ABSTRAK

ANALISIS SINTAKSIS: KONSEP DAN IMPLEMENTASINYA

Anastasia Rita Widiarti
Universitas Sanata Dharma
Yogyakarta

Analisis sintaksis adalah salah satu tahap di dalam proses kompilasi yang sangat penting, yaitu suatu proses untuk menganalisis sederetan *token* yang dihasilkan oleh suatu penganalisis leksikal guna menentukan apakah pola deretan *token* tersebut sesuai dengan aturan sintaksis yang telah ditentukan dalam tatabahasa yang digunakan.

Terdapat berbagai macam metode untuk melakukan analisis sintaksis atau penguraian, yang secara garis besar dapat dibagi ke dalam dua kelompok metode, yaitu metode penguraian dari-atas-ke-bawah dan metode penguraian dari-bawah-ke-atas. Perbedaan kedua metode tersebut terletak pada pola pembentukan pohon urai, yaitu apakah dari akar ke daun, atau dari daun bergerak ke akar. Pada metode penguraian dari-atas-ke-bawah dibahas metode penguraian turun-berulang baik yang mempergunakan pelacakan mundur kembali maupun yang tidak mempergunakan pelacakan mundur kembali, pengurai prakira, dan penguraian untuk tatabahasa LL(1). Sedangkan untuk metode penguraian dari-bawah-ke-atas dibahas tiga metode penguraian yaitu penguraian *shift-reduce*, penguraian operator-*precedence*, dan penguraian LR.

Dari berbagai metode yang dibahas, terdapat satu metode penguraian yang cukup sederhana dan mudah untuk dimengerti maupun untuk diimplementasikan, yaitu metode penguraian LL(1). Dalam metode ini pengurai cukup melihat isi dari suatu *first* simbol, dan bila tidak cukup dilanjutkan dengan melihat isi dari suatu *follow* simbol, maka pengurai dapat menentukan dengan tepat produksi manakah yang akan dipilih apabila terdapat berbagai macam alternatif produksi. Untuk itu metode penguraian LL(1) dipilih untuk dijadikan dasar implementasi suatu program yang dapat melakukan proses analisis sintaksis.

ABSTRACT

SYNTAX ANALYSIS: CONCEPT AND IMPLEMENTATION

Anastasia Rita Widiarti

Universitas Sanata Dharma

Yogyakarta

Syntax analysis is an important stage in compilation process, i.e. a process to analyze a set of tokens produced by a lexical analysis in order to determine whether the pattern of the set of tokens matches with the syntax rule that has been set up in the applied grammar.

There are many methods for syntax analysis or parsing. In general, these methods can be categorized into two: (1) top-down parsing and (2) bottom-up parsing. The difference between these two lies in the mode for developing the parser tree, i.e. whether from root to leaves, or from leaves to root. Top-down parsing includes recursive descent parsing, either using backtracking or without using backtracking, predictive parser, and grammar parsing LL(1). Bottom-up parsing consists of three parsing methods, namely shift-reduce, operator-precedence, and LR parsing.

From the many methods discussed, LL(1) parsing is quite simple and easy to understand as well as to implement. In LL(1) parsing, the "parser" only needs to see the content of a first symbol; if it's not sufficient, it will continue to analyze the content of a follow symbol, and thus the parser can correctly determine which production to be chosen among some various production alternatives, if applicable. Therefore LL(1) parsing is chosen to be the basis of a program implementation that is able to perform a process of syntax analysis.