PEG Grammars and Packrat Parsing

Stanislav Bělehrádek[1] and Vojtěch Dvořáček[2]

[1]*MIS, FIT VUT Brno*
xbeleh04@stud.fit.vutbr.cz
[2]*MIN, FIT VUT Brno*
xdvora0y@stud.fit.vutbr.cz

ABSTRACT

Parsing expression grammars present novel approach for machine language modeling. It is inspired by Top-Down Parsing Language (TDPL). Traditional generative grammars from Chomsky hierarchy, originaly designed for natural language processing, brings difficulties in machine language processing. Above all, they are not generally deterministic, which is required property in computer science. PEGs on the other hand, tackle ambiguity by introducing priority, rather than alternative choice, into grammar rules. This property leads to ambiguity elimination.

Furtheremore, PEGs introduce regular expression-like syntax. This allows us to easily specify lexemes together with recursive language structure and create unified language definition. With such definition, there is no need for separate lexical analysis.[1]

Having such grammar, there is a need for appropriate parsing method as well. Here is where packrat parsing algorithm comes in. It is closely related to traditional recursive descent parsing, extended by backtracking approach. This method tries to match input patern according to given rule priorities and backtracks when rule fails. Unfortunately, parsing complex languages with such method can lead to exponential computiation time. Packrat parsing uses memoization[1] techniques to reduce time complexity caused by backtracking. That is why it is straightforward to implement such parser in lazy functional language. Such approach can finally lead to linear time parsing. [2]

Optimization technique based on storing results of expensive function calls.

References

[1] Ford B., *Parsing Expression Grammars: A Recognition-based syntactic Foundation* Proceedings of the 31st ACM SIGPLAN-SIGACT symposium on Principles of programming languages, Venice, Italy (2004).

[2] Ford B., *Packrat Parsing: Simple, Powerful, Lazy, Linear Time, Functional Pearl* Proceedings of the Seventh ACM SIGPLAN International Conference on Functional Programming, Pittsburgh, PA, USA (2002).