
Problem Set 4 Solutions

Problem 1 Prove that the following language is co-recognizable:

$$A = \{ \langle G_1, G_2 \rangle : G_1, G_2 \text{ are CFGs and } L(G_1) = \overline{L(G_2)} \} .$$

Want: To show that \overline{A} is recognizable, meaning to construct a TM M that recognizes \overline{A} . This means that for all CFGs G_1, G_2 we want

- If $L(G_1) \neq \overline{L(G_2)}$ then $M(\langle G_1, G_2 \rangle)$ accepts
- If $L(G_1) = \overline{L(G_2)}$ then $M(\langle G_1, G_2 \rangle)$ rejects or loops.

Construction: Our program M is the following:

Program $M(\langle G_1, G_2 \rangle)$

Let Σ be the union of the terminal alphabets of G_1 and G_2

For all $w \in \Sigma^*$ do

$d_1(w) \leftarrow 0$; $d_2(w) \leftarrow 0$

If $[M_{\text{cfg}}(\langle G_1, w \rangle)$ accepts] then $d_1(w) \leftarrow 1$

If $[M_{\text{cfg}}(\langle G_2, w \rangle)$ accepts] then $d_2(w) \leftarrow 1$

If $d_1(w) = d_2(w)$ then accept

Correctness: We show that the two conditions in the **Want** above are met. Let G_1, G_2 be any CFGs. Then $L(G_1) \neq \overline{L(G_2)}$ iff there is some string w that is in one of these sets but not the other, in other words iff

$$\exists w \in \Sigma^* : \left[(w \in L(G_1) \text{ and } w \notin \overline{L(G_2)}) \text{ OR } (w \notin L(G_1) \text{ and } w \in \overline{L(G_2)}) \right] .$$

By definition of the complement of a language, the above happens iff

$$\exists w : \left[(w \in L(G_1) \text{ and } w \in L(G_2)) \text{ OR } (w \notin L(G_1) \text{ and } w \notin L(G_2)) \right] .$$

From the Crib Sheet we know that TM M_{cfg} decides language A_{CFG} . From the definition of this language and the definition of decidability we thus have $L(G_1) \neq \overline{L(G_2)}$ iff

$$\exists w : \left[(M_{\text{cfg}}(\langle G_1, w \rangle) \text{ accepts and } M_{\text{cfg}}(\langle G_2, w \rangle) \text{ accepts}) \text{ OR } \right. \\ \left. (M_{\text{cfg}}(\langle G_1, w \rangle) \text{ rejects and } M_{\text{cfg}}(\langle G_2, w \rangle) \text{ rejects}) \right] .$$

The for loop in our program M tries every $w \in \Sigma^*$ in turn, looking for one such that the above conditions are met. If $L(G_1) \neq \overline{L(G_2)}$ then a w satisfying the above conditions exists, and our loop

will eventually hit it, and at that point M will accept and halt. If $L(G_1) = \overline{L(G_2)}$ then there is no such w and so our machine, searching across the infinite set Σ^* , will loop. So the desired conditions are true.

Problem 2 Prove that the following language is undecidable:

$$A = \{ \langle G_1, G_2 \rangle : G_1, G_2 \text{ are CFGs and } L(G_1) = \overline{L(G_2)} \}.$$

The proof is by contradiction.

Assume: A is decidable, meaning there is a TM R that decides A . This means that for all CFGs G_1, G_2 we have

- If $L(G_1) = \overline{L(G_2)}$ then $R(\langle G_1, G_2 \rangle)$ accepts
- If $L(G_1) \neq \overline{L(G_2)}$ then $R(\langle G_1, G_2 \rangle)$ rejects.

Want: To show that ALL_{CFG} is decidable, meaning to construct a TM S such that for all CFGs $G = (V, \Sigma, R, S)$

- If $L(G) = \Sigma^*$ then $S(\langle G \rangle)$ accepts
- If $L(G) \neq \Sigma^*$ then $S(\langle G \rangle)$ rejects.

Why is this what we want? Because if we can do this then we have a contradiction to the fact from the crib sheet that ALL_{CFG} is undecidable, meaning our assumption above must have been false.

Construction: Our program S is the following:

Program $S(\langle G \rangle)$, where $G = (V, \Sigma, R, S)$

Let $G_1 = G$

Let G_2 be the grammer (V_2, Σ, R_2, S_2) where

- $V_2 = \{S_2\}$ and
- R_2 consists of the single rule $S_2 \rightarrow S_2$

If $R(\langle G_1, G_2 \rangle)$ accepts then accept else reject

Correctness: The key point is that $L(G_2) = \emptyset$, and hence $\overline{L(G_2)} = \Sigma^*$. Since $G_1 = G$, this means that

$$L(G) = \Sigma^* \text{ iff } L(G_1) = \overline{L(G_2)}. \quad (1)$$

Given this, we show that the two conditions in the **Want** above are met. For any CFG $G = (V, \Sigma, R, S)$ we have

$$\begin{aligned} L(G) = \Sigma^* &\Rightarrow L(G_1) = \overline{L(G_2)} \quad (\text{by Equation (1)}) \\ &\Rightarrow R(\langle G_1, G_2 \rangle) \text{ accepts} \quad (\text{by the first condition in } \mathbf{Assume}) \\ &\Rightarrow S(\langle G \rangle) \text{ accepts} \quad (\text{because } S(\langle G \rangle) \text{ accepts when } R(\langle G_1, G_2 \rangle) \text{ accepts}) \end{aligned}$$

$$\begin{aligned}
L(G) \neq \Sigma^* &\Rightarrow L(G_1) \neq \overline{L(G_2)} \quad (\text{by Equation (1)}) \\
&\Rightarrow R(\langle G_1, G_2 \rangle) \text{ rejects} \quad (\text{by the second condition in \textbf{Assume}}) \\
&\Rightarrow S(\langle G \rangle) \text{ rejects} \quad (\text{because } S(\langle G \rangle) \text{ rejects when } R(\langle G_1, G_2 \rangle) \text{ rejects}).
\end{aligned}$$

Problem 3 Prove that the following language is undecidable:

$$A = \{ \langle G_1, G_2, G \rangle : G_1, G_2, G \text{ are CFGs and } L(G) = L(G_1) \cap L(G_2) \} .$$

The proof is by contradiction.

Assume: A is decidable, meaning there is a TM R that decides A . This means that for all CFGs G_1, G_2, G_3 we have

- If $L(G_1) = L(G_2) \cap L(G_3)$ then $R(\langle G_1, G_2, G_3 \rangle)$ accepts
- If $L(G_1) \neq L(G_2) \cap L(G_3)$ then $R(\langle G_1, G_2, G_3 \rangle)$ rejects.

Want: To show that ALL_{CFG} is decidable, meaning to construct a TM S such that for all CFGs $G = (V, \Sigma, R, S)$

- If $L(G) = \Sigma^*$ then $S(\langle G \rangle)$ accepts
- If $L(G) \neq \Sigma^*$ then $S(\langle G \rangle)$ rejects.

Why is this what we want? Because if we can do this then we have a contradiction to the fact from the crib sheet that ALL_{CFG} is undecidable, meaning our assumption above must have been false.

Construction: Our program S is the following:

Program $S(\langle G \rangle)$, where $G = (V, \Sigma, R, S)$ and $\Sigma = \{\sigma_1, \dots, \sigma_n\}$

Let $G_1 = G$

Let G_2 be the grammer (V_2, Σ, R_2, S_2) where

- $V_2 = \{S_2\}$ and
- R_2 consists of the rules $S_2 \rightarrow \sigma_1 S_2 \mid \sigma_2 S_2 \mid \dots \mid \sigma_n S_2 \mid \varepsilon$

Let $G_3 = G_2$

If $R(\langle G_1, G_2, G_3 \rangle)$ accepts then accept else reject

Correctness: The key point is that $L(G_2) = L(G_3) = \Sigma^*$, and hence $L(G_2) \cap L(G_3) = \Sigma^*$. Since $G_1 = G$, this means that

$$L(G) = \Sigma^* \text{ iff } L(G_1) = L(G_2) \cap L(G_3) . \quad (2)$$

Given this, we show that the two conditions in the **Want** above are met. For any CFG $G = (V, \Sigma, R, S)$ we have

$$\begin{aligned}
L(G) = \Sigma^* &\Rightarrow L(G_1) = L(G_2) \cap L(G_3) \quad (\text{by Equation (2)}) \\
&\Rightarrow R(\langle G_1, G_2, G_3 \rangle) \text{ accepts} \quad (\text{by the first condition in \textbf{Assume}})
\end{aligned}$$

$$\begin{aligned}
&\Rightarrow S(\langle G \rangle) \text{ accepts} \quad (\text{because } S(\langle G \rangle) \text{ accepts when } R(\langle G_1, G_2, G_3 \rangle) \text{ accepts}) \\
L(G) \neq \Sigma^* &\Rightarrow L(G_1) \neq L(G_2) \cap L(G_3) \quad (\text{by Equation (2)}) \\
&\Rightarrow R(\langle G_1, G_2, G_3 \rangle) \text{ rejects} \quad (\text{by the second condition in } \mathbf{Assume}) \\
&\Rightarrow S(\langle G \rangle) \text{ rejects} \quad (\text{because } S(\langle G \rangle) \text{ rejects when } R(\langle G_1, G_2, G_3 \rangle) \text{ rejects}).
\end{aligned}$$

Problem 4 Prove that the following language is undecidable:

$$A = \{ \langle M_1, M_2 \rangle : M_1, M_2 \text{ are TMs and } M_1(\varepsilon) \text{ halts and } M_2(\varepsilon) \text{ loops} \}.$$

The proof is by contradiction.

Assume: A is decidable, meaning there is a TM R that decides A . This means that for all TMs M_1, M_2 we have

- If $M_1(\varepsilon)$ halts and $M_2(\varepsilon)$ loops then $R(\langle M_1, M_2 \rangle)$ accepts
- If $M_1(\varepsilon)$ loops or $M_2(\varepsilon)$ halts then $R(\langle M_1, M_2 \rangle)$ rejects.

Want: To show that $HALT_{TM}$ is decidable, meaning to construct a TM S such that for all TMs M and all strings w over the input alphabet of M

- If $M(w)$ halts then $S(\langle M, w \rangle)$ accepts
- If $M(w)$ loops then $S(\langle M, w \rangle)$ rejects.

Why is this what we want? Because if we can do this then we have a contradiction to the fact from the crib sheet that $HALT_{TM}$ is undecidable, meaning our assumption above must have been false.

Construction: Here is the program S :

Program $S(\langle M, w \rangle)$

Let M_1 be the TM whose description is: “On input x , run $M(w)$. If it halts, accept.”

Let M_2 be the TM whose description is: “On input x , loop.”

If $R(\langle M_1, M_2 \rangle)$ accepts then accept else reject

Correctness: We first consider the machine M_1 whose code is defined above. If given input $x = \varepsilon$, it runs $M(w)$ and accepts (and thus halts) iff $M(w)$ halts. On the other hand, M_2 is defined to be a machine that, regardless of its input, does not halt. Thus the following conditions are true:

$$M_1(\varepsilon) \text{ halts iff } M(w) \text{ halts} \tag{3}$$

$$M_2(\varepsilon) \text{ loops} \tag{4}$$

Given this, we show that the two conditions in the **Want** above are met. For any TM M and string w over the input alphabet of M we have

$$M(w) \text{ halts} \Rightarrow M_1(\varepsilon) \text{ halts and } M_2(\varepsilon) \text{ loops} \quad (\text{by Equations (3) and (4)})$$

$\Rightarrow R(\langle M_1, M_2 \rangle)$ accepts (by the first condition in **Assume**)

$\Rightarrow S(\langle M, w \rangle)$ accepts (because $S(\langle M, w \rangle)$ accepts when $R(\langle M_1, M_2 \rangle)$ accepts)

$M(w)$ loops $\Rightarrow M_1(\varepsilon)$ loops and $M_2(\varepsilon)$ loops (by Equations (3) and (4))

$\Rightarrow M_1(\varepsilon)$ loops or $M_2(\varepsilon)$ halts (because $M_1(\varepsilon)$ loops)

$\Rightarrow R(\langle M_1, M_2 \rangle)$ rejects (by the second condition in **Assume**)

$\Rightarrow S(\langle M, w \rangle)$ rejects (because $S(\langle M, w \rangle)$ rejects when $R(\langle M_1, M_2 \rangle)$ rejects) .
