
CO2011 Automata, Languages and Computation

Credits: 20 **Convenor:** Prof R M Thomas **Semester:** 1st

Prerequisites:	<i>Essential: CO1011 or MA1101</i>	<i>Desirable: CO1003 (or equivalent)</i>
Assessment:	<i>Coursework: 30%</i>	<i>Three hour exam in January: 70%</i>
Lectures:	<i>30 hours</i>	Problem Classes: <i>5 hours</i>
Surgeries:	<i>5 hours</i>	Classtests: <i>5 hours</i>
		Private Study: <i>105 hours</i>

Subject Knowledge

Aims The aim of this module is to give an understanding of some of the basic theory of language recognition. The module will also aim to provide a general model of computation and thereby to illustrate the limits of the power of computers, both in terms of the problems for which a solution exists and also the problems for which a feasible solution exists.

Learning Outcomes By the end of the module students should be able to describe some abstract models of the process of computation such as finite automata, pushdown automata and Turing machines. They should be able to construct basic arguments couched in terms of these models.

Methods Class sessions together with course notes, exercises and web support. Recommended textbooks for extra information and supplementary reading.

Assessment Marked class tests and written examination.

Skills

Aims To teach students a range of comprehension, writing and problem-solving skills.

Learning Outcomes Students should be able to solve problems and produce reasoned arguments about the power of the computational models studied in the course (using their understanding of these models to solve the problems). They should be capable of writing such arguments clearly and correctly with a proper use of formal notation where appropriate.

Methods Class sessions together with exercises.

Assessment Marked class tests and written examination.

Explanation of Prerequisites There is not much in the way of pre-requisite knowledge required for this module. We need the basic concepts of sets, logic, relations and functions as introduced in CO1011 or MA1101; either of these modules provides what we need here. In order to help understand the motivation, it would be helpful to have done some programming before; however, while previous experience of programming is desirable, it is not essential. Some of the methods in this module are expressed in a sort of pseudocode notation, but there is no actual programming content; a student who had not done programming before could still take this module if he/she wanted to. Such students are welcome to discuss their suitability for the course with the module convenor.

Course Description In this course we are primarily concerned with what computers can do. It turns out that there are problems that cannot be solved by computer, or, at least, by machines corresponding to the mathematical models of computers we shall present. It is clearly sensible to investigate which problems cannot be solved; there is no point trying to program a computer to solve a problem that is unsolvable! A problem may be unsolvable in the sense that no computer program exists that will solve it or in the sense that any program that would solve it would take longer than the lifetime of the universe to run. We will give some precise mathematical models of the process of computation; within these models, we will see what sort of tasks can be performed.

At first sight, it may appear that these models are unduly simple and do not really capture all the subtleties of the process of computation. The advantages of using such models is two-fold. First, they are very simple to reason

about, so that we can reach our conclusions much more simply than (for example) considering actual hardware and software components in fine detail. Second, they have proved to be very robust, in that successive generations of computers have all been shown to be no more powerful than the most general model we will present, and so the analysis based on these models has been useful throughout the history of Computer Science, whereas an analysis based on the specifics of various machines and programming languages quickly becomes obsolete.

Detailed Syllabus

Revision of mathematical pre-requisites (sets, logic, relations, graphs and functions). Strings. Formal languages. Operations on languages. Concatenation of strings. Kleene star.

Finite automata. Language acceptors. Regular languages. Equivalence. Complete automata. The concepts of determinism and non-determinism. The pumping lemma for regular languages. Examples of non-regular languages.

Grammars. Terminals and non-terminals. Regular grammars. Rewrite rules. Equivalence of regular grammars and finite automata. Closure properties of regular languages. Empty moves. Regular expressions. Equivalence of regular expressions and finite automata.

Stacks. Pushdown automata and context-free grammars. Emptying the stack. Parse trees. Leftmost and rightmost derivations. Equivalence of pushdown automata and context-free grammars. Ambiguous grammars. Inherent ambiguity. Closure properties of context-free languages. Are programming languages context-free? Deterministic context-free languages. Parsing. LL-parsers.

Turing machines. Extensions of Turing machines. Non-determinism. Church-Turing Thesis. Decision-making Turing machines. Recursive languages. Existence of Turing acceptable languages that are not recursive. The halting problem. Further examples of unsolvable problems.

Complexity. Space and time complexity and the relationship between them. Decision-making versus acceptance. Polynomial transformations. Determinism versus non-determinism. P and NP. NP-completeness. Examples of NP-complete problems.

Reading List

- [C] H. R. Lewis and C. H. Papadimitriou, *Elements of the Theory of Computation, second edition*; ISBN: 0132727412, Prentice Hall, 1998.
- [C] J. E. Hopcroft, R. Motwani and J. D. Ullman, *Introduction to Automata Theory, Languages and Computation*; ISBN: 0321210298, Addison-Wesley, 2001.
- [C] J. G. Brookshear, *Formal Languages, Automata and Complexity*, Benjamin Cummings, 1989 (out of print, but copies available in the Library).
- [C] D. Kelly, *Automata and Formal Languages - an Introduction*, Prentice Hall, 1995 (out of print, but copies available in the Library).
- [C] D. Wood, *Theory of Computation*, Wiley, 1987 (out of print, but copies available in the Library).
- [C] D. I. A. Cohen, *Introduction to Computer Theory*; ISBN: 0471137723, Wiley, 1996.
- [C] J. Martin, *Introduction to Languages and the Theory of Computation*; ISBN: 0071198547, McGraw-Hill, 2003.
- [C] P. Linz, *An introduction to Formal Languages and Automata*; ISBN: 0763714224, Jones and Bartlett, 2001.

Resources Course notes, web page, study guide, exercises, lecture rooms with board space and two OHPs, past examination papers.

Module Evaluation Course questionnaires, course review.