Theory of Computing

View PDF

Instructor(s):

Gyula Y. Katona
Balázs Patkós
Padmini Mukkamala

Short Description of the Course:
In the first part of this coursewe introduce different theoretical models for computing. There are those that are easy to describe but have limited power, and there are more complicated models with more power. We begin with finite automatons. On the one hand they can be used directly, since it is easy to build them to perform simple tasks. Many electronic devices contain basically one chip, which is really a finite automaton. On the other hand, this model can be used as a programming technique, as well, to help with code optimization and verification. Unfortunately, not every problem can be solved in this way, and we will also examine their limits.

The next model is the pushdown automaton. It has more power but is still relatively easy to build. It is used widely in compilers, for example.

The last important model is the Turing machine. It is considered to be the model of an everyday computer. It is very powerful, and can be used to solve most real life problems, especially when there is no time limit. We will define many variants, and examine how such variations change its scope and ability.

In the second part of this course we answer an important question: *Can every problem be expressed in algorithms? And if there is an algorithm for a particular problem, can we implement it with a given computational model?* It turns out that some interesting problems seem to be much more difficult than others even when implemented on extremely fast computers.

Another question concerns what happens if we have limited resources, if time and/or space for computation is limited, which is often the case in real life.

Scientists have tried to solve some of the most difficult algorithmic problems over centuries, but have failed in some cases. Is it the case that they are not clever enough? Is there no algorithm at all? Or is there something in between these two possibilities?

Aim of the Course:
The main aim of the course is to establish a mathematical background for computational problems. With this background we are able to derive results that have important consequences in many areas of mathematics, engineering, programming and other practical areas of life. The theoretical foundation also gives new tools that can be useful in other disciplines.

Prerequisites:
Some experience with basic proof methods *(e.g. proof by construction, contradiction, and induction)* is recommended.

Very basic knowledge of combinatorics, number theory, sets, logic *(e.g. definitions and basic properties of graphs, binomial coefficients, primes, union, set complement, AND, OR)* is helpful, but not absolutely necessary.

Although course content directly relates to programming no prior knowledge of programming is required.

Detailed Program and Class Schedule:

1. Finite automata, non-deterministic finite automata.
2. Regular expressions, regular and non-regular languages, pumping lemma.
3. Pushdown automata, context-free grammars.
4. Non-context-free languages. Turing Machines and variants.
5. Universal Turing machines, enumerators, Church-Turing Thesis.
6. Algorithmic decidability, reducibility.
7. Undecidable problems; *Midterm Exam*.
8. Post's correspondence problem, domino problem.
9. Storage and time, space and time complexity.
10. The class P, space-time theorem.
11. PSPACE, EXPTIME, NTIME, NSPACE.
12. Polynomial time reduction, NP and coNP.
13. Cook's theorem, NP-complete problems.
14. Applications, review; *Final Exam*.

Sample homework, midterm, and final to be downloaded here

Method of instruction:
Lectures, recitations

Grading:
There will be a homework set roughly every week consisting of 4 exercises (1 easy, 2 medium and 1 hard problem).

There is one in class Midterm 6 exercises for 2 hours (open book, open notes).

The final test is in class, 6 questions for 2 hours. The first question is to state a definition or a theorem (without proof), the second question is to prove one of the theorems which had been proven in the lectures (a list of the possible proofs will be provided before the final). This part of the final is closed books, closed notes. Then four other exercises with open book, open notes.

The final grade is:
5% Classroom participation, 25% Homeworks, 30% Midterm, 40% Final

90% A+
85% A
80% A-
75% B+
70% B
65% B-
etc.

Textbook:
Michael Sipser: *Introduction to the Theory of Computation*, Thomson CourseTechnology, 2006

80% of the course material can be found in the book. We cover about half of the book, Chapters 1-8 with a few exceptions, however part of the material is presented in different approach. Generally there are no reading assignments, but it is always useful to read the corresponding Chapter in the book.

Instructors' bio:

Gyula Y. Katona (born 1965) is an associate professor and head at the Department of Computer Science and Information Theory, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics (BME). He graduated from Eötvös Loránd University as a mathematician in 1991. Receiving his Ph.D. in mathematics in 1997 from the Hungarian Academy of Sciences, Katona spent two years at Ibaraki University, Japan. He also had a visiting appointment for a year at Arizona State University. He is the author of more than 30 papers and co-author of a university textbook on discrete mathematics. He has been teaching Theory of Computing at the Budapest Semesters of Mathematics for several years. His Erd?s number is 2.

Balázs Patkós (born 1978) is head of Department of Combinatorics and its Applications at the Alfréd Rényi Institute of Mathematics, Hungarian Academy of Sciences. He obtained his MSc degree at Eötvös University, Budapest in 2003 and his PhD at Central European University in 2008. After a 3-semester postdoc position at the University of Memphis, he returned to Hungary in 2010. He has been teaching introductory courses in several areas of mathematics at several universities (Eötvös University, Budapest University of Technology and Economics, Central European University, McDaniel College) both in English and in Hungarian. He has been awarded the Youth Prize of the Hungarian Academy of Sciences and several research fellowships by the Academy and the Hungarian National Scientific Fund. His Erd?s number is 2.

Padmini Mukkamala (born 1983) received her Ph.D. in Mathematics from Rutgers, The State University of New Jersey, in 2011. At Rutgers, she received the "Teaching Award" for Fall 2008 based on student feedback surveys. She holds a B.Tech degree in Computer Science from Indian Institute of Technology, Delhi. She has taught courses in Mathematics and Computer Science at Rutgers, IIT-Delhi, McDaniel College and Budapest University of Technology and Economics. Her Erd?s number is 2.