Course program and reading list

Semester: 1 Year: 2021

School:   Efi Arazi School of Computer Science B.Sc

Computability and Complexity

Lecturer:
Dr. Reut Levi  reut.levi1@idc.ac.il

Tutors:
Mr. Yarin Shechter  yarin.shechter@post.idc.ac.il
Mr. Idan Brodet  idan.brodet@post.idc.ac.il

Teaching Assistant:
Mr. Yarin Shechter  yarin.shechter@post.idc.ac.il
Mr. Idan Brodet  idan.brodet@post.idc.ac.il
Mr. Yaniv Rubin  Yaniv.Rubin@post.idc.ac.il
Mr. Gal Sade  Gal.Sade01@post.idc.ac.il
Mr. Daniel Karalnik  daniel.karalnik@post.idc.ac.il

Course No.:  Course Type :  Weekly Hours :  Credit:
644  Lecture  3  4

Course Requirements :  Group Code :
Final Exam  211064402

Language:
Hebrew
Course Description

In this course we will learn how to reason precisely about computation and prove mathematical theorems about its capabilities and limitations. We will start by studying the theory of Computability, which is concerned with the rigorous definition of computational tasks and the analysis of automated procedures that may solve them (the course on automata and formal languages has already exposed you to weak but useful forms of such procedures). This will set the stage for the theory of Computational Complexity, whose goal is to examine what are the resources that are necessary for any algorithm to solve a given task. Topics covered in the course include Turing machines, universality, nondeterminism, the halting problem, recursive and recursively enumerable languages, space and time complexity, the classes P and NP, reducibility between decision problems, the Cook-Levin theorem and NP completeness. Videos from spring 2015 are available at http://www.faculty.idc.ac.il/smozes/complexity/index.html. Videos from 2019 should be available via Moodle.

Course Goals

The goal of the class is to introduce you to the basic concepts of the theory of computation, and to allow you to argue about the limits of computation, and computational hardness. The most important practical skill that the students are expected to pick up during this course is the ability to recognize and interpret computational intractability in case it is encountered (be it in work or in other occasions). Computation is not just a practical tool, but also a scientific tool. Generalizing from abstract models of computation, scientists have come to view many natural phenomena as akin to computational processes. Today, computational models underlie many research areas in Biology, Chemistry, Physics and Economics. Thus, beyond being useful in your day to day profession as programmers, understanding the limitations and capabilities of computation may enhance your ability to appreciate the complexity of the world around you.

The study of computation also raises (and, in some cases, answers) philosophically interesting questions such as: Are there well-defined problems that cannot be solved automatically? Is solving a problem more difficult than verifying a solution? Can “creativity” be automated? Does Randomness increase our computational capabilities? Those of you with philosophical inclinations may be able to appreciate these aspects of the course. Generally speaking, the course is fairly “technical” and in some cases the formalism may seem excessive. It is important to keep in mind that the formalism is merely a means to develop a solid conceptual understanding of various notions related to computation.

Prerequisites

Prerequisite:

643 – Automata And Formal Languages
1. The concept of universal models of computation (such as Turing machines), that capture our intuitive notion of computation and allow us to reason about the capabilities of computers in a technology-independent manner.

2. The existence of intrinsic limits to computation. Computational problems that cannot be solved by any algorithm whatsoever (undecidability), and problems that are solvable but require unreasonable computational resources (computational complexity).

3. The notion of nondeterminism and in particular the conceptual difference between finding a solution and verifying that a given solution is correct.

4. The representation of computational problems, and the distinction/relationships between decision and search problems.

5. The notion of a reduction between computational problems and its implications on the relative complexity of the problems.

6. Basic tools for dealing with intractability.

---

### Grading

Let \( n \) be the number of homework problem sets in this class (roughly one per week).

The Grades for the course will be based on the following weighting:

1. Average grade of best top \( n-2 \) problem sets: \( 15\% \)
2. Mid-term exam: \( 15\% \)
3. Final exam: \( 70\% \).

To be eligible to take the final exam, you are required to hand in \( n-4 \) homework problem sets.

To pass the course you must get at least 60/100 in the final (and of course an average of 60/100 overall).

A bonus of 5 points will be given to the grade of each problem set that was prepared with a word processor.

Note: To allow you flexibility you are automatically granted permission not to submit 2 problem sets during the semester. If you have a qualifying reason for not submitting a problem set (miluim, medical reason, etc.) you should submit the appropriate documents to the student secretariat (Mazkirut Hastudentim). These will not count towards the two problem sets you are allowed not to submit (e.g., if the student secretariat notifies me that you were on Miluim for a week during the semester then you will be excused from submitting 1+2=3 problem sets). **Students who choose to contact the course staff and ask for special permission not to submit a specific problem set will not receive the automatic permission to skip 2 problem sets.**

---

### Learning Outcomes
After completing this class students should be able to:

1. Express computational problems formally as the problem of deciding whether a given string belongs to a certain formal language.

2. Identify and classify problems according to their complexity class, and understand the implications of this classification to the ability to solve a problem.

3. Prove computational hardness of problems using reductions.

4. Suggest methods to overcome computational intractability when it is encountered.

Lecturer Office Hours
Office hours will be announced on the course webpage (Piazza).

Tutor Office Hours
Specific hours to be announced on the course website.

Teaching Assistant
Ms. Shuli Finely
Mr. Idan Brodet
Mr. Yarin Shechter

Additional Notes
Course website: We will use www.piazza.com to host the class website, where you can find notes, problem sets, a discussion group, and announcements.

The website can be found at http://piazza.com/idc.ac.il/spring2021/644/home.

You should have received an invitation to join the website to your IDC email account.

Moodle will only be used for electronically submitting problem sets and accessing Classboost videos.

It is always worth checking the web site for corrections and announcements before starting to work problem sets.

Occasionally students will find mistakes and omissions in the problems' statements after the homework has been posted on the site.

So it is worth checking for updates even after you have started working on the homework.
Homework collaboration policy: Students are encouraged to work together on homework problems. However, you must not take any written notes from your discussion with others. In writing the homework solution to be turned in, a student may not consult any notes or homework solutions prepared together with or by another student, nor any solutions to homework problems posted on the web or otherwise. In other words, you are required to write your homework from scratch on your own without having any other written material with you.

The assignment questions have been carefully selected for their pedagogical value and may be similar to questions on problem sets from past offerings of this course or courses at other universities. Using any preexisting solutions from these other sources is strictly prohibited. At the beginning of each submitted problem set students are required to write down the following collaboration statement:

I collaborated with (names of other students) / used material from (cite external sources of information). I wrote my solution on my own and did not use any written notes while writing my solution.

The collaboration statement will not affect your grade. However, no problem set will be given credit unless it has a collaboration statement.

---

Reading List

The course will closely follow:

1. Introduction to the Theory of Computation. M. Sipser

Other, more advanced, sources that you may find useful are:


For the curious, the following books can be found in the library:


2. The P=NP Question and Godel's Lost Letter. R. Lipton.