Master Degree in Computer Engineering

Final Exam for
Automata, Languages and Computation

January 26th, 2022

1. **[5 points]** Let E be a regular expression and let $L(E)$ be the generated language. Let R be the string reversal operator, extended to languages in the usual way. Using structural induction, construct a regular expression E^R such that $L(E^R) = (L(E))^R$, and prove this relation.

   Solution

   The required construction can be found in Chapter 4 of the textbook, Theorem 4.11.

2. **[8 points]** Consider the following languages, defined over the alphabet $\Sigma = \{a, b, c\}$

$$\begin{aligned} L_1 &= \{w \mid w = a^p b^q c^r,\ p, q, r \geq 1,\ p = r = 2q\} ; \\ L_2 &= \{w \mid w = a^p b^q c^r,\ p, q, r \geq 1,\ p + r = 2q\} . \end{aligned}$$

   State whether L_1 and L_2 are context-free languages, and motivate your answers.

   Solution

   (a) L_1 is not a context-free language. To prove this statement, we use the pumping lemma for context-free languages. Let us start by reformulating the definition of the language as $L_1 = \{w \mid w = a^{2q} b^q c^{2q},\ q \geq 1\}$. Let N be the pumping lemma constant. We choose the string $z = a^{2N} b^N c^{2N} \in L_1$ and consider all possible factorizations $z = uvwxy$ satisfying the conditions $|v| + |x| \geq 1$ and $|vwx| \leq N$. Because of the latter condition, we have that vx can contain occurrences of at most two symbols from Σ, and these two symbols can be either a and b or else b and c, but not a and c. We separately discuss all possible cases in what follows.

   - If vx contains at most one symbol X from Σ, the string $uv^k wx^k y$ with $k = 0$ will not belong to L_1, because there will be some mismatch in the length of the three blocks of a's, b's and c's.
   - If v contains only X and y contains only Y, X and Y from Σ such that $X \neq Y$, then there must be a symbol $Z \in \Sigma$ such that Z does not occur in v and in x. Again, the string $uv^k wx^k y$ with $k = 0$ will not belong to L_1, because there will be some mismatch in the length of the three blocks.
   - If v contains two (distinguishable) symbols X and Y from Σ, it is easy to see that any string $uv^k wx^k y$ with $k \geq 2$ will not belong to L_1, because of alternating occurrences of X and Y. A similar argument holds if x contains two symbol from Σ.

   We thus conclude that L_1 is not a context-free language.

   (b) L_2 is a context-free language. To see this, we reformulate the definition of the language as $L_2 = L_2' \cup L_2''$, where

$$\begin{aligned} L_2' &= \{w \mid w = a^p b^q c^r,\ p, q, r \geq 1,\ p + r = 2q,\ p \text{ is even}\} ; \\ L_2'' &= \{w \mid w = a^p b^q c^r,\ p, q, r \geq 1,\ p + r = 2q,\ p \text{ is odd}\} . \end{aligned}$$

We now define CFGs G' and G'' such that $L(G') = L_2'$ and $L(G'') = L_2''$. Our claim then follows from the closure of context-free languages under the union operator. Grammar G' is implicitly defined by the following productions:

$$\begin{aligned} S &\to S_1 S_2 \\ S_1 &\to aaS_1b \mid aab \\ S_2 &\to bS_2cc \mid bcc \end{aligned}$$

Grammar G'' is implicitly defined by the following productions:

$$\begin{aligned} S &\to S_1 S_2 \\ S_1 &\to aaS_1b \mid a \\ S_2 &\to bS_2cc \mid bc \end{aligned}$$

3. **[5 points]** Consider the CFG G implicitly defined by the following productions:

$$\begin{aligned} S &\to ABA \mid BAB \\ A &\to aA \mid bB \\ B &\to b \mid \varepsilon \end{aligned}$$

Perform on G the following transformations that have been specified in the textbook, in the given order. Report the CFGs obtained at each of the intermediate steps.

(a) Eliminate the ε-productions
(b) Eliminate the unary productions
(c) Eliminate the useless symbols
(d) Produce a CFG in Chomsky normal form equivalent to G.

Solution

We start by observing that $\varepsilon \notin L(G)$, therefore we can construct a new CFG in Chomsky normal form that is equivalent to G. All of the algorithms that need to be applied to the grammar G are reported in Chapter 7 of the textbook.

(a) The set of nullable variables of G is $n(G) = \{B\}$. After elimination of the ε-productions we obtain the intermediate CFG G_1

$$\begin{aligned} S &\to ABA \mid AA \mid BAB \mid AB \mid BA \mid A \\ A &\to aA \mid bB \mid b \\ B &\to b \end{aligned}$$

(b) The only unary production in G_1 is $S \to A$. Thus the set of unary pairs of G_1 is

$$u(G_1) \quad = \quad \{(S, A)\} \cup \{(X, X) \mid X \in \{S, A, B\}\}.$$

After elimination of the unary productions we obtain the intermediate CFG G_2

$$\begin{aligned}
S &\rightarrow ABA \mid AA \mid BAB \mid AB \mid BA \mid aA \mid bB \mid b \\
A &\rightarrow aA \mid bB \mid b \\
B &\rightarrow b
\end{aligned}$$

(c) All nonterminals in G_2 are reachable and generating, that is, there are no useless nonterminals in G_2. Therefore this step does not change the intermediate CFG obtained at the previous step.

(d) The construction of a CFG in Chomsky normal form from G_2 proceeds in two steps. The first step eliminates terminal symbols in the right-hand side of the productions of G_2, in case they appear along with some other symbols. To do this we introduce new nonterminal symbols C_a, C_b and produce the intermediate CFG G_3

$$\begin{aligned}
S &\rightarrow ABA \mid AA \mid BAB \mid AB \mid BA \mid C_aA \mid C_bB \mid b \\
A &\rightarrow C_aA \mid C_bB \mid b \\
B &\rightarrow b \\
C_a &\rightarrow a \\
C_b &\rightarrow b
\end{aligned}$$

The second step factorizes productions of G_3 having right-hand side of length larger than two. To do this we introduce new nonterminal symbols D, E and produce the final CFG G_4

$$\begin{aligned}
S &\rightarrow AD \mid AA \mid BE \mid AB \mid BA \mid C_aA \mid C_bB \mid b \\
D &\rightarrow BA \\
E &\rightarrow AB \\
A &\rightarrow C_aA \mid C_bB \mid b \\
B &\rightarrow b \\
C_a &\rightarrow a \\
C_b &\rightarrow b
\end{aligned}$$

4. **[6 points]** Assess whether the following statements are true or false, providing motivations for all of your answers.

   (a) If L_1 and L_2 are not in CFL, then the language $L_1 \cap L_2$ cannot be in CFL.

   (b) If $L_1 \cup L_2$ is a regular language, then also L_1 and L_2 are regular languages.

   (c) Let Σ be some fixed alphabet and let L_i, $i \geq 1$, be finite languages over Σ. Then the language

   $$L = \cup_{i=1}^{\infty} L_i$$

   is always a regular language.

   (d) The class $\mathcal{P}$ of languages that can be recognized in polynomial time by a TM is closed under intersection with regular languages.

Solution

(a) False. Consider the alphabet $\Sigma = \{a, b, c\}$ and the counterexample $L_1 = \{a^n b^n a^n \mid n \geq 1\}$, $L_2 = \{b^n a^n b^n \mid n \geq 1\}$. It is easy to show that L_1 and L_2 are not in CFL, using the pumping lemma. But the language $L_1 \cap L_2$ is the empty language, which is a regular language and therefore a CFL as well.

(b) False. Consider the alphabet $\Sigma = \{a, b\}$ and the counterexample $L_1 = \{w \mid w \in \Sigma^*,\ \#_a(w) = \#_b(w)\}$, $L_2 = \{w \mid w \in \Sigma^*,\ \#_a(w) \neq \#_b(w)\}$. It is easy to see that $L_1 \cup L_2 = \Sigma^*$ and thus a regular language. However, L_1 and L_2 are not regular languages.

(c) False. Consider the alphabet $\Sigma = \{a, b\}$ and, for each $i \geq 1$, the language $L_i = \{a^i b^i\}$. Each L_i contains exactly one string, therefore each L_i is a finite language. However, $L = \cup_{i=1}^{\infty} L_i = \{a^n b^n \mid n \geq 1\}$, which is not a context-free language and therefore not a regular language.

(d) True. Let L_1 be an arbitrary language in $\mathcal{P}$. By definition of $\mathcal{P}$, there exists some TM M_1 such that $L(M_1) = L_1$ and M_1 processes its input in polynomial time. Let also L_2 be a regular language. It is not difficult to devise a TM M_2 that simulates a DFA for L_2 and that runs in polynomial time. We can now construct a TM M that, given as input a string w, simulates M_1 and M_2 on w in polynomial time. M accepts if both M_1 and M_2 accept, and rejects otherwise. This shows that the intersection language $L_1 \cap L_2$ is in $\mathcal{P}$. Since L_1 and L_2 were chosen arbitrarily, we have shown that the class $\mathcal{P}$ is closed under intersection with regular languages.

5. **[9 points]** For a property $\mathcal{P}$ of the RE languages, define $L_{\mathcal{P}} = \{\mathsf{enc}(M) \mid L(M) \in \mathcal{P}\}$.

(a) Let k be some fixed natural number with $k > 1$. Consider the following properties of the RE languages defined over the alphabet $\Sigma = \{0, 1\}$:

$$\begin{aligned} \mathcal{P}_{<k} &= \{L \mid L \in \text{RE},\ |L| < k\}\ ; \\ \mathcal{P}_{\geq k} &= \{L \mid L \in \text{RE},\ |L| \geq k\}\ . \end{aligned}$$

Assess whether each of the languages $L_{\mathcal{P}_{<k}}$ and $L_{\mathcal{P}_{\geq k}}$ belongs to the classes REC, RE$\smallsetminus$REC, or else does not belong to RE.

(b) Let $\mathsf{enc}(M_1, M_2)$ be a binary string representing some fixed encoding of TMs M_1, M_2. Consider the following language, where '$\cdot$' denotes the concatenation operation between languages:

$$L = \{\mathsf{enc}(M_1, M_2) \mid |L(M_1) \cdot L(M_2)| < k\}.$$

Prove that L does not belong to the class RE.

Solution

(a) Language $L_{\mathcal{P}_{\geq k}}$ is not in REC. To prove this statement, we apply Rice's theorem and show that property $\mathcal{P}_{\geq k}$ is not trivial. First, Σ^* is in RE and has more than k strings. Therefore we have $\Sigma^* \in \mathcal{P}_{\geq k}$ and $\mathcal{P}_{\geq k}$ is not empty. Second, the empty language $\emptyset$ is in RE and has fewer than k strings, since $k \geq 1$. Therefore we have $\emptyset \notin \mathcal{P}_{\geq k}$, and $\mathcal{P}_{\geq k}$ does not contain every RE language. Since $\mathcal{P}_{\geq k}$ is not trivial, we can conclude that $L_{\mathcal{P}_{\geq k}}$ is not in REC, according to Rice's theorem. We now prove that $L_{\mathcal{P}_{\geq k}}$ is in RE. To this end, we specify a nondeterministic TM N such that $L(N) = L_{\mathcal{P}_{\geq k}}$. Let $\mathsf{enc}(M)$ be the input to N.

- Using nondeterminism, N guesses k different strings $w_i \in \Sigma^*$, $1 \leq i \leq k$.
- For each $i = 1, \ldots, k$ in the given order, N simulates M on input w_i.
- If any of the k simulations above does not halt, then N does not halt as well.
- If all of the k simulations halt, N accepts in case every simulation reaches a final state, and rejects otherwise.

It is not difficult to see that $L(N) = L_{\mathcal{P}_{\geq k}}$. Since nondeterministic TMs are equivalent to TMs, we conclude that $L_{\mathcal{P}_{\geq k}}$ is in RE.

Consider now the language $L_{\mathcal{P}_{<k}}$. We observe that $L_{\mathcal{P}_{<k}}$ is the complement language of $L_{\mathcal{P}_{\geq k}}$ with respect to Σ^*. Since $L_{\mathcal{P}_{\geq k}}$ is in RE$\smallsetminus$REC, from a well-known property we conclude that $L_{\mathcal{P}_{<k}}$ cannot be in RE.

(b) Language L is not in RE. To prove this statement, we use the fact that $L_{\mathcal{P}_{<k}}$ is not in RE, as shown in (a), and define a reduction $L_{\mathcal{P}_{<k}} \leq_m L$.

We need to map instances $\mathsf{enc}(M)$ of $L_{\mathcal{P}_{<k}}$ into instances $\mathsf{enc}(M_1, M_2)$ of L. We set $M_1 = M$ and $M_2 = M_\varepsilon$, where M_ε is any TM that recognizes the language $\{\varepsilon\}$. The following chain of logical equivalences shows that the construction represents a valid reduction:

$$
\begin{array}{lll}
\mathsf{enc}(M) \in L_{\mathcal{P}_{<k}} & \text{iff} & |L(M)| < k & \text{(definition of } \mathcal{P}_{<k}) \\
& \text{iff} & |L(M) \cdot \{\varepsilon\}| < k & \text{(definition of concatenation)} \\
& \text{iff} & |L(M_1) \cdot L(M_\varepsilon)| < k & \text{(definition of our reduction)} \\
& \text{iff} & \mathsf{enc}(M_1, M_2) \in L & \text{(definition of } L) \ .
\end{array}
$$