Knuth-Morris-Pratt Algorithm and Boyer-Moore Algorithm Analysis

Chih Chang

Abstract

This paper investigates the pumping property of context free languages. In particular, the pumping property is a modification to a similar property found in regular languages. However, because of Chomsky's Hierarchy, regular languages are properly included in context free languages. Therefore, it can be concluded that the pumping lemma property for regular language is not sufficient for determining if a given language is context free.

Before the proof on a particular language is presented, this paper will introduce the concepts of languages and the pumping lemma properties for both regular and context free languages, discussing their hierarchical differences. Based on the introduced foundations, a particular language will be introduced where the pumping lemma will be applied to investigate if it isn't a context free language.

Index Terms

pumping lemma; regular language; context free languages;

I. Introduction

To begin, we must establish a general language, usually denoted by the capital letter L, that is L is defined over a grammar, G, on a finite alphabet, Σ. Therefore, depending on the grammar, a language L can be classified into a hierarchy of language complexity, formally known as Chomsky's Hierarchy.

A. Context Free Languages and Context Free Grammar

In particular, context free languages are a group of languages that can be defined by context free grammars. A context free grammar is a finite set of variables, also known as nonterminals, which can produce terminal symbols through production rules. However, these production rules cannot be context sensitive; that is, the left side of the production rule can only include a single variable.

More formally, a CFG, G = $\{V, T, P, S\}$ such that V is the set of variables, T is the set of terminal symbols, P is the set of production rules, and S is the starting symbol where S $\in$ V. P must be in the form of A $\rightarrow \alpha$, where A can be any variable from the set V and α is a string of symbols from (V$\cup T$)*.

It is obvious, then, to see that there exist a higher class of languages that are context sensitive, such that production P = $\alpha \rightarrow \beta$. This is the higher level class of languages, which uses phrase structured grammer.

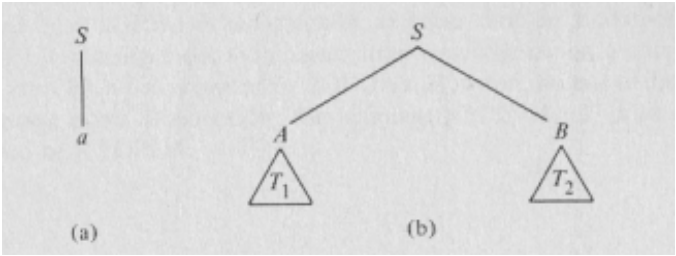
A CFL-equivalent abstract machine is the push down automata, which uses a stack and stack symbols in addition to states to decide acceptance of a language. Formally, a PDA is a machine M = (Q, Σ, Γ, δ, q_0, Z_0, F), where Q is the set of states, Σ is the set of input symbols, Γ is the set of stack symbols, δ is the set of transition functions,

q_0 is the initial state, Z_0 is the initial bottom stack symbol such that $Z_0 \in \Gamma$, and F is the set of final states. δ is a set of transition functions mapping from Q$\times(\Sigma \cup \{\epsilon\}) \times \Gamma$ to finite subsets of Q$\times\Gamma^*$.

Intuitively, the transition function reads the input symbol or make an epsilon move, and depending on the the control's internal state, the function pops the symbol at the top of the stack to determine the next state and symbols to be pushed onto the stack. This is useful for counting the number of a given terminal symbol because the stack and keep a count of a given symbol. This count can then be used to match against another terminal symbol, such that for example, if the language is defined as consecutive 0s concatenated to consecutive 1s with the number of 0s equal to the number of 1s, the stack can push the number of 0s and pop the stack symbols for every consecutive 1s.

B. Pumping lemma for CFL

To begin discussing the pumping property of CFL, we will first introduce the idea of a derivation tree. Strings of a language is derived by the production rules of the grammar, and because CFGs strictly follow the form A $\rightarrow \alpha$, we can represent string derivations using a tree such that the root and every internal nodes are nonterminal symbols and the leaves are terminal symbols. A root will have a branch for each terminal and nonterminal in its production rule. A preorder traversal of the tree will result in the final concatenated string that is derived. And because, by the definition of a CFG explained in the previous section, every derivation must begin with the start symbol S, therefore, the root of every derivation tree must be the nonterminal symbol S.



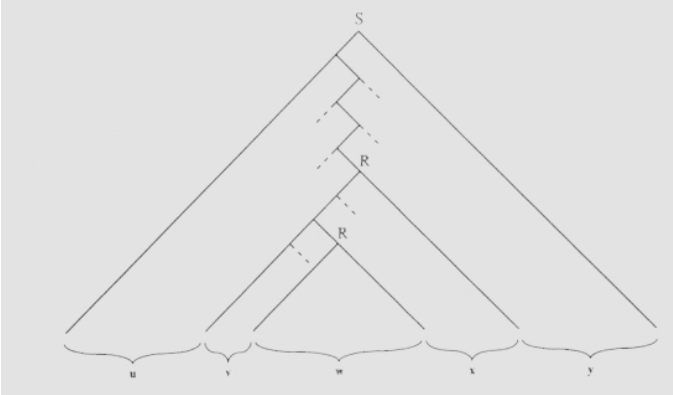
This is the recursive definition of a derivation tree starting with the symbol S.

Fig. 1. CFL Derivation Tree

Now that a derivation tree has been defined, the pumping property of a CFL requires a sufficiently large string whose length is greater than the pumping number n such that its derivation requires a repeated pumping of a certain derivation subtree. Given any arbitrary CFG, we can assume a production rule R $\stackrel{*}{\Rightarrow}$ vRx and R $\stackrel{*}{\Rightarrow}$ w. And we can also assume S $\stackrel{*}{\Rightarrow}$ uRy, where u, v, w, x, y are all terminal strings. In this way, we can clearly see that any string z produced will follow the form z = uv^iwx^iy where i $\in \mathbb{N}$.

The pumping property is derived from z. Formally, for z $\in$ a CFL L such that $|z| \geq n$, z = uvwxy following the properties:

- $|vx| \geq 1$



This is a pictoral representation of the pumping lemma. Given n, R must be reused to generate z = uvwxy.

Fig. 2. Pumping Lemma Derivation Tree

- $|vwx| \leq n$
- for all i $\geq$ 0, $uv^iwx^iy \in L$

The pumping lemma tests if a language is not a context free language by investigating a particular z in L that when specific i is changed that z is no longer in L for every cases of arranging uvwxy that follows the three defined properties.

C. A particular language L

For this particular paper, a particular language is defined L = $\{b_i\#b_{i+1}|b_i$ is i in binary, $i \geq 1\}$. Intuitively, this language is defined over the alphabet $\Sigma = \{0, 1\}$ such that the left hand side of $\#$ is the binary representation of i and the right hand side is the binary representation of i+1.

II. Proof

A. Picking a particular string in L

To prove that L is not a context free language, let z = uvwxy = $1^n0^n\#1^n0^{n-1}1$. This particular string z is in L because the right hand side of $\#$ simply replaces the 2^0th digit of the left hand side with 1 instead of 0, thus representing i+1 in binary.

B. Proof on the particular language L using z

Because $|vwx| \leq n$ and $|vx| \geq 1$, z = uv^iwx^iy in the following cases are considered:

- case $\#$ is not included in vwx

  Because vx cannot be empty and vwx must be a concatenation in order, this case requires that either vwx is strictly contaned some or all parts of the left hand side of $\#$ or the right hand side. In either case, let i = 0 such that only one side is pumped down. And because the other side is not changed, z is no longer in L.
- case $\#$ is in either v or x

Let i = 0 such that # is removed after v and x are pumped down so that z is no longer in L.

- case # is in w

  In this case, w = $0^p\#1^q$, v = 0^k, and x = 1^j, where p + q + k + j $\leq$ n - 1. Therefore, pump v and x up, for example, let i = 2, would multiply the left hand side by some power of 2, depending on k, because the left hand side is increased by addition of 0s to its right-most digits. On the other hand, the right hand side will not be increased by any power of two because the right hand side is increased by addition of 1s to its left-most digits. This is true for any k or j because at least k or j must be at least 1. And because the right hand side is increasing at a different multiple to the left hand side, z is no longer in L.

III. Conclusion

In conclusion, the language defined L = $\{b_i\#b_{i+1}|b_i$ is i in binary, $i \geq 1\}$ is not context free because the pumping lemma can produce a string z' that is not within the language from a string z that is within the language for every considerable cases to split z into uvwxy by applying the pumping lemma properties.

This paper has introduced the properties of CFL and CFG. And based on such foundations, derivation trees are defined which leads to the introduction of the pumping lemma for CFL. And using the definition of converting decimal numbers to their binary representation, the multiplication of the power of 2 was used to prove that the language is not context free.

A future investigation may be to determine if Ogden's lemma can be applied to simplify the proof because Ogden's lemma restricts the cases of where the split can occur by labeling the string with markers. This particular language may not gain much simplification using the Ogden's lemma, but an investigation may still be interesting.

References

[1] W. S. Brainard and L. H. Landweber, *Theory of Computation*, John Wiley & Sons, 1974.

[2] J. Carroll and D. Long, *Theory of Finite Automata with an Introduction to Formal Languages*, Englewood Cliffs, New Jersey: Prentice-Hall, 1989.

[3] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

[4] J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation* Addison-Wesley, 1979.

[5] K. H. Rosen, *Discrete Mathematics and Its Applications*, 6th ed. New York, United States: McGraw-Hill, 2007.