# A Short Note on LL(1) Grammars

Suresh Purini, IIIT-H

Let $G = (N, T, P, S)$ be a context-free grammar (CFG) where $N$ is the set of non-terminals, $T$ is the set of terminals, $P$ is the set of production rules and $S \in N$ is the start symbol.

**Definition 1.** *For a non-terminal symbol $A \in N$ define*

$$FIRST(A) = \{\ a \in T \mid \exists\ A \Rightarrow^{*} a\alpha \textit{ where } \alpha \in (N \cup T)^{*}\ \} \cup \{\ \epsilon \mid \exists\ A \Rightarrow^{*} \epsilon\ \}.$$

**Definition 2.** *For a sentential form $\alpha \in (N \cup T)^{*}$ define*

$$FIRST(\alpha) = \{\ a \in T \mid \exists\ \alpha \Rightarrow^{*} a\beta \textit{ where } \beta \in (N \cup T)^{*}\ \} \cup \{\ \epsilon \mid \exists\ \alpha \Rightarrow^{*} \epsilon\ \}.$$

Note that the Definition 2 actually subsumes the Definition 1. However the definitions are stated seperately for the sake of clarity.

Let us assume that after a sequence of left-most derivation steps, a top-down recursive descent parser has constructed a partial parse tree and $uA\alpha$ be the left-sentential form labeling the leaf nodes of the parse tree (refer Figure 1). Also let the input string be $w = uav$ where $u$ is prefix that had already been derived and $a$ is the correct look-ahead token in the input stream.

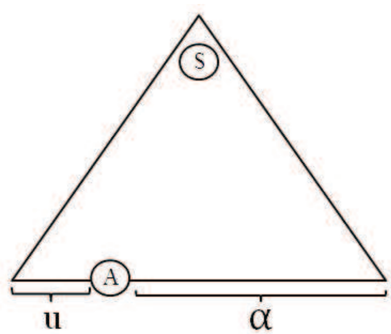


Figure 1: Partially Expanded Parse Tree

Let $A \rightarrow \alpha_1 \mid \cdots \mid \alpha_k$ be the production rules corresponding to the non-terminal symbol A in the CFG $G$. In order to avoid back-tracking the top-down parser wants to deterministically choose one of the $k$ alternate production rules to expand the parse tree using the look-ahead symbol $a$ as the extra information. We shall first take a **Naive Approach** to solve this problem and later have to refine it to get the **LL(1) Approach**. Throughout our discussion $uA\alpha$ is the current left-sentential form and $a$ is the look-ahead symbol.

**Naive Approach.** *Use the production rule* $A \rightarrow \alpha_i$ *to expand the parse tree if* $a \in FIRST(\alpha_i)$ *and* $a \notin FIRST(\alpha_j)$ *for* $1 \leq j \neq i \leq k$.

If we observe carefully there is a subtle flaw in the Naive Approach. Consider the case where $a \in FIRST(\alpha_i)$, $a \notin FIRST(\alpha_j)$ and $\epsilon \in FIRST(\alpha_j)$. It is quite possible for us to have a derivation like $S \Rightarrow^* uA\alpha \Rightarrow u\alpha_j\alpha \Rightarrow^* u\epsilon\alpha \Rightarrow^* ua\beta$ (refer Figure 2). However this derivation also implies the derivation $S \Rightarrow^* uA\alpha \Rightarrow^* uAa\beta$ indicating that $a \in FOLLOW(A)$.

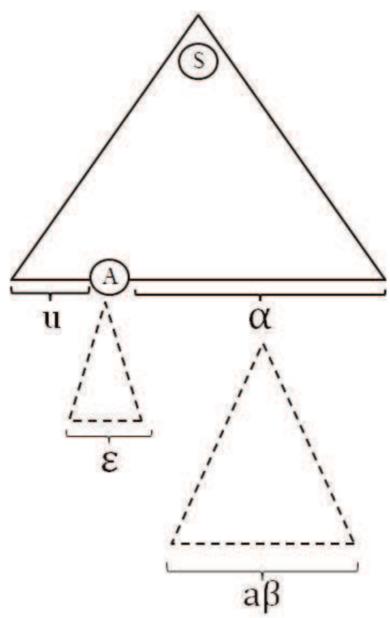


Figure 2: Alternate Parse Tree Expansion

**Claim 1.** *Given a left-sentential form $uA\alpha$, we can use the substituion rule $A \Rightarrow \alpha_i$ to expand the parse tree only if at least one of the following two conditions hold.*

1. $a \in FIRST(\alpha_i)$
2. $\epsilon \in FIRST(\alpha_i)$ *and* $a \in FOLLOW(A)$

With the previous claim in mind we give the following defintion.

**Definition 3.**

$$FIRST^+(A \to \alpha) = \begin{cases} FIRST(\alpha) & \epsilon \notin FIRST(\alpha) \\ FIRST(\alpha) \cup FOLLOW(A) & \epsilon \in FIRST(\alpha) \end{cases}$$

**LL(1) Approach.** *Use the production rule $A \to \alpha_i$ to expand the parse tree if $a \in FIRST^+(\alpha_i)$ and $a \notin FIRST^+(\alpha_j)$ for $1 \leq j \neq i \leq k$.*

**Remark:** A grammar is not LL(1)-parsable if there exists two productions $A \to \alpha_i$ and $A \to \alpha_j$ such that $FIRST^+(A \to \alpha_i) \cap FIRST^+(A \to \alpha_j) \neq \phi$.