Practice Exam 3

CS181: Fall 2021

1 Problem

Let G be a CFG with variables V, and terminals Σ. Let L be its language. Which of the following are true?

1. If a string x is in the language L, then exists a way to derive x from the start symbol in at most $2|x|+1$ steps.

   Solution. False. The statement would be true if we were guaranteed that G is in Chomsky Normal Form (CNF), but we are not. Here are two counterexamples. Counterexample 1: $S \to A, A \to B, B \to C, C \to D, D \to 1$. Counterexample 2: $S \to 1AAAA, A \to \epsilon$. These are both valid grammars that accept only the string, 1, but deriving this string takes 5 steps. Note that we could change these counterexamples to make them require any number of steps. □

2. If G is in addition in Chomsky normal form, $x \in L$ and x is not the empty string, then there exists a derivation of x takes at most $2|x|-1$ steps.

   Solution. True. Every rule of type $A \to BC$ increments a number of symbols by 1, and rules of type $A \to a$ change one variable to one terminal. Hence, to get x, we need to add $|x|-1$ new symbols, and change all $|x|$ variables to terminals. In total, it's $2|x|-1$ steps for all derivations. □

3. If G is in addition in Chomsky normal form, $x \in L$ and x is not the empty string, then every derivation of x takes at least $2|x|-1$ steps.

   Solution. True. Observe that argument in the previous problem also shows that the derivation of x *requires* $2|x|-1$ steps. □

4. Call G loop-free if there do not exist $\alpha, \beta \in (\Sigma \cup V)^*$ such that $\alpha \Rightarrow_* \beta$ and $\beta \Rightarrow_* \alpha$. If G is in CNF, then G is loop-free.

   Solution. True. There is no way to get $\alpha \Rightarrow_* \beta$ if β has fewer symbols than α. Hence, if there is a loop $\alpha \Rightarrow_* \beta \Rightarrow_*$, the strings have the same length. But in this case, only rules of the form $A \to a$ may be applied. But this rules increase the number of terminals, so a loop is not possible. □

2 Problem

In the following, let $F, G : \{0,1\}^* \to \{0,1\}$ denote context-free functions with corresponding languages L, L' respectively. Which of the following are true?

1. The language $\{xy : x \in L, y^{Reverse} \in L\}$ is a context-free language.

   Solution. True. If L is CF, then so is $L^{Reverse} = \{x : x^{Reverse} \in L\}$. Indeed, we can have a grammar that is identical to that of L but have rules whose right hand sides are written in reversed order. Finally, the language in the problem is $L \circ L^{Reverse}$ which is also CF because we can combine the two grammars with rule $S \to S_1 S_2$. $\square$

2. The function $H(x) = F(x) + G(x) \mod 2$ is context-free.

   Solution. False. If this were CF, then a complement of any CF language would also be CF due to $\overline{L} = L \oplus \{0,1\}^*$. As we know, this is not true in general, so $H(x)$ is not necessarily CF. $\square$

3. The function $H(x) = F(x) \wedge G(x)$ is computable.

   Solution. True. Even though $H(x)$ is not CF, it is still computable. Because F and G are both computable, they have the corresponding Turing machines which we can run sequentially to compute H. $\square$

4. The function $H(x) = 1$ if and only if x can be written as $x = x_1 \circ x_2 \circ \ldots \circ x_t$ where for all $1 \le i \le t$, $x_i \in L$ or $x_i \in L'$. Then, H is context-free.

   Solution. True. Let S_1 and S_2 be the start symbols of L and L', respectively. Then, a grammar for H contains all rules and symbols of the two grammars as well as rule $S \to SS \mid S_1 \mid S_2$ where S is the new start symbol. $\square$

3 Problem

Let $F, G : \{0,1\}^* \to \{0,1\}$ be two functions and suppose that F reduces to G (in the sense defined in class: there exists a computable function $\mathcal{R}$ such that $F(x) = G(\mathcal{R}(x))$ for all x).

True or false: If G is context-free, then F is context-free. Explain your answer in one or two sentences.

Solution. False. Let F be any computable function that is not CF, e.g., $F(x) = 1$ if and only if $x = 0^n 1^n 0^n$, and G be any nontrivial function, e.g., $G(x) = 1$ if and only if x is nonempty. There is a reduction $\mathcal{R}$: a TM can easily compute $F(x)$ and map it to either 1 or ε. $\square$

4 Problem

Write down the definitions of the following properties of a proof system V for a language L:

1. Effectiveness

   Solution. For any $x \in \{0,1\}^*$ and $w \in \{0,1\}^*$ $V(x,w)$ halts and outputs either 0 or 1. □

2. Soundness

   Solution. If $x \notin L$, then for any w, $V(x,w) = 0$. □

3. Completeness

   Solution. If $x \in L$, then there exists w such that $V(x,w) = 1$. □

5 Problem

Suppose we are given a TM M. The following question is about the reduction we saw in class for mapping the computation of M to a QIS.

1. Describe how we mapped a configuration to an integer in class.

   Solution. This problem is not relevant to the material taught this year, as we did not go into detail about the proof of GFIT. □

2. Describe the arithmetic formula we designed in class for $LegalStep(m,n)$.

   Solution. This problem is not relevant to the material taught this year, as we did not go into detail about the proof of GFIT. □

6 Problem

Give a CFG that generates the following language: $L = \{x;y : |x| \neq |y|, x \neq \varepsilon, y \neq \varepsilon, x, y \in \{0,1\}^*\}$.

Solution. The following grammar generates L.

$$\begin{aligned}
S &\to TA \mid AT \\
T &\to BTB \mid B;B \\
A &\to AB \mid B \\
B &\to 0 \mid 1,
\end{aligned}$$

where S is a start symbol, T generates strings of the form $x;y$ where $|x| = |y|$, A generates nonempty strings, and B generates an arbitrary terminal. □

7 Problem

Prove or disprove that the following language is context-free: $L = \{0^m 10^n 10^{mn} | m \geq 1, n \geq 1\}$.

Solution. This language is not CF which we'll show with the pumping lemma.

Let $p > 0$ be given by the PL. Choose $x = 0^p 10^p 10^{p^2} \in L$. Suppose $x = abcde$ such that

1. $|bcd| \leq p$,
2. $|bd| > 0$,
3. $ab^i cd^i e \in L$ for every $i = 0, 1, 2, \ldots$.

Neither b nor d contains 1's. Otherwise, ab^2cd^2e would contain at least 3 ones, and would not be in L. Hence, b and d contain only zeros. Since $|bcd| \leq p$, b and d cannot contain zeros from all the three blocks. Consider the following cases:

1. b and d contain zeros only from the first two blocks. For ab^2cd^2e, the number of zeros in the first block is $p_1 \geq p$, the same holds for the second block, $p_2 \geq p$, and at least one p_1, p_2 is strictly greater than p. Hence, $p_1 p_2 > p^2 = p_3$ where p_3 is the number of zeros in the third block, so $ab^2cd^2e \notin L$.

2. b and d contain k_2 zeros from the second block, and k_3 zeros from the third block. Again, let p_1, p_2, p_3 be the numbers of zeros in each block of the string $x' = ab^2cd^2e$. If $k_2 = 0$ (i.e., b and d only contain zeros in from the third block), then $k_3 > 0$ and

$$p_1 p_2 = p^2 < p^2 + k_3 = p_3,$$

so $x' \notin L$. On the other hand, if $k_2 > 0$, then $k_3 < p$ and

$$p_1 p_2 \geq p(p+1) = p^2 + p > p^2 + k_3 = p_3,$$

so $x' \notin L$. In any case, $x' \notin L$ which contradicts the PL, so L is not CF.

□

8 Problem

Given a TM M, call a state of M useless if the TM doesn't ever move to that state no matter the input. Call a TM M wasteful if M has a useless state.

Define $W : \{0,1\}^* \to \{0,1\}$ as the function that takes as input a TM M and $W(M) = 1$ if M is wasteful and 0 else. (We assume as in class that all binary strings represent TMs.)

Prove that W is uncomputable. You can use any result proved in class or in the homework problems but point out what you are using.

Solution. Notice that we cannot use Rice's theorem because wastefulness is a property of a TM, not of a language: there may be two equivalent TMs, only one of which is wasteful. Therefore, we need to use reductions, so we will reduce $HALTONZERO$ to $\neg W$ ($\neg W$ is computable if and only if W is). Suppose, we are given a TM M, and we want to decide whether M halts on string 0. We construct a new TM $N = \mathcal{R}(M)$ as follows. N will have a special symbol ☺ in its working alphabet that is not used in the alphabet of M. Given input x, N runs $EVAL(M, 0)$. Add a new dummy state to N that is not used before nor while computing $EVAL$. After computing $EVAL$, N writes ☺ on the tape, and visits all the states of N in sequential order without moving the head or writing anything. The symbol ☺ is needed to indicate that N needs to just switch to the next state instead of doing actual computation (because some states of N are used for computing $EVAL$). After visiting the last state, N halts and outputs 1.

If M does not halt on 0, then $N(x)$ will loop when computing $EVAL$ and will never reach the dummy state no matter what its input x is. On the other hand, if M halts on 0, then $N(x)$ would visit all of its states after invoking $EVAL$. Therefore, M halts on 0 if and only if $\mathcal{R}(M) = N$ is not wasteful, so $HALTONZERO(M) = \neg W(\mathcal{R}(M))$. Therefore, $\neg W$ is uncomputable, and so is W. □

9 Problem

Let $HZ = \{M \in \{0,1\}^* : M \text{ halts on } 0\}$. (Here we view every binary string as corresponding to a TM just as we did in class.)

Prove that there exists an effective, sound, and complete proof system for HZ.

Solution. It suffices to define a verifier for the proof system and prove that it satisfies effectiveness, soundness, and completeness. Given a TM M, and a positive integer $t > 0$, the verifier V is defined as follows. Note that we view t as an attempted proof that M halts on 0.

$V(M, t)$:

1. Simulate $M(0)$ for t steps.
2. If $M(0)$ halts within t steps, then output 1. Otherwise output 0.

Effectiveness: Clearly, $V(M, t)$ halts for every M and $t > 0$.

Soundness: Suppose M does not halt on 0. Then, for any $t > 0$, $V(M, t)$ halts and outputs 0.

Completeness: Suppose M halts on 0. Then there is some $t^* > 0$ for which $M(0)$ halts within t^* steps. Thus, $V(M, t^*)$ halts and outputs 1. □

10 Problem

Consider the following Python program:

```
def Collatz(n):
    if n==1: return 1
    if n % 2 == 0:
            return Collatz(n/2)
    else:
            return Collatz(3*n+1)
```

Write a QIS for the statement "Collatz(n) halts for every $n > 0$".

Solution. This problem is out of the scope of the class this year. It requires the use of some tricks that were taught last year in the proof of GFIT. $\square$