The Language $0^n\#0^{2n}\#0^{3n}$ is not Context-Free

Proof Paper Example

James Logan Mayfield

February 24, 2017

1 Problem

In this paper we consider the language $L = \{0^n\#0^{2n}\#0^{3n} \mid n \geq 0\}$ for the alphabet $\{0, \#\}$. The simplest string in this language is just two consecutive hash-tags: $\#\#$. This occurs when the number n is zero and each string of zeros is empty. As n increases then the length of the string grows to $6n + 2$ as we distribute $6n$ zeros around the two hash-tags such that the group to the left contains n zeros, between the hash-tag is $2n$ zeros, and finally to the right we see the final $3n$ zeros. In general, we're looking at strings containing two hash-tags surrounded by sequences of zeros where the length of each sequence increases as a multiple of the length of the first sequence. Our aim is to prove the following theorem about L:

Theorem 1. *The language $L = \{0^n\#0^{2n}\#0^{3n} \mid n \geq 0\}$ over the alphabet $\{0, \#\}$ is not a context-free Language.*

It is unlikely that the language L is context-free due to the strict distribution of zeros across more than two substrings. While context-free languages can often have matched pairs such as $0^n\#0^n$ and even $0^n\#0^{2n}$, matched triples are typically too much for context-free grammars and push-down automaton. For example, Sipser showed that $a^nb^nc^n$ is not context-free [1]. The language examined in this paper is ultimately a variation on that same pattern.

2 Proof Sketch

To prove Theorem 1 we can contradict the pumping lemma for context-free languages which states that all context-free languages have a pumping length p such that any string longer than p has one or more parts that can be repeated or removed, i.e. pumped, such that the resultant string is still in the language [1]. By choosing the string $s = 0^p\#0^{2p}\#0^{3p}$ we can show that any decomposition of s allowable under the pumping lemma can be pumped such that the result is not in L and thereby contradict the pumping lemma. This choice of s will force the pump-able parts of the string to cover at most part of two of the three substrings containing zeros. This means that adding or removing these parts violates the essential condition that each of the three substrings of zeros contains a multiple number of the first.

3 Proof of Theorem 1

The overall structure of the proof of Theorem 1 is that of a proof by contradiction. We'll assume language L is context-free and thereby subject to the pumping lemma. This will necessitate a case analysis for the decomposition of a chosen string from L. The first level of this proof emphasizes a high-level case analysis showing that all possible decompositions can be pumped out of the language L. A second level of analysis is provided in order to give a more detailed treatment of each case. The second level of analysis will appear in Section 3.1.

Proof. To prove Theorem 1 we will derive a contradiction of the pumping lemma for context-free languages which states that for any context-free language A there exists a pumping length p such that for any string

s with length at least p there exists a decomposition $s = uvxyz$ where: the length of vxy is at most p, the combined length of v and y is grater than 0, and for all i, the string uv^ixy^iz is in A[1].

Assume that language L is a context-free language and has pumping length p. Let string s be $0^p\#0^{2p}\#0^{3p}$. By the pumping lemma we can decompose s into $uvxyz$ under the constraint that substring vxy is no longer than p and the combined length of v and y is not zero. This gives rise to the following cases:

i. vxy is contained within the first sequence of p zeros.

ii. vxy straddles the first and second sequence of zeros and therefore contains the first hash-tag.

iii. vxy is contained within the second sequence of $2p$ zeros

iv. vxy straddles the second and third sequence of zeros and therefore contains the second hash-tag

v. vxy is contained within the third sequence of $3p$ zeros.

The first, third, and fifth cases all generalize to pumping within only one of the subsequences of zeros. The second and fourth cases generalize to pumping across two of the three sequences. No case allows for pumping within all three of the subsequences at once. Without loss of generality, we will only consider the first and second cases as representative examples of all five cases.

When the string vxy contains only zeros from the first sequence of p zeros then v and y must contain some number of zeros k for k grater than zero. When $i = 2$, the string uv^2xy^2z is $0^{p+2k}\#0^{2p}\#0^{3p}$ which is not L and we've reached a contradiction of the pumping lemma.

When the string vxy straddles the end of the first sequence of zeros and the start of the second sequence then one of the following is true for vxy:

i. v is empty, x contains the first zeros and the hash-tag, and y some number of zeros k from the second sequence. Similarly, v may be k zeros from the first sequence, x contains the hash-tag, and y is empty. In both cases, we can select $i = 2$ to add k zeros to only one of the sequences of zeros and arrive at the same contradiction we found above.

ii. Either x and v are both empty or x and y are both empty. In each case the non-empty variable, y and v respectively, is a hash-tag surrounded by some number of zeros. Letting $i = 2$ will therefore add a third hash-tag into the string. Such a string is not in L and we've reached another contradiction of the pumping lemma.

iii. Neither v nor y is empty. If in this case either v or y contains the hash-tag, then pumping the string with $i = 2$ adds a third hash-tag to the string to produce a string not in L. If x contains the hash-tag, then v contains some number of zeros k and y contains some number of zeros l. The resultant string is $0^{p+k}\#0^{2p+l}\#0^{3p}$ and is not in L. In both cases we contradict the pumping lemma.

We have shown that for any decomposition of the string $0^p\#0^{2p}\#0^{3p}$ allowable under the pumping lemma there exists at least one way to pump the string out of the language L and have thereby contradicted the pumping lemma. It follows then that L cannot be a context-free language.

□

3.1 A More Detailed Case Analysis

We'll now engage in a more detailed analysis of the first and second cases of the decomposition of $s = 0^p\#0^{2p}\#0^{3p} = uvxyz$ under the constraints of the pumping lemma. Recall that the first case occurs when vxy is fully contained within the first sequence of p zeros. The second case occurs when vxy contains the first hash-tag and therefore straddles the first and second sequences of zeros.

When vxy is contained within the first p zeros then the substring u is 0^i, vxy is 0^j, and z is $0^k\#0^{2p}\#0^{3p}$ $i + j + k = p$ This gives us $s = uvxyz = 0^{i+j+k}\#0^{2p}\#0^{3p}$. We can further decompose vxy as to isolate the strings v and y. The substring v is 0^l and y is 0^m such that $l + m$ is greater than zero but less than j. This leaves $j - (l + m)$ zeros for x. By the pumping lemma uv^2xy^2z must be in the language L. However, $uv^2xy^2z = 0^{i+2l+2m+j-(l+m)+k}\#0^{2p}\#0^{3p}$ and $i + l + m + j + k$, the number of zeros in the first sequence, is

strictly greater than $p = i + j + k$ because $l + m$ cannot be zero. This string is not in the language L and we've contradicted the pumping lemma.

When vxy straddles the first and second sequence of zeros and contains the first hash-tag then we must consider three possible scenarios with respect to v and y: v is not empty but y is, neither v nor y is empty, and v is empty but y is not. In general, we can say the following: u is 0^i, vxy is $0^j\#0^k$, and z is $0^l\#0^{3p}$ where $i + j = p$ and $k + l = 2p$. The case where one of v and y is empty share the same logic so without loss of generality, let us consider the case where y is empty but v is not such that $s = uvxz$. The substring v may contain the hash-tag. If it does then v is $0^j\#0^m$ and x is 0^o where $m + o = k$. If the language L is context-free then uv^2xz must be in L according to the pumping lemma. However, the string uv^2xz is $0^i0^j\#0^m0^j\#0^m0^o0^l\#0^{3p}$ or $O^p\#0^{m+j}\#0^{m+o+l}\#0^{3p}$ which is most definitely not in L and contradicts the pumping lemma. If v does not contain the hash-tag then v contains a non-zero number of zeros m such that x is $0^o\#0^k$ and $m + o = j$. Once again, the string uv^2xz should be in the language L but uv^2xz is $0^{i+2m+o}\#0^{k+l}\#0^{3p}$. The first sequence of $i + 2m + o$ zeros contains exactly $p + m$ zeros and uv^2xz is not in the language L, contradicting the pumping lemma. Where neither v nor y is empty then either v, x, or y contains the hash-tag. In the event that v or y contains the hash-tag then we have the same problem as when y was empty and v contained the hash-tag. By choosing $i = 2$, then we create uv^2xy^2z that contains three hash-tags and contradict the pumping lemma by generating a string not in L. If x contains the hash-tag, then v must be m zeros and y must be o zeros. For $i = 2$ we have string uv^2xy^2z equal to $0^{i+2m+(j-m)}\#0^{2o+(k-o)}\#0^{3p}$ where $i + m + j$ is strictly greater than p and the string is no longer in the langauge L. Thus, in all cases we can find an i such that uv^ixy^iz is not in the language, contradicting the pumping lemma for context-free languages.

4 Conclusion

In showing that the language L is not context-free we utilized a common strategy of choosing a string to pump that separates two portions of the string that share a key relationship. In this case it was a relationship of length. This is the exact strategy used by Sipser to show that $a^nb^nc^n$ is not context-free.

References

[1] Michael Sipser. *Introduction to the Theory of Computation.* Cengage Learning, 3rd edition, 2013.