Elements of the Theory of Computation (MS013A). Solutions to Selected Problems in L & P

Lars Kristiansen

April 13, 2005

Section 1.8

Problem 1.8.2

Problem (a). We have $\emptyset^* \cup a^* \cup b^* \cup (a \cup b)^* = (a \cup b)^*$.

Explanation: It is easy to see that any string in the set $\emptyset^* \cup a^* \cup b^* \cup (a \cup b)^*$ is a string over the alphabet $\{a, b\}$. Hence, we have

$$\emptyset^* \cup a^* \cup b^* \cup (a \cup b)^* \subseteq (a \cup b)^* \qquad (\dagger)$$

It is trivial that

$$(a \cup b)^* \subseteq \emptyset^* \cup a^* \cup b^* \cup (a \cup b)^* \qquad (\ddagger)$$

It follows from $(\dagger)$ and $(\ddagger)$ that $\emptyset^* \cup a^* \cup b^* \cup (a \cup b)^* = (a \cup b)^*$.

Problem (b). We have $(\,(a^*b^*)^*\,(b^*a^*)^*\,)^* = (a \cup b)^*$.

Explanation: We have

(1) $(a^*b^*)^* = (a \cup b)^*$ and $(b^*a^*)^* = (a \cup b)^*$

(2) $\alpha^*\alpha^* = \alpha^*$ for any regular expression α

(3) $\alpha^{**} = \alpha^*$ for any regular expression α

Hence, $(\,(a^*b^*)^*\,(b^*a^*)^*\,)^* \overset{(1)}{=} (\,(a \cup b)^*\,(a \cup b)^*\,)^* \overset{(2)}{=} (a \cup b)^{**} \overset{(3)}{=} (a \cup b)^*$.

Problem (c). We have $(a^*b)^* \cup (b^*a)^* = (a \cup b)^*$.

We have $(a^*b)^* \cup (b^*a)^* \subseteq (a \cup b)^*$ $(\dagger)$ as any string over the alphabet $\{a, b\}$ is in the set $(a \cup b)^*$. We will argue that we also have $(a \cup b)^* \subseteq (a^*b)^* \cup (b^*a)^*$ $(\ddagger)$. It follows from $(\dagger)$ and $(\ddagger)$ that $(a^*b)^* \cup (b^*a)^* = (a \cup b)^*$.

Note that

$$(a^*b)^* = \{e\} \cup \{\, w \in \{a, b\}^* \,:\, w \text{ ends with } b \,\}$$

and

$$(b^*a)^* = \{e\} \cup \{\, w \in \{a, b\}^* \,:\, w \text{ ends with } a \,\}\,.$$

Since any nonempty string in the set $(a \cup b)^*$ either ends with an a or a b, we have $(a \cup b)^* \subseteq (a^*b)^* \cup (b^*a)^*$, that is, $(\ddagger)$ holds.

Problem 1.8.3

Problem (a). $b^* \cup b^*ab^* \cup b^*ab^*ab^* \cup b^*ab^*ab^*ab^*$.

Problem (b). The following solutions are acceptable:

- $b^*(\,ab^*ab^*ab^*\,)^*$
- $b^* \cup (\,b^*ab^*ab^*ab^*\,)^*$.

Problem (c). $(\,b \cup ab \cup aab\,)^*aaa(\,b \cup ab \cup aab\,)^*$.

Problem 1.8.5

(a) and (b) are true. (c) is false as $a^*b^* \cap b^*c^* = b^*$. (d) is also false.

Section 2.2

Problem 2.2.1

The automaton shown on the left accepts a, aa, and e, but not aab. The automaton shown on the right accepts e, ab, $abab$, and aba, but not $abaa$.

Problem 2.2.2

The automaton to the left accepts the language a^*. Here is two alternative expressions for the language accepted by the automaton to the right:

- $\emptyset^* \cup (ab \cup aba)(ab \cup aba)^*$
- $\emptyset^* \cup a(ba \cup aaa)^*(b \cup ba)$

Problem 2.2.3

We give the automata formally.

Problem (a). $(K, \Sigma, \Delta, s, F)$ where $K = \{s, q_1, q_2, p_2, q_3, p_3\}$, $\Sigma = \{a, b\}$,

$$\Delta = \{(s,a,q_1),(q_1,a,q_1),(s,e,q_2),(q_2,a,p_2),(p_2,b,q_2),(q_2,e,q_3),(q_3,b,p_3),(p_3,a,q_3)\}$$

and $F = \{q_1, q_3\}$.

Problem (b). $(K, \Sigma, \Delta, s, F)$ where $K = \{s, q\}$, $\Sigma = \{a, b\}$,

$$\Delta = \{(s,a,s),(s,a,q),(q,b,s)\}$$

and $F = \{s\}$.

To see that this (surprisingly simple) automaton indeed is correct, note that we have $(\alpha^*\beta^*)^* = (\alpha \cup \beta)^*$ for any regular espressions α and β. Thus, we have $((ab \cup aab)^*a^*)^* = (ab \cup aab \cup a)^* = (ab \cup a)^*$. It is easy to see that the automaton accepts the language given by $(ab \cup a)^*$.

Problem (c). $(K, \Sigma, \Delta, s, F)$ where $K = \{s, q, p\}$, $\Sigma = \{a, b\}$,

$$\Delta \;=\; \{(s, e, q), (q, a, q), (q, b, p), (p, e, q)\}$$

and $F = \{s, p\}$.

To see that the automaton is correct, note that

$$((a^*b^*a^*)^*b)^* \;=\; ((a \cup b \cup a)^*b)^* \;=\; ((a \cup b)^*b)^* \;=\; (a^*b)^* \,.$$

Problem 2.2.6

Problem (a). $(K, \Sigma, \Delta, s, F)$ where $K = \{s, p, q, r\}$, $\Sigma = \{a, b\}$,

$$\Delta \;=\; \{(s, a, p), (p, a, r), (p, b, s), (p, b, q), (q, a, s), (r, b, s)\}$$

and $F = \{s\}$.

Problem (b). $(K, \Sigma, \delta, s, F)$ where $K = \{s, q_1, q_2, p_1, p_2, p_3, r\}$, $\Sigma = \{a, b\}$, δ is give by the table below, and $F = \{s, p_1, p_2\}$.

q	a	$\delta(q, a)$
s	a	q_1
s	b	r
q_1	a	q_2
q_1	b	p_1
q_2	a	r
q_2	b	s
p_1	a	p_2
p_1	b	r
p_2	a	p_3
p_2	b	p_1
p_3	a	r
p_3	b	p_1
r	a	r
r	b	r

Problem 2.2.7

Problem (a). $(K, \Sigma, \Delta, s, F)$ where $K = \{q_0, q_1, q_2, q_3, q_4, q_5\}$, $\Sigma = \{a, b\}$,

$$\Delta \;=\; \{(q_0, a, q_0), (q_0, b, q_0), (q_0, a, q_1), (q_1, a, q_2), (q_2, b, q_3), (q_3, a, q_4), (q_4, b, q_5)\}$$

$s = q_0$, and $F = \{q_5\}$.

Problem (b). $(K, \Sigma, \delta, s, F)$ where $K = \{q_0, q_1, q_2, q_3, q_4, q_5\}$, $\Sigma = \{a, b\}$, δ is given by the table below, $s = q_0$, and $F = \{q_5\}$.

q	a	$\delta(q, a)$
q_0	a	q_1
q_0	b	q_0
q_1	a	q_2
q_1	b	q_0
q_2	a	q_2
q_2	b	q_3
q_3	a	q_4
q_3	b	q_0
q_4	a	q_2
q_4	b	q_5
q_5	a	q_0
q_5	b	q_0

Problem 2.2.9

(a) The automaton to the left. If we use the algorithm given in the proof of Theorem 2.2.1, we end up with the nondeterministic automaton $(K, \Sigma, \delta, s, F)$ where $K = \{\{q_0, q_1\}, \{q_0, q_1, q_2, q_4\}, \{q_0, q_1, q_3, q_4\}\}$, $\Sigma = \{a, b\}$, δ is given by the table below, $s = \{q_0, q_1\}$, and $F = \{\{q_0, q_1, q_2, q_4\}, \{q_0, q_1, q_3, q_4\}\}$.

q	a	$\delta(q, a)$
$\{q_0, q_1\}$	a	$\{q_0, q_1\}$
$\{q_0, q_1\}$	b	$\{q_0, q_1, q_2, q_4\}$
$\{q_0, q_1, q_2, q_4\}$	a	$\{q_0, q_1, q_2, q_4\}$
$\{q_0, q_1, q_2, q_4\}$	b	$\{q_0, q_1, q_3, q_4\}$
$\{q_0, q_1, q_3, q_4\}$	a	$\{q_0, q_1, q_3, q_4\}$
$\{q_0, q_1, q_3, q_4\}$	b	$\{q_0, q_1, q_2, q_4\}$

If we inspect the automaton, we will realize that the automaton accepts the language

$$L = \{ w \in \{a, b\}^* \mid w \text{ contains at least one } b \} .$$

The nondeterministic automaton M given below accepts L, but has only two states. This shows that the algorithm does not always give the smallest eqiuvalent nondeterministic automaton.

Let $M = (K, \Sigma, \delta, s, F)$ where $K = \{s, q\}$, $\Sigma = \{a, b\}$, δ is given by the table below, and $F = \{q\}$.

q	a	$\delta(q, a)$
s	a	s
s	b	q
q	a	q
q	b	q

Section 2.4

Problem 2.4.4

Let $L = \{a^\ell b a^m b a^{\ell+m} : n, m \geq 1\}$. We prove that L is not a regular language.

Assume that L is regular. By Theorem 2.4.1 (a pumping theorem) there exists a fixed number $n \geq 1$ such that any string $w \in L$ can be rewritten as $w = xyz$ such that (i) $y \neq e$, (ii) $|xv| \leq n$, and (iii) $xy^iz \in L$ for any $i \geq 0$. Let $u = a^nbaba^{n+1}$ where n is the fixed number given by Theorem 2.4.1. (So, u denote a particular fixed string.) Then, we have $u \in L$, and thus there exist x, y, z such that $u = xyz$ and (i), (ii), and (iii) hold. As (i) and (ii) hold, there will be only a's in y, and there will be at least one a in y, that is, we have $y = a^j$ for some $j \geq 1$. As (iii) holds, we have $xyyz \in L$. Now, $xyyz = a^{n+j}baba^{n+1}$, and hence, according to the definition of L, we have $xyyz \notin L$. This yield the desired contradiction. We have $xyyz \in L$ and $xyyz \notin L$. We conclude that L cannot be regular.

Section 3.1

In this section S always denote the start symbol of the grammar, and as usual, uppercase letters denote nonterminals, lowercase letters denote terminals, and e denotes the empty string.

Problem 3.1.2

- $S \to aAa$ (rule 1)
- $S \to bAb$ (rule 2)
- $S \to e$ (rule 3)
- $A \to SS$ (rule 4)

$$S \overset{2}{\Rightarrow} bAb \overset{4}{\Rightarrow} bSSb \overset{2}{\Rightarrow} bSbAbb \overset{4}{\Rightarrow} bSbSSbb \overset{3}{\Rightarrow} bSbSbb \overset{3}{\Rightarrow} bSbbb \overset{1}{\Rightarrow}$$
$$baAabbb \overset{4}{\Rightarrow} baSSabbb \overset{3}{\Rightarrow} baSabbb \overset{3}{\Rightarrow} baabbb\,.$$

Problem 3.1.3

Problem (a). $S \to aSa$, $S \to bSb$, and $S \to c$.

Problem (b). $S \to aSa$, $S \to bSb$, and $S \to e$.

Problem (c). $S \to aSa$, $S \to bSb$, $S \to a$, $S \to b$, and $S \to e$.

Problem 3.1.5

For later reference, we enumerate rules of the grammar.

1. $S \to aB$
2. $S \to bA$
3. $A \to a$
4. $A \to aS$

5. $A \to BAA$
6. $B \to b$
7. $B \to bS$
8. $B \to ABB$

Problem (a).

$$S \overset{1}{\Rightarrow} aB \overset{7}{\Rightarrow} abS \overset{1}{\Rightarrow} abaB \overset{7}{\Rightarrow} ababS \overset{2}{\Rightarrow} ababbA \overset{3}{\Rightarrow} ababba\,.$$

Problem (b). For any $\alpha \in V$ and any $w \in V^\star$, let $(\#_\alpha\, w)$ denote the number of occurrences of α in w. Let $(\#_{\alpha,\beta}\, w) = (\#_\alpha\, w)+(\#_\beta\, w)$. E.g., we have $(\#_a\, aBAA) = 1$ and $(\#_{a,A}\, aBAA) = 3$. Let $L = \{w \in \Sigma^\star : (\#_a\, w) = (\#_b\, w)\}$. We prove that $L = L(G)$ by proving $L(G) \subseteq L$ and $L \subseteq L(G)$. (The proof of the latter inclusion is difficult.)

Lemma 1 *If $S \overset{\star}{\Rightarrow} w$, then $(\#_{a,A}\, w) = (\#_{b,B}\, w)$.*

The inclusion $L(G) \subseteq L$ follows easily Lemma 1: Assume $w \in L(G)$. By the definition of $L(G)$, we have $S \overset{\star}{\Rightarrow} w$. By Lemma 1, we have $(\#_{a,A}\, w) = (\#_{b,B}\, w)$. Since $w \in \{a,b\}^*$ (w does not contain nonterminals), we have $(\#_a\, w) = (\#_b\, w)$, and hence $w \in L$. The lemma is proved by induction on the number of steps in the derivation $S \Rightarrow \cdots \Rightarrow w$. We skip the details. This completes the proof of the inclusion $L(G) \subseteq L$.

In order to prove $L \subseteq L(G)$ we need the next lemma.

Lemma 2 *Let $w \in \{a,b\}^\star$.*

1. *If $(\#_a\, w) = (\#_b\, w)$, then $S \overset{\star}{\Rightarrow} w$.*
2. *If $(\#_a\, w) = (\#_b\, w) + 1$, then $A \overset{\star}{\Rightarrow} w$.*
3. *If $(\#_a\, w) + 1 = (\#_b\, w)$, then $B \overset{\star}{\Rightarrow} w$.*

The inclusion $L \subseteq L(G)$ follows easily from Clause 1 of the lemma: Assume $w \in L$. By the definition of L, we have $(\#_a\, w) = (\#_b\, w)$, and then the lemma says that $S \overset{\star}{\Rightarrow} w$. Hence, by the definition of $L(G)$, we have $w \in L(G)$. This proves the inclusion $L(G) \subseteq L$. We turn to the proof of the lemma.

Proof of Lemma 2. We use induction on the length of w.

Assume $|w| = 1$. Proof of 1: Clause 1 holds trivially since $(\#_a\, w) \neq (\#_b\, w)$. Proof of 2: Assume $(\#_a\, w) = (\#_b\, w) + 1$. Then $w = a$, and we have $A \overset{\star}{\Rightarrow} w$ by as the grammar contains the rule $A \to a$. Proof of 3: Assume $(\#_a\, w) + 1 = (\#_b\, w)$. Then $w = b$, and we have $B \overset{\star}{\Rightarrow} w$ as we have the rule $B \to b$.

Assume $|w| > 1$. The proof slits into the cases $w = ax$ and $w = bx$. We will carry out the details for the case $w = ax$. (The proof when $w = bx$ is symmetric.) So assume $w = ax$. For any $v \in \{a,b\}^*$ such that $|v| < |w|$, we have

(IH1) If $(\#_a\, x) = (\#_b\, x)$, then $S \overset{\star}{\Rightarrow} x$.

(IH2) If $(\#_a\, x) = (\#_b\, x) + 1$, then $A \overset{\star}{\Rightarrow} x$.

(IH3) If $(\#_a\, x) + 1 = (\#_b\, x)$, then $B \stackrel{\star}{\Rightarrow} x$.

by the induction hypothesis.

First we argue that Clause 1 of the lemma holds. Assume $(\#_a\, ax) = (\#_b\, ax)$. It follows that $(\#_a\, x)+1 = (\#_b\, x)$. By (IH3), we have $B \stackrel{\star}{\Rightarrow} x$. The grammar contains the rule $S \to aB$. Hence, we have $S \Rightarrow aB \stackrel{\star}{\Rightarrow} ax$, that is, $S \stackrel{\star}{\Rightarrow} ax$.

Next we argue that Clause 2 of the lemma holds. Assume $(\#_a\, ax) = (\#_b\, ax) + 1$. This implies $(\#_a\, x) = (\#_b\, x)$. By (IH1), we have $S \stackrel{\star}{\Rightarrow} x$. The grammar contains the rule $A \to aS$. Hence, we have $A \stackrel{\star}{\Rightarrow} ax$ since $A \Rightarrow aS \stackrel{\star}{\Rightarrow} ax$.

Finally, we prove Clause 3. Assume $(\#_a\, ax) + 1 = (\#_b\, ax)$. This implies $(\#_a\, x) + 2 = (\#_b\, x)$. There exists a splitting $x = yz$ such that $(\#_a\, y) + 1 = (\#_b\, y)$ and $(\#_a\, z) + 1 = (\#_b\, z)$. (There will always be at least one splitting satisfying the two equations.) By (IH3), we have $B \stackrel{\star}{\Rightarrow} y$ and $B \stackrel{\star}{\Rightarrow} z$. Hence,

$$B \stackrel{8}{\Rightarrow} ABB \stackrel{2}{\Rightarrow} aBB \stackrel{\star}{\Rightarrow} ayB \stackrel{\star}{\Rightarrow} ayz = ax$$

and we have $B \stackrel{\star}{\Rightarrow} ax$ as desired. This completes the proof of Lemma 2. ■

Problem 3.1.7

Let $L = \{w \in \{a,b\}^* : |w| \text{ is even}\}$. It is easy to see that L is a regular language. Thus, it is sufficient to show that $L(G) = L$.

First, we prove the inclusion $L(G) \subseteq L$. Assume $S \stackrel{*}{\Rightarrow} w$ where $w \in V^*$. Further, assume that w can be derived from S in n steps. We prove by induction on n that w contains an even number of terminal symbols.

Induction start, $n = 0$: Then $w = S$, and thus, w contains 0 terminals. (0 is an even number.)

Induction step, $n > 0$: Then we have a derivation on the form $S \Rightarrow w' \stackrel{*}{\Rightarrow} w$ where the first rule applied is one of the rules $S \to e$, $S \to aSb$, $S \to aSa$, $S \to bSa$, $S \to bSb$. The proofs splits into cases, one for each of the five rules. *Case $S \to e$:* $w = w' = e$, and thus w contains 0 nonterminals. *Case $S \to aSb$:* $w' = aSb$, and $w = axb$ where $x \in \{a,b\}^*$ can be derived from S in $n-1$ steps. By the induction hypothesis x contains an even number of terminals, and hence w, which has two additional terminals, also contains an even number of terminals. The cases for the rules $S \to aSa$, $S \to bSa$, and $S \to bSb$ are similar to case for the rule $S \to aSb$.

This proves any string in the language $L(G)$ is of even length, and hence we have $L(G) \subseteq L$. We will now prove the inclusion $L \subseteq L(G)$.

Let $w \in L$. (We will prove that $w \in L(G)$.) Then $|w| = 2n$ for some $n \geq 0$. We will prove by induction on n that $S \stackrel{*}{\Rightarrow} w$.

Induction start $n = 0$: Then $w = e$, and we have the derivation $S \Rightarrow e$.

Induction step $n > 0$: Then there exists $x \in \{a,b\}^*$ such that w is of one of the forms (i) $w = axb$, (ii) $w = axa$, (iii) $w = bxa$, or (iv) $w = bxb$. *Case (i) $w = axb$:* We have $|x| = 2n - 2$, and thus, by the induction hypothesis, we have $S \stackrel{*}{\Rightarrow} x$. Further, since we have the rule $S \to aSb$, we get $S \Rightarrow aSb \stackrel{*}{\Rightarrow} axb = w$. This proves that $S \stackrel{*}{\Rightarrow} w$. *Case (ii) $w = axa$:* Similar to (i), apply the rule $S \to aSa$ in the start of the derivation. *Case (iii) $w = bxa$:* Similar to (i), apply the rule $S \to bSa$ in the start of the derivation. *Case (iv) $w = bxb$:* Similar to (i), apply the rule $S \to bSb$ in the start of the derivation.

This, proves that any $w \in L$ can be derived from the start symbol S, that is, we have $L \subseteq L(G)$.

Problem 3.1.9

Problem (a). $S \to aSb$, $S \to aS$, and $S \to e$.

Problem (b). $S \to aSd$, $S \to A$, $S \to B$, $A \to aAc$, $A \to C$, $B \to bBd$, $B \to C$, $C \to bBc$, and $C \to e$.

Problem (b). Let G be the grammar given by the rules $S \to e$, $S \to SaSbSbS$, $S \to SbSaSbS$, and $S \to SbSbSaS$.

Let $L = \{w \in \{a,b\}^* \ : \ w \text{ has twice as many } b\text{'s as } a\text{'s}\}$. It is easy to see that $L(G) \subseteq L$. To verify that $L \subseteq L(G)$ observe that if $w \in L$, then there will exist $v, x, y, z \in L$ such that at least one of the following equalities holds

- $w = vaxbybz$
- $w = vbxaybz$
- $w = vbxbyaz$.

E.g. the string $bbbaba \in L$ has the splitting $bbbaba = vbxbyaz$ where $x = bba$ and $v = y = z = e$ and $v, x, y, z \in L$. Thus, the string can be derived by an derivation starting with an application of the rule $S \to SbSbSaS$.

Problem (d). $S \to Ab$, $A \to aAa$, $A \to aAb$, $A \to bAa$, $A \to bAb$, and $A \to a$.

Problem (e). $S \to AcS$, $S \to aTa$, $S \to bTb$, $T \to aTa$, $T \to bTb$, $T \to cU$, $U \to AcU$, $U \to c$, $A \to aA$, $A \to bA$, $A \to a$, and $A \to b$.

To see that the grammar is correct observe that we e.g. can derive the string $AcAcAcabbcAcAccbba$.

$$\begin{aligned} S \Rightarrow AcS \Rightarrow AcAcS \Rightarrow AcAcAcS \Rightarrow AcAcAcaTa \Rightarrow AcAcAcabTba \Rightarrow \\ AcAcAcabbTbba \Rightarrow AcAcAcabbcUbba \Rightarrow AcAcAcabbcAcUbba \Rightarrow \\ AcAcAcabbcAcAcUbba \Rightarrow AcAcAcabbcAcAccbba \, . \end{aligned}$$

Further, observe that the string has the form $AcAcAcwcAcAccw^R$, and that any string in the language $\{a,b\}^+$ can be derived from the nonterminal A.

Problem (e). $S \to e$, $S \to AASb$, $A \to a$, and $A \to e$.

Section 3.5

Problem 3.1.9

Problem (a). Let $L = \{a^p \ : \ p \text{ is a prime}\}$. We show that L is not a context-free language.

Assume that L is context-free, and let w be a sufficiently long string in L. Theorem 3.5.3 says w can be rewritten as $w = uvxyz$ such that (i) $|vy| \geq 1$ and (ii) $uv^n xy^n z \in L$ for any $n \geq 0$. Let k denote the length of vy, and let ℓ denote the length of uxz, that is, $k = |vy| \geq 1$ and $\ell = |uxz|$. By (ii) we have $uv^{\ell+2}xy^{\ell+2}z \in L$. Now, $|uv^{\ell+2}xy^{\ell+2}z| = k(\ell+2) + \ell = (k+1)(\ell+2)$. Thus, according to the definition of L, the number $(k+1)(\ell+2)$ is a prime. But according to (i) we have $k \geq 1$, and hence $(k+1)(\ell+2)$ is not a prime. Thus we have a contradiction. (Recall that a number a is prime iff $a \neq (b+2)(c+2)$ for any $b, c \geq 0$.)

Since our assumption that L is context-free implies a contradiction, we conclude that L is not a context-free language.

Problem (c). Let $L = \{www : w \in \{a,b\}^*\}$. We prove that L is not a context-free language.

Lemma 3 *The language* $\{a^n ba^n ba^n b : n \geq 0\}$ *is not context-free.*

Proof of Lemma 3. Use Theorem 3.5.3. We skip the details. ■

Assume L is a context-free language. Let $L_0 = L \cap a^*ba^*ba^*b$. Then we have $L_0 = \{a^n ba^n ba^n b : n \geq 0\}$. By Theorem 3.5.2, L_0 is a context-free language, by the lemma above, L_0 is not a a context-free language. Since the assumption that L is context-free implies a contradiction, we conclude that L is not a context-free language.

Problem (d). Let

$$L = \{w \in \{a,b,c\}^* : w \text{ has equal numbers of } a\text{'s, } b\text{'s and } c\text{'s.}\} .$$

We prove that L is not a context-free language.

Lemma 4 *The language* $\{a^n b^n c^n : n \geq 0\}$ *is not context-free.*

The lemma is proved in Example 3.5.2 on page 146.

Assume L is context-free. Let $L_0 = L \cap a^*b^*c^*$. Then $L_0 = \{a^n b^n c^n : n \geq 0\}$ By Theorem 3.5.2 L_0 is a context-free language, by the lemma above L_0 is not a a context-free language. Since the assumption that L is context-free implies a contradiction, we conclude that L is not a context-free language.