1 Top-down Parsing (LL(1))

1.1 Definitions

• LL(1) stands for Left-to-right scan of input, Leftmost derivations, 1 symbol of lookahead.

• An LL(1) parser is a top-down parser; it begins from the start symbol and finds a derivation for the input string.

• A grammar that can be parsed using the LL(1) algorithm is called an LL(1) grammar.

• The LL(1) prediction function for a grammar \( G = (N, \Sigma, P, S) \) is defined as:

\[
\text{Predict}(A, a) = \{(A \rightarrow \gamma) \in P : a \in \text{First}(\gamma) \lor (\text{Nullable}(\gamma) \land a \in \text{Follow}(A))\}
\]

Where:

\[
\text{Nullable}(\gamma) = \top \text{ iff } \gamma \Rightarrow \epsilon
\]

\[
\text{First}(\gamma) = \{b \in \Sigma : \gamma \Rightarrow b\beta \text{ for some } \beta\}
\]

\[
\text{Follow}(A) = \{c \in \Sigma : S' \Rightarrow \alpha Ac\beta \text{ for some } \alpha, \beta\}
\]

• A grammar is LL(1) iff \(|\text{Predict}(A, a)| \leq 1 \ \forall A \in N, a \in \Sigma.\]

1.2 Exercises

Consider the following context-free grammar \( G \) and its associated predictor table:

\[
\begin{align*}
S' & \rightarrow \cdot S \cdot \\
S & \rightarrow aXYb \\
S & \rightarrow XY \\
X & \rightarrow pX \\
X & \rightarrow \epsilon \\
Y & \rightarrow q \\
Y & \rightarrow \epsilon
\end{align*}
\]

| | \( \Rightarrow \) | \( a \) | \( b \) | \( p \) | \( q \) |
|---|---|---|---|---|
| \( S' \) | 0 | | | | |
| \( S \) | 2 | 1 | 2 | 2 |
| \( X \) | 4 | 4 | 3 | 4 |
| \( Y \) | 6 | 6 | | 5 |
1. Perform a top-down parse of the string \( \vdash appqb \vdash \) and draw the parse tree.

2. Consider the grammar \( G' \), which is identical to \( G \) except with rule 1 \( (S \rightarrow aXYb) \) replaced by the rule \( S \rightarrow Sab \). Why is it impossible to parse this grammar with an LL(1) parser? That is, what aspect of the grammar makes it not LL(1)?