CS 4510 Automata and Complexity 3/1/2021

Lecture 10

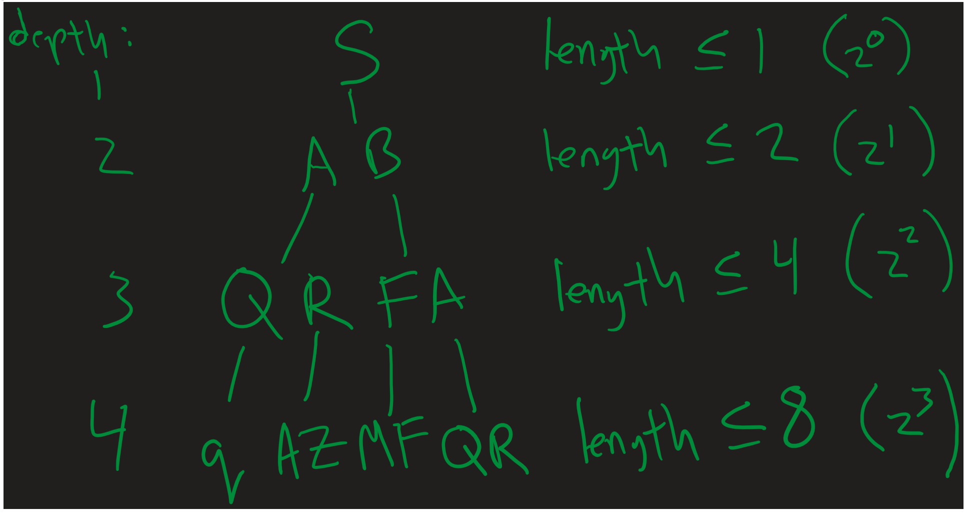
Lecturer: Frederic Faulkner *Scribe(s):*

1 The Context-Free Pumping Lemma

It's back! In the same way that the pumping lemma provided us with a characterization of regular languages, the context-free pumping lemma provides us with a characterization of context-free languages. We can then prove that certain languages are not context-free by showing that they do not obey the context-free pumping lemma.

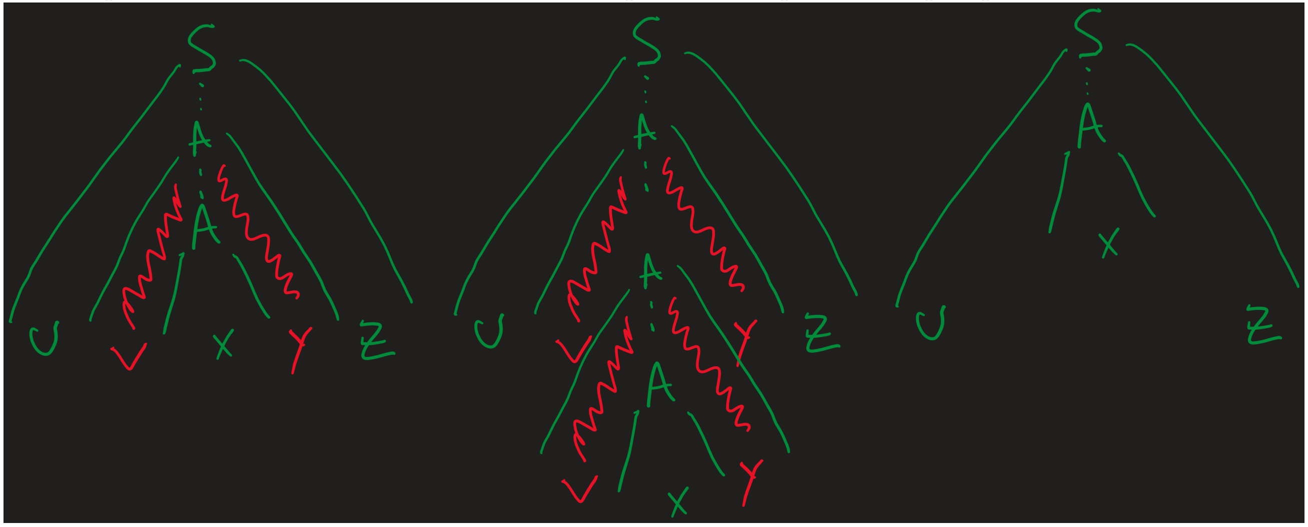
Once again, we are interested in long strings from context-free languages. The key idea is that if we pick a long enough string, we know that its derivation tree is going to be tall.

For example, consider a Chomsky normal form Grammar. We start with a terminal S (length = 1). Now, after one application of a rule, our new string has length at most 2. Suppose that our new string is AB. We replace A using some rule and B with some other other rule. Our tree is now depth 3, and our string is at most length 4. (It might be shorter if we used a $A \to a$ style of rule, but it cannot be longer.) Finally, suppose that in our new length 4 (or shorter) string, we replace each nonterminal again. How long can our string be? Since we can replace each nonterminal by at most 2 nonterminals, our new string is length 8 or shorter. See the picture below.



If we have a string of length 9 from this grammar, we know that its tree must have depth 5 (or greater), since we just proved that trees that do not achieve depth 5 cannot produce strings of length greater than 8. Similarly, suppose we wished to talk about strings whose tree height was at least h. How long a string should we pick? Well, we know that trees of height $h-1$ can only generate strings of length 2^{h-2}. So a string of length, say, $2^{h-2}+1$ must have a tree of height at least h.

The calculation we just did was particular to Chomsky Normal Form, but we could have performed a similar calculation for any context-free grammar. The most important fact is just this: *long enough strings must have tall derivation trees.* So let's pick a long string s from some arbitrary grammar G. Since s is long, it has a tall derivation tree T. Inside of T there must be some long path from S to the final string, since if all paths from S are short the entire tree would be short. But as we follow that path, if it is long enough, we will be forced to repeat some non-terminal. We can then exploit that repetition to pump S.



Effectively, if s is long enough then in the derivation $S \overset{*}{\Rightarrow} s$ has intermediate steps $S \overset{*}{\Rightarrow} uAz \overset{*}{\Rightarrow} uvAyz \overset{*}{\Rightarrow} uvxyz = s$ for some nonterminal A. So we have that $A \overset{*}{\Rightarrow} vAy$. But then also $A \overset{*}{\Rightarrow} vvAyy$, $A \overset{*}{\Rightarrow} vvvvAyyyy$, etc. So we have that, for example $S \overset{*}{\Rightarrow} uAz \overset{*}{\Rightarrow} uvAyz \overset{*}{\Rightarrow} uvvAyyz \overset{*}{\Rightarrow} uvvxyyz = uv^2xy^2z$. So we have pumped s, on the basis of nothing except that it is a long string in a context-free language.

1.1 Getting Formal

Definition 1 *Informally, all the long strings in a context-free grammar have a pair of repeatable substrings that are close together.*

Formally, every context-free language L has a pumping length p such that for long strings $s \in L$ ($|s| \geq p$), we can break s into five parts u, v, x, y, and z such that

- $|vxy| \leq p$
- $v \circ y \neq \varepsilon$
- $uv^ixy^iz \in L$ *for* $i \in \mathbb{Z}^{\geq 0}$

Here are some things to notice about the context-free pumping lemma. First, you must pump v and y an equal number of times. Secondly, since we have that $v \circ y \neq \varepsilon$, either v or y is not the empty string - but the other might be empty! Thirdly, unlike for the pumping lemma, we have no restriction on the length of the first "piece" of s (no length restriction on u). So unlike the pumping lemma, we cannot just examine the beginning of the string to find the pumpable part; v and y might come from the middle or the end of the string.

Fourthly, however, we do at least have that $|vxy| \leq p$, so we know that v and y are located near each other in the string; it is not possible for v to come from the beginning but for y to come from the end.

2 Non-context-free languages

We will now look at some examples of how to use the context-free pumping lemma to prove that a language is not context free.

Theorem 2 $L_1 = \{0^i 1^i 2^i \mid i \in \mathbb{Z}^{\geq 0}\}$ *is not context-free.*

Proof.

- Suppose L_1 is context-free. Then according to the pumping lemma, L_1 has some pumping length p.
- Pick a long string s from L_1. For example, $s = 0^p 1^p 2^p$, since $s \in L_1$ and $|s| \geq p$.
- According to the pumping lemma, we can split s into five parts, i.e. $s = uvxyz$, such that
  - $|vxy| \leq p$
  - $v \circ y \neq \varepsilon$
  - $uv^i xy^i z \in L$ for all nonnegative integers i.
- However, consider $uvvxyyz$........

Hmm, what does $xyyz$ look like? Back when we were doing pumping lemma proofs for regular languages, this step was very simple because we just had to consider the beginning of the string. However, now we must consider many more possibilities. For example,

- $v = 0^k$, $y = 0^m$
- $v = 0^k$, $y = 0^m 1^n$
- $v = 1^k$, $y = 1^m 2^n$

and many other cases besides. Of course, if you structure your proof carefully, you can reason about many of these cases simultaneously so your proof may still be short. BUT the point is that your proof must account for all of these cases, either by handling each case separately, or by some clever grouping that simplifies the proof.

As an example, we will group them into 3 cases:

1. Perhaps at least one of v or y contains a 0. Then it is not possible for either to contain a 2 (why?). So $uvvxyyz$ will have more 0's than 2's and is not in L.
2. Perhaps at least one v or y contains a 2. Similarly then, neither v nor y can contain a 0, so $uvvxyyz$ has more 2's than 0's and is not in L.

3. If v and y contain neither 0's nor 2's, then it must be the case that v and y consist only of 1's. In this case, $uvvxyyz$ has more 1's than 2's or 0's, so is not in L.

This completes the proof, since we have shown that no matter how we split $s = uvxyz$, s cannot be pumped.

Note an important point: why can't vy contain both 0's and 2's? Remember that the pumping lemma says that $|vxy| \leq p$. But in order to have both 0's and 2's, vxy would have to span the entire section of 1's, and so would have length greater than p (since there are p 1's.) In some sense, this length requirement for vxy forces v and y to be "close" to each other in s.

Theorem 3 $L = \{ww \mid w \in \Sigma^*\}$ *is not context-free.*

Proof.

- Suppose that L is context-free. Then it has some pumping length p.
- Let $s = 0^p10^p1$. ($|s| \geq p$, $s \in L$).
- According to the pumping lemma, we can split $s = uvxyz$ such that $|vxy| \leq p$, $v \circ y \neq \varepsilon$, and $uv^ixy^iz \in L$ for all nonnegative i.
- hmm...

This proof, as attempted, will fail, because s can be pumped! Let $u = 0^{p-1}$, $v = 0$, $x = 1$, $y = 0$, and $z = 0^{p-1}1$. Then, no matter which i we choose, $uv^ixy^iz = 0^{p+i-1}10^{p+i-1}1$, which is in L and so is NOT a contradiction.

Let's try again.

Proof.

- Suppose that L is context-free. Then it has some pumping length p.
- Let $s = 0^p1^p0^p1^p$. ($|s| \geq p$, $s \in L$).
- According to the pumping lemma, we can split $s = uvxyz$ such that $|vxy| \leq p$, $v \circ y \neq \varepsilon$, and $uv^ixy^iz \in L$ for all nonnegative i.
- Cases:
  - vxy falls entirely on the left hand side of s. Then $uxz = 0^{p-i}1^{p-j}0^p1^p$. Now the first half of the word ends in 0 but the second half ends in 1, so it is not in L. Contradiction.
  - vxy falls entirely on the right hand side of s. Then $uxz = 0^p1^p0^{p-i}1^{p-j}$. Now the first half of the word ends begins with a 0 but the second half begins with a 1, so it is not in L. Contradiction.
  - vxy straddles the center. Then $uxz = 0^p1^i0^j1^p$ which is not in L, since the two halves of the the string either disagree about the number of 0's or the number of 1's (or both). Contradiction.