A Context-Free Grammar for a Repeated String

Zach Tomaszewski
Department of Information and Computer Sciences
University of Hawaii at Manoa
Honolulu, HI 96822
Email: ztomasze@hawaii.edu

Abstract

A language is not always on the same level of the Chomsky Hierarchy as its complement. This paper explores one such example, showing that while the language of ww for some string w is not context-free, its complement is context-free. A grammar is provided for the complement language, and a proof is given to show that the provided grammar indeed produces the complement language.

Keywords

context-free grammar; context-free language; duplicate substrings;

I. Introduction

In computer science, the Chomsky hierarchy classifies classes of languages in terms of the grammars that describe them. Each level in the hierarchy has a corresponding class of theoretical automata that are capable of recognizing the languages of that level. At the lowest and simplest level are regular grammars (Type 3), which describe languages that can be recognized by finite automata. Context-free grammars (Type 2) describe languages that can be recognized by non-deterministic pushdown automata. The languages described by context-sensitive grammars (Type 1) can be recognized by linearly-bounded automata. Finally, at the top of the hierarchy are unrestricted or phrase-structure grammars (Type 0), which describe recursively-enumerable languages that can be recognized by a Turing machine [1] [2] [3]. Higher level language classes include those below them as subsets. For example, a regular grammar is also a context-free grammar, though the reverse is not true.

Because the characteristics of each grammar type are so well-known, it is often useful to know the class of a given language. Frequently, this classification is not obvious when simply looking at the language's definition.

For example, consider the language $L_a = \{w | w \in \{\mathtt{a}, \mathtt{b}\}^+$ and w contains at least one $\mathtt{a}\}$. This is a regular language. It is possible to define a finite automaton that reads the string from left-to-right, switching from a initial non-accepting to an accepting state upon reading an $\mathtt{a}$.

A language of even-length palindromes, such as $L_p = \{ww^R | w \in \{\mathtt{a}, \mathtt{b}\}^+\}$, is a well-known example of a non-regular language [4]. A finite automaton cannot recognize an arbitrary string from L_p because a finite automaton has no data store. It cannot record which symbols it has seen in the first half of the string in order to match them up with the second half. However, a pushdown automaton is capable of this task, since it can use an internal stack of unbounded size to record select symbols that it has read. This makes L_p a context-free language.

Now consider the very similar language of duplicated strings, $L_d = \{ww | w \in \{\mathtt{a}, \mathtt{b}\}^+\}$. Here, the second w is no longer reversed. This means a pushdown automaton, equipped with a single stack and a reading head that moves only left-to-right over the symbols, is unable to recognize an arbitrary string from L_d. Although it can store symbols from the first half of the string on its stack, the symbols would be stored in the wrong order to then match them with symbols the automaton reads in the second half of the string. This means that, although L_d looks to be very simple, it is not a context-free language [5].

Now consider L as the complement of L_d. That is, $L = \{$the set of all strings over $\{\mathtt{a}, \mathtt{b}\}$ not of the form ww for some $w \in \{\mathtt{a}, \mathtt{b}\}^+\}$. At first glance, it seems that recognizing a string from L should be as complex a task as recognizing one from L_d. However, this is not so. As is often the case, a language may not be in the same class as its complement [6].

This paper shows that L is a context-free language. It will do this by constructing a context-free grammar G_L and then proving that the language $L(G_L)$ produced by the grammar is equivalent to L.

II. Examination of the Problem

Before constructing the grammar for L, it may be useful to more closely examine the problem. Specifically, of all the possible strings produced over the alphabet $\{\mathtt{a}, \mathtt{b}\}$, which of them might have the form ww? And, of those strings, what simple change would then prevent the ww form?

Since $|ww| = 2 * |w|$, the length of ww must be even. Therefore, no odd-length strings can be of the form ww.

However, when generating an even-length string, it is impossible to have the form ww if at least one symbol in the first w substring differs at the same position in the second w substring. That is, suppose we divide any string $|ww|$ into two strings w and w' of equal length. Then, for any i where $1 \leq i \leq |w|$, if the symbol in position i in w is changed to differ from the symbol in position i in w', we no longer have the string $|ww|$. (See Figure 1.)

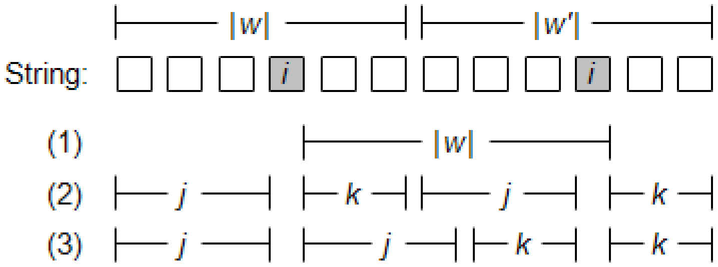

Figure 1. The distance relationships between two differing symbols in a not-ww string.

The key observation is that these two differing symbols are a distance of $|w|$ apart [7], as shown on line (1) of Figure 1. $|w|$ is equal to half the length of the entire string.

Also, there must be the same number j of symbols before position i in w as there are symbols before position i in w'. The same relationship is true of k, the number of symbols after position i in each substring. Both j and k may be integers 0 or greater, and $j + 1 + k = |w|$, as shown on line (2).

A final insight is that, if L is a context-free language, we can rely on the nondeterminism of a pushdown automaton to recognize a string from L. Specifically, we can correctly ”guess” the locations of at least two symbols that must differ and then generate the remaining arbitrary symbols in pairs around each of these pivotal symbols. That is, we can generate j pairs of symbols around the first pivotal symbol and k pairs around the second pivotal symbol. This is shown on line (3) in Figure 1.

III. The Grammar

These observations lead to the following grammar G_L for producing L:

$S \rightarrow E|U|\epsilon$
$E \rightarrow AB|BA$
$A \rightarrow ZAZ|\mathtt{a}$
$B \rightarrow ZBZ|\mathtt{b}$
$U \rightarrow ZUZ|Z$
$Z \rightarrow \mathtt{a}|\mathtt{b}$

IV. The Proof

To know that this grammar is correct, we must prove $L(G_L) = L$, which is to say that the language produced by the grammar is identical to L. To prove this equivalence, we must consider the relation in both directions. First, the grammar must produce only strings found in L. Second, every string in L must be produced by the grammar.

A. $L(G_L) \subset L$

The first rule of the grammar is $S \rightarrow E|U|\epsilon$. ϵ is not equivalent to ww if w includes at least one symbol. We can then consider E and U separately.

Given that $Z \rightarrow \mathtt{a}|\mathtt{b}$, the recursive production rule $U \rightarrow ZUZ|Z$ means that U can produce a string of one terminal symbol, or three terminal symbols, or five terminal symbols, etc. In short, U can produce only odd-length strings, which necessarily cannot be of the form ww.

Finally, consider $E \rightarrow AB|BA$. Either of these choices have the same general form. Examining the derivation from AB as an example, we can see that A will produce the pivotal terminal $\mathtt{a}$ surrounded by j pairs of arbitrary symbols, where $j \geq 0$. B will produce a similar structure of the pivotal terminal $\mathtt{b}$ surrounded by k pairs of arbitrary symbols, where $k \geq 0$. Since A and B both produce odd-length strings, the entire AB string is of even-length. As shown in the discussion above, this production also places $j + k$ symbols between the a and b pivotal symbols. Since these two differing symbols are a distance of $|w|$ apart, the resulting complete string may not be of the form ww.

The derivation of BA is identical to that of AB except that the pivotal a and b terminals are reversed.

Therefore, since every string derived from this grammar is not of the form ww, every generated string is in L.

B. $L \subset L(G_L)$

To show that every string $x \in L$ can be derived from G_L, we will examine odd and even lengths of x separately.

1) Odd Lengths: For the odd lengths, we can use proof by induction on $|x|$.

Basis: $|x|$ = 1. In this case, the derivation $S \Rightarrow U \Rightarrow Z$ can then produce any single symbol in the alphabet. Thus, any string in L of length 1 can be derived from G_L.

Induction: Assume n is odd, $|x| = n$ for some $x \in L$, and $S \Rightarrow U \overset{*}{\Rightarrow} x$. We can then derive $y \in L$, where $|y| = n + 2$, which is the next odd length. From the grammar, we can derive $S \overset{*}{\Rightarrow} ZUZ$. Since $U \overset{*}{\Rightarrow} x$ by the induction hypothesis, this produces ZxZ. Each Z can then be replaced by any symbol from the alphabet. This produces a string y that is two symbols longer than x. Furthermore, it allows for all possible single-symbols prefixes to x and all possible single-symbol postfixes to x.

Therefore, any odd-length string in L is also in $L(G_L)$.

2) Even Lengths: It is possible to see that every even-length $x \in L$ has a derivation in G_L by preforming a reverse derivation as follows. Start with any even-length string $x \in L$. As discussed in the Exploration section above, such as string must contain at least two different symbols that are a distance of $|x|/2$ apart. Select two such symbols in the string. Replace the a of this symbol pair with an A and the b with a B. Replace all other symbols with a Z. This will produce a string of either $Z_1..Z_jAZ_1...Z_jZ_1...Z_kBZ_1...Z_k$ or $Z_1..Z_jBZ_1...Z_jZ_1...Z_kAZ_1...Z_k$, where $j \geq 0$, $k \geq 0$, and $j + k + 1 = |x|/2$. Such a string is derivable from G_L, thus showing that any even-length string in L is also in $L(G_L)$.

3) Zero Length: Finally, if $|x| = 0$, then x is ϵ. This string in L is produced by G_L thanks to the $S \rightarrow \epsilon$ rule.

V. Conclusion

By constructing a context-free grammar, this paper has shown that the set of strings not of the form ww is a context-free language. A proof has also been provided that the language produced by this grammar is indeed equivalent to the original language.

Others have proposed an equivalent grammar for an alphabet of arbitrary size, rather than the limited alphabet of $\{\mathtt{a}, \mathtt{b}\}$ [7]. Future work might also explore whether the complement of the language of substrings repeated an arbitrary number of times (rather than only twice, as done here) is also a context-free language.

References

[1] K. Sugihara. (2011, Apr.) ICS241 Lecture Notes #22. [Online]. Available: http://pearl.ics.hawaii.edu/ sugihara/course/ics241s11/notes/04-12n22.html

[2] Chomsky Hierarchy. (2012, Apr.) [Online]. Available: http://en.wikipedia.org/w/index.php?title=Chomsky_hierarchy&oldid=488076701

[3] J. E. Hopcroft and J. D. Ullman. "Chomsky Hierarchy", in *Introduction to Automata Theory, Languages, and Computation*, 1st ed. Addison-Wesley Publishing Company, 1979, ch. 9.

[4] M. Sipser. "Regular Languages", in *Introduction to the Theory of Computation*, 1st ed. Boston, MA: PWS Publishing Company, 1997, ch. 1.

[5] M. Sipser. "Context-Free Languages", in *Introduction to the Theory of Computation*, 1st ed. Boston, MA: PWS Publishing Company, 1997, ch. 2.

[6] J. E. Hopcroft and J. D. Ullman. "Properties of Context-Free Languages", in *Introduction to Automata Theory, Languages, and Computation*, 1st ed. Addison-Wesley Publishing Company, 1979, ch. 6.

[7] M. Greenstreet. (2008, Nov.) CpSc421 Homework 8 Solutions. [Online.] Available: http://www.ugrad.cs.ubc.ca/ cs421/hw/8/a.pdf