CECS 329, Exam 2 Spring 2024, Dr. Ebert

Directions: Solve AT MOST SIX problems. Closed Notes but you may use a non-programmable scientific calculator. Solve each problem on a separate sheet of paper. For example, if you decide to solve all six problems, then you should submit six pages, one for each problem. Make sure your name is on each page.

Unit-2 LO's

LO5. Solve the following.

a. What does it mean for a unary function $f(x)$ to be URM-computable? Explain. (5 pts)

b. Provide the instructions of a URM program that computes the function $f(x, y) = x \bmod y$. You may assume that $y \geq 1$. (15 pts)

c. Describe the role each register plays in computing $f(x)$. (5 pts)

LO6. Solve the following problems.

a. Provide the URM program P whose Gödel number equals

$$2^{30} + 2^{74} + 2^{112} + 2^{137} - 1.$$

Show all work. (10 pts)

b. Let P be a URM program that accepts two inputs x and y. What does it mean for P to be universal? (5 pts)

c. A universal program P_U is simulating a program that has 113 instructions and whose Gödel number is

$$x = 2^3 + 2^{179} + 2^{191} + 2^{196} + 2^{224} + 2^{268} + \cdots + 2^{c_{113}} - 1.$$

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

$$\sigma = 2^5 + 2^{14} + 2^{16} + 2^{23} + 2^{28} - 1,$$

then provide the next configuration of the computation *and* its encoding. (10 pts)

LO7. Answer the following. Note: correctly answering two of the three constitutes a pass.

a. Describe what it means to be a positive instance of the `Self Accept` decision problem. (5 pts)

b. In applying the diagonalization method towards proving the undecidability of `Self Accept`, we defined a computable function $g(x)$ in terms of $f(x)$, the decision function for `Self Accept`, which we assume to be total computable. Provide the formula for computing $g(x)$ and describe what needs to be established about $g(x)$ in order for the diagonalization method to be successful. (10 pts)

c. Suppose URM program P_{60} computes $\phi_{60}(x)$ and that

$$\phi_{60}(x) = \begin{cases} \lfloor x/3 \rfloor & \text{if } x \text{ is divisible by 3} \\ \uparrow & \text{otherwise} \end{cases}$$

What is the value of $g(60)$? Show that $g \neq \phi_{60}$. (10 pts)

LO8. An instance of `Self Output` is a Gödel number x and the problem is to decide if P_x outputs its own Gödel number x, i.e. there is an input y for which $P_x(y) = x$. Consider the function

$$g(x) = \begin{cases} 1 & \text{if } P_x \text{ outputs its own Gödel number } x \\ 0 & \text{otherwise} \end{cases}$$

a. Evaluate $g(16)$, $g(78)$, and $g(81)$, where (3 pts each)
   i. $\phi_{16}(y) = y^2$.
   ii. $\phi_{78}(y) = 2y$.
   iii. $\phi_{81}(y) = 5y$.

b. Prove that the function $g(x)$ is not URM computable. In other words, there is no URM program that, on input x, always halts and either outputs 1 or 0 as output, depending on whether or not P_x outputs its own Gödel number x. Do this by writing a program P that uses g and makes use of the `self` programming construct. (10 pts)

c. Use a proof by cases to show that your program P from part b behaves inconsistently with how g evaluates P's Gödel number. Conclude that g is not total computable and hence `Self Output` is undecidable. (6 pts)

Advanced Problems

A1. Solve/answer the following.

(a) Formally state Kleene's 2nd Recursion Theorem. (10 pts)

(b) Explain the main application of Kleene's 2nd Recursion Theorem in relation to computer programming. (5 pts)

(c) The proof of Kleene's 2nd Recursion Theorem involves defining a URM program $P = ABC$, where A, B, and C are subprograms that are concatenated together to form P. Explain the role played by each subprogram. Please include information on the effect each has on the machine's registers R_1 and R_2 (assuming the URM model of computation). (15 pts)

A2. Answer the following.

(a) Prove that there is a total computable function $k(x)$ for which $\phi_{k(x)}(y) = x$. (10 pts)

(b) Use the program encoding functions to provide a mathematical formula for the value of $k(x)$ on some input x. Hint: you may find useful the geometric series formula

$$1 + a + a^2 + \cdots + a^k = (a^{k+1} - 1)/(a - 1). \text{ (20 pts)}$$

A3. An instance of `Empty` is a Gödel number x and the problem is to decide if P_x has an empty domain, meaning that it never halts on any input. Alicia wants to use the diagonalization method the prove that `Empty` is undecidable. Letting d_{Empty} denote the decision function for `Empty`, she defines the antagonist function $g(x)$ as

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

(a) For a diagonalization proof to work, what must be achieved by $g(x)$? (10 pts)

(b) Does $g(x)$ succeed? If yes, provide a proof. Otherwise, provide an example where g fails. (15 pts)

A4. Consider the function $m(x)$ which, on input x, returns the least y for which P_y computes function ϕ_x. In other words, P_y is a minimal program for ϕ_x. Prove that $m(x)$ is not a computable function. Hint: assume $m(x)$ is computable and write a program that uses the self-programming construct to create a contradiction. (30 pts)

Unit-1 LO's

LO1. Solve the following.

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

(b) Let $S = \{7, 12, 15, 19, 21, 35, 47, 48\}$ be an instance of `Set Partition (SP)`. Provide $f(S)$, where $f : \texttt{SP} \rightarrow \texttt{SS}$ is the mapping reduction from `SP` to `Subset Sum` provided in lecture.

(c) Using the problem instances in part b, verify that f maps a positive instance of `SP` to a positive instance of `SS`.

LO2. An instance of `Composite` is a positive integer $n \geq 2$ and the problem is to decide if there is a number m, such that $2 \leq m < n$ and for which m divides evenly into n.

(a) For a given instance n of `Composite`, describe a certificate in relation to n.

(b) Provide a semi-formal verifier algorithm that takes as input i) an instance n of `Composite`, and ii) a certificate for n as defined in part a, and decides if the certificate is valid for n.

(c) Which is a better size parameter for instance n: n itself or parameter b that represents the number of bits needed to represent n? Explain.

LO3. Recall the mapping reduction $f(\mathcal{C}) = (S, t)$, where f maps an instance of `3SAT` to an instance of the `Subset Sum` decision problem. Given `3SAT` instance

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

answer the following questions about $f(\mathcal{C})$. Hint: to answer these questions you are *not* required to draw the table, but you might find it helpful.

(a) What is the value of t?

(b) How many numbers (counting repeats) are in S? What is the largest (in terms of numerical value) number in S?

(c) Determine a satisfying assignment for $\mathcal{C}$ and use it to identify a subset A of S that sums to t. List all the members of A. Hint: there are multiple possible answers, but the subset you choose must correspond with your chosen satisfying assignment.

LO4. Answer the following questions. Correctly answering at least two of the three is sufficient for passing LO4.

(a) Provide the definition of what it means for a decision problem to be NP-complete. (6 pts)

(b) Describe the three main steps that must be completed in order to establish that a decision problem L is a member of NP. Clearly define all technical terms. Hint: your three steps should make reference to two different technical terms that need defining.

(c) Provide the chain of mapping reductions that is needed to establish that the Vertex Cover (VC) decision problem is NP-complete, assuming that IS $\leq_m^p$ VC is one of the reductions, and the other reductions all appeared as examples in either the Mapping Reducibility or Computational Complexity lecture.