The Pumping Lemma

If a language L can be recognized by a finite automata, then there is a number p (the pumping length) where, if s is any string L of length at least p, then s may divided into three pieces s = xyz, satisfying the following conditions:

1. For each $i \geq 0, xy^i z \in L$
2. $|y| > 0$
3. $|xy| \leq p$

Mike Solves a Pumping Lemma Problem (Annotated Text Version)

I figured maybe having something you can refer to might be more useful than another live example. Say I want prove that the language ADDER cannot be recognized by a finite automaton (note that in this example, my language has the characters “0” “1” “+” “=” rather than “A” and “B” like usual):

ADDER: x=y+z where x y and z are binary integers and x is the sum of y and z

So…“1=0+1” is in the language, but “++++=” and “111=1+1” is not.

Ok, I am going to prove this language cannot be recognized by a finite automaton. As usual, I’ll assume it can be recognized, and then prove a contradiction with the pumping lemma.

The next step is figuring out a string to pump. This is often the only hard part of these proofs. I can choose any string that is in the language. What I want is a string that when we expand the first part of the string with the pumping lemma, I’ll get a contradiction. This string always contains p somewhere. After some consideration, I choose this string:

$$10^p=10^p+0$$

You should see that that this string is definitely part of ADDER. Ok…now to apply the pumping lemma. By the lemma (and the assumption that ADDER can be recognized by a finite automaton), I know that this string (being longer than p) must be able to be divided into xyz parts. By #3 of the pumping lemma, I know that |xy| ≤ p. What this means is that y must be part of the digits before the equals sign.

Now I don’t know what x and y are exactly. It could be the case that x is empty and y almost every character to the left of the equals sign. Or, it could be that y is only one character long (we know it is not 0 characters long, by #2).

So I don’t know is if y contains the 1 or just consists of 0s. But turns out it doesn’t matter. Because when I expand y (say we think about the string xyyyz) it will no longer be the case that it equals 10^p+0.

So by the definition of ADDER, it seems that xyyyz is not in ADDER. But by #1 of the pumping lemma, xyyyz must be in ADDER. This is a contraction. So we’ve proved that ADDER cannot be recognized by a finite automaton. Make sense?

If you want to test your knowledge, prove that this language cannot be recognized by a finite automaton:

$$(AB)^n A^n$$

So the string “ABABAA” is in the language the strings “AAA” and “ABABA” are not