Computability Final Test — 2012-02-06

Notes.

- Write your name and matriculation number on each of your sheets.
- Solve no more than four (4) exercises. This will be strictly enforced: including more than 4 answers will result in the immediate failure of the test.
- Significantly wrong answers will result in negative scores.
- Always provide a justification for your answers.
- To achieve higher scores (≥ 27) you have to solve the exercise marked with $\star$ below.

Reminder: when equating results of partial functions (as in $\phi_i(3) = \phi_i(5)$), we mean that either 1) both sides of the equation are defined, and evaluate to the same natural number, or 2) both sides are undefined.

Exercise 1. *Comment on this statement by Mr. Rouge Hareng: is it correct?*

> *We proved that A is semantically closed. Let $f(x) = x$ and $g(x) = undefined$ for all x. It is then easy to check that $f \in \mathcal{F}_A$ and $g \notin \mathcal{F}_A$. Also, g is a finite restriction of f. So, by Rice-Shapiro ($\Rightarrow$), $A \notin \mathcal{RE}$.*

Solution (sketch). Everything is correct, but for the last step, which is wrong. In order to apply Rice-Shapiro ($\Rightarrow$), one has to consider *all* the possible finite restrictions of f, and not only the specific g Mr. Hareng chose. $\square$

Exercise 2. *Let $f \in (\mathbb{N}^2 \to \mathbb{N})$ be a recursive total function. Prove whether there exists a total recursive $g \in (\mathbb{N} \to \mathbb{N})$ such that $\mathsf{ran}(g) = \mathsf{ran}(f)$.*

Solution (sketch). Yes, take $g(n) = f(\mathsf{proj1}(n), \mathsf{proj2}(n))$. Then, $\mathsf{ran}(g) = \mathsf{ran}(f)$ because ... $\square$

Exercise 3. *Let $A = \{i \mid \mathsf{dom}(\phi_i)$ finite $\wedge\ \mathsf{ran}(\phi_i)$ finite$\}$ and $B = \{i \mid \mathsf{dom}(\phi_i)$ finite$\}$. Prove that $B \leq_m A$.*

Solution (sketch). Follows immediately from $A = B$, since a function with finite domain must also have a finite range. $\square$

Exercise 4. *Consider the sets $A = \{i \mid \forall x \in \{5, 6, 7\}.\ \phi_i(x) = 8\}$ and $B = \{i \mid \forall x \in \mathbb{N}.\ \phi_i(x) = 8\}$. Does $A \in \mathcal{RE}$? Does $B \in \mathcal{RE}$?*

Solution (sketch). $A = \{i \mid \phi_i(5) = 8 \ \wedge \ \phi_i(6) = 8 \ \wedge \ \phi_i(7) = 8\}$ so it is defined using a conjunction of three $\mathcal{RE}$ properties: indeed a semi-verifier for the first would be

$$S = \lambda i.\ \mathbf{Eq}(\mathbf{Eval1}\ i\ \ulcorner 5 \urcorner)\ulcorner 8 \urcorner$$

The other semi-verifiers only differ for the involved numeric constants. Hence, $A \in \mathcal{RE}$.

$B \notin \mathcal{RE}$ by Rice-Shapiro ($\Rightarrow$). B is semantically closed (because ...). $f(x) = 8$ is a recursive function in $\mathcal{F}_B = \{f\}$, but no finite restriction g of it can belong to $\mathcal{F}_B$, because otherwise $g = f$, implying that g is total, hence not finite. ☐

Exercise 5. *Let $A = \{i \mid \phi_i(3) = 5\}$, $B = \{i \mid \phi_i(5) = 30\}$. For each function f below, state whether it is a m-reduction for $A \leq_m B$. When it is such, prove it; otherwise, justify why it is not an m-reduction by providing at least an informal argument.*

$$f_1(n) = \#(\lambda x.\ \phi_n(3) + 25) \quad f_2(n) = \#(\lambda x.\ \phi_n(3) - 25) \quad f_3(n) = \#(\lambda x.\ \phi_n(5) - 25)$$

$$f_4(n) = \#\left(\lambda x.\ \begin{cases} 30 & \textit{if } \phi_n(3) = 5 \wedge x = 5 \\ 31 & \textit{otherwise} \end{cases}\right)$$

$$f_5(n) = \#\left(\lambda x.\ \begin{cases} 5 & \textit{if } \phi_n(5) = 30 \\ \textit{undefined} & \textit{otherwise} \end{cases}\right)$$

Solution (sketch). f_1 works: if $n \in A$, then $\phi_n(3) = 5$, hence $\phi_{h(n)}(5) = 5 + 25 = 30$, so $h(n) \in B$. Also, if $n \notin A$, then $\phi_n(3) = y \neq 5$ (y possibly undefined), hence $\phi_{h(n)}(5) = y + 25 \neq 30$, so $h(n) \notin B$.

f_2 does not work: if $n \in A$, then $\phi_n(3) = 5$, Hence $\phi_{h(n)}(5) = \phi_n(3) - 25 = 5 - 25 \neq 30$ and so $h(n) \notin B$ instead of $h(n) \in B$.

f_3 does not work: if $n \in A$, then $\phi_n(3) = 5$, but $\phi_n(5)$ is unconstrained (it could be anything, including undefined). Hence $\phi_{h(n)}(5) = \phi_n(5) - 25$ could be anything, and we can not conclude $h(n) \in B$.

f_4 is not well-defined. The body of the $\#(\lambda x.\ b(n, x))$ is not recursive. Indeed, if it were such, we could build a verifier for "$\phi_n(3) = 5$" by just computing $b(n, 5)$ and comparing it to 30. This would contradict the fact that "$\phi_n(3) = 5$" is not recursive, as one can prove using Rice.

f_5 is similar to f_3: $n \in A$ does not imply anything about $\phi_n(5)$. ☐

Exercise 6. *Show that $\bar{\mathsf{K}} \leq_m A = \{i \mid \forall x > 300.\ \phi_i(x) = x + \phi_i(x - 1)\}$*

Solution (sketch). Take

$$h(n) = \#\left(\lambda x.\ \phi_n(n)\right)$$

If $n \in \bar{\mathsf{K}}$, then $\phi_{h(n)}$ is always undefined, and since $\textit{undefined} = x + \textit{undefined}$ (for all x) we have $h(n) \in A$. Instead, if $n \in \mathsf{K}$, we have that $\phi_{h(n)}(x) = \phi_n(n) = y \neq \textit{undefined}$ for all x. In that case, $\phi_{h(n)}(301) = y \neq 301 + y = 301 + \phi_{h(n)}(300)$, so $h(n) \notin A$.

☐

Exercise 7. *Let $A = \{i \mid \exists x.\ \phi_i(x) > \phi_i(x+1)\}$. Show that $A \leq_m \mathsf{K}$ and that $\mathsf{K} \leq_m A$.*

Solution (sketch). $A \leq_m \mathsf{K}$ follows from $A \in \mathcal{RE}$. Indeed, A is defined via an existential quantification of the predicate $p(x,i) = \phi_i(x) > \phi_i(x+1)$ which is $\mathcal{RE}$ since it can be semi-verified by using a universal program to compute both sides and then comparing the results.

$\mathsf{K} \leq_m A$ is obtained e.g. using the reduction

$$h(n) = \#\left(\lambda x. \begin{cases} (1-x)^2 & \text{if } n \in \mathsf{K} \\ undefined & \text{otherwise} \end{cases}\right)$$

Indeed, the above is well-defined because ...(hence it is recursive and total), and is a reduction because ... □

Exercise 8. *State whether there exists a recursive bijection f between $\mathbb{N}$ and $P = \{p \in \mathbb{N} \mid p \text{ prime}\}$.*

Solution (sketch). Yes, $f(n) = p_n$ where p_n is the n-th prime number is computable, and is obviously a bijection. Justifying that f is recursive is a programming exercise. □

Exercise 9. *Given a total $f \in \mathcal{R}$, let*

$$A_f = \{i \mid \forall x \in \mathbb{N}.\ \phi_i(f(x)) = \phi_i(f(x+1))\}$$

Define, <u>when possible</u>, three total functions $f, g, h \in \mathcal{R}$ such that

$$A_f \in \mathcal{R} \qquad A_g \in \mathcal{RE} \setminus \mathcal{R} \qquad A_h \notin \mathcal{RE}$$

Solution (sketch).

- Using $f(x) = 0$ we have $A_f = \mathbb{N} \in \mathcal{R}$.
- Defining such g is *not* possible. Indeed, if g is a constant total function, we have $A_g = \mathbb{N} \in \mathcal{R}$. Otherwise, assume $g(a) \neq g(a+1)$ for some $a \in \mathbb{N}$. Then, we prove $A_g \notin \mathcal{RE}$ by establishing $\bar{\mathsf{K}} \leq_m A_g$. Indeed, the following $l(n)$ is a reduction:

$$l(n) = \#\left(\lambda y. \begin{cases} y & \text{if } n \in \mathsf{K} \\ undefined & \text{otherwise} \end{cases}\right)$$

  - If $n \in \bar{\mathsf{K}}$, then $\phi_{l(n)}(y) = undefined$ for all y, hence $\forall x.\phi_{l(n)}(g(x)) = undefined = \phi_{l(n)}(g(x+1))$, so $l(n) \in A_g$.
  - If $n \in \mathsf{K}$, then $\phi_{l(n)}(y) = y$ for all y, hence $\phi_{l(n)}(g(a)) = g(a) \neq g(a+1) = \phi_{l(n)}(g(a+1))$, hence $\neg\forall x.\phi_{l(n)}(g(x)) = \phi_{l(n)}(g(x+1))$, so $l(n) \notin A_g$.

- Any non-constant h works, by part (g) above.
  Alternative solution: take $h(x) = x$. In that case, $A_h = \{i \mid \forall x \in \mathbb{N}.\ \phi_i(x) = \phi_i(x+1)\}$ is not $\mathcal{RE}$. Indeed, A_h is semantically closed (it is defined only in terms of ϕ_i). The always undefined function $g(x) = \mathit{undefined}$ (which has a finite domain) belongs to $\mathcal{F}_{A_h}$, while the identity function $f(x) = x$ is a recursive extension of g that does not belong to $\mathcal{F}_{A_h}$, since otherwise we would get $x = x+1$ for all x. So, by Rice-Shapiro ($\Leftarrow$) $A_h \notin \mathcal{RE}$.

$\square$

Exercise 10. ⋆ *Recall the* WHILE *imperative programming language, which has the following syntax. Below, c are commands (statements), b are boolean expressions, e are arithmetic expressions, and x are variables.*

$$
\begin{array}{ll}
e ::= & 0 \mid 1 \mid x \mid e+e \mid e-e \mid e*e \\
b ::= & \mathsf{true} \mid \mathsf{false} \mid e=e \mid e \le e \mid e<e \mid e \ge e \mid e>e \mid b \wedge b \mid b \vee b \mid \neg b \\
c ::= & x := e \mid c \,;\, c \mid \mathsf{if}\ b\ \mathsf{then}\ c\ \mathsf{else}\ c \mid \mathsf{while}\ b\ \mathsf{do}\ c \mid \mathsf{for}\ x := e\ \mathsf{to}\ e\ \mathsf{do}\ c
\end{array}
$$

Now, restrict the syntax of WHILE *as follows. Name this restriction* W.

$$
\begin{array}{ll}
e ::= & \cdots \textit{(unchanged)} \cdots \qquad\qquad b ::= \cdots \textit{(unchanged)} \cdots \\
c ::= & \underline{x := x \mid x := e \,;\, c} \mid \mathsf{if}\ b\ \mathsf{then}\ c\ \mathsf{else}\ c \mid \mathsf{while}\ b\ \mathsf{do}\ c \mid \mathsf{for}\ x := e\ \mathsf{to}\ e\ \mathsf{do}\ c
\end{array}
$$

(Changes are underlined). The set W*-definable functions is then the set of functions implementable in* W*, using the inherited* WHILE *semantics.*

Question: state whether the set of W*-definable functions is larger, smaller, or equal to $\mathcal{R}$ (and justify your claim).*

Solution (sketch). Intentionally omitted. $\square$