CS340: Theory of Computation — Sem I 2019-20

Lecture Notes 10: Non Context-Free Languages

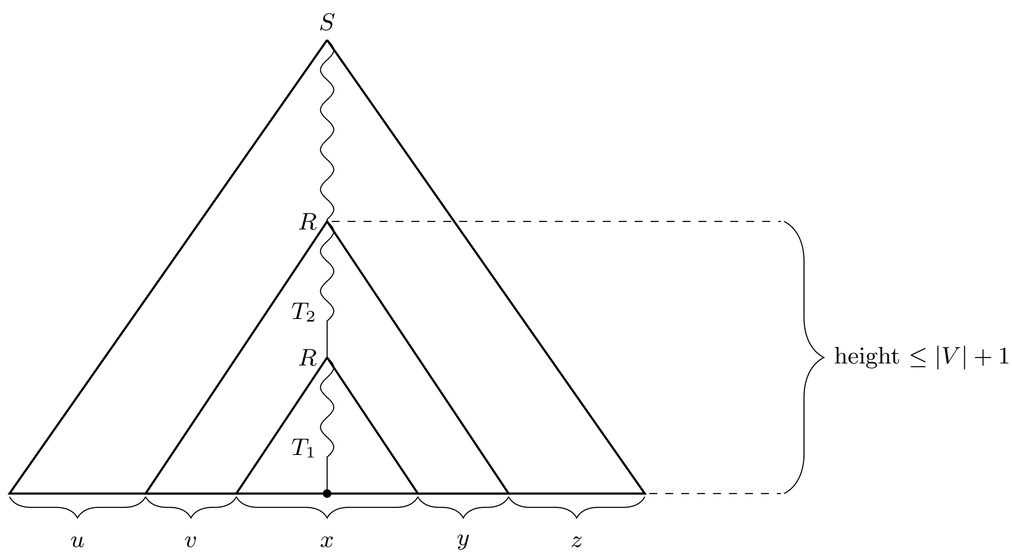
Raghunath Tewari — IIT Kanpur

1 Pumping Lemma for Context-free Languages

We will prove a pumping lemma for context-free languages. Let L be a CFL and $G = (V, \Sigma, P, S)$ be a CFG such that $L = L(G)$. Let w be a string in L. Consider a smallest parse tree of w with respect to G (say $T_{G,w}$). Few observations:

- A path from the root to a leaf in $T_{G,w}$ is a sequence of variables ending with a terminal/ϵ.
- The height of a tree is the maximum number of edges on a path from the root to a leaf node.
- Let d be the maximum degree of a node in $T_{G,w}$. If the height of the tree is h, then $|w| \leq d^h$.
- Recall that w is the concatenation of the terminal symbols at the leaves of $T_{G,w}$, from left to right.

If $|w| \geq d^{|V|+1}$, then height of $T_{G,w}$ is at least $|V| + 1$ (no. of nodes is at least $|V| + 2$) and there exists a path in $T_{G,w}$ from root to a leaf on which it has at least $|V| + 1$ variables. Consider the lowest $|V| + 1$ variables on that path. By pigeon hole principle there exists a variable R which appears twice on that portion of the path. We define a partition of $w = uvxyz$ as illustrated in the figure below.

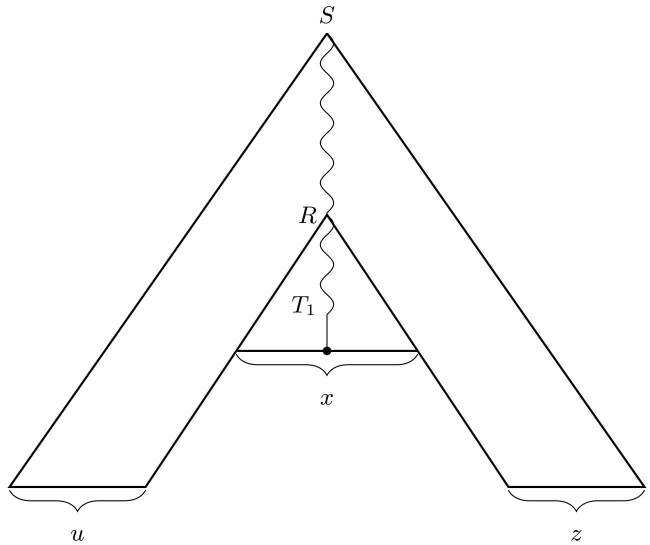


In the above parse tree for w, T_1 is the subtree rooted at the bottom R and it generates the string x and T_2 is the subtree rooted at the top R and it generates the string vxy.

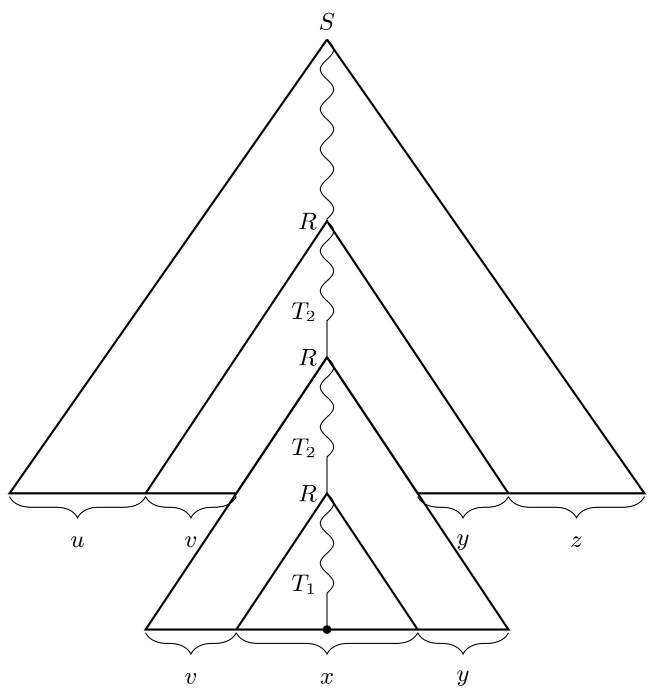
Observation 1. *Suppose there are two internal nodes in a parse tree labelled with the same variable say A, and say T_1^A and T_2^A are the subtrees rooted at these two nodes respectively. If we replace T_1^A with T_2^A or vice versa then we will still get a parse tree for* some *string in the language of the grammar (essentially the string formed by concatenating the leaves from left to right).*

- Since height of T_2 is at most $|V|+1$, therefore $|vxy| \le d^{|V|+1}$.
- Moreover since $T_{G,w}$ is the smallest parse tree of w with respect to G, therefore T_1 cannot be substituted for T_2 to get the same string w. This implies that both v and y cannot be the empty string. Therefore $|vy| > 0$.

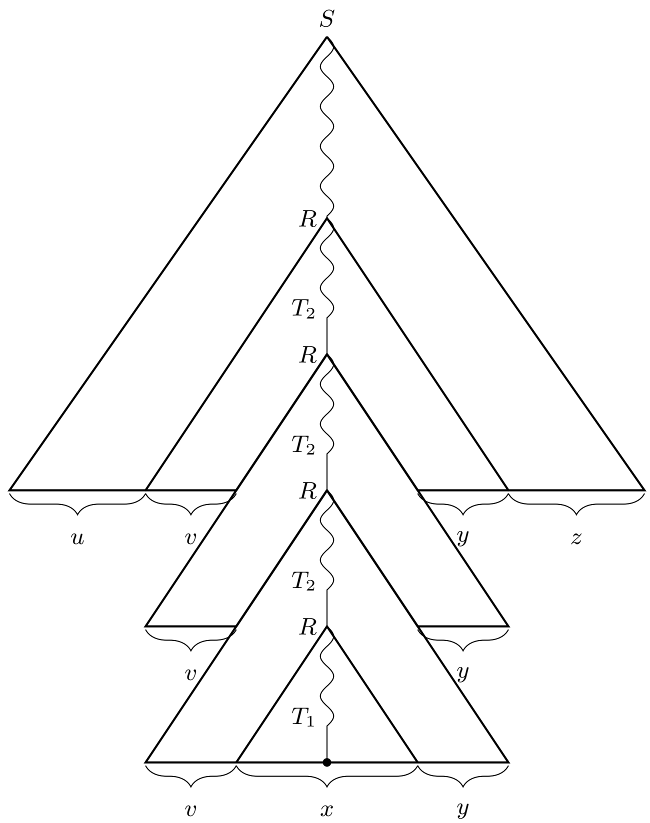
By Obversation 1 if we replace T_2 with T_1 we get the parse tree of the string uxz and hence this string is in L. The parse tree is shown below.



Similarly if we replace the T_1 with T_2 we get the parse tree of the string uv^2xy^2z and hence this string is in L. The parse tree is shown below.



Once again if we replace the T_1 with T_2 in the above parse tree we get the parse tree of the string uv^3xy^3z and hence this string is in L as well. The parse tree is shown below.



We can generalize and extend the above argument to show that for all $i \geq 0$, $uv^ixy^iz \in L$. Now by setting $p = d^{|V|+1}$ we get the following theorem.

Theorem 2 (Pumping Lemma for Context-free Languages)**.** *Let L be a context-free language. Then there exists an integer $p > 0$, such that for all $w \in L$ of length at least p, there exists a partition of $w = uvxyz$ such that $|vxy| \leq p$, $|vy| > 0$, and for all $i \geq 0$, $uv^ixy^iz \in L$.*

Remark. The choice of p for a CFL L is solely dependent on the CFG that we choose for L. Recall that $p = d^{|V|+1}$. Here d is the maximum number of symbols in the right hand side of a substitution rule in the CFG and V is of course the variable set of the CFG. Hence a different grammar for the same language might give a different p.

To prove that languages are not context-free, the pumping lemma will be used in its contrapositive form.

Theorem 3 (Contrapositive form of Pumping Lemma for CFLs)**.** *Let L be a language. If*

- $\forall p \geq 0$, *(opponent's move)*
- $\exists w \in L$ *with* $|w| \geq p$, *such that,* *(your move)*
- $\forall$ *possible partitions of* w *as* $w = uvxyz$, *satisfying* *(opponent's move)*
    - $|vxy| \leq p$, *and*
    - $|vy| > 0$,
- $\exists i \geq 0$ *such that* $uv^ixy^iz \notin L$, *(your move)*

then L is not context-free.

2 Examples of Non Context-free Languages

1.
$$L_1 = \{a^n b^n c^n \mid n \geq 0\}$$

Given p, choose $w = a^p b^p c^p$. Now for any partition $w = uvxyz$, set $i = 2$. We show below that $w' = uv^2xy^2z$ is not in L_1.

Consider the string vxy. Since $|vxy| \leq p$, therefore vxy cannot contain all three symbols. More specifically, it does not contain either a or c. Assume that it does not contain c's. Also since v and y cannot both be empty, therefore w' will have more number of either a's or b's than the number of c's. Hence $w' \notin L_1$. The case when w' does not contain a's is analogous.

2.
$$L_2 = \{ww \mid w \in \{a, b\}^*\}$$

Given p, choose $w = a^p b^p a^p b^p$. Clearly $w \in L_2$ and has length at least p. Now for any partition $w = uvxyz$, consider the following cases.

Case 1: vxy has only a's or only b's. We set $i = 2$ and let $w' = uv^2xy^2z$. Assume vxy lies in the first block of a's. Let $|vy| = k$. Now $0 < k \leq p$. As a result the first half of w' is $a^{p+k}b^{p-k/2}$ and second half of w' is $b^{k/2}a^p b^p$. Clearly the strings are not equal and hence $w' \notin L_2$.

If vxy lies in any other block, the argument is analogous.

Case 2: vxy has both a's and b's. We set $i = 0$ and let $w' = uxz$. Assume vxy straddles the first boundary between a's and b's. Let $vy = a^{k_1}b^{k_2}$. Note that Now $0 < k_1 + k_2 \leq p$. Then $w' = a^{p-k_1}b^{p-k_2}a^p b^p$. Then the first half of w' is $a^{p-k_1}b^{p-k_2}a^{\frac{k_1+k_2}{2}}$ and the second half is $a^{p-\frac{k_1+k_2}{2}}b^p$. Clearly the strings are not equal and hence $w' \notin L_2$.

If vxy straddles any other boundary, the argument is analogous.

Remark. Note that in the above proof we could have fixed $i = 0$ or $i = 2$ for both the cases. But that would make the argument a little more tedious. Also the above proof illustrates the fact that i can vary on a case by case basis.

Exercise 1. Prove that the following languages are not context-free.

(a) $L_1 = \{a^n b^m c^n d^m \mid n, m \geq 0\}$
(b) $L_2 = \{0^n 1^{n^2} \mid n \geq 0\}$
(c) $L_3 = \{0^n \mid n \text{ is prime}\}$