Lecture Notes in Computer Science

Edited by G. Goos and J. Hartmanis

94

Semantics-Directed Compiler Generation

Proceedings of a Workshop
Aarhus, Denmark, January 14–18, 1980

Edited by Neil D. Jones

Springer-Verlag
Berlin Heidelberg New York 1980

Editor

Neil D. Jones
Datalogisk Afdeling
Matematisk Institut
Aarhus Universitet
8000 Aarhus C
Denmark

AMS Subject Classifications (1970): 68B10
CR Subject Classifications (1974): 4.12, 5.24

ISBN 3-540-10250-7 Springer-Verlag Berlin Heidelberg New York
ISBN 0-387-10250-7 Springer-Verlag New York Heidelberg Berlin

Printing and binding: Beltz Offsetdruck, Hemsbach/Bergstr.
2145/3140-543210

INTRODUCTION

For several reasons it is becoming more and more common to provide formal definitions of the semantics of new programming languages, using techniques such as denotational semantics, attribute and affix grammars, algebraic semantics, operational definitions and axiomatic definitions. The construction of such a definition tends to expose ambiguities and unexpected implications of proposed language features, and thus can be a significant aid when designing a new language. A precise definition of the "meaning" of a program is of course essential when implementing a compiler or other language processor, and such definitions have in some cases guided the development of implementations. Further, the existence of formal definitions of source and object languages makes it possible to formally prove the correctness of compilers.

A number of translator-writing systems have been devised to systematize and simplify the task of compiler construction. These are usually syntax-directed, and provide in addition to parsing some means of manipulating symbol tables, parse tree attributes, etc. Still, compiler writing is at present largely handcraft - construction of such a large and complex piece of software requires considerable creativity, is quite prone to errors, and involves an enormous amount of work.

Correctness proofs for real compilers seem to be more of an ideal than a reality at this time, since construction of a correctness proof seems to require even more creativity and labor than construction of the compiler itself.

Clearly both problems would be alleviated if there were a closer connection between the semantic definition of the language and the structure of its compiler, just as parsing problems were much simplified after a firm connection was made between syntax definition and parser structure.

An ideal solution would be a true compiler generator, which if given definitions of the syntax and semantics of a programming language would automatically produce a compiler of acceptable compile-time and run-time efficiency. The purpose of the workshop was to bring together researchers whose work brings us closer to this goal.

The papers presented at the workshop naturally fall into four categories. The first group contains three papers with a common goal: to produce a compiler from the denotational semantics of a programming language. The second group is concerned with the use of abstract algebra to define semantics, to specify compilers and to prove them correct. The third group has to do with several aspects of attribute or affix grammars. These are a powerful and natural medium for expressing compilers, and thus provide a promising output language for compiler generators. The last group contains three papers which are related to compiler generation but not in the earlier categories, including one on the formal semantic definition of the ADA programming language. The definition is intended to serve

"for the validation of implementations and as a guideline for implementors ... (and) as an input for a compiler generator when the technology becomes available." As such it is likely to stimulate further research in automating the compiler generation process.

The "Workshop on Semantics-Directed Compiler Generation" was held January 14-18 at the University of Aarhus, Denmark. The meeting was made possible by grants from the Danish Research Council (Forskningsråd) and the Aarhus University Computer Science Department (Datalogisk Afdeling). Local arrangements were handled by an organizing committee consisting of Neil Jones, Peter Mosses and Mogens Nielsen and by the workshop secretary, Karen Møller. The department deserves a round of thanks for providing (in addition to funds) the use of its reproduction, secretarial and library facilities, and for providing an excellent milieu for work and discussion. The contributions of a number of individuals are warmly acknowledged, including Karen Møller, Mogens Nielsen, Lene Rold and of course the participants in the workshop, without whose professional expertise the workshop would not have been possible.

CONTENTS