SUBJECT OUTLINE

41080 Theory of Computing Science

Course area	UTS: Information Technology
Delivery	Spring 2023; City
Credit points	6cp
Requisite(s)	37181 Discrete Mathematics AND 48024 Programming 2
Result type	Grade and marks

Attendance: 3hpw, on campus

Subject coordinator

Dr Youming Qiao

Room: CB11.07.214

Email: Youming.Qiao@uts.edu.au

Teaching staff

Dr Youming Qiao

Email: Youming.Qiao@uts.edu.au

Subject description

This subject introduces the theory of computation, including topics from the theories of automata, rewriting, and parsing. Students learn to compute orders of complexity for various practical problems.

An understanding of computational complexity can lead to the design of economical and feasible products and services. For example, for an online taxi service, strategically simplifying the computation can dramatically reduce the cost of solving the very large-scale passenger-driver matching problem.

Similarly, mastering the fundamental theory of computation enables one to construct security schemes that are both convenient for users and tough against attackers. Also, in programming, the knowledge of how high-level statements are translated into operational instructions helps one identify performance bottlenecks to achieve optimal efficiency by appropriate design of the algorithm.

Subject learning objectives (SLOs)

Upon successful completion of this subject students should be able to:

1. Build models for real-world applications and plan computations realisable on computers
2. Formulate the Turing machine computation model using formal definition and higher-level operation description
3. Distinguish some decidable/non-decidable problems by proof in the Turing machine computation framework
4. Analyse algorithm efficiency and improve when possible

Course intended learning outcomes (CILOs)

This subject also contributes specifically to the development of the following Course Intended Learning Outcomes (CILOs):

- Design Oriented: FEIT graduates apply problem solving, design and decision-making methodologies to develop components, systems and processes to meet specified requirements. (C.1)
- Technically Proficient: FEIT graduates apply abstraction, mathematics and discipline fundamentals, software, tools and techniques to evaluate, implement and operate systems. (D.1)

Teaching and learning strategies

Each week, material will be delivered via a combination of preparatory pre-work, a collaborative lecture developing and expanding upon the pre-work, and a lab/tutorial exploring the concepts developed in the pre-work and lecture via scaffolded practical activities.

The pre-work will aim to engage the students with the core knowledge, which will then be challenged in the ctive learning sessions via active problem solving, peer learning activities and the like. The lab/tutorial sessions will provide a venue to apply this knowledge in aggregate and to extend it, accompanied by a tight feedback loop with their instructors. The lab/tutorial sessions will also provide a place in which to ensure concepts and skills required for the assignments are encountered and giving room to develop with high stakes nature of assessment. The lab/tutorial sessions will also include weekly opportunities for structured formative assessment and feedback, as separate from the summative assessment constituted in the assignments.

The subject will develop students' analytical and logical skills, along with their knowledge of the theory of computation, via active and collaborative learning activities exploring the connexions between the theory of computation and practical computer science.

The students' learning will be assessed via two connected assignments, both of which will build on and extend knowledge and skills developed and scaffolded in the pre-work, lectures and tutorials. The first assignment will present a simplified task focusing on an aspect of the theory of computation with tangible practical applications, such as basic parsing. This assignment will be completed individually. The second assignment will be a small-group assignment, where students will collaboratively enlarge upon the core concepts introduced in the first assignment, working together to solve a more advanced and realistic version of the problem presented in the first assignment, such as constructing a compiler or interpreter for a simple programming language. The second assignment will require students to communicate and to coordinate their efforts to complete the task. Together, the assignments will provide students with a concrete instantiation of the theory of computation in a real-world form.

The students' understanding of the underlying theory will be assessed via a written assignment, focused on developing correct and coherent solutions to domain relevant problems, rather than examining simple factual recall.

Content (topics)

- Regular languages and automata
- Regular expressions
- Context free languages
- Turing Machines
- Decidability
- Reducability
- Time/Space Complexity

Program

Week/Session	Dates	Description
1	7-11 Aug	**Pre-work & Lecture: Subject introduction and mathematical fundamentals.** The program will be introduced to students. Students will have a clear idea of the arrangement of this course: • what to expect in the following sessions • how the lab tutorials and course projects look like • what to prepare before the lab tutorials • the assignments and their deadlines and criteria for marking • how to seek help when encountering problems in learning the subject The pre-work will also provide the students a general overview of the subject: the main problems to study, how they are connected to each other, and how they are connected to the digital computers in the real world and the practical problems to solve. In the first lecture, the students will be able to address any initial issues, and will be given a first, gentle, problem focused learning session.

		The pre-work & lecture will prepare the students with necessary mathematical background knowledge and basic skills / tools for later study.
		Before week 1, introductory pages will be put on Canvas.
2	14-18 Aug	**Pre-work & Lecture: Automata and regular languages (1)**
		The pre-work and lecture will introduce the notion of automata, linking to regular languages. Students will study practical interesting examples of automata. The link between automata and language will be studied. Students will be given interesting challenges of designing automata that recognise particular languages. Students will get familiarised with the idea of language family and automata acceptance.
		Lab
		The lab tutorial will introduce the students programming skills /tools that will be used in the later lab tutorials and course projects.
		Students are encouraged to (re-)familiarise themselves with the basic concepts, such as sets, Boolean operations, in discrete mathematics. The background warming-up will facilitates the introduction of this highly abstract subject. The pre-work is not mandatory. The essential notions will be studied in class.
		Assignment 1 released on 14 August.
3	21-25 Aug	**Pre-work & Lecture: Automata and regular languages (2)**
		The pre-work and lecture will further explore the notion of regular languages and automata. The structure and attributes of the family of regular languages will be discussed. The properties will be studied linking finite state automata. Students will benefit from the understanding of the fundamental concepts and the way of formal reasoning (methods of proof).
		Lab
		In the lab tutorial, students will be guided to build concrete examples of automata. Students will gain experience of how automata works and how they produce languages.
		Students are encouraged to have pre-reading of automata and hand-on experience of regular expressions using common tools, such as [grep/sed/vi]. Some experiments used in the lab tutorial will be available on Canvas prior to the class (also applies to the following weeks).
		Students are provided with online quizzes with feedback on Canvas in the first three weeks.
4	28 Aug - 1 Sept	**Pre-work & Lecture: Automata and regular languages (3)**
		The pre-work and lecture will develop the final theory components of regular languages, highlighting the equivalence between finite state automata, regular expression and regular grammars. Completing the introduction to the central concepts and characterisations of formal languages more generally. The students will also be introduced to the fact that some languages are not regular.
		Lab

		The lab tutorial will allow students to explore different types of finite automata, as well as experiment with regular expressions. Pre-reading of how to sketch the link between regular expression and automata will facilitate the study of this week.
5	4-8 Sept	**Pre-work & Lecture: Context free languages** The pre-work and lecture will introduce students to context free languages. Students will learn the formal definition of context free languages, the notion and attributes of ambiguity, and the Chomsky normal form. Practical examples will be studied to show the link between context free languages and practical programming languages. **Lab** In the lab tutorial, the student will be guided into a framework helping them start with the course project – implementing a simple programming language. Students are encouraged to pre-study lab materials and examples of CFL grammar (will be available on Canvas). **Assignment 2 released on 4 Sept.**
6	11-15 Sept	**Pre-work & Lecture: Context free languages** The pre-work and lecture will continue the study of context free languages, touching on how context free languages are used in practice and what restrictions are necessary to produce efficient parsing algorithms. **Lab** In the lab, the student will focus on the conversion methods necessary to produce a grammar suitable for LL(1) parsing.
7	18-22 Sept	**Pre-work & Lecture: Turing machines** The pre-work and lecture will introduce the abstract model of a modern digital computer – Turing machine, along with the formal definition of algorithm, which is the foundation for the future study. **Lab** In the lab tutorial, the student will perform a case study of a Turing machine. They will use basic components to construct Turing machines to perform certain computation tasks. The idea of “computation” will be formalised as well as materialised. Pre-reading of Turing machine, and some background knowledge about the Turing machine and development of digital computers will be helpful.
	25-29 Sept	STUVAC

8	2-6 Oct	**Pre-work & Lecture:Decidability and limits to computation** The pre-work and lecture will study the capacity (and limits) of modern digital computers. The notion of decidability will be formally introduced. The limits of Turing machines will be demonstrated using examples. **Tutorial** Students will practise proving interesting and challenging examples related to the decidability. The practice will help students build the skill of formal reasoning and proof. For pre-work, general knowledge about the existence of non-decidable problems is useful.
9	9-13 Oct	**Pre-work & Lecture: Reducibility** The pre-work and lecture will introduce the concept of reducibility. The students will learn the structure of the family of computable problems, and obtain an overall idea about how to establish the connection between two problems. **Tutorial** Students will practise proving interesting and challenging examples related to reducibility. General pre-reading about reducibility and computational complexity will be helpful.
10	16-20 Oct	**Pre-work & Lecture: Computational complexity and problem classes** The pre-work and lecture will introduce students to computational complexity. The study will be focused on time complexity, with spatial complexity being briefly mentioned. The notion of big-O and small-o will be introduced. In terms of complexity, three classes, P, NP and NP-complete, will be introduced. **Lab** Examples of solving problems belonging to different complexity classes will be studied. The students will get experience on how the computation cost grows with the problem size, for different types of problems. For pre-work, students are encouraged to test algorithms of different time complexity, which are available on Canvas.
11	23-27 Oct	**Pre-work & Lecture: Complexity** The pre-work and lecture will continue the study of complexity. Examples on intractable problems, and how to show they are intractable, i.e. NP-complete, will be studied. Alternative approaches will be briefly introduced. **Lab** Students will continue experimenting with algorithms of different complexity. They will study the cost versus problem size quantitatively. Pre-work is not needed for this week.

12	30 Oct - 3 Nov	**Lecture: Advanced Topics** In this lecture, some advanced topics will be introduced, such as the zero-knowledge proof notion and its connection with cryptography, and quantum computing. **Lab** The lab will help the students to build a deeper understanding of the new notions introduced in the class. **Assignment 3 released on 3 November.**

Assessment

Please refer to the Policy and the Procedures on Assessment of Coursework Subjects.

There will be two practical assignments (one project, in two stages), and a theory assignment.

Assessment task 1: Practical Individual Project

Intent:

- To ensure a firm ground for learning the advanced topics of the subject
- To allow students to self-assess whether/how the subject suits their learning objectives (this task will be done before the census date)
- To guide students to start with the course project (see Task 2)

Objective(s): This assessment task addresses the following subject learning objectives (SLOs):

1, 2 and 4

This assessment task contributes to the development of the following Course Intended Learning Outcomes (CILOs):

C.1 and D.1

Type: Report

Groupwork: Individual

Weight: 30%

Task: Implement a parser realising the function of a simple digital calculator (or similar functionalities at the same level of technical challenge);

Write a short report on the implementation, linking to the concepts and methods learned in class.

Length: No particular length limits. Generally, the report would be about 4 pages.

Due: 11.59pm Friday 8 September 2023

Assessment task 2: Practical Group Project

Intent: To ensure students master the theory of computation and understand how it is linked with practical computation. The project will be based on assignment 1, but significantly more sophisticated. Completing this project involves reviewing, mastering and applying a large part of the scope covered by this subject. Students need to apply the techniques learned during Week 3-8.

Objective(s): This assessment task addresses the following subject learning objectives (SLOs):

1, 2, 3 and 4

This assessment task contributes to the development of the following Course Intended Learning Outcomes (CILOs):

C.1 and D.1

Type: Project

Groupwork: Group, group assessed

Weight: 30%

Task:

- Implement an interpreter or a compiler of a simple but practical programming language (or task of equivalent technical challenge that expands upon Assessment Task 1)
- Write a report on the implementation steps, technical problems encountered and solutions.

Length: No particular length limits. Generally, the report would be about 10 pages.

Due: 11.59pm Friday 13 October 2023

Assessment task 3: Theory Assignment

Intent: The theory assignment assesses the students' understanding of the theoretical concepts underpinning standard models of computation.

Objective(s): This assessment task addresses the following subject learning objectives (SLOs):

1, 2, 3 and 4

This assessment task contributes to the development of the following Course Intended Learning Outcomes (CILOs):

C.1 and D.1

Type: Project

Groupwork: Individual

Weight: 40%

Task: Demonstrate understanding of key theoretical concepts through developing solutions to long-form problems based on techniques and methods shown in the subject.

Due: 11.59pm Friday 17 November 2023

Assessment feedback

Formative feedback for assignment 1 and 2 will be given by assessing students’ reports by each criterion, including understanding of the problem, design of algorithm, implementation of the computer program and program output (criteria will be made specific in detailed assignment description). Assessment of assignment 2 will also take the individual role played in the teamwork into consideration. The feedback will be given within 2 weeks of the respective deadlines.

Minimum requirements

In order to pass the subject, a student must achieve a mark of 50% or more.

Recommended texts

Lewis, H. R., & Papadimitriou, C. H. (1997). *Elements of the theory of computation* (2nd ed.). Prentice-Hall.

Sipser, M. (2012). *Introduction to the theory of computation* (3rd ed.). Cengage Learning.

Arora & Barak (2009): Computational Complexity: A Modern Approach. Cambridge University Press.

Wigderson (2019): Mathematics and Computation: A Theory Revolutionizing Technology and Science. Princeton University Press.

Graduate attribute development

For a full list of the faculty's graduate attributes refer to the FEIT Graduate Attributes webpage.

For the contribution of subjects taken in the Bachelor of Engineering (Honours) or Master of Professional Engineering to the Engineers Australia Stage 1 Competencies, see the faculty's Graduate Attributes and the Engineers Australia Stage 1 Competencies webpage.

Assessment: faculty procedures and advice

Marking criteria

Marking criteria for each assessment task will be available on the Learning Management System: Canvas.

Extensions

When, due to extenuating circumstances, you are unable to submit or present an assessment task on time, please contact your subject coordinator before the assessment task is due to discuss an extension. Extensions may be granted up to a maximum of 5 days (120 hours). In all cases you should have extensions confirmed in writing.

Special consideration

If you believe your performance in an assessment item or exam has been adversely affected by circumstances beyond your control, such as a serious illness, loss or bereavement, hardship, trauma, or exceptional employment demands, you may be eligible to apply for Special Consideration.

Late penalty

For Graded subjects:

Work submitted late without an approved extension is subject to a late penalty of 10 per cent of the total available marks deducted per calendar day that the assessment is overdue (e.g. if an assignment is out of 40 marks, and is submitted (up to) 24 hours after the deadline without an extension, the student will have four marks deducted from their awarded mark). Work submitted after five calendar days is not accepted and a mark of zero is awarded.

For some assessment tasks a late penalty may not be appropriate – these are clearly indicated in the subject outline. Such assessments receive a mark of zero if not completed by/on the specified date. Examples include:

a. weekly online tests or laboratory work worth a small proportion of the subject mark, or
b. online quizzes where answers are released to students on completion, or
c. professional assessment tasks, where the intention is to create an authentic assessment that has an absolute submission date, or
d. take-home papers that are assessed during a defined time period, or
e. pass/fail assessment tasks.

For Pass/Fail subjects:

Work submitted late without an approved extension will only be assessed at the subject coordinator's discretion. Students who do not submit assessment tasks by the due dates may be referred to the Responsible Academic Officer under Student Rule 3.8.2, and a fail result may be recorded for the subject.

Request a review of a result

If you believe an error may have been made in the calculation of your result in an assessment task or the final result for the subject, it is possible to request a review of a result with the Subject Coordinator within five (5) working days of the date of release of the result.

Academic liaison officer

Academic liaison officers (ALOs) are academic staff in each faculty who assist students experiencing difficulties in their studies due to: disability and/or an ongoing health condition; carer responsibilities (e.g. being a primary carer for small children or a family member with a disability); and pregnancy.

ALOs are responsible for approving adjustments to assessment arrangements for students in these categories. Students who require adjustments due to disability and/or an ongoing health condition are requested to discuss their situation with an accessibility consultant at the Accessibility Service before speaking to the relevant ALO.

Statement about assessment procedures and advice

This subject outline must be read in conjunction with the Coursework Assessments Policy and the Courswork Assessments Procedure.

Statement on copyright

Please remember that teaching materials and resources provided to you at UTS are protected by copyright. You are not permitted to re-use those for any purposes (including commercial purposes, in kind benefit or gain) without permission of the copyright owner. Breaching copyright in relation to teaching materials and resources could lead to a legal action being brought against you.

Statement on plagiarism

Plagiarism and academic integrity

At UTS, plagiarism is defined in Rule 16.2.1(4) as: 'taking and using someone else's ideas or manner of expressing them and passing them off as ... [their] own by failing to give appropriate acknowledgement of the source to seek to gain an advantage by unfair means'.

The definition infers that if a source is appropriately referenced, the student's work will meet the required academic standard. Plagiarism is a literary or an intellectual theft and is unacceptable both academically and professionally. It can take a number of forms including but not limited to:

- copying any section of text, no matter how brief, from a book, journal, article or other written source without duly acknowledging the source
- copying any map, diagram, table or figure without duly acknowledging the source
- paraphrasing or otherwise using the ideas of another author without duly acknowledging the source
- re-using sections of verbatim text without using quote marks to indicate the text was copied from the source (even if a reference is given).

Other breaches of academic integrity that constitute cheating include but are not limited to:

- submitting work that is not a student's own, copying from another student, recycling another student's work, recycling previously submitted work, and working with another student in the same cohort in a manner that exceeds the boundaries of legitimate cooperation
- purchasing an assignment from a website and submitting it as original work
- requesting or paying someone else to write original work, such as an assignment, essay or computer program, and submitting it as original work.

Students who condone plagiarism and other breaches of academic integrity by allowing their work to be copied are also subject to student misconduct Rules.

Where proven, plagiarism and other breaches of misconduct are penalised in accordance with UTS Student Rules Section 16 – Student misconduct and appeals.

Avoiding plagiarism is one of the main reasons why the Faculty of Engineering and IT is insistent on the thorough and appropriate referencing of all written work. Students may seek assistance regarding appropriate referencing through UTS: HELPS.

Work submitted electronically may be subject to similarity detection software. Student work must be submitted in a format able to be assessed by the software (e.g. doc, pdf (text files), rtf, html).

Further information about avoiding plagiarism at UTS is available.

Retention of student work

The University reserves the right to retain the original or one copy of any work executed and/or submitted by a student as part of the course including, but not limited to, drawings, models, designs, plans and specifications, essays, programs, reports and theses, for any of the purposes designated in Student Rule 3.9.2. Such retention is not to affect any copyright or other intellectual property right that may exist in the student's work. Copies of student work may be retained for a period of up to five years for course accreditation purposes. Students are advised to contact their subject coordinator if they do not consent to the University retaining a copy of their work.

Statement on UTS email account

Email from the University to a student will only be sent to the student's UTS email address. Email sent from a student to the University must be sent from the student's UTS email address. University staff will not respond to email from any other email accounts for currently enrolled students.