PARALLEL NON-DETERMINISTIC BOTTOM-UP PARSING

Bernard Lang

Center for Research in Computing Technology
Harvard University

ABSTRACT

The development of translator writing systems and extensible languages has led to a simultaneous development of more efficient and general syntax analyzers, usually for context-free (CF) syntax. Our paper describes a type of parser that can be used with reasonable efficiency for any CF grammar, even one which is ambiguous.

Our parser, like most others, is based on the pushdown automaton model, and thus its generality requires it to be non-deterministic (in the automata theoretic sense). An actual implementation of a non-deterministic automaton requires that we explore every computational path that could be followed by the theoretical automaton. This can be done serially [5], following successively every computational path whenever a choice occurs. It leads to the algorithms described in [1], whose time bounds can be exponential functions of the length of the string to be parsed.

We can also use a parallel implementation which consists in following "simultaneously" all the possible computational paths whenever a non-deterministic choice occurs. Then we can merge paths that have ceased to be different after a certain point. Those mergings reduce drastically the amount of computation. The best known example is the top-down algorithm described in [2]. The algorithm we describe in the first part of our paper is a basic bottom-up parser, similar to [2] in its organization. Both parsers can be shown to work within time bounds which are at most the cube of the length of the input string, and are often a linear function of it. The space bounds are at most the square of that length.

The second part of our paper deals with the optimization of the basic parallel bottom-up algorithm, using the properties of weak precedence relations [3]. The various optimization techniques further reduce the amount of computation required and cause frequent occurrence of a "sparse determinism" phenomenon which allows determination of part of the parse-tree before the non-deterministic analysis is ended. These optimizations also considerably lessen the space requirements. Full details can be found in [6].

The parser is easy to generate for any CF grammar, requiring only the computation of precedence tables. It is slower than most deterministic parsers (on the order of ten times); but all examples

we have tried showed it to be more efficient than any other parser having the same generality. We consider that the inefficiency of our parser is reasonable as we intend it as a research tool for the language designer rather than as a part of an industrial compiler.

Language designers, or extensible language users, are often more preoccupied with semantics than syntax and fix the syntax of their language in a "nice" form only when they have found what kind of semantics it is to represent. So they can easily come up with ambiguous syntax [4]. Our parser parses ambiguous languages and detects ambiguities in programs, thus enabling the programmer to eliminate them.

REFERENCES

[1] GRIFFITHS, T.V. & PETRICK, S.R. "On the relative efficiencies of context-free grammar recognizers", CACM 8,5, May 1965, pp. 289-300.

[2] EARLEY, J. "An efficient context-free parsing algorithm", CACM 13, 2, Feb. 1970, pp. 94-102.

[3] ICHBIAH, J.D. & MORSE, S.P. "A technique for generating almost optimal Floyd-Evans productions for precedence grammars", CACM 13,8, Aug. 1970, pp.501-508.

[4] IRONS, E.T. "Experience with an extensible language", CACM 13,1, Jan. 1970, pp. 31-40.

[5] FLOYD, R.W. "Non-deterministic algorithms", JACM 14,1, Oct. 1967 pp. 636-644.

[6] LANG, B. "Parallel non-deterministic bottom-up parsing", Center for Research in Computing Technology, Harvard University, Cambridge, Mass., Aug. 1971.