CSCI 2400 – Models of Computation, Section 3

Solutions to Homework 4

Problem 1. All the solutions below refer to the Pumping Lemma of Theorem 4.8, page 119.

(a) $L = \{a^n b^l a^k : k \geq n + l\}$

Let's assume for contradiction that L is a regular language. We apply the pumping lemma to L. Let m be the parameter of the pumping lemma. We choose to pump the string $a^m b^m a^{2m}$ which is in the language L, since $2m \geq m+m$ (we have $n = m$, $l = m$, and $k = 2m$). Since $xyz = a^m b^m a^{2m}$ and $|xy| \leq m$ we have that the string y is a substring of the first a^m of $a^m b^m a^{2m}$. Therefore, the string y has the form $y = a^p$, for some integer p, $1 \leq p \leq m$ (since $|y| \geq 1$). Now, we pump up y once and we obtain the string $a^{m+p} b^m a^{2m}$ which is a string identical to $a^m b^m a^{2m}$ with the only difference that we replace y with y^2. By the pumping lemma, we have that $a^{m+p} b^m a^{2m}$ is in the language L. However, $a^{m+p} b^m a^{2m}$ is not in the language L since $2m < m + p + m$ (violating the condition $k \geq n + l$, where $n = m+p$, $l = m$, and $k = 2m$). Therefore, we have a contradiction. Subsequently, our original assumption that the language L is regular must be wrong. Therefore, the language L is not regular.

(b) $L = \{ww : w \in \Sigma^*\}$

Let's assume for contradiction that L is a regular language. We apply the pumping lemma to L. Let m be the parameter of the pumping lemma. We choose to pump the string $a^m b^m a^m b^m$ which is in the language L, since $a^m b^m a^m b^m = ww$ with $w = a^m b^m$. Since $xyz = a^m b^m a^m b^m$ and $|xy| \leq m$, we have that the string y is a substring of the first a^m of $a^m b^m a^m b^m$. Therefore, the string y has the form $y = a^p$, for some integer p, $1 \leq p \leq m$ (since $|y| \geq 1$). Now, we pump up y once and we obtain the string $a^{m+p} b^m a^m b^m$ which is a string identical to $a^m b^m a^m b^m$ with the only difference that we replace y with y^2. By the pumping lemma, we have that $a^{m+p} b^m a^m b^m$ is in the language L. However, $a^{m+p} b^m a^m b^m$ is not in the language L since there is no string w such that $a^{m+p} b^m a^m b^m = ww$. Therefore, we have a contradiction. Subsequently, our original assumption that the language L is regular must be wrong. Therefore, the language L is not regular.

(c) $L = \{a^{k^2}\}$

Let's assume for contradiction that L is a regular language. We apply the pumping lemma to L. Let m be the parameter of the pumping lemma. We choose to pump the string a^{m^2} which obviously is in the language L.

For convenience, we rewrite a^{m^2} as

$$a^{m^2} = a^m a^m \cdots a^m,$$

where a^m is repeated m times. Since $xyz = a^m a^m \cdots a^m$ and $|xy| \leq m$, we have that the string y is a substring of the first a^m of $a^m a^m \cdots a^m$. Therefore, the string y has the form $y = a^p$, for some integer p, $1 \leq p \leq m$ (since $|y| \geq 1$). Now, we pump up y once and we obtain the string $a^{m+p} a^m \cdots a^m$ which is a string identical to $a^m a^m \cdots a^m$ with the only difference that we replace y with y^2. By the pumping lemma, we have that $a^{m+p} a^m \cdots a^m$ is in the language L.

On the other hand, we can rewrite $a^{m+p} a^m \cdots a^m$ as

$$\begin{aligned} a^{m+p} a^m \cdots a^m &= a^p a^m a^m \cdots a^m \\ &= a^p a^{m^2} \\ &= a^{m^2+p}. \end{aligned}$$

The string a^{m^2+p} would be in the language L if there would exist an integer k such that $a^{m^2+p} = a^{k^2}$. But there is no such k. To prove this, observe the following:

$$m^2 < m^2 + p < m^2 + 2m + 1 = (m+1)^2,$$

since $p \leq m$. This implies that

$$m^2 < k^2 < (m+1)^2.$$

Taking the square roots we obtain,

$$m < k < m + 1,$$

which is impossible, since m and k are positive integers. Therefore, there in no k such that $a^{m^2+p} = a^{k^2}$. Subsequently, the string a^{m^2+p} is not in the language L.

Therefore, we have a contradiction. Subsequently, our original assumption that the language L is regular must be wrong. Therefore, the language L is not regular.

(d) $L = \{uww^R v : u, v, w \in \Sigma^+, |u| \geq |v|\}$

Let's assume for contradiction that L is a regular language. We apply the pumping lemma to L. Let m be the parameter of the pumping lemma. We choose to pump the string $(ab)^m aa(ba)^m$ which is in the language L, by taking $u = (ab)^m$, $ww^R = aa$, and $v = (ba)^m$ (notice that $|u| = |v|$, and thus the condition $|u| \geq |v|$ holds). We note that in the string $(ab)^m aa(ba)^m$ the rightmost location of the middle of any substring of the form ww^R must appear within the middle substring aa. Since $xyz = (ab)^m aa(ba)^m$ and $|xy| \leq m$, we have that the string y is a substring of the first $u = (ab)^m$. Now let's pump down the string y, by removing it from the string $(ab)^m aa(ba)^m$. The resulting string has the form $u'aa(ba)^m$, where u' is idectical with the substring $u = (ab)^m$ with the difference that y is removed. By the pumping lemma we have that the string $u'aa(ba)^m$ is in the language L.

On the other hand, we have that $|u'| < |u|$ (since we pumped down y, and $|y| \geq 1$). Notice now that in the string $u'aa(ba)^m$ the rightmost location of the middle of any substring of the form ww^R must appear in the substring aa, right after u'. For $ww^R = aa$, the string $u'aa(ba)^m$ can be written in the form $u'ww^Rv$, with $|u'| < |u| = |v|$. Therefore, the condition $|u'| \geq |v|$ of language L is violated. The same violation occurs even when we identify the string ww^R as any other possible substring of $u'aa(ba)^m$ (which could possibly span u' and v), since the middle of that substring appears in or at the left from aa. Therefore, the string $u'aa(ba)^m$ is not in the language L.

Therefore, we have a contradiction. Subsequently, our original assumption that the language L is regular must be wrong. Therefore, the language L is not regular.

Problem 2.

(a) $L = \{w : n_a(w) \neq n_b(w)\}$

Not regular.

Intuitive explanation: For any string in this language we need to keep count of the numbers of a's and b's so that we compare them. A finite automaton cannot count the numbers of a's and b's in an arbitrary input string, since the number of states is finite and the input string length can be arbitrarily large.

Formal Explanation: Consider the complement language $\overline{L} = \{w : n_a(w) = n_b(w)\}$. By the pumping lemma, the language $\overline{L}$ is not regular (you cannot pump the string $a^m b^m$). Since regular languages are closed under complement, it follows that the language L is not regular either (if L was regular then $\overline{L}$ would be regular too).

(b) $L = \{a^n b^l : n \geq 100, l \leq 100\}$

Regular.

Explanation: You can construct a finitite automaton that accepts this language. The automaton has two parts connected in series. The first part recognizes strings of the form a^n with $n \geq 100$ (this part consists from a sequence of 100 states, each state for a single a, then followed by a loop state for the a's after the 100th a). The second part recognizes strings of the form b^l, where $l \leq 100$ (this part consists from a sequence of 100 states, each state for a single b, and each state is a final state).

(c) $L = \{uww^R v : u, v, w \in \Sigma^+\}$

Regular.

Explanation: The regular expression that describes this language is

$$(a+b)^+(aa+bb)(a+b)^+$$

where the superscript "+" means 1 or more repetitions (the superscript "$*$" means 0 or more repetitions).

(d) $L = \{b^n a^l b^k : n > 5, l > 3, l \geq k\}$

Not Regular.

Intuitive explanation: For any string in this language we need to keep count of the numbers of a's and b's so that we compare them. A finite automaton cannot count the numbers of a's and b's in an arbitrary input string, since the number of states is finite and the input string length can be arbitrarily large.

Formal Explanation: Consider the reverse language $L^R = \{b^k a^l b^n : n > 5, l > 3, l \geq k\}$. By the pumping lemma, the language L^R is not regular (you cannot pump the string $b^m a^m a^4 b^6$). Since regular languages are closed under reversal, it follows that the language L is not regular either (if L was regular then L^R would be regular too).

Problem 3.

(a) $L = \{w : \ w$ starts and ends with the same symbol, and $w \in \{a,b\}^*\}$

The following grammar generates language L, where S is the start variable.

$$S \to aTa \mid bTb$$
$$T \to aT \mid bT \mid \lambda$$

The variables S generate any string that starts and ends with the same symbol. The variable T generates any string made from a's and b's.

(b) The complement of the language $L = \{a^n b^n\}$.

The following grammar generates language L, where S is the start variable.

$$S \to A_1 \mid A_2 \mid A_3 \mid B$$

$$A_1 \to aA_1 \mid a$$
$$A_2 \to A_1 X \mid A_1 X a T$$
$$A_3 \to XbT \mid XaT$$

$$B \to bT$$

$$X \to aXb \mid ab$$
$$T \to aT \mid bT \mid \lambda$$

The variables of the grammar correspond to a case analysis of the possible strings. The variables A_1, A_2 and A_3 generate all the strings that start with an a, and the variable B generates all the strings that start with a b. Therefore, the start variable S covers all the cases.

In order to generate the desired strings we use the helping variables X and T as follows. The variable X generates all the strings of the form $a^n b^n$, which we want to avoid (since these are excluded from the desired language). The variable T generates any string made from symbols a and b (namely, the language $(a + b)^*$).

The three cases of strings starting with a are the following:

- Variable A_1: generates any string of the form a^*.
- Variable A_2: generates any string of the form $a^+a^nb^n$ and $a^+a^nb^na(a+b)^*$.
- Variable A_3: generates any string of the form $a^nb^n(a + b)^+$.

(The + superscript means one or more occurence.) These are all the cases of strings starting with an a. Notice that variables A_2 and A_3 deal with the case where a^nb^n is a substring. The way we remove a^nb^n is by concatenating other strings to the left and right of a^nb^n.

(c) $L = \{w : n_a(w) = 2n_b(w), \text{where } w \in \{a, b\}^*\}$

The following grammar generates language L, where S is the start variable.

$$S \to SS \mid \lambda$$
$$S \to aaSb \mid aSab \mid aSba$$
$$S \to abSa \mid baSa \mid bSaa$$

This is a modification of the grammar of Example 1.12 in page 23 which handles the case $n_a(w) = n_b(w)$. The main difference here is that we add an extra symbol a to each possible position of the righthand side of each production that has a terminal a.

(d) $L = \{w\#x : w^R \text{ is a substring of x, where } w, x \in \{a, b\}^*\}$

The following grammar generates language L, where S is the start variable.

$$S \to AT$$
$$A \to aAa \mid bAb \mid \#T$$
$$T \to aT \mid bT \mid \lambda$$

The strings in the language have the form $w\#uw^Rv$, where u and v are strings of the form $(a+b)^*$ (any string made from symbols a and b). The variable T generates the strings u and v, while variable A generates the string $w\#uw^R$ and the variable S generates the desired string $w\#uw^Rv$.

Problem 4.

(a) The string $s = aab$ has the following two leftmost derivations:

$$S \Rightarrow aaB \Rightarrow aab$$
$$S \Rightarrow AB \Rightarrow AaB \Rightarrow aaB \Rightarrow aab$$

(b) The two derivation trees of string aab are shown in Figure 1.

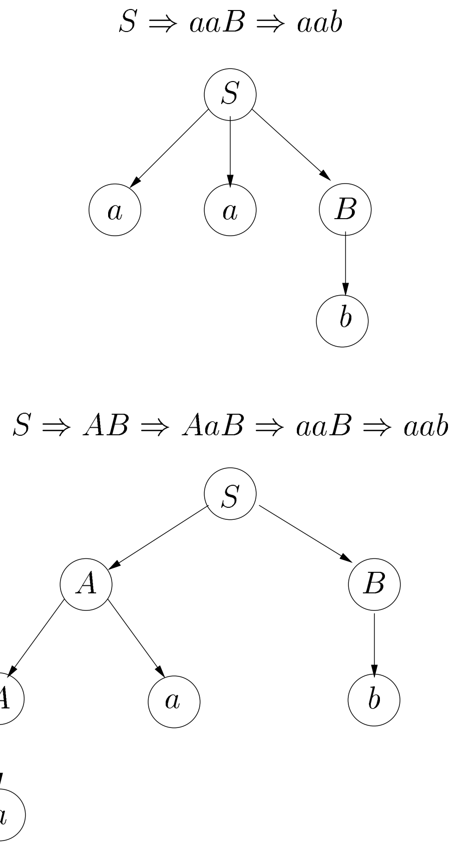

Figure 1: The two derivation trees of string aab

(c) The equivalent unambiguous grammar is the following:

$$\begin{aligned} S &\rightarrow Ab \\ A &\rightarrow a|Aa \end{aligned}$$

This grammar is not ambiguous because at any derivarion step there is only one choice to make. This grammar is equivalent to the previous grammar becase both grammars generate the same language: all the stings that start with one or more a's and end with a single b.

(d) With the new grammar the unique leftmost derivation and derivation tree of the string aab are shown in Figure 2.

$$S \Rightarrow Ab \Rightarrow Aab \Rightarrow aab$$

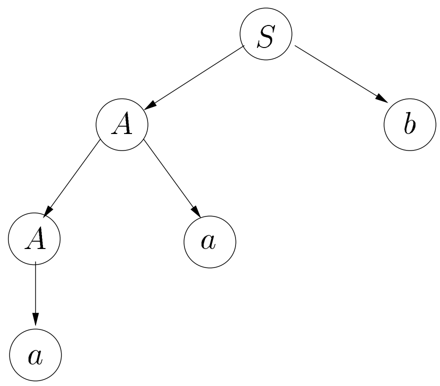

Figure 2: The derivation of string aab with the unambiguous grammar