Course content for MT5413, Complexity Theory

Prerequisites:
An UG course in discrete mathematics

Aims:
To introduce the technical skills to enable the student to understand the different classes of computational complexity, recognise when different problems have different computational hardness, and to be able to deduce cryptographic properties of related algorithms and protocols.

Learning outcomes:
1. Understand the formal definition of algorithms and Turing machines
2. Understand that not all languages are computable and prove simple examples
3. Organise the low-level complexity classes (P, NP, coNP, NP-complete, RP, ZPP, BPP, PSPACE) into a hierarchy and prove simple languages exist in each class
4. Give examples of one-way functions and hardcore functions, and demonstrate that every NP function has a hardcore predicate
5. Use complexity theoretic techniques as a method of analysing communication services
6. Demonstrate independent learning skills

Course content:
Algorithms: Motivation for complexity; languages; deterministic Turing machines; Church-Turing thesis; randomised algorithms.
Computability: Gödel numbers; incomputable languages.
Low-level complexity classes: Class P; 2-SAT; class NP; Cook’s theorem; 3-SAT; coNP; class RP; class BPP; probability amplification; relation between classes; class PSPACE.
One-way functions: One-way functions; one-way permutations; trapdoors; hardcore functions; Goldreich-Levin theorem
Applications of complexity theory to communication: Applications of complexity theory to analysing the efficiency of communication services.