Computation Theory
Exercise Sheet 2

Alistair O'Brien

February 4, 2024

Before Attempting the Problems

The primary goal of this course is to introduce to you *formal notion of computability* and various computational models, such as *register machines*, *Turing machines*, and the *lambda calculus*. It is vitial that you master many of these topics for various other Tripos courses[1].

Please complete these exercises and submit your solutions **96 hours** before our scheduled supervisions to `ajo41@cam.ac.uk` as a **PDF** attachment. Feel free to choose the format of your answers (handwritten, typed, LaTeX'd) – just ensure that they're (mostly) legible[2].

Attempt to complete at least 75% of the set exercises – if you get stuck simply make a note of it in your answer and move onto the next question. We will be able to discuss all the solutions during the supervisions.

These supervisions will focus on both examinable and non-examinable material, the latter being for pedigogical reasons. Each week I will provide various *practical programming exercises* related to the course content. You are welcome to skip these questions, they are simply there for your benefit.

Some of the exercises are taken from Prof. Dawar's exercise sheets which you can find here: https://www.cl.cam.ac.uk/teaching/2324/CompTheory/exercise-sheet.pdf.

[1] In particular, Complexity Theory (IB), Semantics (IB), Denotational Semantics (II), Types (II), Hoare Logic and Model Checking (II), and Category Theory (II)

[2] Don't worry too much about this though, I can't handwrite anything legible either

Exercises

1. Show that decidable sets are closed under union, intersection, and complementation. Do all of these closure properties hold for undecidable languages?

2. A *reduction* $f : S_1 \to S_2$ on sets $S_1, S_2 \subseteq \mathbb{N}$ is a computable function $f : \mathbb{N} \rightharpoonup \mathbb{N}$ such that
$$\forall x \in \mathbb{N}. x \in S_1 \iff f(x) \in S_2.$$
For all $S_1, S_1 \subseteq \mathbb{N}$ and reduction $f : S_1 \to S_2$, show that
$$S_2 \text{ is deciable} \implies S_1 \text{ is decidable}$$
Is the converse, inverse, or contrapositive of this statement true?

3. Show that the set $\{(e, e') \in \mathbb{N}^2 : \phi_e = \phi_{e'}\}$ is undecidable.

4. Compare and contrast register machines with Turing machines: how do they keep track of state, how are programs represented, what form do machine configurations and computations take?

5. (*optional*) Familiarise yourself with the Chomsky hierarchy and explain the connection between regular expressions and Turing machines

6. Briefly describe of three Turing-complete models of computation not covered in the course.
[*hint*: pick some fun ones]

7. Show that the following functions are all primitive recursive. Make sure to give the final form of the function as a composition of primitive functions.
   (a) Truncated subtraction function, $\dot{-} : \mathbb{N}^2 \to \mathbb{N}$, defined by $x \dot{-} y \triangleq \begin{cases} x - y & \text{if } y \leq x \\ 0 & \text{otherwise} \end{cases}$
   (b) Exponentiation, $\exp : \mathbb{N}^2 \to \mathbb{N}$, where $\exp(x, y) = x^y$.
   (c) Conditional branch on zero, $ifzero : \mathbb{N}^3 \to \mathbb{N}$, where
   $$ifzero(x, y, z) \triangleq \begin{cases} y, & \text{if } x = 0 \\ z, & \text{otherwise} \end{cases}$$
   (d) Bounded summation: if $f : \mathbb{N}^{n+1} \to \mathbb{N}$ is primitive recursive, then so is $g : \mathbb{N}^{n+1} \to \mathbb{N}$ where
   $$g(\vec{x}, x) \triangleq \begin{cases} 0, & \text{if } x = 0 \\ f(\vec{x}, 0) & \text{if } x = 1 \\ f(\vec{x}, 0) + \cdots + f(\vec{x}, x + 1), & \text{if } x > 1 \end{cases}$$

8. Explain the motivation and intuition behind minimisation. How does it extend the set of functions computable using primitive recursion? Give three examples of computable partial functions that are not definable using primitive recursion, justifying your answer in each case.

9. Use minimisation to show that the following functions are partial recursive:

   (a) Binary maximum function $\max : \mathbb{N}^2 \to \mathbb{N}$.

   (b) Integer square root function $\sqrt{\cdot} : \mathbb{N} \rightharpoonup \mathbb{N}$ which is only defined if its argument is a perfect square.

10. (*optional*) Write a Turing machine simulator in OCaml. Using your simulator, implement the machine described on slide 64.