INT201, Assignment 2 Overleaf Link

Submission Deadline Dec 16, 23:55 pm. 5% per day late penalty will be applied. Maximum 5 day delay. This assignment assesses learning outcomes: 3,4,5

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)

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)

Question 3

$\mathrm{T} = \{\langle M\rangle |$ 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)

Solution 1.1

CFLs are not closed under complement.

The counterexample:

$$\begin{cases} L_1 = \{a^n b^n c^m \mid n, m \geq 0\} \\ L_2 = \{a^m b^n c^n \mid n, m \geq 0\} \end{cases}$$

Proof:

$$L_1 \cap L_2 = \{a^k b^k c^k \mid k \geq 0\}$$

By the CFL pumping lemma:

$$L_1 \cap L_2 \text{ is not CFL.}$$

If CFLs were closed under complement, then by De Morgan's law:

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

$$L_1 \cap L_2 = \{a^k b^k c^k \mid k \geq 0\} \text{ should be CFL.}$$

Thus, CFLs are not closed under complement.

Solution 1.2

Co-Turing-recognizable languages are closed under concatenation.

Proof:

Let L_1 and L_2 be two co-Turing-recognizable languages. By definition, their complements $\overline{L_1}$ and $\overline{L_2}$ are Turing-recognizable. Let M_1 and M_2 be Turing machines that recognize $\overline{L_1}$ and $\overline{L_2}$, respectively.

We aim to show that the concatenation L_1L_2 is co-Turing-recognizable. This is equivalent to showing that its complement, $\overline{L_1L_2}$, is Turing-recognizable.

First, let us analyze the logical condition for a string w to be in $\overline{L_1L_2}$. By the definition of concatenation:

$$w \in L_1L_2 \iff \exists u, v \text{ such that } w = uv \text{ and } (u \in L_1 \wedge v \in L_2).$$

Taking the negation of this statement, we get:

$$w \in \overline{L_1L_2} \iff \forall u, v \text{ such that } w = uv, \neg(u \in L_1 \wedge v \in L_2).$$

Applying De Morgan's laws, this is equivalent to:

$$w \in \overline{L_1L_2} \iff \forall u, v \text{ such that } w = uv, (u \in \overline{L_1} \vee v \in \overline{L_2}).$$

Since the length of w is finite, there are finitely many ways (specifically $|w|+1$) to split w into substrings u and v. We can construct a Turing machine M' to recognize $\overline{L_1L_2}$

TM M' on input w:

1. Generate all possible partitions of w into two substrings u and v such that $w = uv$. Let these partitions be $(u_1, v_1), (u_2, v_2), \ldots, (u_k, v_k)$, where $k = |w| + 1$.
2. Simulate M_1 and M_2 in parallel on all partitions. specifically, for each partition i ($1 \leq i \leq k$), we check the condition:

   $$\text{Accept}_i \iff (M_1 \text{ accepts } u_i) \vee (M_2 \text{ accepts } v_i)$$

   This is possible because the class of Turing-recognizable languages is closed under union.

3. Monitor the simulation for all k partitions simultaneously.
4. Evaluate the results:
   - If eventually, for **every** partition i, the condition Accept_i becomes true (meaning for every split $w = uv$, either $u \in \overline{L_1}$ or $v \in \overline{L_2}$), then M **accepts**.
   - If there exists a partition where the condition is never met (because $u \in L_1$ and $v \in L_2$, causing M_1 and M_2 to loop or reject), then M will loop (which is consistent with recognizing, not deciding).

Since M recognizes $\overline{L_1 L_2}$, it follows that $L_1 L_2$ is co-Turing-recognizable.

Solution 2

To prove that A_{CFG} is decidable, we construct a Turing machine that decides:

$$A_{CFG} = \{\langle G\rangle \mid G \text{ is a CFG that only generates the letter a}\}.$$

TM M on input $\langle G\rangle$:

1. Construct DFA D_1, which recognizes the language $\{a\}^*$.
2. Construct DFA D_2, which recognizes the language $\Sigma^* \setminus \{a\}^*$ (the complement of the language recognized by D_1). Since regular languages are closed under complement, this is possible.
3. Construct CFG G_1 s.t. $L(G_1) = L(G) \cap L(D_2)$ (the intersection of a CFL and a regular language is still a CFL). $L(G_1)$ represents all strings generated by G that are not "a".
4. Run the CFG emptiness detection (E_{CFG}) on G_1.

- If $L(G_1) \neq \emptyset$, then G generates a string containing a symbol other than letter a. Hence $L(G) \neq \{a\}$, and M rejects.
- If $L(G_1) = \emptyset$, then all strings generated by G consist only of the letter a, and M accepts.

Since M always halts and correctly decides, A_{CFG} is decidable.

Solution 3

The language T is undecidable.

Proof by reduction from $HALT_{TM}$:

For a given TM M and input w, we construct a TM M_w s.t. M accepts w iff $M_w \notin T$.

We claim $M_w =$ "On input x:

1. If $|x| \leq 50$, reject.
2. If $|x| > 50$, run M on input w. If M accepts w, accept x."

- If M accepts w: In step 1.2, M_w will accept all strings x with $|x| > 50$. Since $L(M_w)$ contains strings with length > 50, the condition "$\forall s \in L(M_w), |s| \leq 50$" is violated. Therefore $M_w \notin T$.
- If M does not accept w: In step 1, M_w rejects all strings with $|x| \leq 50$. In step 2, M_w will not accept any x with $|x| > 50$ (it either rejects or loops), so $L(M_w) = \emptyset$. The condition becomes vacuously true, so $M_w \in T$.

Conclusion: If T were decidable by a decider R, we could decide $HALT_{TM}$ by constructing M_w and checking if R rejects $\langle M_w \rangle$. Since $HALT_{TM}$ is undecidable, this leads to a contradiction, T must also be undecidable.