

01 Feb 1986

Lexical Analysis Using Table Look-up

Brian Keith Smith


Missouri University of Science and Technology, smithbrian@mst.edu

Dominic Soda

George Winston Zobrist

Missouri University of Science and Technology

Follow this and additional works at: https://scholarsmine.mst.edu/engman_syseng_facwork

 Part of the [Computer Sciences Commons](#), and the [Operations Research, Systems Engineering and Industrial Engineering Commons](#)

Recommended Citation

B. K. Smith et al., "Lexical Analysis Using Table Look-up," *Proceedings of the 1986 ACM 14th Annual Conference on Computer Science, CSC 1986*, p. 488, Association for Computing Machinery, Feb 1986. The definitive version is available at <https://doi.org/10.1145/324634.325317>

This Article - Conference proceedings is brought to you for free and open access by Scholars' Mine. It has been accepted for inclusion in Engineering Management and Systems Engineering Faculty Research & Creative Works by an authorized administrator of Scholars' Mine. This work is protected by U. S. Copyright Law. Unauthorized use including reproduction for redistribution requires the permission of the copyright holder. For more information, please contact scholarsmine@mst.edu.



Lexical Analysis Using Table Look-Up

Brian Smith
Dominic Soda
George W. Zobrist

Department of Computer Science
University of Missouri-Rolla
Rolla, Missouri 65401

Lexical analysis [Aho 77] is concerned with scanning the input for significant clusters of characters called tokens. The input stream of characters is analyzed by using delimiters and a careful description of the way various types of tokens may be constructed. There are circumstances in which this approach is not sufficiently flexible. It may not be convenient to define the nature of tokens by giving a set of rules of construction.

Such a circumstance arises during the process of building a functional model of a microprocessor system [Marsh 85]. The first step in this process is the lexical analysis of the assembler output of the microcode input of the microprocessor system. In this case the nature of the tokens to be isolated depend on the microprocessor being used and the assembler used to code the system description.

The need to develop a lexical analyzer which allows each of these two components to vary with minimal reprogramming has instigated the development of a lexical analyzer which exploits table lookup to provide the necessary flexibility.

The idea is that the information on the microprocessor and assembler is stored in a set of tables; the opcode table, the blockmarker table, the directive table, etc.. As the lexical scan proceeds these tables provide the intelligence to carry the scan forward or to require the scanner to backtrack. Each of these tables is implemented using dynamic storage allocation so that they may be modified at run time.

The number and nature of the tables needed is microprocessor/assembler dependent. At present a set of tables rich enough to handle several microprocessors is being used. Whether this information together with a microprocessor-dependent table precedence scheme can be provided as input to the scanner is currently being examined [Soda 86].

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.

The lexical analyzer currently in operation outputs tokens for four different statement categories; comments, control section entry/exit points, directives, and executable statements. All other statements are passed over as 'garbage' statements. The tokens output form a data base which allows a program which translates microcode into a high level language to operate on a more conceptual level [Marsh 85].

References

1. Principles of Compiler Design, A. Aho and J. Ullman, Addison-Wesley, Reading, Mass (1977).
2. "A Functional Mapping for Microprocessor System Simulation", W. K. Marshall, G. Zobrist, W. Bach, and A. Richardson, 1985 IEEE Microprocessor Forum, Atlantic City, NJ, April 1985.
3. "Lexical Analysis Using Generalized Table Look-up", D. Soda and G. Zobrist (in preparation), 1986.

Acknowledgements

Partial support for this paper was from IBM, Lexington, KY., Grant Number 459-613.