# Lecture 13: Even More on Context-Free Grammars

5 March 2009

## 1 Grammars in CNF form have compact parsing trees

In this section, we prove that CNF give very compact parsing trees for strings in the language of the grammar.

In the following, we will need the following easy observation.

**Observation 1.1** *Consider a grammar $G$ which is CNF, and a variable $\mathsf{X}$ of $G$ which is not the start variable. Then, any string derived from $\mathsf{X}$ must be of length at least one.*

**Claim 1.2** *Let $\mathcal{G} = (\mathcal{V}, \Sigma, \mathcal{R}, \mathsf{S})$ be a context-free grammar in Chomsky normal form, and $w$ be a string of length one. Furthermore, assume that there is a $\mathsf{X} \in \mathcal{V}$, such that $\mathsf{X} \stackrel{*}{\Longrightarrow} w$ (i.e., $w$ can be derived from $\mathsf{X}$), and let $T$ be the corresponding parse tree for $w$. Then the tree $T$ has exactly $2|w| - 1$ internal nodes.*

*Proof:* A *full binary tree* is a tree were every node other than the leaves has two children. It is easy to verify by easy induction that such a tree with $m$ leaves, has $m - 1$ internal nodes.

Now, the given tree $T$ is not quite a full binary tree. Indeed, the $k$th leaf (from the left) of $T$, denoted by $\ell_k$, is the $k$ character of $w$, and its parent must be a node labeled by a variable, $X_k$, for $k = 1, \ldots, n$. Furthermore, we must have that the parent of $\ell_k$ has only a single child. As such, if we remove the $n$ leafs of $T$, we remain with a full binary tree $T'$ with $n$ leafs (every parent of a leaf $\ell_k$ became a leaf). This tree is a full binary tree, because any internal node, must correspond to a non-terminal derivation of a CNF grammar, and any such derivation has the form $\mathsf{X} \to \mathsf{Y}VZ$; that is, the derivation corresponds to an internal node with two children in $T$. Now, the tree $T'$ has $n - 1$ internal nodes, by the aforementioned fact about full binary trees. As such, $T'$ has $n - 1$ internal nodes. Each leaf of $T'$ is an internal node of $T$, and $T'$ has $n$ such leafs. We conclude that $T$ has $2n - 1$ internal nodes. ■

**Alternative proof for Claim 1.2.**

*Proof:* The proof is by induction on the length of $w$.

If $|w| = 1$ then we claim that $w$ must be derived by a single rule $\mathsf{X} \to c$, where $c$ is a character. Otherwise, if the root corresponds to a rule of the form $\mathsf{X} \to \mathsf{YZ}$, then by the above observation, the generated string for $\mathsf{Y}$ and $\mathsf{Z}$ are each of length at least one, which implies that the overall length of the word is at least 2, a contradiction. (We are using here implicitly the property of a CNF that the start variable can never appear on the right side of a rule, and that the start symbol is the only symbol that can yield the empty string.)

As such, if $|w| = 1$, then the parsing tree has a single internal node, and the claim holds as $2|w| - 1 = 1$.

So, assume that we proved the claim for all words strictly shorter than $w$, and consider the parse tree $T$ for $w$ being derived from some variable $\mathsf{X} \in \mathcal{V}$. Since $|w| > 1$, it must be that the root of the parse tree $T$ corresponds to a rule of the form $\mathsf{S} \rightarrow \mathsf{X}_1\mathsf{X}_2$. Let $w_1$ and $w_2$ be the portions of $w$ generated by $\mathsf{X}_1$ and $\mathsf{X}_2$ respectively, and let $T_1$ and $T_2$ denote the corresponding subtrees of $T$. Clearly $w = w_1 w_2$, $|w_1| > 0$ and $|w_2| > 0$, by the above observation. Clearly, $T_1$ (resp. $T_2$) is a tree that derives $w_1$ (resp. $w_2$) from $\mathsf{X}_1$ (resp. $\mathsf{X}_2$). By induction, $T_1$ (resp. $T_2$) has $2|w_1| - 1$ (resp. $2|w_2| - 1$) internal nodes. As such, $T$ has

$$\begin{aligned} N &= 1 + \binom{\#\text{ internal}}{\text{nodes of } T_1} + \binom{\#\text{ internal}}{\text{nodes of } T_2} \\ &= 1 + (2|w_1| - 1) + (2|w_2| - 1) = 2(|w_1| + |w_2|) - 1 \\ &= 2|w| - 1, \end{aligned}$$

■

The following is the same claim, restated as a claim on the number of derivation used.

**Claim 1.3** *if $G$ is a context-free grammar in Chomsky normal form, and $w$ is a string of length $n \geq 1$, then any derivation of $w$ from any variable $\mathsf{X}$ contains exactly $2n - 1$ steps.*

**Theorem 1.4** *Given a context free grammar $\mathcal{G}$, and a word $w$, then one can decide if $w \in L(\mathcal{G})$ by an algorithm that always stop.*

*Proof:* Convert $\mathcal{G}$ into Chomsky normal form, and let $\mathcal{G}'$ be the resulting grammar tree. Let $n = |w|$. Observe that $w$ has a parse tree using $\mathcal{G}'$ with $2n - 1$ internal nodes, by Claim 1.2. Enumerate all such possible parse trees (their number is large, but finite), and check if any of them is (i) a legal parse tree for $\mathcal{G}'$, and (ii) it derives the word $w$. If we found such a legal tree deriving $w$, then $w \in L(\mathcal{G})$. Otherwise, $w$ can not be generated by $\mathcal{G}'$, which implies that $w \notin L(\mathcal{G})$. ■

# 2 Closure properties for context-free grammars

Context-free languages are closed under the following operations: union, concatenation, star, string reversal, homomorphism, and intersection with a regular language.

Notice that they are *not* closed under intersection (i.e. intersection of two context-free languages). Nor are they closed under set complement (i.e. the complement of a context-free language is not always context-free). We will show these non-closure facts later on, when we have established that some sample languages are definitely not context-free.

**Example 2.1** Here is a quick argument why CFG languages are not closed under intersection. Consider the language $L = \left\{ \mathtt{a}^n\mathtt{b}^n\mathtt{c}^n \;\middle|\; n \geq 0 \right\}$, which (as we stated above is not CFG- a fact we will prove in the near future). However, it is easy to verify that any word $\mathtt{a}^n\mathtt{b}^n\mathtt{c}^n$, for $n \geq 0$, is in the intersection of the languages

$$L_1 = \left\{ \mathtt{a}^*\mathtt{b}^n\mathtt{c}^n \;\middle|\; n \geq 0 \right\} \quad \text{and} \quad L_2 = \left\{ \mathtt{a}^n\mathtt{b}^n\mathtt{c}^* \;\middle|\; n \geq 0 \right\}.$$

In fact, $L = L_1 \cap L_2$. Now, it is easy to verify that $L_1$ and $L_2$ are CFG, since

$$\begin{aligned} S &\to A_{\text{star}} X \\ A_{\text{star}} &\to \mathtt{a} A_{\text{star}} \mid \epsilon \\ X &\to \mathtt{b} X \mathtt{c} \mid \epsilon, \end{aligned}$$

with the start symbol being $S$ is a CFG for $L_1$ (a similar grammar works for $L_2$). As such, both $L_1$ and $L_2$ are CFG, but their intersection $L = L_1 \cap L_2$ is not CFG. Thus, context-free languages are not closed under intersection.

## 2.1 Proving some CFG closure properties

Most of the closure properties are most easily proved using context-free grammars. These constructions are fairly easy, but they will help you become more familiar with the features of context-free grammars.

### 2.1.1 CFGs are closed under union

Suppose we have grammars for two languages, with start symbols $S$ and $T$, respectively. Rename variables (in the two grammars) as needed to ensure that the two grammars do not share any variables. Then construct a grammar for the union of the languages, with start symbol $Z$, by taking all the rules from both grammars together and adding a new rule $Z \to S \mid T$.

### 2.1.2 Concatenation.

Suppose we have grammars for two languages, with start symbols $S$ and $T$. Rename variables as needed to ensure that the two grammars do not share any variable. Then construct a grammar for the union of the languages, with start symbol $Z$, by taking all the rules from both grammars and adding a new rule $Z \to S\,T$.

### 2.1.3 Star operator

Suppose that we have a grammar for the language $L$, with start symbol $S$. The grammar for $L^*$, with start symbol $T$, contains all the rules from the original grammar plus the rule $T \to T\,S \mid \epsilon$.

### 2.1.4 String reversal

Reverse the character string on the righthand side of every rule in the grammar.

### 2.1.5 Homomorphism

Suppose that we have a grammar $G$ for language $L$ and a homomorphism $h$. To construct a grammar for $h(L)$, modify the righthand side of every rule in $G$ to replace each terminal symbol $t$ with its image $h(t)$ under the homomorphism.

## 2.2 CFG are closed under intersection with a regular language

### 2.2.1 Informal description

It is also true that the intersection of a context-free language with a regular language is always context-free. If we are manipulating a context-free language $L$ in a proof, we can intersect it with a regular language to select a subset of $L$ that has some particular form. For example, if $L$ contains all strings with equal numbers of a's and b's, we can intersect it with $\mathtt{a}^*\mathtt{b}^*$ to get the language[1] $\mathtt{a}^n\mathtt{b}^n$.

So, assume we have a CFG $\mathcal{G} = (\mathcal{V}, \Sigma, R, \mathsf{S})$ accepting the context-free language $L_{\mathsf{CFG}}$ and a DFA $\mathtt{M} = (Q, \Sigma, \delta, q_{\text{init}}, F)$ accepting a regular language $L_{\text{reg}}$. Furthermore, the CFG $\mathcal{G}$ is in Chomsky Normal Form (i.e., CNF).

The idea of building a grammar for the intersection language is to write a new CFG grammar, where a variable $\mathsf{X}$ of $\mathcal{G}$ would be replaced by the set of variables

$$\left\{ \mathsf{X}_{q \rightsquigarrow q'} \;\middle|\; q, q' \in Q \text{ and } \mathsf{X} \in \mathcal{V} \right\}.$$

Here, the variable $\mathsf{X}_{q \rightsquigarrow q'}$ represents all strings that can be derived by the variable $\mathsf{X}$ of $\mathcal{G}$, and furthermore if we feed such a string to $\mathtt{M}$ (starting at state $q$), then we would reach the state $q'$. So, consider a rule of the form

$$\mathsf{X} \to \mathsf{YZ}$$

that is in $\mathcal{G}$. For every possible starting state $q$, and ending state $q'$, we want to generate a rule for the variable $\mathsf{X}_{q \rightsquigarrow q'}$. So we derive a substring $w$ for $\mathsf{Y}$. Feeding the $\mathtt{M}$ the string $w$, starting at $q$, would lead us to a state $s$. As such, the string generated from $\mathsf{Y}$in this case, would move $\mathtt{M}$ from $q$ to $s$, and the string generated by $\mathsf{Z}$would move $\mathtt{M}$ from $s$ to $q'$. That is, this rule can be rewritten as

$$\forall q, q', s \in Q \quad \mathsf{X}_{q \rightsquigarrow q'} \to \mathsf{Y}_{q \rightsquigarrow s}\, \mathsf{Z}_{s \rightsquigarrow q'}.$$

If we have a rule of the from $\mathsf{X} \to c$ in $\mathcal{G}$, then we create the rule $\mathsf{X}_{q \rightsquigarrow q'} \to c$ if there is a transition in $\mathtt{M}$ from $q$ to $q'$ that reads the character $c$, where $c \in \Sigma_\epsilon$.

Finally, we create a new start variable $\mathsf{S}_0$, and we introduce the rule $\mathsf{S}_0 \to \mathsf{S}_{q_{\text{init}} \rightsquigarrow q'}$, where $q_0$ is the initial state of $\mathtt{M}$, and $q' \in F$ is an accept state of $\mathtt{M}$.

We claim that the resulting grammar accepts only words in the language $L_{\mathsf{CFG}} \cap L_{\text{reg}}$.

### 2.2.2 Formal description

We have a CFG $\mathcal{G} = (\mathcal{V}, \Sigma, R, \mathsf{S})$ and a DFA $\mathtt{M} = (Q, \Sigma, \delta, q_{\text{init}}, F)$. We now build a new grammar for the language $L(\mathcal{G}) \cap L(\mathtt{M})$. The set of new variables is

$$\mathcal{V}' = \{\mathsf{S}_0\} \cup \left\{ \mathsf{X}_{q \rightsquigarrow q'} \;\middle|\; \mathsf{X} \in \mathcal{R}, q, q' \in Q \right\}.$$

[1] Here, and in a lot of other places, we abuse notations. When we write $\mathtt{a}^n\mathtt{b}^n$, what we really mean is the language $\left\{ \mathtt{a}^n\mathtt{b}^n \;\middle|\; n \geq 0 \right\}$.

$$\mathcal{R}' = \left\{ \mathsf{X}_{q \rightsquigarrow q'} \to \mathsf{Y}_{q \rightsquigarrow s}\, \mathsf{Z}_{s \rightsquigarrow q'} \;\middle|\; \forall q, q', s \in Q \quad (\mathsf{X} \to \mathsf{YZ}) \in \mathcal{R} \right\} \tag{1}$$

$$\bigcup \left\{ \mathsf{S}_0 \to \mathsf{S}_{q_{\text{init}} \rightsquigarrow q'} \;\middle|\; q' \in F \right\} \tag{2}$$

$$\bigcup \left\{ \mathsf{X}_{q \rightsquigarrow q'} \to c \;\middle|\; (\mathsf{X} \to c) \in \mathcal{R} \text{ and } \delta(q, c) = q' \right\}. \tag{3}$$

If $\mathsf{S} \to \epsilon \in \mathcal{R}$ and $q_{\text{init}} \in F$ (i.e., $\epsilon$ is in the intersection language) then we add the rule $\{\mathsf{S}_0 \to \epsilon\}$ to $\mathcal{R}'$.

The new grammar is $\mathcal{G}_{\cap \mathtt{M}} = (\mathcal{V}', \Sigma, \mathcal{R}', \mathsf{S}_0)$.

**Observation 2.2** *The new grammar $\mathcal{G}_{\cap \mathtt{M}}$ is "almost" a CNF. That is, if we ignore rules involving the start symbol $\mathsf{S}_0$ of $\mathcal{G}_{\cap \mathtt{M}}$ then its a CNF.*

### 2.2.3 Correctness

**Lemma 2.3** *Let $\mathcal{G}$ be a context-free grammar in Chomsky normal form, and let $\mathtt{M}$ be a DFA. Then one can construct a grammar is a grammar $\mathcal{G}_{\cap \mathtt{M}} = (\mathcal{V}', \Sigma, \mathcal{R}', \mathsf{S}_0)$, such that, for any word $w \in \Sigma^* \setminus \{\epsilon\}$, we have that $\mathsf{X} \stackrel{*}{\Longrightarrow} w$ and $\delta(q, w) = q'$ if and only if $\mathsf{X}_{q \rightsquigarrow q'} \stackrel{*}{\Longrightarrow} w$.*

*Proof:* The construction is described above, and proof is by induction of the length of $w$.

$\Big[|w| = 1\Big]$**:** If $|w| = 1$ then $w = c$, where $c \in \Sigma$.

Thus, if $\mathsf{X} \stackrel{*}{\Longrightarrow} w$ and $\delta(q, w) = q'$ then $\mathsf{X} \to c$ is in $\mathcal{R}$, which implies that we introduced the rule $\mathsf{X}_{q \rightsquigarrow q'} \to c$ into $\mathcal{R}$, which implies that $\mathsf{X}_{q \rightsquigarrow q'} \stackrel{*}{\Longrightarrow} w$.

Similarly, if $\mathsf{X}_{q \rightsquigarrow q'} \stackrel{*}{\Longrightarrow} w$ then since $\mathcal{G}_{\cap \mathtt{M}}$ is a CNF, and $|w| = 1$, this implies that there must be a derivation $\mathsf{X}_{q \rightsquigarrow q'} \to c$. But this implies, by construction, that $\mathsf{X} \to c$ is a rule of $\mathcal{G}$ and $\delta(q, c) = q'$, as required.

$\Big[|w| > 1\Big]$**:** Assume, that by induction, the claim holds for all words strictly shorter than $w$.

- $\mathsf{X} \stackrel{*}{\Longrightarrow} w$ and $\delta(q, w) = q' \implies \mathsf{X}_{q \rightsquigarrow q'} \stackrel{*}{\Longrightarrow} w$.

  IF $\mathsf{X} \stackrel{*}{\Longrightarrow} w$ and $\delta(q, w) = q'$, then consider the parse tree of $\mathcal{G}$ deriving $\mathsf{X}$ from $w$. Since $\mathcal{G}$ is a CNF, we have that the root of this parse tree $T$ corresponds to a rule of the form $\mathsf{X} \to \mathsf{YZ}$. Let $w_{\mathsf{Y}}$ and $w_{\mathsf{Z}}$ be the two sub-words derived by these two subtrees of the $T$. Clearly, $w = w_{\mathsf{Y}} w_{\mathsf{Z}}$, and since $\mathcal{G}$ is a CNF, we have that $|w_{\mathsf{Y}}|, |w_{\mathsf{Z}}| > 0$ (since any symbol except the root in a CNF derives a word of length at least 1). As such, $|w_{\mathsf{Y}}|, |w_{\mathsf{Z}}| < |w|$. Now, let $q'' = \delta(q, w_{\mathsf{Y}})$. We have that

  $$\mathsf{Y} \stackrel{*}{\Longrightarrow} w_{\mathsf{Y}}, \quad 0 < |w_{\mathsf{Y}}| < |w|, \quad \text{and} \quad q'' = \delta(q, w_{\mathsf{Y}}).$$

  As such, by induction, it must be that $\mathsf{Y}_{q \rightsquigarrow q''} \stackrel{*}{\Longrightarrow} w_{\mathsf{Y}}$. Similarly, since $\delta(q'', w_{\mathsf{Z}}) = q'$, and by the same argument, we have that $\mathsf{Z}_{q'' \rightsquigarrow q'} \stackrel{*}{\Longrightarrow} w_{\mathsf{Z}}$. Now, by Eq. (1), we have the rule $\mathsf{X}_{q \rightsquigarrow q'} \to \mathsf{Y}_{q \rightsquigarrow q''} \mathsf{Y}_{q'' \rightsquigarrow q}$ in $\mathcal{R}'$. Namely,

  $$\mathsf{X}_{q \rightsquigarrow q'} \to \mathsf{Y}_{q \rightsquigarrow q''} \mathsf{Y}_{q'' \rightsquigarrow q} \stackrel{*}{\Longrightarrow} w_{\mathsf{Y}} w_{\mathsf{Z}} = w,$$

implying the claim.

– $\mathsf{X}_{q \rightsquigarrow q'} \stackrel{*}{\Longrightarrow} w \implies \mathsf{X} \stackrel{*}{\Longrightarrow} w$ and $\delta(q, w) = q'$.

If $\mathsf{X}_{q \rightsquigarrow q'} \stackrel{*}{\Longrightarrow} w$, and $|w| > 1$, then consider the parsing tree $T'$ of $w$ from $\mathsf{X}_{q \rightsquigarrow q'}$, and let

$$\mathsf{X}_{q \rightsquigarrow q'} \to \mathsf{Y}_{q \rightsquigarrow q''} \mathsf{Y}_{q'' \rightsquigarrow q}.$$

be the ruled used in the root of $T'$, and let $w_{\mathsf{Y}}, w_{\mathsf{Z}}$ be the two substrings of $w$ generated by these two subtrees. That is $w = w_{\mathsf{Y}} w_{\mathsf{Z}}$. By induction, we have that

$$\mathsf{Y} \stackrel{*}{\Longrightarrow} w_{\mathsf{Y}}, \delta(q, w_{\mathsf{Y}}) = q'', \qquad \text{and} \qquad \mathsf{Z} \stackrel{*}{\Longrightarrow} w_{\mathsf{Z}}, \delta(q'', w_{\mathsf{Z}}) = q'.$$

Now, by construction, the rule $\mathsf{X} \to \mathsf{YZ}$ must be in $\mathcal{R}$. As such $\mathsf{X} \to \mathsf{YZ} \stackrel{*}{\Longrightarrow} w_{\mathsf{Y}} w_{\mathsf{Z}} = w$, and

$$\delta(q, w) = \delta(q, w_{\mathsf{Y}} w_{\mathsf{Z}}) = \delta(\delta(q, w_{\mathsf{Y}}), w_{\mathsf{Z}}) = \delta(q'', w_{\mathsf{Z}}) = q'.$$

Thus, $\mathsf{X} \stackrel{*}{\Longrightarrow} w$ and $\delta(q, w) = q'$, thus implying the claim.

■

**Theorem 2.4** *Let $L$ be a context-free language and $L'$ be a regular language. Then, $L \cap L'$ is a context free language.*

*Proof:* Let $\mathcal{G} = (\mathcal{V}, \Sigma, \mathcal{R}, \mathsf{S})$ be a CNF for $L$, and let $\mathtt{M} = (Q, \Sigma, \delta, q_{\text{init}}, F)$ be a DFA for $L'$. We apply the above construction to compute a grammar $\mathcal{G}_{\cap \mathtt{M}} = (\mathcal{V}', \Sigma, \mathcal{R}', \mathsf{S}_0)$ for the intersection.

- $w \in L \cap L' \implies w \in L(\mathcal{G}_{\cap \mathtt{M}})$.

  If $w = \epsilon$ then the rule $\mathsf{S}_0 \to \epsilon$ is in $\mathcal{G}_{\cap \mathtt{M}}$ and we have that $\epsilon \in L(\mathcal{G}_{\cap \mathtt{M}})$.

  For any other word, if $w \in L \cap L'$ then $\mathsf{S} \stackrel{*}{\Longrightarrow} w$ and $q' = \delta(q_{\text{init}}, w) \in F$ then, by Lemma 2.3, we have that

  $$\mathsf{S}_{q_{\text{init}} \rightsquigarrow q'} \stackrel{*}{\Longrightarrow} w.$$

  Furthermore, by construction, we have the rule

  $$\mathsf{S}_0 \to \mathsf{S}_{q_{\text{init}} \rightsquigarrow q'}.$$

  As such, $\mathsf{S}_0 \stackrel{*}{\Longrightarrow} w$, and $w \in L(\mathcal{G}_{\cap \mathtt{M}})$.

- $w \in L(\mathcal{G}_{\cap \mathtt{M}}) \implies w \in L \cap L'$.

  Similar to the above proof, and we omit it.

■