$LL(*)$ Parsing strategy

The Foundation of the ANTLR Parser Generator

Jan Tuma xtumaj02
Karol Troska xtrosk00

November 7, 2015

Non-deterministic top-down parsers are not efficient. They must find first derivation to proceed. Deterministic top-down parsers are faster, because ambiguity is no longer a problem. $LL(1)$ parsers are quite efficient, but there are occasionally situations when $LL(1)$ parsers are not useful and it is better to look ahead k symbols with $k > 1$. Here it comes to construction of $LL(k)$ parsers that brings up new problems and can cause to unexpected parse-time behavior which introduces many potecional errors.

The goal of this presentation is to describe $LL(*)$ parsing strategy and associated grammar analysis algorithm that constructs $LL(*)$ parsing decisions from ANTLR grammars.

In first part, we will focus $LL(*)$ parsing by explaining how it works for two ANTLR grammar fragments constructed to illustrate the algorithm. We will formally define predicate grammar, to desrice $LL(*)$ parsing precisely. Next part will focus on $LL(*)$ parsers and $LL(*)$ grammar analysis. We will takl about existing parsers and we will try explain how $LL(*)$ parsing works. Finally, we will takl about efficiency of $LL(*)$ parsing.