INT201, Assignment 2

Question 1

Are context-free languages closed under complement? Prove it or provide a counter example. (20 Marks)

Are co-Turing-recognizable languages closed under concatenation? Prove it or provide a counter example. (20 Marks)

(40 Marks Total)

ANSWER:

(1) No. CFLs are not closed under complement.

Counter example: *Let* $L = \{w \in \{a, b, c\}^* \mid w \neq a^n b^n c^n \text{ for any } n \geq 0\}$.

L is a Context-Free Language because it can be constructed as the union of languages checking non-equal counts ($i \neq j$ or $j \neq k$) and invalid ordering. However, the complement $\overline{L} = \{a^n b^n c^n \mid n \geq 0\}$ *is well-known to be* ***not*** *context-free (provable via Pumping Lemma).*

Since L is CFL but $\overline{L}$ *is not, the class is not closed under complement.*

(2) Yes. co-Turing-Recognizable Languages are closed under concatenation.

Proof:

Let L_1, L_2 *be co-recognizable. Then* $A = \overline{L_1}$ *and* $B = \overline{L_2}$ *are recognizable. We need to show that* $\overline{L_1 L_2}$ *is recognizable.* $\overline{L_1 L_2}$ *is defined as:*

$$\overline{L_1 L_2} = \{w \mid \forall x, y \text{ s.t. } w = xy, x \in \overline{L_1} \text{ or } y \in \overline{L_2}\} \tag{1}$$

Suppose the recognizer for A is TM Q, and the recognizer for B is TM E. Now we we can construct a TM M for $\overline{L_1 L_2}$*:*

Given input w, consider all splits $w = xy$*, where* $x = w[0..i]$ *and* $y = w[i+1..|w|-1]$ *for* $i = 0, 1, \ldots, |w| - 1$*. For each split, run the Q on x and E on y in dovetailing fashion. If either one accepts, mark this split as covered. If eventually every split is covered, M accepts w, otherwise rejects.*

If $w \in \overline{L_1 L_2}$*, then for every split* $w = xy$ *we have* $x \in A$ *or* $y \in B$*. Therefore* *at least one recognizer halts and accepts for each split, so all splits become covered and the M accepts. If $w \notin \overline{L_1 L_2}$, which means $w \in L_1 L_2$, then there exists some split $w = xy$ with $x \in L_1$ and $y \in L_2$, which indicates that $x \notin A$ and $y \notin B$. For that split neither recognizer will accept, so the TM M will reject or loop forever.*

Finally, by definition, we can conclude that $\overline{L_1 L_2}$ is recognizable, so $L_1 L_2$ is co-recognizable.

Question 2

Is the language $\mathrm{A}_{\mathrm{CFG}} = \{\langle G\rangle | G$ is a CFG that only generates letter $a\}$ decidable? Prove your conclusion. (e.g., $S \to a|b$ is not valid and $S \to a|aa$ is not valid.)

(30 Marks Total)

ANSWER:

Yes. The language $A_{CFG} = \{\langle G\rangle \mid G$ is a CFG that only generates the letter $a\}$ is decidable. We construct a Turing machine M that decides A_{CFG} as follows:

On input $\langle G\rangle$, where G is a context-free grammar over an alphabet Σ:

1. ***Check that $a \in L(G)$.** Given a CFG G, we first convert it into an equivalent grammar G' in Chomsky Normal Form (CNF). In CNF, any derivation of a string w with length $|w| = n \geq 1$ requires exactly $2n - 1$ steps. Specifically, for the string "a" where $n = 1$, the derivation requires exactly 1 step. We can enumerate all derivations of this length. If any derivation generates "a", we accept this condition; otherwise, we reject.*

2. ***Check that G generates no string other than a.** Let $R = \Sigma^* \setminus \{a\}$. Since Context-Free Languages are closed under intersection with Regular Languages, we can construct a new CFG G'' such that $L(G'') = L(G) \cap L(D) = L(G) \cap R$. Then, we check whether $L(G'')$ is empty. If $L(G'')$ is not empty, it means G generates some invalid string $w \neq a$, so we reject. Otherwise, accept.*

Both steps are decidable, and M always halts. Therefore, A_{CFG} is decidable.

Question 3

T = $\{\langle M \rangle |$ M is a TM and $\forall w \in L(M), |w| \leq 50$ and M accepts w within 50 steps$\}$
Is the language T decidable? Prove your conclusion.
(30 Marks Total)

ANSWER:

No, the language is undecidable.

Proof: Assume that T is decidable. Then, there exists a TM R_T that decides T , where: $R_T(\langle M \rangle) = \begin{cases} accept, & if\ \langle M \rangle \in T \\ reject, & if\ \langle M \rangle \notin T \end{cases}$

We construct a new Turing Machine S to decide A_{TM} using R_T as a subroutine. S takes an input $\langle M, w \rangle$, where M is a TM and w is a string. Inside S, we construct a new Turing Machine M'. M' ignores its own input x and simulates M on w. If M accepts w, then M' accepts x.If M rejects or loops on w, M' loops or rejects x.

As a result, if M accepts w, M' accepts all inputs, which means $L(M') = \Sigma^*$. Since Σ^* contains strings with length > 50, the condition fails. $\implies \langle M' \rangle \notin T$. If M does not accept w then M' accepts nothing, which means $L(M') = \emptyset$. The condition is true for an empty set. $\implies \langle M' \rangle \in T$.

Now, the decider S uses R_T to check $\langle M' \rangle$:

- *If R_T rejects $\langle M' \rangle$, then $\langle M' \rangle \notin T$, which implies that M accepted w. Therefore, S accepts.*
- *If R_T accepts $\langle M' \rangle$, then $\langle M' \rangle \in T$, which implies that M did not accept w. Therefore, S rejects.*

Thus, S decides A_{TM}. Since A_{TM} is known to be undecidable, our assumption that T is decidable must be false. Therefore, T is undecidable.