Class Test 1

IIT Kharagpur, CSE Dept., Autumn 2022

CS41001: Theory of Computation — 5th September, 2022

Time = 1 hour — Total marks = 20

Answer all questions. Provide concise answers. State all assumptions you make.

1. Prove exactly one of the following statements. [6]

   (a) Every infinite regular set contains a subset that is not recursively enumerable.

   Solution: An infinite regular set is countable. But it has uncountably many subsets. Number of Turing machines is countable since each Turing machine can be encoded (uniquely) as a natural number. A subset of the set of all Turing machines corresponds to the set of all *r.e.* langauges, which is countable. Hence at least one of the subsets of a regular set must be non *r.e.* .

   (b) Prove that every infinite *r.e.* set contains an infinite recursive subset.

   Solution: We know that a set is recursive iff there exists an enumeration machine enumerating its strings in *lexicographic order*. (Here, lexicographic order of strings in Σ^* is an arrangement such that strings are in non-decreasing order of length and strings of the same length are in lexicographic order.)

   Let A be an infinite *r.e.* set over alphabet Σ and let $\mathcal{M}$ be an enumeration machine that enumerates A. Let $\mathcal{N}$ be an enumeration machine that simulates $\mathcal{M}$ and does the following whenever $\mathcal{M}$ enters the enumeration state:

   - Suppose x is the first string that $\mathcal{M}$ enumerates. Enumerate x and continue simulating $\mathcal{M}$, remembering x.
   - Repeat: if $\mathcal{M}$ enumerates a string y such that x precedes y in a lexicographic order of strings, then enumerate y; set $x \leftarrow y$ (i.e., replace x on the tape with y). Otherwise, ignore y and continue simulating $\mathcal{M}$.

   Observe that, for any string x, $\mathcal{M}$ always enumerates a string y that comes after x in the lexicographic order as A is infinite.

   The strings enumerated by $\mathcal{N}$ are in lexicographic order and therefore $L(\mathcal{N})$ is recursive.

2. Prove **or** disprove exactly one of the following. [6]

   (a) Is it decidable for a given TM $\mathcal{M}$ whether $L(\mathcal{M}) = L(\mathcal{M})^{\mathbf{R}}$. (For a set $A \subseteq \Sigma^*$, define $A^{\mathbf{R}} = \{w^{\mathbf{R}} \mid w \in A\}$ where $w^{\mathbf{R}}$ denotes w reversed.)

   Solution: Let $\mathsf{REV} = \{\mathcal{M} \mid \mathcal{L}(\mathcal{M}) = \mathcal{L}(\mathcal{M})^{\mathbf{R}}\}$. We know that REV is not recursive (undecidable) iff its complement is not, for otherwise by the fact that recursive sets are closed under complementation, both would be recursive. We show that $\neg\mathsf{REV}$ is undecidable via a reduction from HP. Let $(\mathcal{M}, x)$ be an instance of HP. Construct a TM $\mathcal{N}$ over Σ with $|\Sigma| > 1$ that on input y, does the following.

   - Run $\mathcal{M}$ on x.
   - If $\mathcal{M}$ halts and $y = a_1a_2$, then accept and halt. (Here $a_1, a_2 \in \Sigma$ and $a_1 \neq a_2$).
   - Reject otherwise.

We choose Σ of size > 1 for otherwise the problem is trivially decidable. Every language over the unary alphabet is closed under reversal. Now, if $\mathcal{M}$ does not halt on x, then $\mathcal{L}(\mathcal{N}) = \emptyset$ and trivially $\mathcal{L}(\mathcal{N}) = \mathcal{L}(\mathcal{N})^{\mathbf{R}}$. Otherwise, $\mathcal{L}(\mathcal{M}) = \{a_1a_2\}$. Since $(a_1a_2)^{\mathbf{R}} = a_2a_1 \notin \mathcal{L}(\mathcal{N})$, $\mathcal{L}(\mathcal{N}) \neq \mathcal{L}(\mathcal{N})^{\mathbf{R}}$. Hence REV is undecidable.

Alternate solution using Rice's theorem. Let P be a property on *r.e.* sets defined as

$$P(A) = \begin{cases} T & \text{if } A = A^{\mathbf{R}} \\ F & \text{otherwise} \end{cases}$$

Again, we consider languages over Σ of size > 1 for otherwise the problem is trivially decidable. Trivially, $P(\emptyset) = T$. Also $P(\{a_1a_2\}) = F$ for some set $a_1, a_2 \in \Sigma$ with $a_1 \neq a_2$ since $(a_1a_2)^{\mathbf{R}} = a_2a_1 \notin \mathcal{L}(\mathcal{N})$. We have exhibited two sets, one for which the P holds and the other for which it does not. It follows that P is a non-trivial property and hence undecidable, by Rice's theorem.

(b) Given CFG G, it is undecidable whether $L(G)$ is deterministic context-free.

Solution: This follows from Griebach's theorem. Define a property on CFLs $\mathcal{P}(L(G)) = \top$, if $L(G)$ is DCFL and $\perp$ otherwise. Every regular language is a DCFL and so tthe property is true for regular sets. The property is non-trivial – the language $\{0^n1^n2^n \mid n \geq 0\}$ is a CFL but not a DCFL. DCFLs are closed under quotienting (requires proof). Hence $\mathcal{P}$ is undecidable.

3. Given a context-free grammar G, show that it is undecidable whether

(a) $L(G) = L(G)L(G)$. (For a set A, $AA = \{xy \mid x, y \in A\}$, where xy denotes concatenation of x and y). 4

Solution: Let $\mathsf{CC} = \{G \mid L(G) = L(G)L(G)\}$. We show that $\neg\mathsf{HP} \leq_m \mathsf{CC}$. Given an instance $(\mathcal{M}, x)$ of $\neg\mathsf{HP}$, define G such that $L(G) = \neg\mathsf{VALCOMPS}_{\mathcal{M},x}$. Let Δ be the alphabet of $\mathsf{VALCOMPS}_{\mathcal{M},x}$. If $(\mathcal{M}, x) \in \neg\mathsf{HP}$, then there are no valid computation histories i.e., $\neg\mathsf{VALCOMPS}_{\mathcal{M},x} = \Delta^*$ and hence $L(G) = L(G)L(G) = \Delta^*$. If $(\mathcal{M}, x) \notin \neg\mathsf{HP}$, then $L(G) = \neg\mathsf{VALCOMPS}_{\mathcal{M},x} \neq \Delta^*$ but $L(G)L(G) = \Delta^*$ (this follows from the fact that you can always split a valid computation history into two parts that are themselves not valid histories). Therefore CC is undecidable.

(b) G is ambiguous. (A grammar G is ambiguous if there exists a string in $L(G)$ with two different derivations in G.) 4

Solution: Let $\mathsf{AMB} = \{G \mid G \text{ is an ambiguous CFG }\}$. We describe a reduction $\mathsf{PCP} \leq_{\mathsf{m}} \mathsf{AMB}$. Let $A = (w_1, w_2, \ldots, w_k)$ and $B = (x_1, x_2, \ldots, x_k)$ be an instance of PCP defined over alphabet Σ. Let $a_1, a_2, \ldots, a_k \notin \Sigma$ be k new distinct symbols and let $\Sigma' = \Sigma \cup \{a_1, \ldots, a_k\}$. Define a context-free grammar $G = (N = \{S, S_A, S_B\}, \Sigma', P, S)$ where P consists of the following productions:

$$S \to S_A \mid S_B,$$

$$S_A \to w_iS_Aa_i \mid w_ia_i \quad \text{for } 1 \leq i \leq k,$$

$$S_B \to x_iS_Ba_i \mid x_ia_i \text{ for } 1 \leq i \leq k.$$

We now show that $(A, B) \in \mathsf{PCP}$ iff G is ambiguous.

$(A, B) \in \mathsf{PCP} \implies \mathsf{G} \in \mathsf{AMB}$: If $(A, B) \in \mathsf{PCP}$, there exists a sequence $i_1, i_2, \ldots, i_n$ such that $w_{i_1}w_{i_2}\cdots w_{i_n} = x_{i_1}x_{i_2}\cdots x_{i_n} = y$. Then the string $ya_{i_n}a_{i_{n-1}}\cdots a_{i_1}$ has two derivations in G, namely

$$\begin{aligned} S \to S_A & \to w_{i_1} S_A a_{i_1} \to w_{i_1} w_{i_2} S_A a_{i_2} a_{i_1} & \to \cdots \to w_{i_1} w_{i_2} \cdots w_{i_n} a_{i_n} \cdots a_{i_2} a_{i_1} \\ & & = y a_{i_n} a_{i_{n-1}} \cdots a_{i_1} \\ S \to S_B & \to x_{i_1} S_B a_{i_1} \to x_{i_1} x_{i_2} S_B a_{i_2} a_{i_1} & \to \cdots \to x_{i_1} x_{i_2} \cdots x_{i_n} a_{i_n} \cdots a_{i_2} a_{i_1} \\ & & = y a_{i_n} a_{i_{n-1}} \cdots a_{i_1} \end{aligned}$$

$G \in$ AMB $\implies (A, B) \in$ PCP: Suppose that there are 2 derivations for a string y in G. y must end with a sequence of a_i's. One of the derivations must be via S_A and the other from S_B. Suppose the two derivations terminate at $w_{i_1} w_{i_2} \cdots w_{i_n} a_{i_n} \cdots a_{i_2} a_{i_1}$ and $x_{i_1} x_{i_2} \cdots x_{i_m} a_{i_m} \cdots a_{i_2} a_{i_1}$. Then we have $y = w_{i_1} w_{i_2} \cdots w_{i_n} a_{i_n} \cdots a_{i_2} a_{i_1} = x_{i_1} x_{i_2} \cdots x_{i_m} a_{i_m} \cdots a_{i_2} a_{i_1}$. The sequence of a_i's at the end should match, thus implying that $n = m$. It now follows that $w_{i_1} w_{i_2} \cdots w_{i_n} = x_{i_1} x_{i_2} \cdots x_{i_n}$ thus implying that $i_1, i_2, \ldots, i_n$ is a solution for the PCP instance.