Context Free Languages

Last Updated: February 15th, 2023

1 Introduction

Context free languages are foundational for defining several types of computing languages that occur in practice; including programming languages, markup languages, and languages for communication protocols. CFL's were first studied in relation to natural-language processing in the 1950's.

Their importance stems from the following.

1. CFL's are significantly more expressive than regular languages in that they are capable of defining recursive languages that may have unlimited recursive depth.
2. a CFL can be recognized by a pushdown automaton (PDA). Unlike DFA's a PDA has unlimited memory, albeit in the form of a stack whose access is limited to i) reading the top of the stack, ii) pushing on to the stack, and iii) popping the top of the stack. PDA's are also interesting because their nondeterministic counterparts (NPDA's) are more powerful than PDA's, and the set of languages accepted by an NPDA is equal to the set of CFL languages.

Although every regular language is also a CFL (see the exercises), the converse is not true. For example, it can be proved that the language

$$L = \{a^n b^n | n \geq 0\}$$

is a CFL but is *not* regular. Intuitively, L is not regular because a DFA M would have to keep track of the number a's read and them make sure that the same number of b's are subsequently read. However, the number of a's read can grow without bound and exceed M's finite and bounded memory capacity. Example 7 below shows that L is a CFL.

2 Context-Free Grammars

A **Context-Free Grammar (CFG)** is a 4-tuple (V, Σ, R, S), where

1. V is a finite set of variables
2. Σ is a finite set that is disjoint from V, called the **terminal set**
3. R is a finite set of rules where each **rule** has the form

   $$A \to s,$$

   where $A \in V$ and $s \in (V \cup \Sigma)^*$. Variable A is referred to as the **head** of the rule, while s is referred to its **body**.
4. $S \in V$ is the **start variable**

Example 1. Consider the set of rules

$$R = \{S \to SS, S \to aSb, S \to \varepsilon\}.$$

Then we may use this set of rules to define a CFG $G = (V, \Sigma, R, S)$, where

$$V = \{S\},$$

$$\Sigma = \{a, b\},$$

and variable S is the start variable.

For brevity we may list together rules having the same head as follows.

$$S \to SS \mid aSb \mid \varepsilon.$$

Here, each of the rule bodies is separated by a vertical bar. □

Example 2. One common use of CFG's is to provide grammatical formalism for natural languages. For example, consider the set of rules R:

$$\langle \text{SENTENCE} \rangle \to \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle$$

$$\langle \text{NOUN-PHRASE} \rangle \to \langle \text{COMPLEX-NOUN} \rangle \mid \langle \text{COMPLEX-NOUN} \rangle \langle \text{PREP-PHRASE} \rangle$$

$$\langle \text{VERB-PHRASE} \rangle \to \langle \text{COMPLEX-VERB} \rangle \mid \langle \text{COMPLEX-VERB} \rangle \langle \text{PREP-PHRASE} \rangle$$

$$\langle \text{PREP-PHRASE} \rangle \to \langle \text{PREP} \rangle \langle \text{COMPLEX-NOUN} \rangle$$

$$\langle \text{COMPLEX-NOUN} \rangle \to \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle$$

$$\langle \text{COMPLEX-VERB} \rangle \to \langle \text{VERB} \rangle \mid \langle \text{VERB} \rangle \langle \text{NOUN-PHRASE} \rangle$$

$$\langle \text{ARTICLE} \rangle \to \text{a} \mid \text{the}$$

$$\langle \text{NOUN} \rangle \to \text{trainer} \mid \text{dog} \mid \text{whistle}$$

$$\langle \text{VERB} \rangle \to \text{calls} \mid \text{pets} \mid \text{sees}$$

$$\langle \text{PREP} \rangle \to \text{with} \mid \text{in}$$

Here, the variables are the ten parts of speech delimited by $\langle\ \rangle$, Σ is the lowercase English alphabet, including the space character, and $\langle \text{SENTENCE} \rangle$ is the start variable.

Example 3. A CFG may also be used to define the syntax of a programming language. One fundamental language component to any programming language is that of an *expression*. The following rules imply a CFG for defining expressions formed by a single terminal `a`, parentheses, and the two arithmetic operations $+$ and $\times$. Here E stands for `expression`, T for `term`, and F for `factor`.

$$E \to E + T \mid T$$
$$T \to T \times F \mid F$$
$$F \to (E) \mid a$$

We have $V = \{E, T, F\}$, $\Sigma = \{+, \times, a, (,)\}$, and E is the start variable.

2.1 Grammar derivations

Let $G = (V, \Sigma, R, S)$ be a CFG, then the language $D(G) \in (V \cup \Sigma)^*$ of **derived words** is structurally defined as follows.

Atom $S \in D(G)$.

Compound Rule Suppose $s \in D(G)$, s is of the form uAv for some $u, v \in (V \cup \Sigma)^*$, $A \in V$, and $A \to \gamma$ is a rule of G, then

$$u\gamma v \in D(G).$$

In this case we write $s \Rightarrow u\gamma v$, and say that s **yields** $u\gamma v$. In words, to get a new derived word, take an existing derived word and replace one of its variables A with the body of a rule whose head is A.

The subset $L(G)$ of derived words $w \in D(G)$ for which $w \in \Sigma^*$ is called the **context-free language (CFL)** associated with G. Thus, the words of $L(G)$ consist only of terminal symbols.

2.2 The Derivation relation

Let u and v be words in $(V \cup \Sigma)^*$. We say that u **derives** v, written $u \stackrel{*}{\Rightarrow} v$ if and only if either $u = v$ or there is a sequence of words $w_1, w_2, \ldots, w_n$ such that

$$u = w_1 \Rightarrow w_2 \Rightarrow w_3 \Rightarrow \cdots \Rightarrow w_n = v.$$

Such a sequence is called a **derivation sequence** from u to v.

$$L(G) = \{w \in \Sigma^* | S \stackrel{*}{\Rightarrow} w\}.$$

Example 4. Use the CFG from Example 1 to derives the word `aabbaababb`.

$$S \to SS \mid aSb \mid \varepsilon.$$

Solution.

2.3 Derivation parse trees

Determining if an arbitrary word belongs to $L(G)$ is of fundamental importance. But in addition, it is sometimes important to know the structure of the grammar's derivation of the word. For example, if a CFG generates arithmetic expressions, then knowing the structure of the derivation allows one to readily evaluate the expression (assuming the expression terminals have assigned values and the expression operations are properly defined). A **parse tree** for a word $w \in L(G)$ is a tree whose structure and node labels reflect the derivation w, where, from left to right, the leaves of the tree are labeled with the letters of w. Indeed, consider the derivation sequence

$$S = w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow w_n = w.$$

Then the parse tree for w can be defined in a step-by-step manner. To begin the parse tree T_1 for $S = w_1$ consists of a single node labeled with S.

Now suppose a parse tree T_k has been associated with w_k, the k th word of the derivation. Moreover, assume that, from left to right, the leaves of T_k are labeled in one-to-one correspondence with the symbols of w_k. Moreover, assume that w_k has the form $w_k = uAv$, where A is substituted for a word γ, so that $w_{k+1} = u\gamma v$. Then T_{k+1} is obtained from T_k by assigning the leaf node labeled with A a number of children equal to the length of γ and for which, from left to right, the i th child is labeled with the ith symbol of γ.

Example 5. Use the CFG from Example 3 to derive the expression $a \times (a + a)$, and provide the parse tree associated with the derivation.

$$E \to E + T \mid T$$

$$T \to T \times F \mid F$$

$$F \to (E) \mid a$$

Solution.

2.4 Ambiguity

Given a CFG G, and a word $w \in L(G)$, there may be several different derivations of w from start symbol S. Many of these derivations however will yield identical parse trees. But in the event that two different derivation sequences of w from S yield two different parse trees, then we call G **ambiguous**. It turns out that an easy way to check for ambiguity is to check that no word w has more than one *leftmost derivation*.

Given grammar $G = (V, \Sigma, R, S)$ and word $w \in L(G)$, a derivation sequence $S = w_0 \Rightarrow w_1 \Rightarrow \cdots \Rightarrow w_{n-1} \Rightarrow w_n = w$ is called a **leftmost derivation** of w provided that, for every $0 \leq i \leq n-1$, the yielding of w_i from w_{i-1} was obtained by replacing the leftmost variable A of w_{i-1} with the body of one of a rule whose head is A. Therefore, if w has more than one leftmost derivation, it must be the case that a different sequence of rules were used to derive w. When this happens we call G ambiguous, since some words in the grammar have more than one parsing structure.

Example 6. Show that the grammar defined by the following rules is ambiguous.

$$\langle \text{SENTENCE} \rangle \to \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle$$

$$\langle \text{NOUN-PHRASE} \rangle \to \langle \text{COMPLEX-NOUN} \rangle \mid \langle \text{COMPLEX-NOUN} \rangle \langle \text{PREP-PHRASE} \rangle$$

$$\langle \text{VERB-PHRASE} \rangle \to \langle \text{COMPLEX-VERB} \rangle \mid \langle \text{COMPLEX-VERB} \rangle \langle \text{PREP-PHRASE} \rangle$$

$$\langle \text{PREP-PHRASE} \rangle \to \langle \text{PREP} \rangle \langle \text{COMPLEX-NOUN} \rangle$$

$$\langle \text{COMPLEX-NOUN} \rangle \to \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle$$

$$\langle \text{COMPLEX-VERB} \rangle \to \langle \text{VERB} \rangle | \langle \text{VERB} \rangle \langle \text{NOUN-PHRASE} \rangle$$

$$\langle \text{ARTICLE} \rangle \to \text{a} \mid \text{the}$$

$$\langle \text{NOUN} \rangle \to \text{trainer} \mid \text{dog} \mid \text{whistle}$$

$$\langle \text{VERB} \rangle \to \text{calls} \mid \text{pets} \mid \text{sees}$$

$$\langle \text{PREP} \rangle \to \text{with} \mid \text{in}$$

Solution.

Solution Continued.

Example 7. Provide a CFG G for which

$$L(G) = \{a^n b^n | n \geq 0\}.$$

Provide a derivation of a^3b^3 and draw its parse tree.

Example 8. Provide a CFG G for which

$$L(G) = \{x^i y^j z^k | i, j, k \geq 0 \text{ and } j = 2i \text{ or } k = 2j\}.$$

Example 9. Use the CFG from the previous example to provide a derivation of $x^2y^4z^3$ and draw its parse tree.

Exercises

1. For the CFG defined in Example 1, provide a derivation for the following words.

   a. ababab

   b. aaababbbab

   c. aababaabbbaabb

2. For the CFG defined in Example 3, provide a derivation and parse tree for the following expressions.

   a. a

   b. $a + a$

   c. $a \times (a \times a)$

   d. $((a))$

3. For the CFG in Example 6, provide a leftmost derivation and parse tree for the sentence "the trainer calls the dog with the whistle", where we assume that it is the *dog* that possesses the whistle.

4. Provide context-free grammars for each of the following languages.

   a. The set of binary words that contain at least three 1's.

   b. The set of binary words that begin and end with the same symbol.

   c. The set of binary words having odd length.

   d. The set of binary words having odd length and for which the middle bit is 0.

   e. The set of binary words that are palindromes (i.e. read the same forward as backwards). For example, 110011 is a palindrome, but 110 is not.

   f. The empty set.

   g. The set of words over $\{\mathrm{a,b}\}^*$ for which there are more a's than b's.

   h. The complement of the language $\{\mathrm{a}^n\mathrm{b}^n | n \geq 0\}$.

   i. The set of words of the form $w\#x$, where w and x are both binary words and w^r is a substring of x, where w^r denotes the reverse of w. For example, $(00111)^r = 11100$.

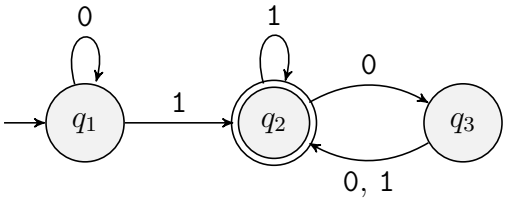
5. Prove that the union of two CFL's is also a CFL. Hint: how to take the "union" of two CFG's?

6. Given a DFA $M = (Q, \Sigma, \delta, q_0, F)$ we may define a CFG $G = (Q, \Sigma, R, q_0)$ for which $L(G) = L(M)$, where, for every two states $q_1, q_2 \in Q$ and $s \in \Sigma$ where $\delta(q_1, s) = q_2$, we have the *state-transition rule*

   $$q_1 \to sq_2.$$

   Also, for every $q \in F$, we have the ε-rule $q \to \varepsilon$. The idea behind G is that a derivation mimics a computation of M on some input word w. The ε-rules allow for only terminal words to be derived that are accepted by M. We may thus conclude that all regular languages are context free.

Provide the CFG in the case that M is the DFA shown below.



Provide a derivation of the word $w = 010100 \in L(M)$.

Exercise Solutions

1. We have the following derivations.

   a. ababab

   $$\text{S} \Rightarrow \text{SS} \Rightarrow \text{aSbS} \Rightarrow \text{abS} \Rightarrow \text{abSS} \Rightarrow \text{abaSbS} \Rightarrow \text{ababS} \Rightarrow \text{ababaSb} \Rightarrow \text{ababab}.$$

   b. aaababbbab

   $$\text{S} \Rightarrow \text{SS} \Rightarrow \text{aSbS} \Rightarrow \text{aaSbbS} \Rightarrow \text{aaSSbbS} \Rightarrow \text{aaaSbSbbS} \Rightarrow \text{aaabSbbS} \Rightarrow \text{aaabaSbbbS}.$$
   $$\Rightarrow \text{aaababbbS} \Rightarrow \text{aaababbbaSb} \Rightarrow \text{aaababbbab}.$$

   c. aababaabbbaabb

   $$\text{S} \Rightarrow \text{SS} \Rightarrow \text{aSbS} \Rightarrow \text{aSSbS} \Rightarrow \text{aaSbSbS} \Rightarrow \text{aabSbS} \Rightarrow \text{aabSSbS}$$
   $$\Rightarrow \text{aabaSbSbS} \Rightarrow \text{aababSbS} \Rightarrow \text{aababaSbbS} \Rightarrow \text{aababaaSbbbS} \Rightarrow \text{aababaabbbS}$$
   $$\Rightarrow \text{aababaabbbaSb} \Rightarrow \text{aababaabbbaaSbb} \Rightarrow \text{aababaabbbaabb}.$$

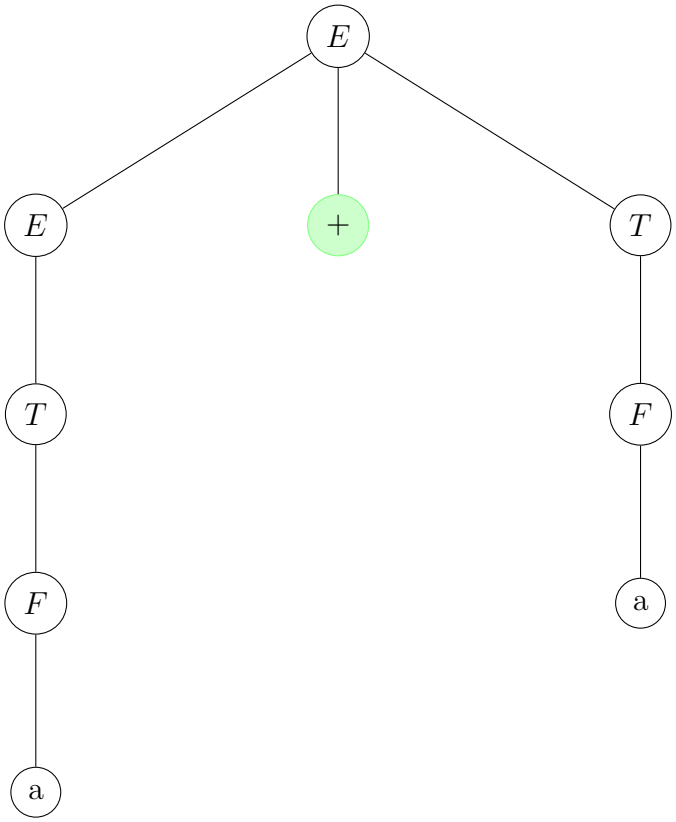
2. For the CFG defined in Example 3, provide a derivation and parse tree for the following expressions.

   a. a

   $$E \Rightarrow T \Rightarrow F \Rightarrow \text{a}.$$

b. $a + a$

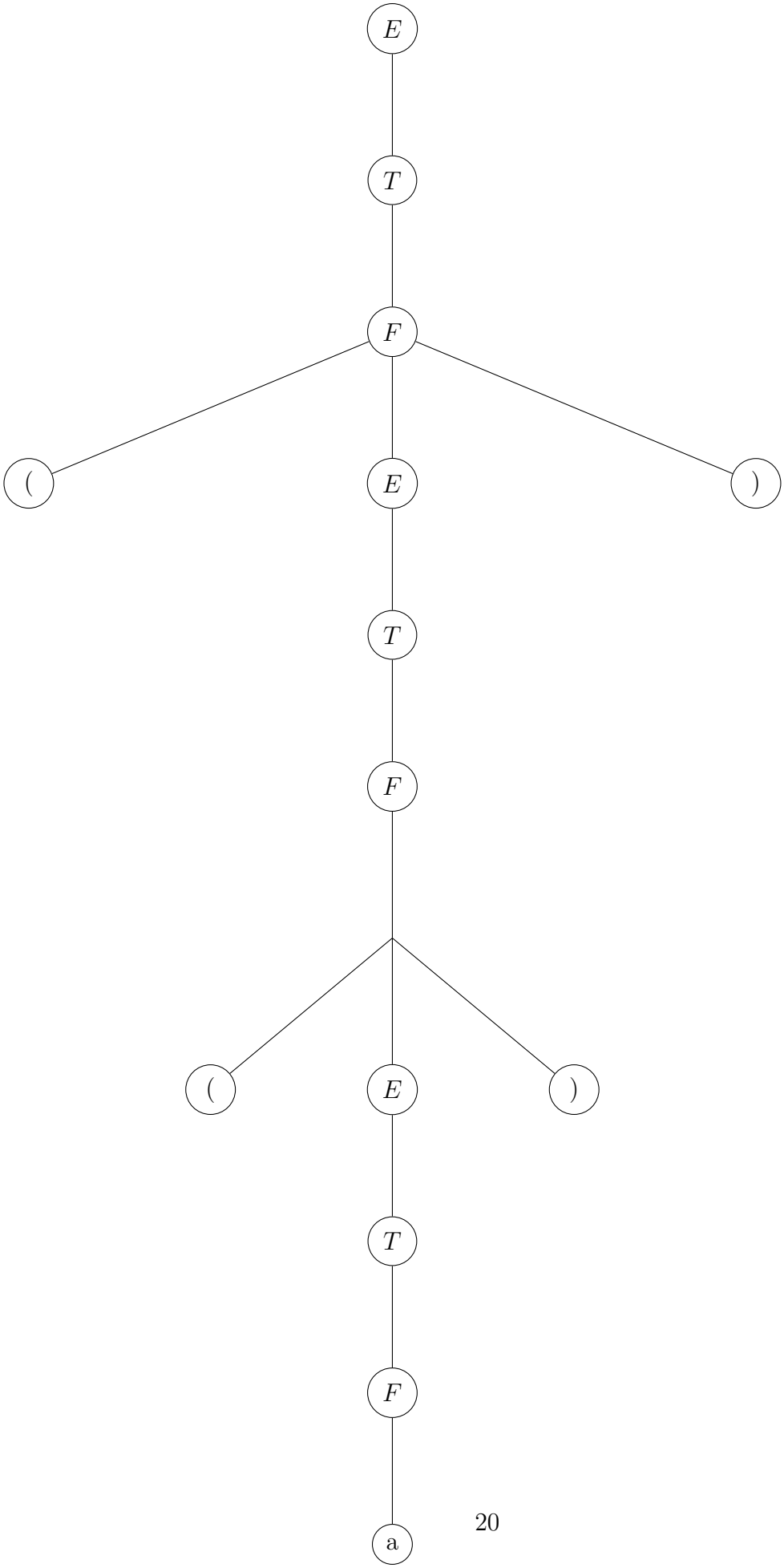
$$E \Rightarrow E + T \Rightarrow T + T \Rightarrow F + T \Rightarrow \mathrm{a} + T \Rightarrow \mathrm{a} + F \Rightarrow \mathrm{a} + \mathrm{a}.$$



c. $a \times (a \times a)$

d. $((a))$

$$E \Rightarrow T \Rightarrow F \Rightarrow (E) \Rightarrow (T) \Rightarrow (F) \Rightarrow ((E)) \Rightarrow ((T)) \Rightarrow ((F)) \Rightarrow ((\mathrm{a})).$$


E
T
F
(
E
)
T
F
(
E
)
T
F
a

3. The following is a leftmost derivation.

$$\langle \text{SENTENCE} \rangle \Rightarrow \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle \Rightarrow$$

$$\langle \text{COMPLEX-NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \Rightarrow$$

$$\langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \Rightarrow$$

$$\text{the } \langle \text{NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \Rightarrow$$

$$\text{the trainer } \langle \text{VERB-PHRASE} \rangle \Rightarrow$$

$$\text{the trainer } \langle \text{COMPLEX-VERB} \rangle \Rightarrow$$

$$\text{the trainer } \langle \text{VERB} \rangle \langle \text{NOUN-PHRASE} \rangle \Rightarrow$$

$$\text{the trainer calls } \langle \text{NOUN-PHRASE} \rangle \Rightarrow$$

$$\text{the trainer calls } \langle \text{COMPLEX-NOUN} \rangle \langle \text{PREP-PHRASE} \rangle \Rightarrow$$

$$\text{the trainer calls } \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \langle \text{PREP-PHRASE} \rangle \Rightarrow$$

$$\text{the trainer calls the } \langle \text{NOUN} \rangle \langle \text{PREP-PHRASE} \rangle \Rightarrow$$

$$\text{the trainer calls the dog } \langle \text{PREP-PHRASE} \rangle \Rightarrow$$

$$\text{the trainer calls the dog } \langle \text{PREP} \rangle \langle \text{COMPLEX-NOUN} \rangle \Rightarrow$$

$$\text{the trainer calls the dog with } \langle \text{COMPLEX-NOUN} \rangle \Rightarrow$$

$$\text{the trainer calls the dog with } \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \Rightarrow$$

$$\text{the trainer calls the dog with the } \langle \text{NOUN} \rangle \Rightarrow$$

$$\text{the trainer calls the dog with the whistle.}$$

4. The following are the rule sets (answers may vary!) that show each language is a CFL.

   a. $S \to Z1Z1Z1B$
   $Z \to 0Z \mid \varepsilon$
   $B \to 0B \mid 1B \mid \varepsilon$

   b. $S \to 0B0 \mid 1B1 \mid \varepsilon$
   $B \to 0B \mid 1B \mid \varepsilon$

   c. $S \to 0B \mid 1B$
   $B \to 00B \mid 01B \mid 10B \mid 11B \mid \varepsilon$

   d. $S \to 0S0 \mid 0S1 \mid 1S0 \mid 1S1 \mid 0$

   e. $S \to 0S0 \mid 1S1 \mid 0 \mid 1 \mid \varepsilon$

   f. Any CFG (V, Σ, R, S) for which $R = \emptyset$

g. We say that a word w in {a,b}* is **minimally balanced** iff it has an equal number of a's and b's and there is no prefix of w that also has an equal number of a's and b's. For example aaababbabb is minimally balanced, but aabbaaabbb is not since prefix aabb also has an equal number of a's and b's. Now consider a word w where there are more a's than b's. If w begins with an 'a', then no prefix of w (including w) can be minimally balanced. On the other hand, if it begins with a 'b', then it can be written as $w = m_1 m_2 \cdots m_k a u$ where each m_i is a minimally-balanced word, $k \geq 1$, and u is a word with at least as many a's as b's. Therefore, we may derive w with the following CFG rules.

$S \rightarrow \mathrm{a}A \mid M\mathrm{a}U$
$A \rightarrow \mathrm{a}A \mid \varepsilon$
$M \rightarrow \mathrm{a}M\mathrm{b} \mid \mathrm{b}M\mathrm{a} \mid MM \mid \varepsilon$
$U \rightarrow UU \mid A\mathrm{a}AU A\mathrm{b}A \mid A\mathrm{b}AU A\mathrm{a}A \mid \varepsilon$

The rule $S \rightarrow \mathrm{a}A$ can generate words having only a's, while the rule $S \rightarrow M\mathrm{a}U$, allows one to generate one or more balanced words using M, followed by an unbalanced word that begins with 'a'. The rules headed by U are almost the same as the rules headed by M (which in turn is similar to the rules in Example 1), with the exception that they use multiple occurrences of variable A so that additional a's can be inserted where needed.

h. The desired CFL is the union of the following CFL's: i) one or more a's and no b's, ii) one or more a's followed by one or more b's, followed by an a, followed by any word from {a,b}*, iii) $\{a^m b^n | 0 < m < n \text{ or } 0 < n < m\}$, iv) one or more b's, followed by any word from {a,b}*. Take the union of their corresponding CFG's.

i. $S \rightarrow TB$
$T \rightarrow 0T0 \mid 1T1 \mid \#$
$B \rightarrow 0B \mid 1B \mid \varepsilon$

5. Suppose L_1 and L_2 are CFL's, where $L_1 = L(G_1)$ and $L_2 = L(G_2)$, where $L_1 = (V_1, \Sigma_1, R_1, S_1)$ and $L_2 = (V_1, \Sigma_1, R_1, S_1)$. Without loss of generality, we may assume that $V_1 \cap V_2 = \emptyset$, which in turn implies that $R_1 \cap R_2 = \emptyset$. Then define the grammar

$$G = (V_1 \cup V_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, R_1 \cup R_2 \cup \{S \rightarrow S_1 \mid S_2\}, S),$$

where $S \notin V_1 \cup V_2$. The idea is that the rule $S \rightarrow S_1$ will allow for all words in L_1 to be derived, and the same is true for rule $S \rightarrow S_2$ and L_2. Moreover, since V_1 and V_2 are disjoint, it is impossible to derive any other word that does not belong in either L_1 or L_2. Therefore, $L(G) = L_1 \cup L_2$ is a CFL. □

6.