it free or is there a license fee? By using different graph resources, students will encounter different approaches to editing the graphs. There is no single high-quality comprehensive graph-editing environment that is available for free. As a matter of practice, the differences between packages and tools don't seem to bother students much. They adapt easily to the different modes of editing the graphs; e.g., for trees and networks, there are excellent online demonstration resources for spanning trees (MSTs) and network flows [Ikeda 2000; 2000b; Gough 2006].

CSLI's Tarski's World is invaluable for presenting quantification in the treatment of logic [Barwise and Etchemendy 1993]. Initial use of this resource without quantification generally precedes the treatment of quantification. When the students advance to using universal and existential quantifiers and variables, it is useful to encourage them not to label the objects in the "World," the impact of the quantification is then more dramatic and powerful in the learning process.

How high a learning curve will the students face? Is it efficient in terms of time? Does it have a graphic or a syntactic interface? The simplest introduction to logic gates is Calderwood's introduction in the form of automations [Calderwood 2006] Hacker's [2002; 2006] excellent combinatorial circuit exercise is easy for students to use for single-output circuits. Circuit simplification exercises are also supported by Hacker's software, which appeals to students because they can produce professional-looking results and replicate examples and exercises from any textbook. Students can design a circuit by entering the Boolean function or by directly building the circuit model and testing the logic, after which they can see the Boolean function equivalent to the circuit they built. Timing diagrams are available for the hardware-oriented student. Hacker's [2006] Web Release Version 4.0, January 2006 is for practice with logic gates and combinatorial circuits. Select module WinBoolean for more advanced practice, selecting from modules BoolTut and WinCounter.

Is there a resource for material the instructor wants to cover? If an interactive resource on acyclic hypergraphs and GYO or Graham reduction [Ullman 2006or Hasse diagrams and partial order is desired, there doesn't seem to be a good specific resource available. Graph-Magics [Ciubatii 2005] is one of the best graph-editing tools (but requires licensing). It supports general graph-editing, demonstrations and the development of exercises on shortest path, minimal spanning tree, maximum flow, minimum cut, Eulerian and Hamiltonian paths, the Chinese postman problem, maximal subset of independent vertices, maximum clique, graph-coloring, graph center, graph median, and so on. With parameterization, different types of graphs (bipartite, grid, tree, etc.) can be generated quickly; graph representation can also be explored with Graph-Magics. If typing vertices (as in resource allocation graphs for deadlock modeling) is desired, a number of different vertex shapes is available.

General-purpose modeling, not necessarily focused on discrete mathematics, is offered by systems like Mathematica, Maple, and MATLAB/Mathworks. A new version of *Combinatorica* became available with Mathematica 4.2, and additional support is provided by a textbook, called *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica* [Pemmaraju and Skiena 2003].

Ubiquitous tools, like Microsoft Excel or Microsoft Word, are sometimes very useful. There are a number of simple exercises that can be performed using Microsoft Excel. Microsoft Word's color selection facilities permit convenient experimentation with color vectors. Critical path concepts can be explored using Microsoft Project.

In addition to Ross' paper, included in this issue, we now list software for exploring automata theory and formal languages. JFLAP [Rodger and Finley 2006; www.jflap.org] is an extensive package for experimenting with several types of automata, grammars, regular expressions, and parsing methods. The automata one can build with JFLAP include nondeterministic finite automata, pushdown automata, and multitape Turing machines. The parsers in JFLAP include brute-force parsing, LL(1) parsing, and SLR(1) parsing. The grammars we can build with JFLAP include regular grammars, context-free grammars, unrestricted grammars and L-systems. In addition, JFLAP allows us to study the proofs of theorems by studying the conversions of one form of a language to another via an example. For example, we can create a nondeterministic finite automaton (NFA), then step through the creation of the equivalent deterministic finite automaton (DFA), and then step through the creation of a minimal state DFA. JFLAP is a versatile tool and is easy to learn to use.

The following two systems study automata in a nongraphical manner. Sutner [1994] developed a package called *automata* (based on Mathematica) to study finite automata. Finite automata are entered as tuples in Mathematica code. Sutner's package implements several algorithms to generate and manipulate finite automata, such as converting a regular expression to a finite automaton. Hannay [1992; 2002] developed tools for experimenting with several types of automata, including finite automata, pushdown automata, and Turing machines. With each of these tools, the user types in the formal definition of the machine, which is either in the form of a long mathematical tuple or a tabular format.

There are several tools that represent automata as graphs, but each tool only addresses one type of automata. Stallman et al. [1991] developed a graph tool to interactively draw an FA; AMS [Lee 1990] is a tool for simulating FA; Grinder [2002] developed one for experimenting with finite state automata; and McDonald [2002] invented a tool for experimenting with pushdown automata. JCT [Robinson et al. 1999] is for experimenting with both finite automata and Turing machines.

We mention several other software tools for teaching automata. Turing's World [Barwise and Etchemendy 1993]. is a user manual with software for simulating TM's. In Turing's World the user can define and simulate sophisticated Turing machines in a building block manner by copying and pasting parts of machines for other machines, and reusing machines as "submachines" by referring to their names. We can also build finite automata with this tool, but the focus is mainly on Turing machines.

Taylor [1998] has written an automata theory textbook that integrates with software, called Deus Ex Machina [Savoiu 1998], which allows us to simulate seven models of computation, with the most extensive number of example files on Turing and vector machines. The machines that come with the software are pushdown automata, finite automata, 1 and 2-tape Turing machines, linear bounded automata, Markov, register machines, and vector machines.

Logo and Prolog are among the programming environments that are useful in discrete mathematics. Logo can be used to introduce turtle graphics, and thus serve as a "readiness" exercise for Lindenmeyer systems, such as offered by the L-System module in JFLAP. Logo is well known as a practice environment for recursion. Prolog offers practice with sets, lists, recursion, and infix/prefix notation. SQL can be used to explore equivalence relations implicit in uses of the GROUP BY syntax, as Wu [2005] has shown.

A number of resource lists are available, including the Math Forum and Drexel University, the Utah State University National Library of Virtual Manipulatives for Interactive Mathematics [Cannon et al. 2006], formal methods' education resources. An annotated list of applets compiled by Susanna Epp is appended to this article [Epp 2006].

When it comes to the advanced treatment of logic, after a discussion at the "Teaching Formal Methods" symposium in Ghent, Belgium, in 2005, Raymond Boute, University of Ghent, commented, "Perhaps you recall my critique on Maple: sets, and other concepts traditionally categorized under 'discrete math', are apparently added as an afterthought by some programmer. Indeed, sets containing a small number of elements are specified by listing these elements, but the sets are considered dependent on the order in the list. Ideal to cause permanent misconceptions in students! - Thus far, Isabelle/Isar (by Nipkow and Paulson) seems to be the best tool to support logic for discrete math, but has a few idiosyncrasies that need cleaning up."[Boute 2005; ISAR 2006]. In this connection it should be noted that the representation of sets in Prolog is affected by Prolog ordering.

For the future, we can hope for success in Peter Henderson's ITiCSE working group project [ITiCSE 2002].

REFERENCES

ALMSTRUM, V., DEAN, C. N., GOELMAN, D., HILBURN, T. B., AND SMITH, J. 2000. Support for teaching formal methods, Report of the ITiCSE 2000 Working Group on Formal Methods in Education. http://www.cs.utexas.edu/users/csed/FM/work/final-v5-6.pdf.

BARWISE, J. AND ETCHEMENDY, J. 1993. *Turing's World 3.0 for the Macintosh*. Center for the Study of Language and Information (CSLI).

BARWISE, J., ETCHEMENDY, J., BARKER-PLUMMER, D. AND LIU, A. 2006. http://www-csli.stanford.edu/. See also http://www-csli.stanford.edu/hp/Tarski3.html. Center for the Study of Language and Information (CSLI). BOUTE, R. 2005. Personal communication. Dec.

BRUCE, K., DRYSDALE, S., KELEMAN, C., AND TUCKER, A. 2003. Why Math? *Commun. ACM 46*, 9 (Sept.), 40-44**.**

CALDERWOOD, D. 2006. *Introduction to Logic Gates*. College of the Redwoods, Eureka, CA http://isweb.redwoods.cc.ca.us/INSTRUCT/CalderwoodD/diglogic/index.htm.

CANNON, L., HEAL, R., DOWOOD, J., AND EDWARDS, L. 2006. http://nlvm.usu.edu/en/nav/index.html. See also http://nlvm.usu.edu/en/nav/frames_asid_153_g_4_t_1.html?open=instructions. Utah State University, Logan.

CIUBATII, D. 2005. http://www.graph-magics.com. See also http://www.graph-magics.com/gallery.php.

COPI, I. M. AND COHEN, C. 2005. *Introduction to Logic.* 12th (ed.), Pearson Prentice-Hall, Englewood Cliffs, N.J.

DEHAVEN, S. AND KEEGAN, C. 2004. Bluestorm, the logic course. http://www.thelogiccourse.com/bluestorm/. See also Topic 8, Categorical Logic.

DEVLIN, K. 2003. Why universities require computer science students to take math. *Commun. ACM 46*, 9 (Sept.), 37-39.

EPP, S. 2006. Discrete mathematics applets, etc. http://condor.depaul.edu/~sepp/DMappletsEtc.htm.

Formal Methods Educational Resources. http://www.cs.indiana.edu/formal-methods-education/.

GOUGH, G. 2006. Spanning tree demos: Greedy algorthms, Kruskal's algorithm: Tech. Rep., Dept of Computer Science, Univ. of Manchester, Manchester, UK. http://www.cs.man.ac.uk/~graham/cs2022/greedy/.

GRINDER, M. 2002. Animating automata: A cross-platform program for teaching finite automata. In *Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education*. ACM, New York, 63-67.

HACKER, C. AND SITTE, R. 2002. A computer-based interactive teaching software for the tracing of logic levels in a digital circuit. *Global J. Eng. Edu. 6,* 1, 85-90.

HACKER, C. 2006. Demonstration WinLogiLab. Griffith Univ., Gold Coast, Queensland, Australia. http://www.gu.edu.au/school/eng/mmt/WinLLab.html.

HANNAY, D. 1992. Hypercard automata simulation: Finite state, pushdown and Turing machines. *SIGCSE Bull. 24*, 2 (June), 55-58.

HANNAY, D. 2002. Interactive tools for computation theory. SIGCSE Bull. 34, 4, (Dec.), 68-70.

HENDERSON, P. B. 2003. Mathematical reasoning in software engineering education. *Commun. ACM 46*, 9 (Sept.), 45-50.

IKEDA, K. 2000. Prim's algorithm. Tech. Rep., Dept. Information Science and Intelligent Systems, Univ. of Tokushima. http://www-b2.is.tokushima-u.ac.jp/~ikeda/suuri/dijkstra/Prim.shtml.

IKEDA, K. 2000. Max flow, min cut, Ford Fulkerson algorithm. Tech. Rep., Dept. Information Science and Intelligent Systems, Univ. of Tokushima. http://www-b2.is.tokushima u.ac.jp/~ikeda/suuri/maxflow/Maxflow.shtml.

INDIANA UNIVERSITY. 2006. Formal methods. Education resources: http://www.cs.indiana.edu/formal-methods-education/. See also tools page: http://www.cs.indiana.edu/formal-methods-education/Tools/.

ISAR. 2006. Intelligent semi-automated reasoning. http://isabelle.in.tum.de/Isar/.

ITiCSE. 2002. http://www.cs.geneseo.edu/~baldwin/math-thinking/iticse2002wg.html.

KEMERLING, G. 2002. Categorical logic and categorical syllogisms. http://www.philosophypages.com/lg/e08a.html.

KENTLER, C. J., MANORANJAN, V.S., AND CAMPBELL, J.K. 2006. Categorical logic. Showing Venn diagrams: http://www.sci.wsu.edu/math/Lessons/Logic/categoricalLogic.html.

KHALIFA, Y., NOVACKY, G., AND UTHAISOMBUT, P. 2006. Project Discrete Math (PDM). http://oldwww.cs.pitt.edu/PDM//index.html.

LAFLAMME, C. 2006. Chromatic number of a graph, the four color map problem, edge chromatic number of a graph**,** the dominating number of a graph, and other games and tutorials. In *Colorful Mathematics*, LAF Software, Calgary, Alberta. http://www.math.ucalgary.ca/~laf/colorful.html.

LEE, M. C. 1990. An abstract machine simulator. In *Proceedings of the Third International Conference on Computer Assisted Learning*.129-141.

MALKEVITCH, J. 2003. Optimal cell configurations, maximizing call capacity of a frequency band. Tech. Rep. Mathematics and Computing Dept., York College, CUNY, New York. http://www.york.cuny.edu/~malk/tidbits/tidbit-cellphone.html.

MALKEVITCH, J. 2006. Mathematical models for cell phone technology. In *Colorful Mathematics*. Part IV, American Mathematical Society. http://www.ams.org/featurecolumn/archive/colorapp5.html.

MATH FORUM. 2006. http://mathforum.org/library/topics/graph_theory/.

MAWATA, C. P. 2004. Graph theory lessons. Tech. Rep., Dept. of Mathematics, Univ. of Tennessee, Chattanooga.http://www.utc.edu/Faculty/Christopher-mawata/petersen/lesson8.html.

MCDONALD, J. 2002. {\tt Interactive pushdown automata animation}. In *Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education*. ACM, New York, 376-380.

MERLOT. 2006. http://www.merlot.org/Home.po.

NORTH CAROLINA STATE UNIVERSITY. 1991. Tool for graph algorithms. Tech. Rep. TR-91-27, North Carolina State Univ.

NEZAMI, A. R.1998. DiscMath: An intelligent tutoring system for discrete mathematics. Master's dissertation, University of New Brunswick.

PEMMARAJU, S. AND SKIENA, S. 2003. *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*. Cambridge University Press. http://www.wolfram.com/news/skiena.html.

ROBINSON, M., HAMSHAR, J., NOVILLO, J., AND DUCHOWSKI, A. 1999. A Java-based tool for reasoning about models of computation through simulating finite automata and Turing machines. In *Proceedings of the 30th SIGCSE Technical Symposium on Computer Science Education*. ACM, New York, 105-109.

RODGER, S. H. 2005. www.jflap.org.

RODGER, S. H. AND FINLEY, T. W. 2006. *JFLAP: An Interactive Formal Languages and Automata Package*. Jones and Barlett Publishers, Sudbury, MA

ROSEN. 2006. Rosen's resource list. http://www.mhhe.com/math/advmath/rosen/r5/student/demos/index.html.

SAVOIU, N. 1998. Software deus ex machine. http://www.ics.uci.edu/$\sim$savoiu/dem/.

SHIN, S.-J. AND LEMON, O. 2003. Diagrams. In *The Stanford Encyclopedia of Philosophy* (Winter ed.), E. N. Zalta (ed.). http://plato.stanford.edu/entries/diagrams/.

STALLMAN, M., CLEAVELAND, R., AND HEBBAR, P. 1991. GDR: A visualization tool for Graph Algorithms. http://www4.ncsu.edu/~mfms/GDR-1.0.

SUTNER, K. 1992. Implementing finite state machines. In Proceedings of the *DIMACS 1992 Workshop on Computational Support for Discrete Mathematics*. American Mathematical Society, Providence, R.I.

TAYLOR, R. 1998. *Models of Computation and Formal Languages*, Oxford University Press, Oxford, UK.

ULLMAN, J. D. 2006. Hypergraphs. Stanford University. http://www-db.stanford.edu/~ullman/cs345notes/slides01-3.pdf.

WU, P. Y. 2005. Aggregates in databases. http://home.earthlink.net/~inforef/i3450sqlaggr.htm.

APPENDIX: DISCRETE MATHEMATICS APPLETS ETC.

Prepared by Susanna S. Epp, Department of Mathematical Sciences, DePaul University, Chicago, IL 60614

Note: These links were live when last checked. If one appears to be dead, it may be possible to find the item by using a keyword search to discover its new location. Please send comments, corrections, and suggestions for new items to sepp@condor.depaul.edu.

Interactive Mathematics Activities:
www.cut-the-knot.org/Curriculum/index.shtml
<http://www.cut-the-knot.org/Curriculum/index.shtml>
This website has links to over 450 interactive mathematics activities arranged by mathematical topic. The activities listed under combinatorics, algebra, logic, fractals, combinatorial games, probability, social science, and puzzles & games are almost entirely discrete mathematical, and many of those listed under arithmetic, miscellaneous demonstrations, fallacies, math magic, and mathematical droodles also involve discrete mathematics.

David Eck Labs:
http://math.hws.edu/TMCM/java/index.html
David Eck writes: "On this page you'll find a set of lab worksheets and Java applets that are meant to help people learn about computer science. They were written for use with my introductory computer science textbook [The Most Complex Machine: A Survey of Computers and Computing], but they can also be used independently of that text. The labs and applets are free for personal use. In addition, the applets can be freely used for non-commercial purposes, including courses that do not use my textbook. I ask that teachers use the labs as an official part of a course only if they adopt my textbook for that course (but I will consider giving permission for other uses)."

Doug Ensley Interactive Flash Exercises:
www.ship.edu/~deensl/DiscreteMath/flash/
<http://www.ship.edu/~deensl/DiscreteMath/flash/>
This website contains 75 exercises, which were written to accompany /Introduction to Discrete Mathematics: Mathematical Reasoning with Puzzles, Patterns and Games/, by Doug Ensley and Winston Crawley. Topics include puzzles, patterns, and mathematical thinking (including sentential and predicate logic); number puzzles and sequences; a primer of mathematical writing (proof and disproof); sets and Boolean algebra; functions and relations; combinatorics; probability; and graphs and trees.

***State University of California Reference Notes*:**
www.math.csusb.edu/notes
<http://www.math.csusb.edu/notes>/
This website contains links to notes on the following topics: set theory, symbolic logic, methods of proof, basic set theory proofs, functions, relations, binary operations, and groups. A number of the notes pages contain applets.

***The Shodor Education Foundation Mathematical Activities*:**
www.shodor.org/interactivate/lessons/index.html
<http://www.shodor.org/interactivate/lessons/index.html>
This website contains interactive activities on Number and Operation, Geometry and Measurement, Function and Algebra, and Probability and Data Analysis.

***Derive Labs*:**
www.brookscole.com/cgi-wadsworth/course_products_wp.pl?fid=M20b&product_isbn_issn=0534359450&discipline_number=1
<http://www.brookscole.com/cgi-wadsworth/course_products_wp.pl?fid=M20b&product_isbn_issn=0534359450&discipline_number=1>

These labs were developed by Nancy Hagelgans at Ursinus College to accompany parts of /Discrete Mathematics with Applications./

Martindale's

*** 'The Reference Desk'*:**
www.martindaleconter.com
<http://www.martindaleconter.com/>
This is a huge reference website. The following are some of the subsections that contain java applets for discrete mathematics:
(www.martindalecenter.com/Calculators2_6_AD.html#COMP-DISCRETE
<http://www.martindalecenter.com/Calculators2_6_AD.html#COMP-DISCRETE>),
encryption/cryptography
(www.martindalecenter.com/Calculators2_6_EH.html#COMP-ENC
<http://www.martindalecenter.com/Calculators2_6_EH.html#COMP-ENC>),
Fibonacci numbers
(www.martindalecenter.com/Calculators2_6_EH.html#COMP-FIBON
<http://www.martindalecenter.com/Calculators2_6_EH.html#COMP-FIBON>),
graph theory (www.martindalecenter.com/Calculators2_6_EH.html#COMP-GR-TH
<http://www.martindalecenter.com/Calculators2_6_EH.html#COMP-GR-TH>),
logic), and number theory
(www.martindalecenter.com/Calculators2_6_NZ.html#COMP-NT
<http://www.martindalecenter.com/Calculators2_6_NZ.html#COMP-NT>).

***Basic Set Definitions*:**
www.geocities.com/basicmathsets/
<http://www.geocities.com/basicmathsets/>
This website, by Martin Selditch, contains interactive tutorials and exercises about definitions of set, union, intersection, and subset.

Venn Diagrams

· ***Venn Diagram Definition*:**
http://www.shodor.org/interactivate/activities/vdiagram/indexflash.html
This applet, from the Shodor Educational Foundation, asks a user to place an item in the correct region of a Venn diagram. For instance, if the three regions represent even integers, palindromic integers, and perfect squares, respectively, the number 121 should be placed in the intersection of the regions representing palindromic integers and perfect squares but not in the intersection of all three regions. Feedback is provided about the correctness of the user's answer.

· ***Venn Diagram Regions*:**
http://www.math.csusb.edu/notes/quizzes/venn1/venn1.html
This applet, from Dan Rinne, California State University, San Bernardino, displays Venn diagram for sets /A/, /B/, and /C/ and contains exercises testing a user's ability to identify various regions in the diagram, such as /A/ Ç (/B^c / È /C/).

· ***Probability Calculator*:**
http://www.stat.sc.edu/~west/applets/Venn.html

This applet allows a user to move Venn diagram representations for sets /A/ and /B/ inside a universe /U/, and it calculates the probabilities of various related sets, such as /A /È/ /B and (/A/ Ç /B/)/^c /,assuming that the probability of /A/ is 0.075 and the probability of /B/ is 0.2.

· ***Survey of Venn Diagrams*:**
http://www.combinatorics.org/Surveys/ds5/VennEJC.html
This webpage does not contain applets, but it provides excellent information, pictures, and information about Venn diagrams involving four or more sets.

***ASL Committee on Logic Education*:**
http://www.phil.ucalgary.ca/asl-cle
If you scroll down the page, you will see a list of sites for freshman logic courseware and another list of sites with some resources and/or courses using computers in logic or with logic.

Complete, Interactive, Free, Online Textbooks for Introductory Symbolic Logic:

§ ***Logic Café*:**
www.oakland.edu/phil/cafe/intro.htm
<http://www.oakland.edu/phil/cafe/intro.htm>
The Logic Café was developed by John F. Halpin, Oakland University. Its chapters and tutorials include Basic Concepts, Sentence Logic, Truth Tables, SL Symbolizations, SL Derivations, Predicate Logic, PL Semantics and Symbolizations, PL Derivations, Predicate Logic with Identity. The Logic Café contains tutorials, reference material, and many exercises with immediate feedback about the correctness of students' answers.

****blogic**:**
* / //www.nyu.edu/classes/velleman/blogic/Logic
<http://www.nyu.edu/classes/velleman/blogic/Logic>////blogic/
<http://www.nyu.edu/classes/velleman/blogic/Logic/>was developed by J. David Velleman, New York University. Its chapters and tutorials include Boolean connectives in online search-strings, logic circuits, propositional logic with truth-tables, modal logic and counterfactuals with possible-worlds, diagrams, the logic of frequencies and probabilities, and the language of quantification. blogic**contains tutorials, reference material, and many exercises with immediate feedback about the correctness of students' answers.

Truth Tables

· ***Truth Table Constructor*:**
* *www.brian-borowski.com/Truth/
<http://www.brian-borowski.com/Truth/>
At this site, by Brian Borowski, one can Input a logical expression see the associated truth table. The applet can be set so that it shows the truth values of the complete expression and of all sub-expressions.

· ***Truth Table Practice*:**
www.math.csusb.edu/notes/quizzes/tablequiz/tablepractice.html
<http://www.math.csusb.edu/notes/quizzes/tablequiz/tablepractice.html>

This applet, from California State University, San Bernardino, presents a user with a blank truth table for parts of a logical expression in /P/ and /Q/, asks the user to fill in truth values for the various columns, and indicates whether or not the user's answer is correct. The user may then click a button to obtain a new problem.

***Tarski's* * World Applets*:**
The use of Tarski's World for teaching logic was developed by Jon Barwise and John Etchemendy. (Seewww-csli.stanford.edu/hp/CVandNR.pdf
<www-csli.stanford.edu/hp/CVandNR.pdf> and www-csli.stanford.edu/hp/#LOFOL
<http://www-csli.stanford.edu/hp/#LOFOL>.) Web addresses for Java applets implementing two-dimensional versions of Tarski's World are given below.

*** *Tilominos**:**
* www.scs.ryerson.ca/~quigley/Theses/Completed/2002/Moxam/Tilomino/test/
<http://www.scs.ryerson.ca/~quigley/Theses/Completed/2002/Moxam/Tilomino/test/>
or www.scs.ryerson.ca/~danziger/mth599/Tilomino/
<http://www.scs.ryerson.ca/~danziger/mth599/Tilomino/>
Version 1 of this applet was created by Neil Deakin (1997) under the supervision of Sophie Quigley at Ryerson Polytechnic University, and version 2 was created by Neil Deakin, Sophie Quigley, and Wesley Moxam (2002).

*** *Tarski's** World:**
*www.bu.edu/linguistics/UG/course/lx502/local/tarski.html
<http://www.bu.edu/linguistics/UG/course/lx502/local/tarski.html>
This applet was created by professor Robert Stärk of the Swiss Federal Institute of Technology (ETH).

Logic Circuits

*** *xLogicCircuits**:**
*http://math.hws.edu/TMCM/java/xLogicCircuits/index.html
This applet, part of David Eck's collection (see above), "lets you build simulated logic circuits from AND, OR, and NOT gates and see how they behave. This is not a serious circuit design tool. It's an educational tool for learning the basics about logic circuits."

*** *circuitbuilder**:**
*http://www.jhu.edu/~virtlab/logic/log_cir.htm
This webpage, from the Johns Hopkins University Virtual Laboratory website, includes a description of the operation of digital logic circuits and a link to an applet that allows a user to construct circuits and see the associated input-output tables.

***Prolog*:**
http://www.calvin.edu/~rpruim/courses/m156/F99/prolog/
These materials, developed by Randall Pruim, Calvin College, "were used in conjunction with the predicate logic part of a discrete math course." They include "a prolog interpreter that can be run over the WWW, an example from Monty Python, examples, exercises, ..."

Logic Proof Developer/Checkers

****Logic Daemon*:**
http://logic.tamu.edu/
<http://logic.tamu.edu/quickhelp.html>
"The Daemon Proof Checker checks [logic] proofs and can provide hints for students attempting to construct proofs in a natural deduction system for sentential (propositional) and first-order predicate (quantifier) logic. The Quizmaster provides a variety of exercises, from questions about basic concepts such as validity, to wff construction and translation, to proofs, truth tables, and countermodels." The system and exercises are based on /Logic Primer/ <http://logic.tamu.edu/Primer/> (MIT Press, 2000) and were developed by Colin Allen and Chris Menzel.

****Oliver*:**
www.csis.pace.edu/~scharff/SOFTWARE/OLIVER/oliver.html
<http://www.csis.pace.edu/~scharff/SOFTWARE/OLIVER/oliver.html>
Oliver stands for OnLine Inference and VERification System for Propositional Logic proofs. Oliver provides a web-based interface for learning propositional logic proofs, and accepts any valid direct proof. Oliver provides instant feedback to whether each step is correct or not and encourages experimentation. Users may login into the system with any login name and password. (It seems necessary, however, to log in separately for each problem.) Andy Wildenberg (Cornell College) and Christelle Scharff (Pace University) wrote this article about the system: http://fie.engrng.pitt.edu/fie2002/papers/1613.pdf.

Proof Developer Assistants

· ***Proof Designer: *www.cs.amherst.edu/~djv/pd/pd.html**
<http://www.cs.amherst.edu/~djv/pd/pd.html>
http://www.cs.amherst.edu/~djv/pd/pd.html
This applet was written by Dan Velleman of Amherst College "to write outlines of proofs in elementary set theory, under the guidance of the user." Proof Designer's approach to proof-writing is similar to the approach used in Velleman's book /How to Prove it <http://us.cambridge.org/titles/catalogue.asp?isbn=0521446635>/.

· ***Flash Applications: *www.ship.edu/~deensl/DiscreteMath/flash/**
<http://www.ship.edu/~deensl/DiscreteMath/flash/>
Several of the Flash applications on Doug Ensley's website (see above) give assistance for development of proofs and counterexamples.

· ***Find the Flaw*: www.math.toronto.edu/mathnet/falseProofs/**
<http://www.math.toronto.edu/mathnet/falseProofs/>
This page, from the University of Toronto Mathematics Network, contains examples of false proofs whose flaw can be discovered through interactive activity.

Prime Numbers

*** *Sieve of Eratosthenes:***
www.faust.fr.bw.schule.de/mhb/eratclass.htm
<http://www.faust.fr.bw.schule.de/mhb/eratclass.htm>:

This is a demonstration applet from Dr. Hans-Bernhard Meyer, Faust-Gymnasium, Germany. A table of the integers from 1 to 400 is shown. The user clicks on successive prime numbers in sequence to see all the multiples of the prime successively crossed out.

****The Prime Pages: ***
www.utm.edu/research/primes/
<http://www.utm.edu/research/primes/>
This website, from Chris Caldwell, the University of Tennessee at Martin, contains a great deal of information about prime numbers, many links, and some applets.

****Notes and Literature on Pri**me Numbers*:**
www.math.utah.edu/~alfeld/math/prime.html
<http://www.math.utah.edu/~alfeld/math/prime.html>
This website, created by Peter Alfeld of the University of Utah, contains a variety of information about prime numbers and links to several applets. One, The Prime Machine, allows the user to obtain data to explore simultaneously the prime number theorem, the distribution of prime twins, and the Goldbach conjecture.

Euclidean Algorithm:
http://aleph0.clarku.edu/~djoyce/java/elements/bookVII/propVII2.html
This webpage, from David Joyce's website on the history of mathematics at Clark University, contains the original Euclidean algorithm: Euclid's Proposition 2 from Book VII of the /Elements/. It includes a Java applet that represents numbers by line segments and allows the user to experiment with segments of different lengths.

***Visible Euclidean Algorithm and Fast Modular Exponentiation *:**
www.math.umn.edu/~garrett/js/gcd.html
<http://www.math.umn.edu/~garrett/js/gcd.html>
and www.math.umn.edu/~garrett/crypto/a01/FastPow.html
<http://www.math.umn.edu/~garrett/crypto/a01/FastPow.html>
These applets, by Paul Garrett at the University of Minnesota, find the greatest common divisor of two numbers using the Euclidean algorithm (displaying the intermediate steps) and compute /b^e / mod /n/ for large values of /b/, /e/, and /n/. (Other of Garrett's applets can be viewed at
www.math.umn.edu/~garrett/px/index.shtml
<http://www.math.umn.edu/~garrett/px/index.shtml>.)

Chinese Remainder Theorem
· www.math.mtu.edu/mathlab/COURSES/holt/dnt/chinese1.html
<http://www.math.mtu.edu/mathlab/COURSES/holt/dnt/chinese1.html>
This website, from Michigan Tech University, gives a statement of the Chinese remainder theorem with a crucial condition missing. It includes an applet that takes /a/_1, /a/_2 , /m/_1 , and /m/_2 as input, and finds all values of /x/ (mod /m/_1 /m/_2) that satisfy the pair of congruences /x/ ===/a/_1 (mod /m/_1) and /x/ ===/a/_2 (mod /m/_2). Students explore the applet to figure out the missing condition in the theorem statement.

· http://banach.millersville.edu/~bob/math478/ChineseRemainder.html
This website, from J. Robert Buchanan, Millersville University, allows a user to input as many as 10 simultaneous congruences of the form /x/===/a/ (mod /m/) and obtain a solution.

***Caesar Cipher*:**
http://www.shodor.org/interactivate/activities/caesar/index.html
In this applet letters of the alphabet are coded as: A ® 0, B ® 1, . . . , Z ® 25, "and then the numbers are changed via an affine (linear) transformation to new, coded numbers. The coding function has the form: Y = A * X + B, where X is the uncoded number, A and B are constants (known to allies, but unknown to enemies) and Y is the calculated, coded number. The arithmetic is done mod 26 to ensure that we get numbers back that can be translated back to letters before sending the coded message." The applet allows a user to enter text and specify the values of A and B. It then displays the converted text. The applet contains a link to Caesar Cipher II, in which the user enters a text, which is converted automatically using a Caesar cipher. The user is then asked to determine A and B. Finally, there is a link to Caesar Cipher III, in which only a converted text is given and the user is challenged to decode it.

***RSA Cryptography*:**
http://cisnet.baruch.cuny.edu/holowczak/classes/9444/rsademo/
This Java applet demonstrates the basics of RSA Public Key cryptography. It was written by Richard Holowczak a faculty member in the CIS Department at Baruch College, CUNY.

***The Monty Hall Problem*:**
www.stat.sc.edu/~west/javahtml/LetsMakeaDeal.html
<http://www.stat.sc.edu/~west/javahtml/LetsMakeaDeal.html>
This applet, by R. Webster West, Dept. of Statistics, Univ. of South Carolina, lets the user play multiple times, keeping track of the number of times the user won by switching and the number of times the user won by not switching. To get a good sense for the probabilities, one must play the game quite a few times.

***Pascal's Triangle*:**
http://ptri1.tripod.com <http://ptri1.tripod.com/>
This page contains many illustrations and interactive activities for Pascal's triangle.

***Catalan Numbers*:**
http://mathforum.org/advanced/robertd/catalan.html
This page, by Robert M. Dickau, is not an applet, but it contains a set of very nice pictures illustrating the many ways Catalan numbers occur in various situations. Other mathematical figures by the same author are at
http://mathforum.org/advanced/robertd/index.html.

***Tromino** Puzzle Information and Applet*:**
www.amherst.edu/~nstarr/puzzle.html
<http://www.amherst.edu/~nstarr/puzzle.html>

Norton Starr of Amherst College developed this applet to make it convenient for people to experiment with the problem of tiling a deficient checkerboard with L-trominoes (L-shaped dominoes).

Tower** of Hanoi** and Related Items

****Do-It-Yourself Version*:**
www.mazeworks.com/hanoi/
<http://www.mazeworks.com/hanoi/>
Bob Kirkland and David Herzog created an applet that allows the user to move disks one by one from pole to pole in a Tower of Hanoi puzzle.

****Tower** of Hanoi and the Reve's Puzzle*:**
www.hcc.hawaii.edu/~sam/
<http://www.hcc.hawaii.edu/~sam/>
Sam Rhoads at Honolulu Community College has applets that demonstrate the Tower of Hanoi Puzzle and the Reve's Puzzle (a version of the tower of Hanoi that uses four poles). The user can specify the number of disks, the movement of the disks is shown (rather rapidly), and the number of moves is displayed.

***Missionaries and Cannibals*:**
www.plastelina.net/games/game2.html
<http://www.plastelina.net/games/game2.html>
and www.oswego.edu/~lwu/ai1/a6/ssmc_tree.png
<http://www.oswego.edu/~lwu/ai1/a6/ssmc_tree.png>*
*The first website contains an applet for trying to solve the missionaries and cannibals puzzle. The second website, from Craig Graci’s Artificial Intelligence course at the State University of New York at Oswego, contains a graph that shows the solution.

***The Wolf, the Goat, and the Cabbage*:**
http://perso.wanadoo.fr/jeux.lulu/html/anglais/loupChe/loupChe1.htm
This is an applet for trying to help* *a farmer get a wolf, a goat, and
a cabbage from one side of a river to the other. It is from the French
website “Lulu’s games.” The Leaping sheep puzzle applet from this
website is also good
(http://perso.wanadoo.fr/jeux.lulu/html/anglais/lomouton/mouton.htm#
<http://perso.wanadoo.fr/jeux.lulu/html/anglais/lomouton/mouton.htm>).
Both puzzles may be solved by drawing a graph and finding a path from
the initial state to the desired final state.

***Transitive Closure: ***
www.cs.nmsu.edu/~ipivkina/TransClosure/index.html
<http://www.cs.nmsu.edu/~ipivkina/TransClosure/index.html>
This applet is from Inna Pivkina, New Mexico State University. It calculates the transitive closure of a relation represented as an adjacency matrix.

***Sortin**g Algorithms*:**

- ***xSortLab**:***

math.hws.edu/TMCM/java/xSortLab/index.html
<http://math.hws.edu/TMCM/java/xSortLab/index.html>
This applet, part of David Eck’s collection (see above), illustrates the actions of BubbleSort, SelectionSort, InsertionSort, MergeSort, and QuickSort. By choosing “Visual Sort,” the user can step through the execution of the algorithms for an input of 16 elements. By choosing “Timed Sort,” the user can compare the run times of these algorithms for various input sizes.

· ***Sam Rhoads: ***
www.hcc.hawaii.edu/~sam/
<http://www.hcc.hawaii.edu/~sam/>
This applet counts the number of comparisons and the number of assignment statements used to execute various sorting algorithms for inputs of various sizes up to n = 100. It also displays the individual steps of the algorithms, allowing the user to choose the display at three different speeds (all of which are fairly rapid).

***Java Applets for Data Structures and Algorithms*:**
www.cosc.canterbury.ac.nz/people/mukundan/dsal/appldsal.html
<http://www.cosc.canterbury.ac.nz/people/mukundan/dsal/appldsal.html>*
*This website, by R. Mukundan, University of Canterbury, Christchurch, New Zealand, contains a variety of applets. Recursion: Tower of Hanoi, N-Queens Problem. Sorting Algorithms: Bubble Sort, Insertion Sort, Selection Sort, Merge Sort, Quick Sort. Searching Algorithms: Linear Search, Binary Search. Lists, Stacks and Queues: The Stack, The Queue, The Cyclic Queue, The Linked List, The Stack as a Linked List, The Queue as a Linked List, Trees and Graphs: Binary Tree Traversal, Binary Search Tree, Graph Traversal.

***Graph Theory Animations*:**
http://delab.csd.auth.gr/c_graph/anim.htm
This website, from the Data Engineering Laboratory of Aristotle University in Greece, contains animations for Dijkstra's algorithm, Kruskal’s algorithm, Prim’s algorithm, Floyd’s algorithm, breadth-first search, depth-first search, and topological sort. A few of the links are not operational, but most work fine.

***Backtracking*:**
www.faust.fr.bw.schule.de/mhb/backtrack/backtren.htm
<http://www.faust.fr.bw.schule.de/mhb/backtrack/backtren.htm>
This applet, from Dr. Hans-Bernhard Meyer, Faust-Gymnasium, Germany, shows users a puzzle for which the solution requires backtracking. They may either try to solve the puzzle on their own or click on “automatically” to see a visual representation of the solution.

***Finite-State Automata*:**
www.cs.montana.edu/webworks/projects/fsa-old/fsa.html
<http://www.cs.montana.edu/webworks/projects/fsa-old/fsa.html>
This Java applet allows a user to view what happens when strings areinput to various automata. The applet also makes it possible to modify existing automata and create new ones. It was developed by the Webworks Team

<http://www.cs.montana.edu/webworks/webworks-home/team.html> at Montana State University.

Turing Machine Simulator:
http://ironphoenix.org/tril/tm/
This Java applet simulates the action of a Turing Machine. It was developed by Suzanne Britton, a Canadian programmer-analyst, and includes brief descriptions about Turing machines and instructions for using the applet.

***Earliest Known Uses of Symbols and Terms*:**
http://hometown.aol.com/jeff570/mathsym.html and
http://hometown.aol.com/jeff570/mathword.html
These two websites are the work of Jeff Miller, Gulf High School, New Port Richey, Florida. They do not contain applets, but they do contain the earliest uses Miller has discovered of a very large number of mathematical symbols and mathematical terms.

***Online Encyclopedia of Integer Sequences*:**
www.research.att.com/~njas/sequences/
<http://www.research.att.com/~njas/sequences/>
This is another important webpage that does not contain applets. ”Since the mid-1960's Neil Sloane has been collecting integer sequences from every possible source. His goal is to have all interesting number sequences in the table. At the present time the table contains over 100000 sequences…. The main table is a collection of number sequences arranged in lexicographic order. The entry for each sequence gives: the beginning of the sequence; its name or description; any references or links; any formulae; cross-references to other sequences; the name of the person who submitted it, etc.”

Webpage last revised: 24 March 2006