

COMP 481: Automata, Formal Languages, and Computability
 Spring 2009
 Solutions to Homework Assignment #4

1. (a) $(a \cup b(ab)^*(b \cup aa))(ba \cup (a \cup b)(ab)^*(b \cup aa))^*$
 (b) $(a \cup b)a^* \cup (a \cup b)a^*b((a \cup ba \cup bb)a^*b^*)^*(a \cup ba \cup bb)a^*$
 (c) $b^* \cup b^*a(ab + aab^*a)^*aab^* \cup b^*a(ab \cup aab^*a)^*b(b^* \cup b^*a(ab \cup aab^*a)^*aab^*)^*(bab^* \cup (bb \cup bab^*a)(ab \cup aab^*a)^*aab^*)$

2. (a) Assume A were regular, and let n be the constant guaranteed by the Pumping Lemma. Choose string $w = 1^n 0 1^n$. Clearly, $|w| = 2n + 1 \geq n$. Further, if we take $k = n$, then the string is in the language. Any way of writing $w = xyz$, where $|xy| \leq n$ and $|y| \geq 1$ would have $x = 1^r$, $y = 1^s$, and $z = 1^{n-r-s} 0 1^n$, where $r + s \leq n$ and $s \geq 1$. If we take $i = 0$, then $xy^i z = 1^{n-s} 0 1^n$. In this string, $k \leq n - s$; therefore, in any way of writing this string as $1^k v$, the substring v contains at least $n - k$ 1's, which is strictly greater than k (which is at most $n - s$, where $s \geq 1$). Hence, $xy^0 z \notin A$. Therefore, the language A is not regular.
 (b) Notice that any string $w \in B$ must start with 1 (since $k \geq 1$). In order for y to have at least k 1s, we need to have at least two 1s in w ; that way, only the first 1 is considered the k 1s ($k = 1$), and the rest of 1s will be assigned to y . In other words, this is the language of all strings that (1) start with 1, and (2) have at least one additional 1. A regular expression for this language is $1(0 \cup 1)^* 1(0 \cup 1)^*$.

3. (a) Let $G = C \cap ab^*c^*$. Then, $G = \{ab^n c^n : n \geq 0\}$. You can prove, using the Pumping Lemma for example, that G is not regular (we are not showing the proof here, but you must show it). Therefore, C is not regular, since if it were, G would be regular (intersection of two regular languages).
 (b) Let p be a constant, and let $w \in C$ be a string where $|w| = m \geq p$. Then, we have either $w = ab^q c^q$, where $2q + 1 = m$, or $w = a^r b^s c^t$, where $r + s + t = m$ and $r \neq 1$. In the first case, Let $x = \varepsilon$, $y = a$, and $z = b^q c^q$. Then, for every $i \geq 0$, we have $xy^i z = a^i b^q c^q$, which is a string in C . In the second case, if $r = 0$, then for every $i \geq 0$, $xy^i z \in C$. If $r = 2$, let $x = \varepsilon$, $y = aa$, and $z = b^s c^t$. Since no $i \geq 0$ makes the number of a 's 1, then for every $i \geq 0$, we have $xy^i z \in C$. If $r \geq 3$, then take $x = \varepsilon$, $y = a$, $z = a^{r-1} b^s c^t$, and we have $xy^i z \in C$ for all $i \geq 0$. Therefore, for any string $w \in C$, where $|w| \geq p$, the string w satisfies the conditions of the Lemma. Of course, any string $w \in C$, where $|w| < p$, satisfies the Pumping Lemma vacuously.
 (c) The Pumping Lemma states that if a language L is regular, then it satisfies certain conditions. However, it does not state that if L is *not* regular, then it does *not* satisfy these same conditions. In other words, a non-regular language may satisfy the conditions of the Pumping Lemma.

4. Let $D = a^*bc^*$, which is regular. We have

$$D_{\frac{1}{3}-\frac{1}{3}} = \{a^n c^n : n > 0\} \cup \{a^i b c^j : i + j \equiv_2 1\}.$$

$D_{\frac{1}{3}-\frac{1}{3}} \cap a^*c^* = \{a^n c^n : n > 0\}$, which is not regular (we've seen the proof in class). Therefore, $D_{\frac{1}{3}-\frac{1}{3}}$ is not regular.

5. Let M be the FA. The following is a decision procedure **Q**:

Let $R = \Sigma^{20}\Sigma^*$. (This is an r.e. that generates all strings w , where $|w| \geq 20$).

Convert R into an equivalent FA M' .

Construct an FA, M^* , that recognizes $L(M) \cap L(M')$.

Run the procedure **P** that decides where $L(M^*) = \emptyset$ (which we saw in class).

If **P** prints YES, then **Q** prints NO; else, **Q** prints YES.

6. (a) This is the union of two languages one of which has fewer a 's than b 's, and the other has more a 's than c 's. These two languages are generated from start variables S_1 and S_2 , respectively, in the following CFG for the language.

$$\begin{aligned} S &\rightarrow S_1 | S_2 \\ S_1 &\rightarrow S_1 c | T_1 \\ T_1 &\rightarrow a T_1 b | T_1 b | b \\ S_2 &\rightarrow a S_2 | T_2 \\ T_2 &\rightarrow a T_2 c | a U_2 \\ U_2 &\rightarrow b U_2 | \varepsilon \end{aligned}$$

(b) In this language, the main idea is that for every a you can generate either one b or two b 's, and this is achieved by the following CFG.

$$S \rightarrow a S b | a S b b | \varepsilon$$

(c) The main idea in this language is that either both m and n are even or both of them are odd (if one is even and the other is odd, the subtraction will result in an odd number).

$$\begin{aligned} S &\rightarrow S_1 | S_2 \\ S_1 &\rightarrow a a S_1 b b | a a S_1 | a b \quad (\text{odd } a\text{'s and odd } b\text{'s}) \\ S_2 &\rightarrow a a S_2 b b | a a S_2 | \varepsilon \quad (\text{even } a\text{'s and even } b\text{'s}) \end{aligned}$$

(d) In this language, we need to generate either at least one a or one b for every c generated.

$$S \rightarrow a S c | b S c | a S | b S | \varepsilon$$

(e) In this language, every string w satisfies the property that every prefix of w has at least as many a 's as b 's. The following CFG generates the language.

$$S \rightarrow a S b S | a S | \varepsilon$$

7. The string $aaba$, for example, has two different parse trees; it's easy to show them (make sure you can find two such different parse trees).

8. **Step 1: handling nullable variables.** The nullable variables are S , A , C , and D . The CFG we obtain after handling these variables is:

$$\begin{aligned} S &\rightarrow AaA|Aa|aA|a|CA|C|A|BaB \\ A &\rightarrow aaBa|CDA|C|D|A|CD|CA|DA|aa|DC \\ B &\rightarrow bB|bAB|bB|bb|aS|a \\ C &\rightarrow Ca|a|bC|b|D \\ D &\rightarrow bD|D \end{aligned}$$

Step 2: handling the unit productions. These are all rules of the form $X \rightarrow Y$, where both X and Y are non-terminals. The CFG we obtain after handling these rules is:

$$\begin{aligned} S &\rightarrow AaA|Aa|aA|a|CA|aaBa|CDA|Ca|bC|b|bD|CD|DA|aa|DC|BaB \\ A &\rightarrow aaBa|CDA|Ca|a|bC|b|bD|CD|CA|DA|aa|DC \\ B &\rightarrow bB|bAB|bB|bb|aS|a \\ C &\rightarrow Ca|a|bC|b|bD \\ D &\rightarrow bD \end{aligned}$$

Step 3: handling mixed bodies. These are rules that have a at least one non-terminal and one terminal to the right of the arrow. The CFG we obtain after handling these rules is:

$$\begin{aligned} S &\rightarrow AXA|AX|XA|a|CA|XXBX|CDA|CX|YC|b|YD|CD|DA|XX|DC|BxB \\ A &\rightarrow XXBX|CDA|CX|a|YC|b|YD|CD|CA|DA|XX|DC \\ B &\rightarrow YB|YAB|YB|YY|XS|a \\ C &\rightarrow CX|a|YC|b|YD \\ D &\rightarrow YD \\ X &\rightarrow a \\ Y &\rightarrow b \end{aligned}$$

Step 4: handling long rules. In this step, we break down and rule that has more than two non-terminals on the right. The CFG we obtain after handling these rules is:

$$\begin{aligned} S &\rightarrow AE|AX|XA|a|CA|XF|CH|CX|YC|b|YD|CD|DA|XX|DC|BI \\ A &\rightarrow XF|CH|CX|a|YC|b|YD|CD|CA|DA|XX|DC \\ B &\rightarrow YB|YJ|YB|YY|XS|a \\ C &\rightarrow CX|a|YC|b|YD \\ D &\rightarrow YD \\ X &\rightarrow a \\ Y &\rightarrow b \\ E &\rightarrow XA \end{aligned}$$

$$\begin{aligned}
F &\rightarrow XG \\
G &\rightarrow BX \\
H &\rightarrow DA \\
I &\rightarrow XB \\
J &\rightarrow AB
\end{aligned}$$

The final CFG is in Chomsky Normal Form.

9. (a) The language is the union of two languages, $L_1 = \{a^i b^j c^k : i < j + k\}$ and $L_2 = \{a^i b^j c^k : i > j + k\}$, each of which is generated starting from start variables S_1 and S_2 , respectively, in the following CFG.

$$\begin{aligned}
S &\rightarrow S_1 | S_2 \\
S_1 &\rightarrow aS_1c | S_1c | T_1c | U_1c | U_1 \\
T_1 &\rightarrow aT_1b | T_1b | \varepsilon \\
U_1 &\rightarrow aU_1b | U_1b | b \\
S_2 &\rightarrow aS_2c | T_2 \\
T_2 &\rightarrow aT_2b | aT_2 | a
\end{aligned}$$

- (b) $L = \cup_{i=0}^4 L_i$, where

$$L_i = \{a^m b^n : 3m \leq 5n \leq 4m, m \equiv_5 0\}.$$

So, for language L_0 we have strings of the form $a^{5p} b^n$ and must have $15p \leq 5n \leq 20p$ or $3p \leq n \leq 4p$. Thus, L_0 can be written as

$$L_0 = \{a^{5p} b^{3p+k} : 0 \leq k \leq p\}.$$

The CFG for this language is:

$$S_0 \rightarrow a^5 S_0 b^3 | a^5 S_0 b^4 | \varepsilon.$$

For the language L_1 , we have $3(5p + 1) \leq 5n \leq 4(5p + 1)$, which is equivalent to $3p + 1 \leq n \leq 4p$. From that, we get

$$3(p - 1) + 4 \leq n \leq 4(p - 1) + 4.$$

The CFG for this language is obtained by adding to the grammar for L_0 the following rule:

$$S_1 \rightarrow a^6 S_0 b^4.$$

Using similar argument, we can get CFGs for L_2 , L_3 and L_4 by adding to the grammar for L_0 the following rules, respectively:

$$S_2 \rightarrow a^7 S_0 b^5$$

$$S_3 \rightarrow a^3 S_0 b^2$$

$$S_4 \rightarrow a^4 S_0 b^3$$

The final CFG for L is:

$$S \rightarrow S_0 | a^6 S_0 b^4 | a^7 S_0 b^5 | a^3 S_0 b^2 | a^4 S_0 b^3 | S_0$$

$$S_0 \rightarrow a^5 S_0 b^3 | a^5 S_0 b^4 | \varepsilon.$$

(c) The language L can be written as the union of three languages:

$$L_1 = \{u0v1w : u, v, w \in \{0, 1\}^*, |v| = |u| + |w|\}$$

$$L_2 = \{u1v0w : u, v, w \in \{0, 1\}^*, |v| = |u| + |w|\}$$

$$L_3 = \{x : |x| \text{ is odd}\}.$$

Further, if we denote by $X = \{u0v : u, v \in \{0, 1\}^*, |u| = |v|\}$ and $Y = \{u1v : u, v \in \{0, 1\}^*, |u| = |v|\}$, we can break down the languages further as:

$L_1 = XY$, $L_2 = YX$, and $L_3 = X \cup Y$. A CFG for X has the rules:

$$A \rightarrow 0|0A0|0A1|1A0|1A1$$

and a CFG for Y has the rules:

$$B \rightarrow 1|0B0|0B1|1B0|1B1.$$

Hence, the CFG for the language is:

$$S \rightarrow A|B|AB|BA$$

$$A \rightarrow 0|0A0|0A1|1A0|1A1$$

$$B \rightarrow 1|0B0|0B1|1B0|1B1$$

10. Let $G = (V, \Sigma, R, S)$ be a CFG for L . A CFG for $f(L)$ can be obtained by replacing every terminal σ in a rule by $f(\sigma)$.