POSITION STATEMENT

Mathematical Computer Science Courses

D. Loveland
Duke University
Durham, North Carolina

In order to put my views of the mathematical aspects of computer science training in perspective, I have made a gross (and uneven) course outline of this portion of the curriculum. Obviously, it is not a revolutionary plan; basic programs of any value deal with material of lasting value. I would emphasize the mathematics for applications more (probability, statistics, optimization theory) at the possible expense of modern algebra. A major contribution we can make over Curriculum 68 is to give more attention to suggesting priorities, both over the curriculum and within specific courses. For example, I suggest logic and computability be compressed into one course with less detail than a standard mathematical logic course or computability course (including Curriculum 68, Course A7). For most students the basic concepts are worthwhile and the next level of detail is irrelevant. The automata course is altered from Curriculum 68 and Optimization Theory and Analysis of Algorithms courses are added.

COURSES:

Core:

1. Calculus (2 semesters)
2. Linear Algebra
3. Introduction to Discrete Mathematics
4. Numerical Calculus
5. Probability
6. Statistics
7. Optimization Theory

Undergraduate Electives:

8. Applied Algebra
9. Numerical Mathematics
10. Multivariate Calculus

Graduate Courses:

G1. Logic and Computability
G2. Formal Languages and Automata
G3. Adv. Numerical Mathematics
G4. System Simulation
G5. Analysis of Algorithms

COURSE OUTLINES

1. Calculus

Standard course

2. Linear Algebra

Usual sophomore level course. Three lectures on the definition of group, semi-group, etc. desirable, as is the use of computers for exercises if programming can be an assumed prerequisite.

3. Introduction to Discrete Mathematics

Boolean algebra: set algebra, propositional logic (non-axiomatic) Relations, functions. Graphs: directed, undirected, trees, etc. Combinations: permutations, combinations, transformation groups, Polya's theorem on counting.

Note: Combinatorial mathematics replaces group theory, except for a discussion of transformation groups.

4. Numerical Calculus (Prereq. 1, 2 comp. programming)

Course B4, Cu 68. However, more emphasis on differential equations as this course replaces the sophomore d.e. course.

5. Probability (Prereq. 1)

Both discrete and continuous theory treated. Computer use for exercises strongly desired.

6. Statistics (Prereq. 5, comp. programming)

Use of computer strongly desired.

7. Optimization Techniques (Prereq. 4)

Convexity, n-space geometry, Lagrange multipliers, Simplex method. Linear and convex programming. Computational aspects are to be emphasized.

ELECTIVES (top priority)

8. Applied Modern Algebra (Prereq. 2, 3)

Semi-groups, groups, (polynomial) rings, finite fields. Boolean algebras, lattices. Applications to switching algebra, logic, coding theory, formal languages or automata.

9. Numerical Mathematics (Prereq. 4)

Course I8, Cu 68.

10. Multivariate Calculus (Prereq. 4 or DE)

Usual course.

GRADUATE LEVEL COURSES

G1. Logic and Computability (Prereq. 3)

Propositional calculus, predicate calculus, Axiomatic bases, Validity, Completeness. Peano axioms. Turing machine (or equivalent abstract machine, e.g. Shepherdson-Sturgis machines). Universal machines, halting problem. Recursive functions, normal form theorem, Godel numbering, Church-Turing thesis. Recursive and r.e. sets, relations. Decision procedures, Presburger decision procedure, Fischer-Rabin theorem (commentary). Elements of abstract complexity theory: essentially difficult functions, speed-up theorem (commentary).

Note: Alternatively, one could have this course follow G2 and, instead of abstract complexity, do the Hartmanis-Stearns-Lewis time and tape complexity results. We believe this course should precede G2, however, as it is more fundamental.

G2. Formal Languages and Automata (Prereq. G1)

Overview of Chomsky language hierarchy. Finite automata: representation, equivalence of deterministic and non-determistic finite automata, decision problems. Nerode-Myhill Theorem. Regular languages, Kleene theorem. Elements of sequential machine structure theory: equivalence, incomplete machines, minimal state machines, decomposition. Context-free languages: decision problems, embedding lemma, normal forms. Relationship to programming languages. Pushdown automata. Introduction to parsing methods.

Note: This is a more basic and extensive course than "Formal Languages and Syntactic Analysis" Cu 68, Course A1. A course or seminar on syntactic analysis can pick up the parsing, etc. in the detail suggested in Course A1 outline. A course in sequential machines covering sequential machine structure theory in detail (see Cu 68, Course I7) can be given on demand, or integrated with the switching theory course.

G3. Advanced Numerical Mathematics (Prereq. 10)

Course I9, Cu 68 with more emphasis on algorithm implementation and error analysis.

Note: A serious alternative or addition should be "Numerical Statistics" which studies the algorithms (including implementation) used in statistical data analysis.

G4. System Simulation

Course A4, Cu 68.

G5. Analysis of Algorithms

Divide and recurse technique, application to real and matrix multiplication sorting. Sorting and order statistics algorithms. Tree search techniques. Graph connectedness and path transversal techniques. Substring recognition algorithms. NP-complete and NP-hard problems. Alternately, Fast fourier transform.