CECS 329, Learning Outcome Assessment 11, April 27th, Spring 2023, Dr. Ebert

NO NOTES, BOOKS, ELECTRONIC DEVICES, OR INTERPERSONAL COMMUNICATION ALLOWED. Submit solutions to at most 2 LO problems on separate sheets of paper.

Problems

LO7. Answer and solve the following.

(a) When simulating a program P_x on input y via a universal program P_U, which of the following is a valid reason for why P_U must know the maximum register index used by P_x?

i. The decoding of any instruction of P_x directly depends on the maximum register index.

ii. The location of input y within a configuration directly depends on the maximum register index.

iii. **X** The location of the program counter within a configuration directly depends on the maximum register index.

iv. None of the above are valid reasons for why P_U must know the maximum register index used by P_x.

(b) A universal program P_U is simulating a program that has 116 instructions and whose Gödel number is

$$x = 2^{15} + 2^{46} + 2^{87} + 2^{109} + 2^{c_5} + \cdots + 2^{c_{116}} - 1.$$

If the current configuration of the computation of P_x on some input has encoding

$$\sigma = 2^5 + 2^{15} + 2^{17} + 2^{21} + 2^{24} - 1,$$

then provide the next configuration of the computation *and* its τ encoding.

Solution.

Current configuration: $\tau^{-1}(\sigma) = (5, 9, 1, 3, 2)$, Program counter equals 2, and $\beta(I_2) = 46 - 15 - 1 = 30$ which implies a `Transfer` instruction, since 30 mod $4 = 2$. Moreover, $28/4 = 7$, which yields $I_2 = T(4, 1)$. Finally, the next configuration is

$$(3, 9, 1, 3, 3),$$

and

$$\sigma_{\text{next}} = \tau(3, 9, 1, 3, 3) = 2^3 + 2^{13} + 2^{15} + 2^{19} + 2^{23} - 1.$$

LO8. Do the following.

(a) In one or more complete sentences, describe what is asserted by the Church-Turing Thesis.

Solution. Any [functional] deterministic step-by-step procedure that can be executed by a human using pencil and paper can be represented by a URM program.

(b) Consider the function

$$g(x) = \begin{cases} 1 & \text{if } P_x(x) \downarrow \\ 0 & \text{otherwise} \end{cases}$$

In other words, $g(x) = 1$ iff the program having Gödel number x halts when its own Gödel number x serves as input, and it outputs 0 otherwise. We want to prove that $g(x)$ is undecidable, meaning there is no URM program that computes g. To do this, let's assume that $g(x)$ is computable by some URM program G. Then define the function $f(x)$ as follows. Define function $f(x)$ as

$$f(x) = \begin{cases} 1 & \text{if } g(x) = 0 \\ \uparrow & \text{otherwise} \end{cases}$$

Provide an informal description of the steps you would take to compute $f(x)$ using pencil and paper. Why does this imply that $f(x)$ is URM computable?

Solution. On input x, first simulate program G on input x to obtain $g(x)$. If the output is 0, then return 1. Otherwise loop forever.

(c) Let e denote the Gödel number of the URM program F that computes $f(x)$ from part b. In other words $F = P_e$. Show that a contradiction arises when we try to compute $f(e)$ using F. Hint: consider the two cases when i) $f(e) = 1$ and ii) $f(e) =\uparrow$, meaning that $f(e)$ is undefined.

Solution. If $f(e) = 1$, then $g(e) = 0$ which means $F(e) = P_e(e) \uparrow$, in which case $f(e)$ would be undefined, a contradiction. On the other hand, if $f(e)$ is undefined, then $F(e) = P_e(e) \uparrow$, which means that $g(x) = 1$, in which case $F(e) = P_e(e) \downarrow$, a contradiction. Therefore, since the URM-computability of F depends on that of g, we must conclude that g is *not* URM computable.

LO9. An instance of the decision problem `One-to-One` is a Gödel number x, and the problem is to decide if function ϕ_x is a one-to-one function, meaning that, for every z in the range of ϕ_x, there is *exactly one* y for which $\phi_x(y) = z$. Consider the function

$$g(x) = \begin{cases} 1 & \text{if } \phi_x \text{ is one-to-one} \\ 0 & \text{otherwise} \end{cases}$$

(a) Evaluate $g(x)$ for each of the following Gödel number's x. Note: 2 out of 3 correct is considered passing. **Justify your answers**.

i. $x = e_1$, where e_1 is the Gödel number of the program that computes the function $\phi_{e_1}(y) = \text{sgn}(y)$. Hint: recall that $\text{sgn}(y)$ equals 1 if $y > 0$, and equals 0 otherwise.
Solution. $g(e_1) = 0$ since, e.g. $\text{sgn}(2) = \text{sgn}(3) = 1$ which means there is more than one input that produces the output 1.

ii. $x = e_2$, where e_2 is the Gödel number of the program that computes the function $\phi_{e_2}(y) = y^2$.
Solution. $g(e_2) = 1$ since the output of y^2 is produced by the unique natural number input y.

iii. $x = e_3$, where e_3 is the Gödel number of the program that computes $g(x)$ (assuming that $g(x)$ is URM computable).

Solution. $g(e_3) = 0$ since (similar to part a) g has infinitely many inputs, but only two outputs.

(b) Prove that $g(x)$ is not URM computable. In other words, there is no URM program that, on input x, always halts and either outputs 1 or 0, depending on whether or not ϕ_x is a one-to-one function. Do this by writing a program P that uses g and makes use of the `self` programming concept. Then show how P creates a contradiction.

Solution. Consider the following program P.

> Input $y \in \mathcal{N}$.
>
> If $g(\texttt{self}) = 1$, Return 0.
>
> Return y.

Let e denote P's Gödel number. If $g(e) = 1$, then it means P is one-to-one, but in this case the program returns 0 for each input y, which is a contradiction since the range of P is finite.

If $g(e) = 0$, then it means P is not one-to-one, but in this case the program returns y on each input y and y is the unique input that produces output y.

Therefore, g cannot be computable and hence the `One to One` decision problem is undecidable.

LO10. Answer the following.

(a) Provide the definition of what it means to be a mapping reduction from decision problem A to decision problem B.

(b) Is $f(n) = n^2 + 3n + 5$ a valid mapping reduction from the `Even` decision problem to the `Odd` decision problem? Justify your answer.

LO11. Recall the mapping reduction $f(\mathcal{C}) = (G, k)$, where f maps an instance of `3SAT` to an instance of the `Clique` decision problem. Given `3SAT` instance

$$\mathcal{C} = \{(\overline{x}_1, x_2, x_5), (\overline{x}_2, x_3, \overline{x}_4), (x_1, x_2, x_4), (\overline{x}_1, \overline{x}_3, \overline{x}_5)\},$$

answer the following questions about $f(\mathcal{C})$. Hint: to answer these questions you do *not* need to draw G.

(a) How many vertices does G have? Justify your answer.

(b) How many edges does G have? Hint: six different group pairs must be considered.

(c) Is (G, k) a positive instance of `Clique`? If yes, what size clique must if have? Justify your answer.