CST Part IB Lent 2024
Computation Theory
Exercise Sheet

Exercise 1. Show that the following arithmetic functions are all register machine computable.

(a) First projection function $p \in \mathbb{N}^2 \to \mathbb{N}$, where $p(x, y) \triangleq x$

(b) Constant function with value $n \in \mathbb{N}$, $c \in \mathbb{N} \to \mathbb{N}$, where $c(x) \triangleq n$

(c) Truncated subtraction function, $\_ \dot{-} \_ \in \mathbb{N}^2 \to \mathbb{N}$, where $x \dot{-} y \triangleq \begin{cases} x - y & \text{if } y \leq x \\ 0 & \text{if } y > x \end{cases}$

(d) Integer division function, $\_div\_ \in \mathbb{N}^2 \to \mathbb{N}$, where

$$x \, div \, y \triangleq \begin{cases} \textit{integer part of } x/y & \text{if } y > 0 \\ 0 & \text{if } y = 0 \end{cases}$$

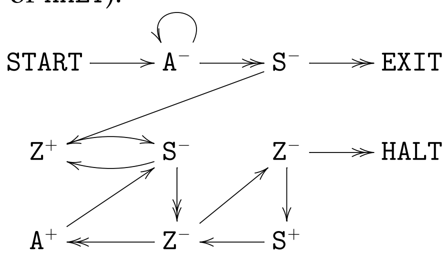
(e) Integer remainder function, $\_mod\_ \in \mathbb{N}^2 \to \mathbb{N}$, where $x \, mod \, y \triangleq x \dot{-} y(x \, div \, y)$

(f) Exponentiation base 2, $e \in \mathbb{N} \to \mathbb{N}$, where $e(x) \triangleq 2^x$.

(g) Logarithm base 2, $\log_2 \in \mathbb{N} \to \mathbb{N}$, where $\log_2(x) \triangleq \begin{cases} \textit{greatest } y \textit{ such that } 2^y \leq x & \text{if } x > 0 \\ 0 & \text{if } x = 0 \end{cases}$

Exercise 2. Let $\phi_e \in \mathbb{N} \rightharpoonup \mathbb{N}$ denote the unary partial function from numbers to numbers computed by the register machine with code e. Show that for any given register machine computable unary partial function $f \in \mathbb{N} \rightharpoonup \mathbb{N}$, there are infinitely many numbers e such that $\phi_e = f$. (Two partial functions are equal if they are equal as sets of ordered pairs; which is equivalent to saying that for all numbers $x \in \mathbb{N}$, $\phi_e(x)$ is defined if and only if $f(x)$ is, and in that case they are equal numbers.)

Exercise 3. Consider the list of register machine instructions whose graphical representation is shown below. Assuming that register `Z` holds 0 initially, describe what happens when the code is executed (both in terms of the effect on registers `A` and `S` and whether the code halts by jumping to the label `EXIT` or `HALT`).



Exercise 4. Show that there is a register machine computable partial function $f : \mathbb{N} \rightharpoonup \mathbb{N}$ such that both $\{x \in \mathbb{N} \mid f(x)\downarrow\}$ and $\{y \in \mathbb{N} \mid (\exists x \in \mathbb{N}) \; f(x) = y\}$ are register machine undecidable.

Exercise 5. Suppose S_1 and S_2 are subsets of $\mathbb{N}$. Suppose $f \in \mathbb{N} \to \mathbb{N}$ is register machine computable function satisfying: for all x in $\mathbb{N}$, x is an element of S_1 if and only if $f(x)$ is an element of S_2. Show that S_1 is register machine decidable if S_2 is.

Exercise 6. Show that the set of codes $\langle e, e' \rangle$ of pairs of numbers e and e' satisfying $\phi_e = \phi_{e'}$ is undecidable.

Exercise 7. For the example Turing machine given on slide 64, give the register machine program implementing $(S, T, D) := \delta(S, T)$, as described on slide 70. [Tedious!—maybe just do a bit.]

Exercise 8. Show that the following functions are all primitive recursive.

(a) Exponentiation, $exp \in \mathbb{N}^2 \to \mathbb{N}$, where $exp(x, y) \triangleq x^y$.

(b) Truncated subtraction, $minus \in \mathbb{N}^2 \to \mathbb{N}$, where $minus(x, y) \triangleq \begin{cases} x - y & \text{if } x \geq y \\ 0 & \text{if } x < y \end{cases}$

(c) Conditional branch on zero, $ifzero \in \mathbb{N}^3 \to \mathbb{N}$, where $ifzero(x, y, z) \triangleq \begin{cases} y & \text{if } x = 0 \\ z & \text{if } x > 0 \end{cases}$

(d) Bounded summation: if $f \in \mathbb{N}^{n+1} \to \mathbb{N}$ is primitive recursive, then so is $g \in \mathbb{N}^{n+1} \to \mathbb{N}$ where

$$g(\vec{x}, x) \triangleq \begin{cases} 0 & \text{if } x = 0 \\ f(\vec{x}, 0) & \text{if } x = 1 \\ f(\vec{x}, 0) + \cdots + f(\vec{x}, x - 1) & \text{if } x > 1. \end{cases}$$

Exercise 9. Recall the definition of Ackermann's function *ack* (slide 102). Sketch how to build a register machine M that computes $ack(x_1, x_2)$ in $R0$ when started with x_1 in $R1$ and x_2 in $R2$ and all other registers zero. [Hint: here's one way; the next question steers you another way to the computability of *ack*. Call a finite list $L = [(x_1, y_1, z_1), (x_2, y_2, z_2), \ldots]$ of triples of numbers *suitable* if it satisfies

(i) if $(0, y, z) \in L$, then $z = y + 1$

(ii) if $(x + 1, 0, z) \in L$, then $(x, 1, z) \in L$

(iii) if $(x + 1, y + 1, z) \in L$, then there is some u with $(x + 1, y, u) \in L$ and $(x, u, z) \in L$.

The idea is that if $(x, y, z) \in L$ and L is suitable then $z = ack(x, y)$ and L contains all the triples $(x', y', ack(x, y'))$ needed to calculate $ack(x, y)$. Show how to code lists of triples of numbers as numbers in such a way that we can (in principle, no need to do it explicitly!) build a register machine that recognises whether or not a number is the code for a *suitable* list of triples. Show how to use that machine to build a machine computing $ack(x, y)$ by searching for the code of a suitable list containing a triple with x and y in it's first two components.]

Exercise 10. For each $n \in \mathbb{N}$, let g_n be the function mapping mapping each $y \in \mathbb{N}$ to the value $ack(n, y)$ of Ackermann's function at $(n, y) \in \mathbb{N}^2$.

(a) Show for all $(n, y) \in \mathbb{N}^2$ that $g_{n+1}(y) = (g_n)^{(y+1)}(1)$, where $h^{(k)}(z)$ is the result of k repeated applications of the function h to initial argument z.

(b) Deduce that each g_n is a primitive recursive function.

(c) Deduce that Ackermann's function is total recursive.

Exercise 11. If you are *still* not fed up with Ackermann's function $ack \in \mathbb{N}^2 \rightarrow \mathbb{N}$, show that the λ-term $\mathsf{ack} \triangleq \lambda x.\, x\,(\lambda f\, y.\, y\, f\,(f\,\underline{1}))\,\mathsf{Succ}$ represents ack (where Succ is as on slide 123).

Exercise 12. Let I be the λ-term $\lambda x.\, x$. Show that $\underline{n}\mathsf{I} =_\beta \mathsf{I}$ holds for every Church numeral $\underline{n}$. Now consider

$$\mathsf{B} \triangleq \lambda f\, g\, x.\, g\, x\, \mathsf{I}\,(f\,(g\, x))$$

Assuming the fact about normal order reduction mentioned on slide 115, show that if partial functions $f, g \in \mathbb{N} \rightharpoonup \mathbb{N}$ are represented by closed λ-terms F and G respectively, then their composition $(f \circ g)(x) \equiv f(g(x))$ is represented by $\mathsf{B}\, F\, G$.