Discrete Mathematics & Combinatorics

Alan Tucker, SUNY Stony Brook (chair)
Doug Baldwin, SIGCE/Discrete and SUNY Geneseo
Karen Collins, Wesleyan University & SIAM
Susanna Epp, DePaul University
Diana Thomas, Montclair State University

This report is divided in two parts, the first and major part about Discrete Mathematics courses, typically a lower-division offering, and the second part about Combinatorics courses, typically an upper-division offering.

I. Discrete Mathematics Courses

Audience for the course: Students majoring in computer science form the primary audience for Discrete Mathematics courses; however, at some institutions the course also serves as a transition to abstract mathematics for mathematics and mathematics education majors and is sometimes taken by students majoring in other technical subjects. A major challenge for the course, addressed below, is how to accommodate both computer science majors and mathematics majors, given that the mathematical backgrounds of these two groups may differ. For information about separate transition courses that use discrete mathematics topics, see this Guide’s report on transition courses.

History of the course. For computer science students, discrete mathematics is intended to provide a mathematical foundation for further study of some or all of the following areas: programming languages, computer organization, database theory, data structures, analysis of algorithms, computability, formal methods in software engineering, and network security including cryptography. This course originated in the first ACM undergraduate curriculum recommendations in 1968. By 1986 there was a movement to broaden the audience for the course, and an MAA report from a committee consisting of mathematicians and computer scientists recommended that “discrete mathematics should be part of the first two years of the standard mathematics curriculum at all colleges and universities.”[1] The joint ACM/IEEE Computing Curricula 2001 report provides the latest description of the discrete math knowledge that all computer science students should have.[2] An updated report is expected to be published late in 2013. In 2007 the ACM Special Interest Group on Computer Science Education (SIGCSE) published the results of a three-year effort to develop syllabi for a Discrete Mathematics course that would achieve the goals of the 2001 curriculum. The result was two syllabi, each for a one-semester course.[3] Both syllabi suggest devoting nine weeks to basic logic, proof techniques, sets, functions, and relations, and an additional two weeks for the basics of counting and recurrence relations. The syllabi diverge for the final two weeks, where graphs and trees are included in one syllabus and discrete probability in the other. Suggestions for computer science applications are given throughout both syllabi.

Relationship with colleagues in computer science. At institutions where mathematics and computer science are located in separate departments, there has been considerable movement of the introductory Discrete Mathematics course out of mathematics and into computer science. Thus mathematics departments wishing to develop or retain Discrete Mathematics as a service for computer science are urged to consult closely with faculty in computer science in designing or updating their offerings. This is especially helpful in determining which computer science applications to include in the Discrete Mathematics course and which applications the computer science department is happy to take charge of itself. It is also desirable for mathematics departments to ensure that course instructors have enough knowledge of computer science to be able to indicate relationships between the course topics and future computer science topics. As is the case for most mathematics service courses, the lengthy list of topics desired by a client department can create tension between including all the material and ensuring that students attain the proficiency desired by the department being served. Another reason for communication between mathematics and computer science faculty is to alleviate this tension.

Prerequisites. The preparation of students for Discrete Mathematics courses varies significantly among institutions. Prerequisites are typically at the level of precalculus but vary from two semesters of calculus to college algebra (or the equivalent at the high school level), and some students' algebra skills are rather weak. Thus decisions about the appropriate balance between breadth, depth, and what can be expected from a one-semester course will vary from one institution to another.

Cognitive learning goals. Discrete Mathematics courses have the following primary cognitive learning goals. These goals are developed in almost all the course topics.

Familiarity with the role logical reasoning plays in mathematics: Students should become aware of the basic principles of logical reasoning. They should come to understand the role of precise definitions for deciding whether a given object satisfies a given mathematical term, appreciate the importance of knowing both examples and non-examples of mathematical terms in order to explore statements about them, and know how to use definitions effectively in proofs. They should be familiar with the basic structures of direct proofs, indirect proofs, and proofs by mathematical induction and be able to apply them in a variety of situations. They should be able both to follow elementary mathematical arguments and to identify mistakes in them.

Experience with mathematical exploration and conjecture: Students should experience the challenge and pleasure of exploring new concepts, looking for patterns, making conjectures, and trying to decide whether they are true or false.

Communication: Students should be able to discuss mathematical ideas coherently with their fellow students and express themselves clearly when giving a proof or counterexample.

Connections with applications: Students should come to appreciate connections between mathematical concepts (such as those between general relations on sets and functions or between propositional logic and set properties) and between abstract mathematical ideas and concrete applications (such as those between propositional logic and digital logic circuits or between abstract graphs and scheduling problems).

Pedagogy. Because many of the cognitive learning goals aim to improve students' ability to evaluate mathematical statements, justify their truth or falsity, and express themselves clearly and precisely, a chief teaching recommendation is to ensure students' active participation and give them frequent feedback on their work.

Addressing students' varied backgrounds: Students come to a Discrete Mathematics course with a variety of backgrounds and abilities. All the cognitive learning goals are important, but the level at which students will achieve them will vary from student to student and from institution to institution. Instructors can address the fact that students absorb ideas and develop new skills at different rates by incorporating review of basic logical principles when introducing new topics and by mixing, throughout the course, straight-forward, concrete problems with ones that are more challenging and abstract. The majority of students coming into a Discrete Mathematics course, even those who have studied calculus, will have had little or no previous experience with the modes of thought listed as cognitive learning goals. For them, one course is just a start on the road to deep understanding. Follow-up courses will need to extend and reinforce the mental abilities they develop in this course.

Syllabus

1. ***Basic Logic and Introduction to Proof***

   A. *Propositional and predicate logic*. Logical connectives and truth tables; logical equivalences; tautology; valid and invalid arguments; universal and existential statements; translating from formal to informal language; arguments with quantified statements.
   B. *Methods of proof.* Conditional proof; counterexample; direct proof using a generic element; proof by contradiction; proof by contraposition.
   C. *Elementary number properties.* Divisibility; rational and irrational numbers; division (aka quotient-remainder) theorem; the Euclidean algorithm; floor and ceiling.

2. ***Sequences, Mathematical Induction, and Recursion***

   A. *Sequences.* Basic properties of sequences; explicit and recursive definitions; summation and product notation; closed form.
   B. *Mathematical induction.* Ordinary mathematical induction; arithmetic and geometric sequences; strong mathematical induction; the well-ordering principle.
   C. *Recursively defined sequences.* Examples and the recursive paradigm; introduction to solving recurrence relations; general recursive definitions; structural induction.

3. ***Set Theory, Functions, and Relations***

   A. *Set Theory*. Basic definitions and set identities; the element method of proof; disproof by counterexample; algebraic proofs; Russell's Paradox (optional).

B. *Functions*. Definitions of relations and functions; examples of functions defined on general sets; one-to-one, onto, and inverse functions; composition of functions; cardinality.
C. *Properties of relations.* Reflexive, symmetric, and transitive properties of relations; equivalence relations; partial order relations.

4. ***Combinatorics and Probability***

A. *Combinatorics.* The multiplication and addition rules; combinations; the pigeonhole principle; Pascal's formula; the binomial theorem.
B. *Probability*. Probability axioms; expected value; conditional probability and Bayes' formula; independent events and binomial probabilities.

5. ***Graphs and Trees***.

A. *Graphs*: Definitions and examples; trails, paths, and circuits; matrix representations of graphs; graph isomorphisms.
B. *Trees.* Examples and basic properties; rooted trees; spanning trees.

6. ***Applications and syllabus variations***

A. *Guiding principles:* All Discrete Mathematics courses should contain applications, but syllabi for courses designed exclusively for computer science majors may emphasize them to a greater extent than courses designed for a mixture of mathematics, computer science, and other majors. However, because consensus is lacking about which applications are essential and because all cannot be incorporated into a semester course, decisions about which to include have to be made locally, ideally in consultation with colleagues from computer science.
B. *Partial list of applications:* Using De Morgan's laws to rewrite the termination condition for a loop; digital logic circuits; binary notation; circuits for addition; logic programming; proof by resolution; proving algorithm correctness; Boolean algebra; the halting problem; existence of non-computable functions; relational databases; modular arithmetic with applications to cryptography; hashing functions; encoding and decoding functions; Hamming distance function; Boolean functions; IP addresses; the discrete representation of real numbers and floating point arithmetic; using Bayes' theorem to compute a real-world probability; computing the number of bits in the binary representation of an integer; use of graphs as models; PERT and CPM scheduling; traveling salesman problem; graph coloring; graph algorithms (e.g., minimum spanning trees, shortest path algorithm, binary tree traversal); searching and sorting algorithms; introduction to the analysis of algorithms; formal languages and regular expressions; finite-state automata.

Remarks
Various orderings of the material are possible according to the taste of the instructor, the needs of the computer science faculty, and the interests of the students. For example:

- Certain basic properties of graphs can be explored using only properties of even and odd integers;
- Some instructors find it effective to intermix the presentation of basic logical principles with methods of proof whereas others prefer to have students practice basic logical principles before starting mathematical proof, and some of the latter precede the study of mathematical proof by having students do natural deduction using logical notation;
- Much of the combinatorial material can be included at almost any point after the basic set operations have been explored;
- Some versions of the course include programming assignments or the use of Internet applets, an extensive listing of which can be found here.

II. Combinatorial mathematics courses

A large number of mathematics departments now offer a course in combinatorial mathematics covering graph theory and enumeration. The major question for an instructor in teaching this course is the balance between problem-solving and theory. Typically the course's primary focus is on problem-solving with some associated theory. Graph theoretic and enumerative problem-solving skills are critical for computer science and operations research as well as much of discrete probability. Since some combinatorial problem-solving topics are part of the Common Core high school curriculum, an applied focus is appropriate for prospective high school teachers. However, based on the philosophy of the mathematics major and preferences of an instructor, a valuable course for students can be designed that puts the primary focus on theory. Graph theory, in particular, is a good setting for strengthening proof skills because of the visual nature of the subject and its proofs. That said, the focus of the following discussion is on a problem-solving approach to the course.

The history of (upper-division) Combinatorics courses has two components. First, research in combinatorics has a long history and so as the mathematics major became more theoretical 50 years ago it was natural that theory-oriented Combinatorics courses were created. The development of computer science around the same time motivated the creation of combinatorial problem-solving courses. The course first appeared in CUPM recommendations as a Discrete Methods course in the 1981 CUPM report, Recommendations for a General Mathematical Sciences Major.

Unlike calculus problems, combinatorial problems are typically not solvable with a core set of theorems and formulas, although graph theory does involve many useful theorems. Most combinatorial problems, especially enumeration problems, are solved through a careful logical analysis of possibilities with simple ad hoc structures. *Thus the main cognitive goal of a Combinatorics course is the development of these combinatorial reasoning skills.* This process involves reasoning in a framework where most problems require solutions that are individually crafted from first principles. A more theoretically oriented Combinatorics course will also contain work with proofs, especially in the graph theory part of the course.

Within the general context of solving problems from first principles, there are usually several different approaches for solving a combinatorial problem; for example, breaking the whole problem into more tractable subparts; solving an appropriate subcase and extending the analysis to other subcases; solving a complementary problem; or finding a way to re-state the problem in terms of a previously solved problem. This leads to the second main cognitive goal for such courses: facility with such multiple approaches in problem-solving. First, students need to be 'de-programmed' of their habit of trying to fit a problem into one of a class of known problem types, each of which has a standard solution procedure. Then they have to develop the experience and confidence to try multiple approaches to solving a problem.

While particular concepts and techniques learned in this course may be forgotten, the combinatorial reasoning skills in these two cognitive goals will be used extensively throughout most quantitative careers.

Combinatorial topics have a variety of interesting motivating applications, but such applications usually need to be greatly simplified in order not to become overwhelmed by practical specifics.

Instruction in combinatorial problem-solving lends itself naturally to an interactive classroom where students can suggest different approaches to a problem and other students can be asked whether a particular approach seems promising. This framework helps students collectively to learn to develop skill with multiple solutions to a problem. Naturally, most students initially have very difficulty reasoning from first principles without formulas to plug into. Thus much class time needs to be devoted to analyzing mistakes in combinatorial reasoning and learning from them. A dividend of such discussions is the development of mathematical communication skills.

Syllabus. The enumeration and graph theory parts can be covered in either order. While many texts start with enumeration, graph theory is easier for most students because the graph diagrams are helpful in organizing combinatorial reasoning. Since the course normally emphasizes general combinatorial reasoning rather than particular theorems or techniques, a large degree of flexibility is possible in the choice of topics.

1. ***Enumerative Combinatorics***
   A. *Elementary counting principles.* Tree diagrams; sum and product rules; decomposing problems into subcases and sequences of sub-problems.
   B. *Permutations and combinations.* Binomial coefficients; Pascal's triangle; multinomial coefficients; combinations with repetition; distribution problems; probability problems; binomial identities. Comment: there should be an increasing complexity of problem-solving that becomes the main focus of this part of the course.
   C. *Generating functions.* Modeling selection and arrangement with constrained repetition.
   D. *Inclusion-Exclusion principle.* A generalization of solving a problem by counting a complementary outcome.
   E. *Recurrence relations,* Building an array of recurrence models and learning ways to

solve some of them; Fibonacci numbers and their applications;

Optional Topics: Generating functions, partitions, Polya's enumeration formula, combinatorial designs, Ramsey theory.

2. ***Graph Theory***

A. *Basic properties of graphs.* Paths, circuits, and connectedness; isomorphism; planarity and dual graphs; digraphs.
B. *Trees.* Basic properties; spanning trees; searching problems. Comments: trees are particularly important for computer science students.
C. *Graph coloring.* Chromatic number; map coloring, some coloring theorems.
D. *Eulerian and Hamiltonian circuits.* Euler circuit theorem; existence and non-existence of Hamiltonian circuits.

Optional Topics: tournaments; matching; network flows; matroids.

Prerequisites are a matter of mathematical maturity rather than content knowledge. Often, a sophomore mathematics course, such as linear algebra, is required as a proxy for this maturity prerequisite.

A two-course sequence has a natural structure, covering enumerative material in one course and graph theory in the other.

Applications are discussed primarily in the graph theory part of the course. However, specifics of most applications are usually a bit messy, and so most applications are presented in a sketchy way without concrete examples. However, it should be emphasized that this course is valuable for future applied coursework and careers because it develops the reasoning skills that are fundamental to solving problems and developing software in a range of applied fields.

Technology typically plays a small role in this course since the focus is on combinatorial problem-solving and associated logical reasoning. Computer problems such as representing graphs in a computer or enumerating all combinations of a set are covered in computer science courses.

Resources:

Remark*: The presence of a text on this list is not meant to imply an endorsement of that text, nor is the absence of a particular text from the list meant to be an anti-endorsement. The texts are chosen to illustrate the sorts of texts that support various types of Discrete Mathematics and combinatorial mathematics courses. Please note that some of the books listed below were written by the authors of this report.*

Possible texts for *Discrete Mathematics Courses*

1. Bender, Edward A. and S. Gill Williamson, *Lectures in Discrete Mathematics*.

   This book is freely available online.

2. Chartrand, Gary and Ping Zhang, *Discrete Mathematics*. Waveland, 2011.

   This book has a full chapter on planar graphs and graph colorings.

3. Ensley, Douglas E. and J. Winston Crawley, *Discrete Mathematics: Mathematical Reasoning and Proof with Puzzles, Patterns, and Games*, Wiley, 2005.

   As indicated in the subtitle, this book uses puzzles and games as a basis for exploring the standard discrete mathematical topics.

4. Epp, Susanna S., *Discrete Mathematics with Applications*, Cengage Learning, 4th edition 2011.

   This is a widely used text. A briefer version is available as *Discrete Mathematics: An Introduction to Mathematical Reasoning.*

5. Ferland, Kevin, *Discrete Mathematics*, Cengage Learning, 2009.

6. Graham, Ronald L. and Donald Knuth and Oren Patashnik, *Concrete Mathematics: A Foundation for Computer Science*, Addison-Wesley Professional, 1994.

   While more advanced than for a typical lower-division course, this book is valuable for reference and to recommend to good students.

7. Gries, David and Fred Schneider, *A Logical Approach to Discrete Math*, Springer, 1993.

8. Hunter, David J., *Essentials of Discrete Mathematics,* 2nd Ed, Jones and Bartlett, 2011.

   Chapters in this text are labeled by the modes of thinking they promote.

9. Johnsonbaugh, Richard, *Discrete Mathematics,* 7th Ed. Pearson, 2008.

This is a significantly updated version of one of the original texts for a Discrete Mathematics course.

10. Rosen, Kenneth H., *Discrete Mathematics and Its Applications*, 7th Ed., McGraw-Hill, 2011.

    This encyclopedic text is widely used, especially in computer science departments. The separately available *Student's Solutions Guide for Discrete Mathematics and Its Applications* is popular.

11. Scheinerman, Edward A., *Mathematics: A Discrete Introduction*, 3rd Ed., Cengage Learning, 2012.

Other relevant references for *Discrete Mathematics Courses*

12. Report: *Committee on Discrete Mathematics in the First Two Years*, M. Siegel chairperson, Mathematical Association of America, 1986.

13. *Computing Curricula 2001: Computer Science*, Association for Computing Machinery, 2001.

14. *On the Implementation of a Discrete Mathematics Course*, A SIGCSE Committee Report, B. Marion & D. Baldwin, co-chairs, Association for Computing Machinery, 2007.

Possible Textbooks for *Combinatorial Mathematics Courses:* There are a number of possible textbooks. Only the more time-tested texts (in their third or later editions) are given here:

15. Brualdi, Richard, *Introductory Combinatorics*, 5th Ed., Prentice-Hall, 2009.

    Balances theory and applications.

16. Bogart, Kenneth, *Introductory Combinatorics*, 3rd Ed., Harcourt Brace Jovanovich, 2000.

    Balances theory and applications.

17. Roberts, Fred and Barry Tesman, *Applied Combinatorics*, 3rd Ed., Prentice-Hall, 2009.

    Balanced, covers a wide range of material.

18. Tucker, Alan, *Applied Combinatorics*, 6th Ed., John Wiley & Sons, 2012.

    More emphasis on problem-solving than theory.