LL(k)

Canonical name	LLk
Date of creation	2013-03-22 19:00:54
Last modified on	2013-03-22 19:00:54
Owner	CWoo (3771)
Last modified by	CWoo (3771)
Numerical id	8
Author	CWoo (3771)
Entry type	Definition
Classification	msc 68Q05
Classification	msc 68Q42
Classification	msc 03D10
Related topic	LRk

Given a word u and a context-free grammar G, how do we determine if $u \in L(G)$?

There are in general two ways to proceed. Start from u, and proceed backward to find v such that $v \Rightarrow u$. Keep going until a derivation $\sigma \Rightarrow^* u$ is found. This procedure is known as the bottom-up parsing of u. The other method the top-down approach: begin with the starting symbol σ, and work its way down to u, so $\sigma \Rightarrow^* u$.

As with the bottom-up approach, finding a derivation of u from the top-down may be time consuming, if one is lucky enough to find a derivation at all.

There is a class of grammars, known as the $LL(k)$ grammars, which make the top-down parsing of a word natural and direct. The first L in $LL(k)$ means scanning the symbols of u from left to right, the second L stands for finding a leftmost derivation ($\Rightarrow_L$) for u, and k means having the allowance to look at up to k symbols ahead while scanning.

Definition. Let $G = (\Sigma, N, P, \sigma)$ be a context-free grammar such that $\sigma \to \sigma$ is not a production of G, and $k \geq 0$ an integer. Suppose $u \in L(G)$ with a $X \to U_1$ a production in a leftmost derivation of u:

$$\sigma \Rightarrow_L^* UXU_2 \Rightarrow_L UU_1U_2 \Rightarrow_L^* u.$$

Let $n = |U| + k$ and v be the prefix of u of length n (if $|u| < n$, then set $v = u$).

Then G is said to be $LL(k)$ if for any $w \in L(G)$, with v as a prefix, such that there is a production $X \to W_1$ in a leftmost derivation of w:

$$\sigma \Rightarrow_L^* UXW_2 \Rightarrow_L UW_1W_2 \Rightarrow_L^* w,$$

implies that $W_1 = U_1$.

In a leftmost derivation D_u of a word u, call a prefix v of u is a *leftmost descendant* of a production $P \to U$ if $\sigma \Rightarrow^* vPU' \Rightarrow vUU' \Rightarrow^* u$ is D_u. Then the definition above can be restated in words as follows:

Given a leftmost derivation D_u of a word u, a production used in D_u is uniquely determined up to k symbols beyond the prefix of u which is a leftmost descendant of the production. In other words, if D_u and D_w are leftmost derivations of u and w which agree on k symbols beyond the common prefix v, where v is both a leftmost descendant of $X \to U$ used in D_u, and a leftmost descendant of $X \to W$ used in D_w, then $X \to U$ and $X \to W$ are the same production, i.e. $U = W$.

Every $LL(k)$ is unambiguous. Furthermore, every $LL(k)$ grammar is `http://planetmath.org/LRk`$LR(k)$.

Given a context-free grammar G and $k \geq 0$, there is an algorithm deciding whether G is $LL(k)$.

Examples

- The grammar G over $\Sigma = \{a, b\}$, with productions $\sigma \to a^2\sigma b^2$, $\sigma \to a$ and $\sigma \to \lambda$ is $LL(2)$ but not $LL(1)$. It is not hard to see that $L(G)$ is the set $\{a^m b^n \mid n \text{ is even, and } n \leq m \leq n+1\}$. On the other hand, the grammar G' over Σ, with productions

  $$\sigma \to aX, \quad \sigma \to \lambda, \quad X \to aYb, \quad X \to \lambda, \quad Y \to aXb, \quad Y \to b$$

  also generates $L(G)$, but is $LL(1)$ instead.

- The grammar G over $\{a, b, c\}$, with productions

  $$\sigma \to X, \quad \sigma \to Y, \quad X \to aXb, \quad X \to ab, \quad Y \to aYc, \quad Y \to ac$$

  is not $LL(k)$ for any $k \geq 0$.

Definition A language is said to be $LL(k)$ if it is generated by an $LL(k)$ grammar. The family of $LL(k)$ languages is denoted by $\mathscr{LL}(k)$.

It is easy to see that an $LL(0)$ contains no more than one word. Furthermore, it can be shown that

$$\mathscr{LL}(0) \subset \mathscr{LL}(1) \subset \cdots \subset \mathscr{LL}(k) \subset \cdots,$$

and the inclusion is strict. If $\mathscr{LL}(k)'$ denotes the family of λ-free $LL(k)$ languages, then

$$\mathscr{LL}(0)' = \mathscr{LL}(1)' = \cdots = \mathscr{LL}(k)' = \cdots.$$

Given two $LL(k)$ grammars G_1 and G_2, there is an algorithm that decides if $L(G_1) = L(G_2)$.

References

[1] A. Salomaa, *Formal Languages*, Academic Press, New York (1973).