Max Shi

CS 334

Professor Bhatt

October 3, 2019

I pledge my honor that I have abided by the Stevens Honor System.

Problem Set 5

1. Let L be the language described in the problem. Assuming L is a regular language, it must satisfy the pumping lemma. Let string s = 0^{2^p}. Thus, |s|>p and s is in L. Thus, in the string xyz, by conditions 2 and 3 of the pumping lemma, |y|>0 and |xy|<=p. In other words, y will have at least one 0 and at most p 0's, as in the latter case, x = ε and y = 0^p. Thus, 0<|y|<=p. For each string 0^{2^p} in the language, the next string in the language is $0^{2^{(p+1)}}$, which is equal to $0^{2^p}0^{2^p}$. In other words, the next string in the language will have 2^p more 0s in it. However, pumping xyz to xyyz only adds anywhere from 1 to p 0s to the string, which is always less than 2^p, according to the pumping lemma. Therefore, this next string will be out of the language, violating the pumping lemma and making L non-regular.
2. Assume the language B is regular. Thus, the complement of language B is regular, as regular languages are closed under the complement operation, as shown in the previous problem set. Because the language B describes all strings where there is any number of 0's, then a different number of 1's, the complement would accept all strings that do not match this pattern, including the string of any number of 0's followed by an equal number of 1's. Let this language be language C = {0^i1^i, i >= 0}. The complement of B is C ⋃ (the rest of the strings not accepted by B). As regular languages are closed under union, C must also be regular. However, as proved by a contradiction of the pumping lemma, the language C is not regular. Thus, a contradiction occurs, and therefore, the language B is not regular.
3.
    3.1. Min length = 4 – the language requires the first 3 zeroes, then at least 1 one to fulfill the 1* portion of the regular expression in order to be pumped and still remain within the language. There is only one string of length <= 3 in the language, which is 000, which cannot be pumped to stay within the language. Any string of length >= 4 in the language must have at least one 1 at the end, and this 1 can be pumped.

    3.2. Min length = 1 – any string of length 1 (i.e. the strings 0 and 1) are in the language, and can be pumped and stay within the language. These strings pumped will give a string of any number of 0's or any number of 1's, respectively, which are both within the language. Any string with length >= 1 must be either all 0's, all 1's, or some number of 0's then some number of 1's, and in each case, the 0, the 1, or either the 0 or the 1 can be pumped, respectively, and still stay within the language.

    3.3. Min length = 1 – very similar to the last problem, where the strings of any number of 0's or any number of 1's are both within the language. The regular expression of 10*1, which would have a minimum pumping length of 3, does not affect the minimum pumping length, as all strings within this regular expression are caught by the expression 1*0*1*, which is a concatenation of 0*1*0*1*, which is the regular expression of the left side of the union. Thus, the left side of the union encompasses the right side, and degenerates it into just the left side of the union for the sake of the minimum pumping length, and any string of length >= 1 must have either a 0 or a 1, both of which can be pumped in the string and stay within the language.

3.4. Min length = 1 – Although all strings in the language must be a multiple of 2, there are no strings in the language that have length 1. Thus, for all strings in the language with length >= 1, they must contain the string 01, and thus can be pumped by (01) to stay within the language.

3.5. Min length = 3 – Any string of length >= 3 in the language must have at least one 1 in the string. This 1, no matter where it is among the 0s, can be pumped to stay within the language. However, of the strings with length < 3 in the language, it includes the string 00, which cannot be pumped.

4.

4.1. S -> 0S0 | 1S1 | 1 | 0 | ε

4.2. S -> 0A0 | 1A1 | ε
A -> 0A | 1A | ε

4.3. S -> 0M | M0 | M0M
M -> ε | 0M1 | 1M0 | 0M | M0