

Tutorial 2

UNDECIDABLE PROBLEMS ABOUT CFLs, PCP

Guidelines: Solve all problems in the class. *Do not search for solutions online.*

1. Define a set $\text{VALC-R}_{\mathcal{M},x}$ to be the set of all strings of the form $\# \alpha_0 \# \alpha_1^{\mathbf{R}} \# \alpha_2 \# \alpha_3^{\mathbf{R}} \# \cdots \# \alpha'_N \#$, where $\alpha'_N = \alpha_N^{\mathbf{R}}$ if N is odd and $\alpha'_N = \alpha_N$ otherwise, where $\# \alpha_0 \# \alpha_1 \# \cdots \# \alpha_N \#$ is a valid computation history of \mathcal{M} on input x . That is

$$\# \alpha_0 \# \alpha_1 \# \cdots \# \alpha_N \# \in \text{VALCOMPS}_{\mathcal{M},x} \Leftrightarrow \# \alpha_0 \# \alpha_1^{\mathbf{R}} \# \alpha_2 \# \alpha_3^{\mathbf{R}} \# \cdots \# \alpha'_N \# \in \text{VALC-R}_{\mathcal{M},x}.$$

Show that $\text{VALC-R}_{\mathcal{M},x}$ can be expressed as the intersection of two context-free languages.

Hint: Consider two possibilities for i – odd and even.

Solution: Let

$$L_1 = \{ \# \alpha_0 \# \alpha_1^{\mathbf{R}} \# \alpha_2 \# \alpha_3^{\mathbf{R}} \# \cdots \# \alpha'_N \# \mid \alpha_i \xrightarrow[\mathcal{M}]{1} \alpha_{i+1} \text{ for odd } i \}$$

and

$$L_2 = \{ \# \alpha_0 \# \alpha_1^{\mathbf{R}} \# \alpha_2 \# \alpha_3^{\mathbf{R}} \# \cdots \# \alpha'_N \# \mid \alpha_i \xrightarrow[\mathcal{M}]{1} \alpha_{i+1} \text{ for even } i \}.$$

Clearly, $L_1 \cap L_2$ consists of strings of the form $\# \alpha_0 \# \alpha_1^{\mathbf{R}} \# \alpha_2 \# \alpha_3^{\mathbf{R}} \# \cdots \# \alpha'_N \#$ with both conditions $\alpha_i \xrightarrow[\mathcal{M}]{1} \alpha_{i+1}$ for odd i and $\alpha_i \xrightarrow[\mathcal{M}]{1} \alpha_{i+1}$ for even i , being satisfied. That is, $\alpha_0 \xrightarrow[\mathcal{M}]{1} \alpha_1 \xrightarrow[\mathcal{M}]{1} \alpha_2 \xrightarrow[\mathcal{M}]{1} \cdots \xrightarrow[\mathcal{M}]{1} \alpha_N$ and so we have $\# \alpha_0 \# \alpha_1 \# \alpha_2 \# \alpha_3 \# \cdots \# \alpha_N \# \in \text{VALCOMPS}_{\mathcal{M},x}$ or equivalently $\# \alpha_0 \# \alpha_1^{\mathbf{R}} \# \alpha_2 \# \alpha_3^{\mathbf{R}} \# \cdots \# \alpha'_N \# \in \text{VALC-R}_{\mathcal{M},x}$ implying that $L_1 \cap L_2 = \text{VALC-R}_{\mathcal{M},x}$. We now build NPDAs $\mathcal{N}_1, \mathcal{N}_2$ accepting L_1, L_2 respectively, thus showing that L_1, L_2 are CFLs.

\mathcal{N}_1 , on input a string $\# \alpha_0 \# \alpha_1^{\mathbf{R}} \# \alpha_2 \# \alpha_3^{\mathbf{R}} \# \cdots \# \alpha'_N \#$ does the following.

1. Go to the next odd position i , scanning (and ignoring) the input string until the $\#$ symbol just before $\alpha_i^{\mathbf{R}}$.
2. While reading $\alpha_i^{\mathbf{R}}$, examine δ to determine (reverse of) the next configuration of \mathcal{M} following α_i . (This can be done using constant amount of memory in the finite control as α_i would differ from its next configuration in atmost three positions. Moreover, \mathcal{M} is deterministic and hence there is atmost one possibility for the next configuration.) Let $\bar{\alpha}_{i+1}$ denote the subsequent configuration. Push $\bar{\alpha}_{i+1}^{\mathbf{R}}$ on the stack. This can be done as $\bar{\alpha}_{i+1}$ is determined in the reverse order, when corresponding symbols in $\alpha_i^{\mathbf{R}}$ are read from the tape.
3. Start reading α_{i+1} – pop the stack one symbol at a time for each input symbol read in order to verify that $\alpha_{i+1} = \bar{\alpha}_{i+1}$.
4. Reject and if there is a mismatch.
5. If end of string is reached, accept. Otherwise, go back to step 1.

The construction of \mathcal{N}_2 is similar. □

2. Prove that it is undecidable whether

- (a) the intersection of two given CFLs is empty.

Hint: Think VALCOMPS or VALC-R.

Solution: Let $\text{EI-CFL} = \{G_1, G_2 \mid G_1, G_2 \text{ are CFGs and } L(G_1) \cap L(G_2) = \emptyset\}$. We show a reduction from $\neg\text{HP}$ to EI-CFL. Let \mathcal{M}, x be an instance of $\neg\text{HP}$. Construct CFGs G_1, G_2 such that $L(G_1) \cap L(G_2) = \text{VALC-R}_{\mathcal{M}, x}$. If \mathcal{M} does not halt on x , then $\text{VALC-R}_{\mathcal{M}, x} = \emptyset$ and hence $L(G_1) \cap L(G_2) = \emptyset$. If \mathcal{M} halts on x , then $\text{VALC-R}_{\mathcal{M}, x} = L(G_1) \cap L(G_2) \neq \emptyset$. Since $\neg\text{HP}$ is undecidable, EI-CFL is also undecidable.

- (b) the intersection of two given CFLs is a CFL.

Hint: Let $\text{VALCOMPS}_{\mathcal{M}, x}^t$ be the set of valid computation histories of \mathcal{M} on input x ending in accepting configurations. If \mathcal{M} is a TM making at least 3 moves, then for any x , $\text{VALCOMPS}_{\mathcal{M}}^t = \cup_{x \in \Sigma^*} \text{VALCOMPS}_{\mathcal{M}, x}^t$ is a CFL if and only if $\mathcal{L}(\mathcal{M})$ is finite. Use this fact.

Solution: Let $\text{I-CFL} = \{(G_1, G_2) \mid G_1, G_2 \text{ are CFGs and } L(G_1) \cap L(G_2) \text{ is a CFL}\}$. Recall the set $\text{FIN} = \{\mathcal{M} \mid \mathcal{L}(\mathcal{M}) \text{ is finite}\}$. We have seen that FIN is not r.e. and hence not decidable. We show a reduction $\text{FIN} \leq_m \text{I-CFL}$. Given a TM \mathcal{M} , modify it in a way that it makes at least 3 moves on every input, without changing the language \mathcal{M} accepts. This can be done by just adding 2 extra states, say, after the start state moving one cell back and forth. Construct CFGs G_1, G_2 such that $L(G_1) \cap L(G_2) = \text{VALCOMPS}_{\mathcal{M}}^t$. Now, $(G_1, G_2) \in \text{I-CFL}$ iff $\text{VALCOMPS}_{\mathcal{M}}^t$ is a CFL iff $\mathcal{L}(\mathcal{M})$ is finite iff $\mathcal{M} \in \text{FIN}$. Since FIN is undecidable, so is I-CFL.

- (c) the complement of a given CFL is a CFL.

Solution: Let $\text{COMP-CFL} = \{G \mid \neg G \text{ is a CFG and } L(G) \text{ is a CFL}\}$. As in the previous problem, we can show $\text{FIN} \leq_m \text{COMP-CFL}$. Given an instance \mathcal{M} of FIN, construct a CFG G such that $L(G) = \neg \text{VALCOMPS}_{\mathcal{M}}^t$. (Arguments similar to those used to show $\neg \text{VALCOMPS}_{\mathcal{M}, x}$ is context-free, can be used to show context-freeness of $\neg \text{VALCOMPS}_{\mathcal{M}}^t$). Now, $\neg L(G) = \text{VALCOMPS}_{\mathcal{M}}^t$ is a CFL iff $\mathcal{L}(\mathcal{M})$ is finite, thus implying that COMP-CFL is undecidable.

3. Consider a silly variant of PCP called SPCP where corresponding strings in both lists are restricted to have the same length. Show that this variant is decidable.

Solution: Let $A = \{w_1, \dots, w_n\}$ and $B = \{x_1, \dots, x_n\}$ denote an instance of SPCP. If there exists a solution, then there is an index $j \in [1, n]$ such that the solution starts with w_j, x_j . Let $|w_j| = |x_j| = \ell$. Since there is a match in the first ℓ positions, it must be the case that $w_j = x_j$. Also, if there is an index j such that $w_j = x_j$, then j is a solution to the SPCP instance (A, B) . Therefore, SPCP can be decided by just checking whether for each $j \in [1, n]$, $w_j = x_j$.

4. Prove that $\{G_1, G_2 \mid G_1, G_2 \text{ are CFGs and } \mathcal{L}(G_1) \cap \mathcal{L}(G_2) \neq \emptyset\}$ is undecidable via a reduction from PCP.

Solution: Let $A = \{w_1, \dots, w_k\}$ and $B = \{x_1, \dots, x_k\}$ denote an instance of PCP over alphabet Σ . Let $\Sigma' = \Sigma \cup \{a_1, \dots, a_k\}$ for some new symbols $a_1, \dots, a_k \notin \Sigma$. Define two CFGs $G_A = (\{S_A\}, \Sigma', P_A, S_A)$ and $G_B = (\{S_B\}, \Sigma', P_B, S_B)$ where P_A consists of the productions

$$S_A \rightarrow w_i S_A a_i \mid w_i a_i \text{ for } 1 \leq i \leq k,$$

and P_B consists of

$$S_B \rightarrow x_i S_B a_i \mid x_i a_i \text{ for } 1 \leq i \leq k.$$

Suppose the PCP instance (A, B) has a solution i_1, \dots, i_m . Let $y = w_{i_1} \cdots w_{i_m} = x_{i_1} \cdots x_{i_m}$. Then $ya_{i_m} \cdots a_{i_1} \in \mathcal{L}(G_A)$ and $ya_{i_m} \cdots a_{i_1} \in \mathcal{L}(G_B)$. As a result $\mathcal{L}(G_A) \cap \mathcal{L}(G_B) \neq \emptyset$.

Now, suppose that $\mathcal{L}(G_A) \cap \mathcal{L}(G_B) \neq \emptyset$. Then there is a string $ya_{i_m} \cdots a_{i_1} \in \mathcal{L}(G_A) \cap \mathcal{L}(G_B)$. Since $ya_{i_m} \cdots a_{i_1} \in \mathcal{L}(G_A)$ it must be the case that $y = w_{i_1} w_{i_2} \cdots w_{i_m}$. Also, y must be equal to $x_{i_1} x_{i_2} \cdots x_{i_m}$ since $ya_{i_m} \cdots a_{i_1} \in \mathcal{L}(G_B)$. Then i_1, \dots, i_m forms a solution to PCP instance (A, B) .

5. Show that PCP is undecidable over the binary alphabet $\{0, 1\}$.

Solution: Denote PCP over alphabet $\{0, 1\}$ as BPCP. We show that $\text{PCP} \leq_m \text{BPCP}$. Let $A = \{w_1, \dots, w_n\}$ and $B = \{x_1, \dots, x_n\}$ denote an instance of PCP over some alphabet Σ . Let $s = |\Sigma|$ and $\Sigma = \{a_1, \dots, a_s\}$. Define a map $f : \Sigma \rightarrow \{0, 1\}^*$ as $f(a_i) = 0^i 1$ for $i \in [1, s]$. Extend this map to strings over Σ^* as $F : \Sigma^* \rightarrow \{0, 1\}^*$ where for any string $y = a_{i_1} a_{i_2} \cdots a_{i_k}$, $F(y) = f(a_{i_1}) f(a_{i_2}) \cdots f(a_{i_k})$. Observe that PCP instance (A, B) has a solution iff the BPCP instance $(\{F(w_1), \dots, F(w_n)\}, \{F(x_1), \dots, F(x_n)\})$ has a solution, when F is one-one. Suppose that $F(y) = F(z)$ for some $y, z \in \Sigma^*$. Let $y = a_{i_1} \cdots a_{i_k}$ and $z = a_{j_1} \cdots a_{j_\ell}$ for some $i_1, \dots, i_k, j_1, \dots, j_\ell \in [1, n]$, then $F(y) = f(a_{i_1}) f(a_{i_2}) \cdots f(a_{i_k}) = 0^{i_1} 1 0^{i_2} 1 \cdots 0^{i_k} 1 = f(a_{j_1}) f(a_{j_2}) \cdots f(a_{j_\ell}) = F(z)$. If $y \neq z$ Let r be the minimum integer such that $a_{i_r} \neq a_{j_r}$. Then $f(a_{i_r}) = 0^{i_r} 1 \neq 0^{j_r} 1 = f(a_{j_r})$ but then this implies that $F(y) \neq F(x)$ contradicting our assumption. Therefore $x = z$ and as a consequence F is 1-1.

6. Show that the language $\text{PF} = \{G \mid G \text{ is a CFG and } L(G) \text{ is prefix-free}\}$ is undecidable.

Solution: We know that PCP is undecidable. This implies $\neg\text{PCP}$ is undecidable as well.

We describe a reduction $\neg\text{PCP} \leq_m \text{PF}$. Let $A = (w_1, w_2, \dots, w_k)$ and $B = (x_1, x_2, \dots, x_k)$ be an instance of $\neg\text{PCP}$ defined over alphabet Σ . Let $a_1, a_2, \dots, a_k, \#, \neg \notin \Sigma$ be $k + 1$ new distinct symbols and let $\Sigma' = \Sigma \cup \{a_1, \dots, a_k, \#, \neg\}$. Define a context-free grammar $G = (N = \{S, S_A, S_B\}, \Sigma', P, S)$ where P consists of the following productions:

$$S \rightarrow S_A \# \neg \mid S_B \#,$$

$$S_A \rightarrow w_i S_A a_i \mid w_i a_i \text{ for } 1 \leq i \leq k,$$

$$S_B \rightarrow x_i S_A a_i \mid x_i a_i \text{ for } 1 \leq i \leq k.$$

Observe that all strings derived from $S \rightarrow S_A \# \neg$ end with $\# \neg$ and all strings derived from $S \rightarrow S_B \#$ end with $\#$. Suppose there are distinct strings $u, v \in L(G)$ such that u is a prefix of v . Then all symbols in u and v upto and including $\#$ must match. That is, we can write $u = u' \#$, $v = v' \# \neg$ such that $u' = v'$, $S_A \xrightarrow[G]{*} v'$ and $S_B \xrightarrow[G]{*} u'$.

We now show that $(A, B) \in \neg\text{PCP}$ iff $L(G)$ is prefix-free. Suppose that $(A, B) \notin \neg\text{PCP}$. For a solution i_1, i_2, \dots, i_m , we have $w_{i_1} w_{i_2} \cdots w_{i_m} = x_{i_1} x_{i_2} \cdots x_{i_m}$. Let $z = w_{i_1} w_{i_2} \cdots w_{i_m} a_{i_m} \cdots a_{i_2} a_{i_1} = x_{i_1} x_{i_2} \cdots x_{i_m} a_{i_m} \cdots a_{i_2} a_{i_1}$. By definition, $L(G)$ contains both $z \# \neg$ and $z \#$ and hence is not prefix-free. Suppose that $\exists u, v \in L(G)$ such that u is a prefix of v . Then, $u = u' \#$, $v = v' \# \neg$ such that $u' = v'$, $S_A \xrightarrow[G]{*} v'$ and $S_B \xrightarrow[G]{*} u'$. The string v' , derived from S_A , must be of the form $w_{i_1} w_{i_2} \cdots w_{i_m} a_{i_m} \cdots a_{i_2} a_{i_1}$. Similarly, u' has the form $x_{i_1} x_{i_2} \cdots x_{i_m} a_{i_m} \cdots a_{i_2} a_{i_1}$. The a_i 's at the end must all match since $u' = v'$. As a result, $w_{i_1} w_{i_2} \cdots w_{i_m} = x_{i_1} x_{i_2} \cdots x_{i_m}$ implying that i_1, i_2, \dots, i_m is a solution for (A, B) i.e., $(A, B) \notin \neg\text{PCP}$. We have shown that $(A, B) \notin \neg\text{PCP} \Leftrightarrow G \notin \text{PF}$ from which it follows that $(A, B) \in \neg\text{PCP} \Leftrightarrow G \in \text{PF}$ and therefore PF is undecidable.

7. For $A, B \subseteq \Sigma^*$, define

$$A/B = \{x \in \Sigma^* \mid \exists y \in B \quad xy \in A\}.$$

(a) Show that if A and B are recursively enumerable, then so is A/B .

Solution: Let $\mathcal{M}_A, \mathcal{M}_B$ be Turing machines accepting A, B respectively. Define a TM \mathcal{N} that on input x does the following.

- For each $y \in \Sigma^*$, simulate \mathcal{M}_B on y on a time-shared basis. That is, simulate \mathcal{M}_B on y_1 for one step and then simulate it on y_2 for one step and continue simulations for some fixed ordering y_1, y_2, \dots of strings in Σ^* .
- If \mathcal{M}_B accepts, then simulate \mathcal{M}_A on xy .
- Halt and accept if \mathcal{M}_A accepts.

If $x \in A/B$, then for some $y \in \Sigma^*$, \mathcal{M}_B accepts y eventually and \mathcal{M}_A accepts xy . Hence A/B is recursively enumerable.

(b) Show that every *r.e.* set can be represented as A/B with A and B being context-free languages.

Solution: Let R be an *r.e.* set and let $\mathcal{M} = (Q, \Sigma, \Gamma, \delta, \vdash, \sqsubset, s, t, r)$ be a Turing machine accepting it. Recall that we defined $\text{VALCOMPS}_{\mathcal{M},c}$ over the alphabet $\Delta = \{\#\} \cup (\Gamma \cup Q)$. For $x = a_1 a_2 \dots a_n$, let $S_{\mathcal{M},x}$ be the starting configuration, given by $s \vdash a_1 a_2 \dots a_n$. We now define the sets A and B over the alphabet Δ as follows:

$$A = \left\{ x \# S_{\mathcal{M},x} \# \alpha_1^{\mathbf{R}} \# \alpha_2 \# \dots \# \alpha'_N \mid \alpha_i \xrightarrow[\mathcal{M}]{1} \alpha_{i+1} \text{ for all odd } i \right\}$$

$$B = \left\{ \# \alpha_0 \# \alpha_1^{\mathbf{R}} \# \alpha_2 \# \dots \# \alpha'_N \mid \alpha_i \xrightarrow[\mathcal{M}]{1} \alpha_{i+1} \text{ for all even } i \text{ and } \alpha_N \text{ contains } t \right\}$$

Here, $\alpha'_N = \alpha_N^{\mathbf{R}}$ if N is odd and $\alpha'_N = \alpha_N$ otherwise.

Convince yourself that $R = A/B$!