CSci 311, Models of Computation
Chapter 8
Properties of Context-Free Languages

H. Conrad Cunningham

29 December 2015

Contents

Acknowledgements: These lecture notes are for use with Chapter 8 of the textbook: Peter Linz. *Introduction to Formal Languages and Automata*, Fifth Edition, Jones and Bartlett Learning, 2012. The terminology and notation used in these notes are similar to those used in the Linz textbook. This document uses several figures from the Linz textbook.

Advisory: The HTML version of this document requires use of a browser that supports the display of MathML. A good choice as of December 2015 seems to be a recent version of Firefox from Mozilla.

Introduction

Chapter 4 examines the closure properties of the family of regular languages, algorithms for determining various properties of regular languages, and methods for proving languages are not regular (e.g., the Pumping Lemma).

Chapter 8 examines similar aspects of the family of context-free languages.

8.1 Two Pumping Lemmas

Because of insufficient time and extensive coverage of the Pumping Lemma for regular languages, we will not cover the Pumping Lemmas for Context-Free Languages in this course. See section 8.1 of the Linz textbook if you are interested in this topic.

8.1.1 Context-Free Languages

Linz Section 8.1 includes the following language examples. The results of these are used in the remainder of this chapter.

1. Linz Example 8.1 shows $L = \{a^n b^n c^n : n \geq 0\}$ is not context free.
2. Linz Example 8.2 shows $L = \{ww : w \in \{a, b\}^*\}$ is not context free.
3. Linz Example 8.3 shows $L = \{a^{n!} : n \geq 0\}$ is not context free.
4. Linz Example 8.4 shows $L = \{a^n b^j : n = j^2\}$ is not context free.

8.1.2 Linear Languages

Linz Section 8.1 includes the following definitions. (The definition of linear grammar is actually from Chapter 3.)

Definition (Linear Grammar): A *linear grammar* is a grammar in which at most one variable can appear on the right side of any production.

A *linear context-free grammar* is thus a context-free grammar that is also a linear grammar.

Linz Definition 8.5 (Linear Language): A context-free language L is *linear* if there exists a linear context-free grammar G such that $L = L(G)$.

Linz Section 8.1 also includes the following language examples.

5. Linz Example 8.5 shows $L = \{a^n b^n : n \geq 0\}$ is a linear language.
6. Linz Example 8.6 shows $L = \{w : n_a(w) = n_b(w)\}$ is not linear.

8.2 Closure Properties and Decision Algorithms for Context-Free Languages

In most cases, the proofs and algorithms for the properties of regular languages rely upon manipulation of transition graphs for finite automata. Hence, they are relatively straightforward.

When we consider similar properties for context-free languages, we encounter more difficulties.

- Some properties do not hold.
- Other properties require more complex arguments.
- Some intuitively simple questions cannot be answered.

Let's consider closure under the simple set operations as we did for regular languages in Linz Theorem 4.1.

8.2.1 Closure under Union, Concatenation, and Star-Closure

Linz Theorem 8.3 (Closure under Union, Concatenation, and Star-Closure): The family of context-free languages is *closed under* (a) *union,* (b) *concatenation, and* (c) *star-closure.*

(8.3a) Proof of Closure under Union:

Let L_1 and L_2 be context-free languages with the corresponding context-free grammars $G_1 = (V_1, T_1, S_1, P_1)$ and $G_2 = (V_2, T_2, S_2, P_2)$.

Assume V_1 and V_2 are disjoint. (If not, we can make them so by renaming.)

Consider $L(G_3)$ where

$$G_3 = (V_1 \cup V_2 \cup \{S_3\}, T_1 \cup T_2, S_3, P_3)$$

with:

$S_3 \notin V_1 \cup V_2$ – i.e, S_3 is a fresh variable

$P_3 = P_1 \cup P_2 \cup \{\ S_3 \rightarrow S_1 \mid\ S_2\ \}$

Clearly, G_3 is a context-free grammar. So $L(G_3)$ is a context-free language.

Now, we need to show that $L(G_3) = L_1 \cup L_2$.

For $w \in L_1$, there is a derivation in G_3:

(1) $S_3 \Rightarrow S_1 \overset{*}{\Rightarrow} w$

Similarly, for $w \in L_2$, there is a derivation in G_3:

(2) $S_3 \Rightarrow S_2 \overset{*}{\Rightarrow} w$

Also, for $w \in L(G_3)$, the first step of the derivation must be either (1) $S_3 \Rightarrow S_1$ or (2) $S_3 \Rightarrow S_2$.

For choice 1, the sentential forms derived from S_1 only have variables from V_1. But V_1 is disjoint from V_2. Thus the derivation

$$S_1 \overset{*}{\Rightarrow} w$$

can only involve productions from from P_1. Hence, for choice 1, $w \in L_1$.

Using a similar argument for choice 2, we conclude $w \in L_2$.

Therefore, $L(G_3) = L_1 \cup L_2$.

QED.

(8.3b) Proof of Closure under Concatenation:

Consider $L(G_4)$ where

$$G_4 = (V_1 \cup V_2 \cup \{S_4\}, T_1 \cup T_2, S_4, P_4)$$

with:

$$S_4 \notin V_1 \cup V_2$$
$$P_4 = P_1 \cup P_2 \cup \{\ S_4 \rightarrow S_1 S_2\ \}$$

Then $L(G_4) = L_1 L_2$ follows from a similar argument to the one in part (a).

QED.

(8.3c) Proof of Closure under Star-Closure:

Consider $L(G_5)$ where

$$G_5 = (V_1 \cup \{S_5\}, T_1, S_5, P_5)$$

with:

$$S_5 \notin V_1$$

$$P_5 = P_1 \cup \{\ S_5 \rightarrow S_1 S_5 \mid \lambda\ \}$$

Then $L(G_5) = L_1^*$ follows from a similar argument to the one in part (a).

QED.

8.2.2 Non-Closure under Intersection and Complementation

Linz Theorem 8.4 (Non-closure under Intersection and Complementation): The family of context-free languages is *not closed under* (a) *intersection and* (b) *complementation.*

(8.4b) Proof of Non-closure under Intersection:

Assume the family of context-free languages is closed under intersection. Show that this leads to a contradiction.

It is sufficient to find two context-free languages whose intersection is not context-free.

Consider languages L_1 and L_2 defined as follows:

$$L_1 = \{a^n b^n c^m : n \geq 0, m \geq 0\}$$
$$L_2 = \{a^n b^m c^m : n \geq 0, m \geq 0\}$$

One way to show that a language is context-free is to find a context-free grammar that generates it. The following context-free grammar generates L_1:

$$S \rightarrow S_1 S_2$$
$$S_1 \rightarrow a S_1 b \mid \lambda$$
$$S_2 \rightarrow c S_2 \mid \lambda$$

Alternatively, we could observe that L_1 is the concatenation of two context-free languages and, hence, context-free by Linz Theorem 8.3 above.

Similarly, we can show that L_2 is context free.

From the assumption, we thus have that $L_1 \cap L_2$ is context free.

But

$$L_1 \cap L_2 = \{a^n b^n c^n : n \geq 0\},$$

which is not context free. Linz proves this in Linz Example 8.1 (which is in the part of this chapter we did not cover in this course).

Thus we have a contradiction. Therefore, the family of context-free languages is not closed under intersection.

QED.

(8.4b) Proof of Non-closure under Complementation:

Assume the family of context-free languages is closed under complementation. Show that this leads to a contradiction.

Consider arbitrary context-free languages L_1 and L_2.

From set theory, we know that

$$L_1 \cap L_2 = \overline{\bar{L_1} \cup \bar{L_2}}.$$

From Linz Theorem 8.3 and the assumption that context-free languages are closed under complementation, we deduce that the right side ($\overline{\bar{L_1} \cup \bar{L_2}}$) is a context-free language for all L_1 and L_2.

However, we know from part (a) that the left side ($L_1 \cap L_2$) is not necessarily a context-free language for all L_1 and L_2.

Thus we have a contradiction. Therefore, the family of context-free languages is not closed under complementation.

QED.

8.2.3 Closure under Regular Intersection

Although context-free languages are not, in general, closed under intersection, there is a useful special case that is closed.

Linz Theorem 8.5 (Closure Under Regular Intersection): Let L_1 be a context-free language and L_2 be a regular language. Then $L_1 \cap L_2$ is *context free.*

Proof:

Let $M_1 = (Q, \Sigma, \Gamma, \delta_1, q_0, z, F_1)$ be an npda that accepts context-free language L_1.

Let $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$ be a dfa that accepts regular language L_2.

We construct an npda

$$\widehat{M} = (\widehat{Q}, \Sigma, \Gamma, \widehat{\delta}, \widehat{q_0}, \widehat{F})$$

that simulates M_1 and M_2 operating simultaneously (i.e., executes the moves of both machines for each input symbol).

We choose pairs of states from M_1 and M_2 to represent the states of $\widehat{M}$ as follows:

$$\widehat{Q} = Q \times P$$
$$\widehat{q_0} = (q_0, p_0)$$
$$\widehat{F} = F_1 \times F_2$$

We specify $\widehat{\delta}$ such that the moves of $\widehat{M}$ correspond to simultaneous moves of M_1 and M_2. That is,

$$((q_k, p_l), x) \in \widehat{\delta}((q_i, p_j), a, b)$$

if and only if

$$(q_k, x) \in \delta_1(q_i, a, b)$$

and

$$\delta_2(p_j, a) = p_l.$$

For moves (q_i, λ, b) in δ_1, we extend δ_2 so that $\delta_2(p_l, \lambda) = p_l$ for all l.

By induction on the length of the derivations, we can prove that

$$((q_0, p_0), w, z) \vdash^*_{\widehat{M}} ((q_r, p_s), \lambda, x),$$

with $q_r \in F_1$ and $p_s \in F_2$ if and only if

$$(q_0, w, z) \vdash^*_{M_1} (q_r, \lambda, x)$$

and

$$\delta^*(p_0, w) = p_s.$$

Therefore, a string is accepted by $\widehat{M}$ if and only if it is accepted by both M_1 and M_2. That is, the string is in $L(M_1) \cap L(M_2) = L_1 \cap L_2$.

QED.

8.2.4 Linz Example 8.7

Show that the language

$$L = \{a^n b^n : n \geq 0, n \neq 100\}$$

is context free.

We can construct an npda or context-free grammar for L, but this is tedious. Instead, we use closure of regular intersection (Linz Theorem 8.5).

Let $L_1 = \{a^{100} b^{100}\}$.

L_1 is finite, and thus also regular. Hence, $\bar{L_1}$ is regular because regular languages are closed under complementation.

From previous results, we know that $L = \{a^n b^n : n \geq 0\}$ is context free.

Clearly, $L = \{a^n b^n : n \geq 0\} \cap \bar{L_1}$.

By the closure of context-free languages under regular intersection, L is a context-free language.

8.2.5 Linz Example 8.8

Show that

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

is not context free.

Although we could use the Pumping Lemma for Context-Free Languages, we again use closure of regular intersection (Linz Theorem 8.5).

Assume that L is context free. Show that this leads to a contradiction.

Thus

$$L \cap L(a^*b^*c^*) = \{a^n b^n c^n : n \geq 0\}$$

is also context free. But we have previously proved that this language is not context free.

Thus we have a contradiction. Therefore, L is not context free.

8.2.6 Some Decidable Properties of Context Free Languages

There exist algorithms for determine whether a context-free language is empty or nonempty and finite or infinite.

These algorithms process the context-free grammars for the languages. They assume that the grammars are first transformed using various algorithms from Linz Chapter 6 (which we did not cover in this course).

The algorithms from Chapter 6 include the removal of

- *useless symbols and productions* (i.e., variables and productions that can never generate a sentence)
- λ-*productions* (i.e., productions with λ on the right side)
- *unit productions* (i.e., productions of the form $A \rightarrow B$)

Linz Theorem 8.6 (Determining Empty Context-Free Languages): Given a context-free grammar $G = (V, T, S, P)$, then there exists an algorithm for *determining whether* or not $L(G)$ is *empty.*

Basic idea of algorithm: Assuming $\lambda \notin L$, remove the useless productions. If the start symbol is useless, then L is empty. Otherwise, L is nonempty.

Linz Theorem 8.7 (Determining Infinite Context-Free Languages): Given a context-free grammar $G = (V, T, S, P)$, then there exists an algorithm for *determining whether* or not $L(G)$ is *infinite*.

Basic idea of algorithm: Remove useless symbols, λ-productions, and unit productions. If there are variables A that repeat as in

$$A \overset{*}{\Rightarrow} xAy$$

then the language is infinite. Otherwise, the language is finite. To determine repeated variables, we can build a graph of the dependencies of the variables on each other. If this graph has a cycle, then the variable at the base of the cycle is repeated.

Unfortunately, *other simple properties are not as easy* as the above.

For example, *there is no algorithm to determine whether two context-free grammars generate the same language.*