On-line course materials

MATH43011 - Computation and Complexity

Year: 4 - **Semester:** 1 - **Credit Rating:** 15

Aims

The course unit aims

- to introduce the main model of computation currently being employed in the theory of computation, Turing machines;
- to introduce the key parameters quantifying computational complexity (deterministic, non-deterministic, time, space) and the relationships between them.

Brief Description

Quite a lot of the mathematics you have studied so far involves using algorithms to solve computational problems. For example, you have probably used Euclid's algorithm to solve the problem of finding the greatest common divisor of two integers. In this course, we abstract a level further, and study the properties of problems and algorithms themselves. The kind of questions we ask are "is there an algorithm to solve EVERY problem?" and "what problems can be solved by an EFFICIENT algorithm?".

Compared with most of mathematics, this area is in its infancy, and many important things remain unknown. The course will take you to the point where you understand the statement of one of the most important open questions in mathematics and computer science: the "P vs NP" problem, for which the Clay Mathematics Foundation is offering a $1,000,000 prize. And who knows, perhaps one day you will be the one to solve it!

Learning Outcomes

On successful completion of the course unit students should

- be familiar with Turing machines and their capabilities and limitations, and be able to construct and analyse examples to solve simple problems;
- have a basic overview of classical complexity theory, including the main parameters quantifying computational complexity classes.
- be able to classify and compare the decidability and complexity of decision problems in simple cases.
- understand the statement of, and have an appreciation of some of the issues and concepts surrounding, the "P vs NP" problem.

Syllabus

0. INTRODUCTION (1 lecture): outline introduction to computability and complexity; course practicalities.

1. COMPUTABILITY (11 lectures): problems and solutions; alphabets and languages; Turing machines; recursiveness and the Church-Turing Thesis; multitape machines; coding machines and non-recursive languages; universal computation; non-determinism.

2. COMPUTATIONAL COMPLEXITY (8 lectures): time and space; linear speed up and space reduction; complexity classes; lower bounds and crossing arguments; space and time hierarchy theorems; tractability and P vs NP; polynomial time reduction.

3. COMPLETENESS (9 lectures): NP-completeness; SAT and the Cook-Levin Theorem; NP-completeness by reduction; further examples of NP-complete languages; NP-intermediacy and Ladner's Theorem; PSpace-completeness; oracles and the Baker-Gill-Solovay Theorem.

4. SPACE COMPLEXITY (3 lectures): Savitch's Theorem; the Immerman-Szelepcsenyi Theorem.

Teaching & Learning Process (Hours Allocated To)

Lectures	Tutorials/ Example Classes	Practical Work/ Laboratory	Private Study	Total
22	11	0	117	150

Assessment and Feedback

- Mid-semester coursework: two take home tests weighting 20%
- End of course examination: two and a half hours weighting 80%.

Further Reading

Printed notes will be supplied and you should not need to refer to any books. But if you would like an alternative viewpoint, the following texts cover most of the course material:

- Bovet and Crescenzi, Introduction to the Theory of Complexity, 1994,
- Papadimitriou, Computational Complexity, 1994,
- Sipser, Introduction to the Theory of Computation (second edition), 2006.

Staff Involved

Dr Mark Kambites - Lecturer

Data source is EPS system
Back To Top