Master Degree in Computer Engineering

Final Exam for Automata, Languages and Computation

February 12th, 2026

1. **[5 points]** Regular languages are closed under intersection. In the textbook and at the course lectures, two alternative proofs of this fact have been reported. Present here both proofs.

   Solution

   The first proof uses the so-called De Morgan laws, and is discussed in Chapter 4 of the textbook, before Theorem 4.8. The second proof uses a construction that takes as input two FAs and provides the so-called intersection FA. The proof is reported in Theorem 4.8, in Chapter 4 of the textbook.

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

$$\begin{aligned} L_1 &= \{a^n b a^n \mid n \geq 1\} \\ L_2 &= \{xby \mid x, y \in \Sigma^+,\ |x| = |y|\} \\ L_3 &= \{xby \mid x, y \in \Sigma^+\} \end{aligned}$$

   For each of the above languages, state whether it belongs to REG, to CFL$\setminus$REG, or else whether it is outside of CFL. Provide a mathematical proof for all of your answers.

   Solution

   (a) L_1 belongs to the class CFL$\setminus$REG.

   We first show that L_1 is not a regular language, by applying the pumping lemma for this class. In what follows, we view each string in L_1 as composed by two blocks of occurrences of symbol a, one at the left of the occurrence of b and the other to its right. These two blocks must have the same length.

   Let N be the pumping lemma constant for L_1. We choose the string $w = a^N b a^N \in L_1$ with $|w| \geq N$. We now consider all possible factorizations of the form $w = xyz$ satisfying the conditions $|y| \geq 1$ and $|xy| \leq N$ of the pumping lemma. Since $|xy| \leq N$, string y can only span over the block of a's placed at the left of b. Therefore we need to consider only one case in our discussion. We choose $k = 2$ in the pumping lemma, and obtain the new string $w_{k=2} = xy^2z = xyyz$, which has the form $a^{N+|y|}ba^N$. Since $|y| \geq 1$, the two blocks of a's do not have the same length, and thus $w_{k=2} \notin L_1$. We conclude that L_1 does not satisfy the pumping lemma, and therefore cannot be a regular language.

   As a second part of the answer, we need to show that L_1 belongs to the class CFL. Consider the CFG G_1 with productions:

$$\begin{aligned} S &\to aSa \mid aBa \\ B &\to b \end{aligned}$$

   It is not too difficult to see that $L(G_1) = L_1$.

(b) L_2 belongs to the class CFL$\setminus$REG.

We first show that L_2 is not a regular language, by applying the pumping lemma for this class. To this end, it is useful to observe that every string $w \in L_2$ has odd length, and the symbol occurring in the middle of w must be b.

Let N be the pumping lemma constant for L_2. We choose the string $w = a^N b a^N \in L_2$ with $|w| \geq N$. We now consider all possible factorizations of the form $w = xyz$ satisfying the conditions $|y| \geq 1$ and $|xy| \leq N$ of the pumping lemma. Since $|xy| \leq N$, string y can only span over the first (left-to-right) N occurrences of symbol a in w, that is, string w cannot include the occurrence of symbol b from w.

We then choose $k = 0$ in the pumping lemma, and obtain the new string $w_{k=0} = xy^0z = xz$, which has the form $a^{N-|y|}ba^N$. We now observe that there is only one occurrence of symbol b in $w_{k=0}$ and, according to the definition of L_2, this symbol must be in the middle of $w_{k=0}$. But this is not the case, since $|y| \geq 1$ and thus $N - |y| < N$. This is a violation of the pumping lemma, and we conclude that L_2 cannot be a regular language.

As a second part of the answer, we need to show that L_2 belongs to the class CFL. Consider the CFG G_2 with productions:

$$\begin{aligned} S &\rightarrow DSD \mid DBD \\ B &\rightarrow b \\ D &\rightarrow a \mid b \end{aligned}$$

It is not too difficult to see that $L(G_2) = L_2$.

(c) L_3 belongs to the class REG.

To see this, we observe that L_3 contains all and only the strings over Σ that satisfy the following two conditions

- length larger than or equal to three;
- at least one occurrence of symbol b, placed at some position different from the first and the last one in the string.

It is then easy to see that the regular expression

$$R = (\boldsymbol{a} + \boldsymbol{b})(\boldsymbol{a} + \boldsymbol{b})^*\boldsymbol{b}(\boldsymbol{a} + \boldsymbol{b})(\boldsymbol{a} + \boldsymbol{b})^*$$

generates L_3.

3. **[5 points]** With reference to the membership problem for context-free languages, answer the following two questions.

(a) Specify the dynamic programming algorithm developed in class for the solution of this problem.

(b) Consider the CFG G defined by the following rules:

$$\begin{aligned} S &\rightarrow AB \mid BC \\ A &\rightarrow BA \mid a \\ B &\rightarrow CC \mid b \\ C &\rightarrow AB \mid a \end{aligned}$$

Assuming as input the string $w = baaba$, trace the application of the above algorithm and report the final parse table.

Solution

(a) The required dynamic programming algorithm is reported in Section 7.4.4 of Chapter 7 of the textbook.

(b) The algorithm constructs a table filling its rows one by one, in a bottom-up way. Each entry in the table is filled with a set of variables of the grammar. On input w and G, the algorithm constructs the table reported below.

$\{S, C, A\}$				
$\emptyset$	$\{S, C, A\}$			
$\emptyset$	$\{B\}$	$\{B\}$		
$\{S, A\}$	$\{B\}$	$\{S, C\}$	$\{S, A\}$	
$\{B\}$	$\{A, C\}$	$\{A, C\}$	$\{B\}$	$\{A, C\}$
b	a	a	b	a

Since the start symbol S belongs to the top-most cell of the parse table, we conclude that the string w is in the language generated by the grammar G.

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

(a) There exist languages L_1 and L_2 in REG such that $L_1 \subsetneq L_2$ and $L_2 \setminus L_1$ is not in REG.

(b) There exist languages L_1 and L_2 in CFL$\setminus$REG such that $L_1 \neq L_2$ and $L_1 \cap L_2$ is in CFL$\setminus$REG.

(c) For every language L_1 in CFL and every language L_2 in REG such that $L_1 \subsetneq L_2$, we have that $L_2 \setminus L_1$ is in REG.

(d) Let $\mathcal{P}$ be the class of languages that can be recognized in polynomial time by a TM, and let $\mathcal{NP}$ be the class of languages that can be recognized in polynomial time by a nondeterministic TM. We have $\mathcal{P} \neq \mathcal{NP}$.

Solution

(a) False. We know from Chapter 4 of the textbook that the class REG is closed under set difference. Hence the language $L_2 \setminus L_1$ must be in REG. Note that the additional condition $L_1 \subsetneq L_2$ is not relevant here, and it has been used in the assignment as a distracting element.

(b) True. Let $L_1 = \{a^n b^n \mid n \geq 0\}$ and $L_2 = \{a^n b^n \mid n \geq 1\}$. Both L_1 and L_2 are known to be in CFL$\setminus$REG. We then observe that $L_1 \cap L_2 = L_2$.

(c) False. For a string w defined over Σ and for any symbol $X \in \Sigma$, let us write $\#_X(w)$ to denote the number of occurrences of X in w. We then define $L_1 = \{w \mid w \in \{a,b\}^*,\ \#_a(w) = \#_b(w)\}$. We know from class that L_1 is in CFL. We also define $L_2 = \{a,b\}^*$, which is in REG. We have $L_1 \subsetneq L_2$ as required. We can now observe that

$$L_2 \setminus L_1 \quad = \quad \{w \mid w \in \{a,b\}^*,\ \#_a(w) \neq \#_b(w)\}\ .$$

It is not difficult to use the pumping lemma for regular languages to show that $L_2 \setminus L_1$ is not in REG.

(d) Neither false nor true. As stated in Chapter 10, the question $\mathcal{P} \neq \mathcal{NP}$ is an important open problem in the theory of the complexity of TMs.

5. **[6 points]** In relation to the theory of undecidability, answer the following questions. All the TMs introduced below are defined over the input alphabet $\Sigma = \{0,1\}$.

Given a string w and a language L, we say that w can be **factorized** in L if, for some pair of string $u, v \in L$, we can write $w = u \cdot v$. Consider the following property of the RE languages

$$\mathcal{P} \quad = \quad \{L \mid L \in \text{RE},\ \text{ for every string } w \in L \text{ we have that } w \text{ cannot be factorized in } L\}$$

and define $L_{\mathcal{P}} = \{\mathsf{enc}(M) \mid L(M) \in \mathcal{P}\}$.

(a) Use Rice's theorem to prove that $L_{\mathcal{P}}$ is not in REC.

(b) Prove that $L_{\mathcal{P}}$ is not in RE.

Solution

(a) We show that the property $\mathcal{P}$ is nontrivial, that is, $\mathcal{P}$ is neither empty nor equal to RE.

- $\mathcal{P} \neq \emptyset$. The language $L_1 = \{011011, 010101, 111000\}$ is in RE, since it is finite. We now have to check that for every string $w \in L_1$, w cannot be factorized in L_1.
  Consider $w = 011001$. If we factorize w into u and v with $|u|, |v| > 0$, we have that $1 \leq |u| \leq 5$ and $1 \leq |v| \leq 5$. But these factorizations should be ruled out, since there aren't strings in L_1 of length ℓ with $1 \leq \ell \leq 5$. The remaining case to be considered is $u = \epsilon, v = w$, or else $u = w, v = \epsilon$. These factorizations should also be ruled out, since $\epsilon \notin L_1$.
  A similar argument can be used for the two remaining strings in L_1. We conclude that $L_1 \in \mathcal{P}$ and thus $\mathcal{P} \neq \emptyset$.
- $\mathcal{P} \neq$ RE. The language $L_2 = \{011, 001, 011001\}$ is in RE, since it is finite. For string $011001 \in L_2$ we have a possible factorization $011 \cdot 001$ with both 011 and 001 in L_2. This is a violation of property $\mathcal{P}$ and then $\mathcal{P} \neq$ RE.

We can now apply Rice's theorem and conclude that, since $\mathcal{P}$ is nontrivial, $L_{\mathcal{P}}$ is not in REC.

(b) We now show that $L_{\mathcal{P}}$ is not in RE. The most convenient way to do this is to consider the complement language $\overline{L_{\mathcal{P}}} = L_{\overline{\mathcal{P}}}$, where $\overline{\mathcal{P}}$ is the complement of the class $\mathcal{P}$ with respect to RE, and can be specified as

$$\overline{\mathcal{P}} \quad = \quad \{L \mid L \in \text{RE},\ \text{there exists a string } w \in L \text{ that can be factorized in } L\}\ .$$

We now define a nondeterministic TM N such that $L(N) = L_{\overline{\mathcal{P}}}$. Since every nondeterministic TM can be converted into a standard TM, this shows that $L_{\overline{\mathcal{P}}}$ is in RE.

Our nondeterministic TM N takes as input the encoding $\mathsf{enc}(M)$ of a TM M and performs the following steps.

- N nondeterministically guesses a string $w \in \Sigma^*$ and a factorization $w = u \cdot v$.
- N simulates M on w, u and v, in any order.
- If all the three computations in the previous step stop, then N also stops. If the three computations have accepted their input, then N accepts. Otherwise N rejects.
- If any of the three computations above does not stop, then N runs for ever and therefore does not accept.

It is not difficult to see that $L(N) = L_{\overline{\mathcal{P}}}$.

Since $L_{\overline{\mathcal{P}}} = \overline{L_{\mathcal{P}}}$ is in RE, if the complement language $L_{\mathcal{P}}$ were in RE as well, then we would conclude that both languages are in REC, from a theorem in Chapter 9 of the textbook. But we have already shown in (a) that $L_{\mathcal{P}}$ is not in REC. We must therefore conclude that $L_{\mathcal{P}}$ is not in RE.

6. **[3 points]** In relation to the theory of intractability, answer the following questions.

   (a) Define the satisfiability problem for Boolean expressions and the formal language SAT.

   (b) Prove that SAT is in the class $\mathcal{NP}$ of all languages that can be recognized in polynomial time by a nondeterministic TM.

Solution

The definition of SAT and the proof that SAT$\in \mathcal{NP}$ are presented in Chapter 10 of the textbook.