Homework 6. Due December 8th, 9:59PM.

CS181: Fall 2020

Guidelines:

- Upload your assignments to Gradescope by 9:59 PM.
- Follow the instructions mentioned on the course webpage for uploading to Gradescope very carefully (including starting each problem on a new page and matching the pages with the assignments); this makes it easy and smooth for everyone. As the guidelines are simple enough, bad uploads will not be graded.
- You may use results proved in class without proofs as long as you state them clearly.
- Most importantly, make sure you adhere to the policies for academic honesty set out on the course webpage. The policies will be enforced strictly. Homework is a stepping stone for exams; keep in mind that reasonable partial credit will be awarded and trying the problems will help you a lot for the exams.
- All problem numbers correspond to our text 'Introduction to Theory of Computation' by Boaz Barak. So, exercise a.b refers to Chapter a, exercise b.

Campuswire: If you are having trouble or are stuck on a problem, don't hesitate to ask on campuswire!

1. Exercise 10.1, Parts (2), (4), (5), (6). There is a typo in the problem: It should say "either prove that H is always context-free or give a counterexample of **context-free** F, G that would make H not context-free." [3 points]

   (You don't have to prove that your counterexamples work. When showing that H is context-free, it suffices to describe a grammar to do it - no need to prove that your grammar satisfies the required properties. Just being correct is good enough. For (2), if it makes things easier, you can design context-free $F, G : \{0,1,2\}^* \rightarrow \{0,1\}$ and you may use the fact that the language $L = \{0^n 1^n 2^n | n \geq 0\}$ is not context-free.)

   Part (2). H need not be context-free. Taking the and of two context-free functions is the same as taking the intersection of the two context-free languages. Let us give two context-free languages whose intersection is not context-free. Consider the language $L = \{0^m 1^n 2^n : m, n \geq 0\}$. Then, L is context-free: You can take the grammar G that is $S \rightarrow AB$, $A \rightarrow 0A|\varepsilon$, $B \rightarrow 1B2|\varepsilon$. Similarly, consider the language $L' = \{0^m 1^m 2^n : m, n \geq 0\}$. Then, L' is also context-free: You can take the grammar G' that is $S \rightarrow AB$, $A \rightarrow 0A1|\varepsilon$, $B \rightarrow 0B|\varepsilon$. Now, $L \cap L' = \{0^n 1^n 2^n | n \geq 0\}$ which is not context-free.

Part (4). Yes, H would also be context-free. You can do this by reversing every the order of the symbols of every rule (so a rule of the form $A \to A_1 A_2 \ldots A_k$ would be changed to $A \to A_k A_{k-1} \ldots A_2 A_1$).

Part (5). Yes, H would also be context-free. The question is equivalent to showing that if we have two context free languages L_1, L_2, then their concatenation $L = L_1 \circ L_2$ is also context-free. To show L is context-free construct a new grammar by duplicating the rules of L_1, L_2 (using different variable names for the two), adding a new start symbol S with a rule $S \to S_1 S_2$ where S_1 is the start symbol of grammar for L_1 and S_2 is the start symbol of grammar for L_2. The resulting grammar will generate the concatenation (as S_1 will lead to a string in L_1, and S_2 will lead to a string in L_2).

Part (6). No, H need not be context-free. For this, we can take F, G to be the constant function: $F(x) = G(x) = 1$ for all x. Here, F, G would be context-free (e.g., take the grammar $S \to 0S|1S|\varepsilon$). However, H as defined would correspond to the language $DOUBLE$ which we showed in class is not context-free.

2. Exercise 10.2. [1 point]

   Proof. Suppose the function was context-free. Then, by the pumping lemma, there exist some numbers p_0, p_1 for which the conclusions of the lemma hold. Now, take the string $s = 1^{2^{p_0}}$. We claim that s cannot be *pumped* as required. Take any partitioning of $s = axyzb$. Then, by condition (2) $|xz| > 0$, and by (3), $|xyz| \le p_1 \le p_0$. Therefore, the length of ax^2yz^2b satisfies $2^{p_0} = |s| < |ax^2yz^2b| \le |s| + p_1 \le 2^{p_0} + p_0$. Now, $2^{p_0} + p_0 < 2^{p_0+1}$. Therefore, we have that $2^{p_0} < |ax^2yz^2b| < 2^{p_0+1}$ which means the length of $|ax^2yz^2b$ cannot be a power of 2 which yields a contradiction to it being in the language (by condition (1) of the lemma). Therefore, our assumption that the language is context-free is false.

3. Give a quantified integer statement to express the following: [1 point]

   (a) "The only solution to $x^a - y^b = 1$ is $x = 3, a = 2, y = 2, b = 3$". You can assume that you have access to a QIS $Power(n)$ which is true if and only if n is a power of a natural number (i.e., $n = x^a$ for some $x, a \in \mathbb{N}$, and $a > 1$).
   You can take the statement

   $$\forall m, n \;\; (Power(m) \wedge Power(n) \wedge (m - n = 1)) \Rightarrow ((m = 9) \wedge (n = 8)).$$

   [Btw, this is called *Catalan's conjecture* which took 160 years to solve. There is a beautiful book devoted to just proving it assuming knowledge of a first course in algebra.]

   (b) "1729 is the smallest natural number that can be expressed as sum of two cubes in two different ways".
   Let us find a statement that $P(n)$ that is true only if n can be expressed as sum of two cubes in two different ways. Note that we have to rule out the two ways being just a rearrangement of one such way. We can take:

   $$P(n) \equiv \exists a, b, c, d (n = a \times a \times a + b \times b \times b) \wedge (n = c \times c \times c + d \times d \times d) \wedge \neg(a = c) \wedge \neg(a = d).$$

   Now, you can write down the desired statement as follows:

$$P(1729) \wedge (\forall n((n < 1729) \Rightarrow \neg P(n)))\,.$$

[The number 1729 is called a *Taxicab* number based on a nice anecdote about the above *fact* and how the famous mathematicial Ramanujan observed it in a casual conversation with G.H. Hardy from a hospital bed.]

4. Exercise 11.2. [1 point]

Part (a). We will show a reduction from NOTEMPTY. Recall its definition: for a TM M, $NOTEMPTY(M) = 1$ if there exists an input x such that $M(x) = 1$ and 0 otherwise. We showed in class that NOTEMPTY is uncomputable (follows from Rice's theorem - NOTEMPTY is a non-trivial semantic property).

Now, let us reduce NOTEMPTY to FINDPROOF. For a machine M, define a new TM V as follows: $V(x, w)$ - (a) Return $EVAL(M, w)$. That is, V ignores the first input and just runs M on w. Now, set $\mathcal{R}(M) = (V, 0)$. Now, it is easy to check that $NOTEMPTY(M) = FINDPROOF(\mathcal{R}(M))$. If there exists a w such that $M(w) = 1$, then $V(0, w) = 1$ so that $FINDPROOF(V, 0) = 1$. Else, both are 0.

This proves that FINDPROOF is uncomputable as NOTEMPTY is uncomputable.

Part (b). We will think of x as a TM and will take w as a number (which we will interpret as an upper bound on the number of steps x takes on input 0). Consider V defined as follows $V(x, w)$:

(a) Run x for at most w steps on 0. If it halts, output 1. If not, output 0.

Then, clearly $V(x, w)$ halts for all inputs x, w (so it is *effective*) as we only run for at most w steps.

We claim that for all x, $HALTONZERO(x) = FINDPROOF_V(x)$; this shows that $FINDPROOF_V$ is uncomputable as HALTONZERO is uncomputable. For, if $HALTONZERO(x) = 1$, then $FINDPROOF_V(x) = 1$ (there is some bound w on the number of steps x takes on 0, and for that bound w, we'll have $V(x, w) = 1$). Similarly, if $HALTONZERO(x) = 0$, then $FINDPROOF_V(x) = 0$ as there is no w for which $V(x, w) = 1$.