

Theory I: Mathematical Foundations of Computing

Course Philosophy: This course provides an introduction to essential mathematical concepts in computing, including logic, induction, discrete structures (sets, functions, and relations), automata and complexity theory. It begins with coverage of logic and induction (where proof skills are stressed) and then moves on to important discrete structures in computer science, which provides the opportunity to present important theoretical concepts (infinite sets, complements, countability, diagonalization, etc.) that will be used in context later in the course. One of the primary outcomes of the course is also to give students experience with proofs, both in learning proof techniques as well as their effective application in a variety of contexts. In this way, students will begin to gain the mathematical maturity needed for approaching more complex problems.

Topics:

- Logic and proof techniques (9 lectures)
 - Propositional logic
 - Truth tables, logical equivalence and consequence
 - Formal proofs and techniques (direct proof, by cases, by contradiction)
 - Predicate logic (with quantification)
 - Applications: Satisfiability, SAT solving (Putnam-Davis), complexity of Putnam-Davis
- Induction (4 lectures)
 - Formal proofs with induction
 - Weak and strong induction
 - Sequences and summations
 - Applications: program proofs, structural induction on trees
- Sets, functions, and relations (4 lectures)
 - Functions (concepts, definitions, and proofs)
 - Sets, set operations, infinite sets, countability, and diagonalization
 - Relations, partial orders (concepts, definitions, and proofs)
 - Introduction to graphs via their use in representing relations
 - Applications: Using sets to construct error-correcting codes
 - Applications: Using relations in modeling social networks
- Introduction to formal languages (1 lecture)
 - Alphabets, strings, languages
- DFAs and NFAs (1 lecture)
 - DFAs and NFAs
- Regular expressions (2 lectures)
 - Regular expressions and equivalence to automata
 - Proofs of non-regularity, pumping lemma (optional)
- Context-free grammars (2 lectures)
 - CFGs, parse trees and ambiguity
 - Applications: compilers
- Turing Machines (3 lectures)
 - TMs and TM programs
 - Undecidability and the Halting problem
Introduction to reductions and examples of reductions to the Halting problem
- NP-completeness (3 lectures)
 - P and NP, examples of NP-complete problems
 - Examples of reductions of NP-complete problems
 - SAT revisited and Cook's theorem

Potential instructors: David Dill, Bob Plummer, Mehran Sahami

Notes: This is a new course encompassing material from CS103 and CS154. The course includes a reduced coverage of logic (from 5 weeks down to 3 weeks) and induction. It then provides coverage of introductory discrete structures material (sets, functions, and relations) now covered in CS103B as means of bridging the proof techniques previously encountered to other techniques (such as diagonalization and enumeration) that will be useful in the analysis of automata. The course then moves onto coverage of a significantly reduced subset of material currently in CS154 – coverage of automata is a much smaller subset than currently covered in CS154, but provides slightly expanded over the coverage currently provided in CS103B. The introduction to NP-completeness at the end of the course gives students the opportunity to understand and appreciate one of the most significant open problems in the field early in their education. The course will be a 5 unit course to reflect its increased coverage of material (and corresponding workload). It will have roughly three 50 minute lectures/week. Given the coverage of basic mathematical concepts, we will try to classify this course as meeting the SoE Mathematics requirements.

There is no ordering constraint between Theory I and Theory II. Despite the numbering scheme here, they may be taken in either order, giving students more flexibility in their program.

CS154 would continue to be offered as a separate course (possibly covering some more advanced material than it currently does) that could be taken by undergraduates interested in pursuing the material on automata further (either as an elective or within a more theoretically-oriented track), as well as continuing to be a breadth requirement for the MS program.

By way of comparison, neither MIT nor CMU require a full course in automata and computability for undergraduates in general. MIT only has such a requirement (6.045) for students pursuing a concentration in Theoretical Computer Science. CMU offers such a course (80-311) as a potential advanced elective that students can choose from (among a list of several courses in the "Fundamentals of Programming" area). Both MIT and CMU do include some basic automata theory (at the level less than or equal to that covered here) in other required introductory courses.

Theory II: Introduction to Probability for Computer Scientists

Course Philosophy: The goal of this course is to provide students with essential foundations of probability theory used in computer science. Applications of theoretical concepts are used to better tie students' mathematical understanding to applications in real systems. Students continue to develop their proof skills in a more numerically oriented context.

Topics:

- Counting and Combinatorics (2-3 lectures)
 - Combinations
 - Permutations
 - Pigeonhole principle
- Probability theory (22-23 lectures)
 - Basic counting, random variables, events
 - Conditional probability
 - Independence and conditional independence
 - Discrete distributions: Uniform, Binomial, Multinomial
 - Continuous distributions: Normal
 - The exponential distribution and Poisson processes
 - Point estimation, expectation and variance
 - Law of large numbers and the Central Limit Theorem
 - Bayes' Theorem
 - Hypothesis testing
 - Applications: analysis of collisions in hashing
 - Applications: PageRank and the random surfer model
 - Applications: data analysis (e.g., comparing system performance)
 - Applications: modeling uncertainty, probabilistic inference
- Introduction to machine learning (4 lectures)
 - Hypothesis spaces, learning as search
 - Fitting data (i.e., changing internal representation) to optimize an objective
 - The Naive Bayes algorithm and Logistic regression
 - Applications: Email spam filtering
 - Applications: Simple recommender systems

Potential instructors: Bob Plummer and Mehran Sahami

Notes: This course is focused on providing a rigorous introduction to probability and its application in computer science. As a result, coverage of logic and discrete structures are moved to other courses, allowing time to provide very significant coverage of probability theory (enough so that STAT 116 is no longer a requirement) and present an introduction to machine learning as a means for showing that computers are capable of "learning"—that is, modifying an internal representation (whether it be logically or probabilistically based)—and providing a "hook" to excite students regarding future possibilities in CS through applications that they are likely to have encountered on a regular basis. The course will be a 5 unit course to reflect its increased coverage of material (and corresponding workload). It will have three 50 minute lectures/week. Given the mathematical nature of this course, we plan on classifying it as meeting the SoE Mathematics requirement.

There is no ordering constraint between Theory I and Theory II. Despite the numbering scheme here, they may be taken in either order, giving students more flexibility in their program.

Theory III (CS161): Data Structures and Algorithms

Course Philosophy: In this course, students are introduced to and given a rigorous treatment of algorithmic techniques and the discrete structures that they are applied to, as well as mathematical tools for their analysis. The course begins with an introduction to complexity theory, and uses this foundation for analyzing algorithms throughout the course. Discrete structures (tree and graphs) and their properties are introduced in conjunction with algorithms that operate on in these contexts. Applications are also emphasized through the various algorithms presented.

Topics:

- Algorithmic complexity and analysis (4 lectures)
 - Asymptotics: Big Oh, Omega, and Theta notation
 - Recurrence relations
 - Master theorem
- Randomization, divide and conquer (2 lectures)
 - Introduction to randomized algorithms
 - Quicksort, divide and conquer
- Heaps and counting sort (1 lecture)
- Hashing (2 lectures)
- Tree and graph definitions and properties (1 lecture)
- Binary Search Trees (1 lecture)
- Greedy Algorithms (2 lectures)
 - Greedy algorithms
 - Min-cost spanning tree
- Dynamic programming (3 lectures)
- Graph algorithms (4 lectures)
 - Graph algorithms, topological algorithms
 - Dijkstra's algorithm for shortest paths
 - Applications
- Blind and heuristic search (A^*) in graphs (1 lecture)

Potential instructors: Leo Guibas, Serge Plotkin, and Tim Roughgarden

Notes: This course is significant modification of our current CS161 course. It provides a slightly slower paced introduction to asymptotics (since students will no longer be getting this introduction in CS103B) and adds more introductory material on trees and graphs (again, currently covered in CS103B). Discussion of order statistics and Red-Black trees are dropped in order to help provide room for the inclusion of this introductory material. Also heuristic search, via the A^* algorithm, is also included as an AI algorithm for analysis. The course will be a 5 unit course to reflect its increased coverage of material (and corresponding workload). It will have roughly two 75 minute lectures/week, but will also require some material to be covered in additional sections/lectures. This course will continue to be classified as a CS depth course.

Prerequisites: This course has both Theory I and Theory II as prerequisites. Importantly, there is no ordering constraint between Theory I and Theory II – they may be taken in any order as long as they are both taken prior to this course.