INCREASING MODULARITY AND LANGUAGE-INDEPENDENCY IN AUTOMATICALLY GENERATED COMPILERS*

Harald GANZINGER


Communicated by M. Paul
Received June 1983

1. Introduction

The aim of this paper is to introduce a method for obtaining modular compiler descriptions that, paraphrasing [26],
- exhibit a semantic processing based on fundamental concepts of languages and compiling;
- are easily modifiable and adaptable to different but related languages;
- are combinations of language-independent modules;
- are subject to automatic compiler generation.

1.1. Related work

The work reported here is based on ideas from (modular) algebraic specifications of abstract data types [31, 2, 6, 23], abstract semantic algebras [25, 26], and compiler descriptions based on attribute grammars [22].

Many papers have utilized ideas of abstract data type theory to improve the structure of semantics definitions and/or compiler descriptions. In [4], following [10] and [24], the fundamental algebraic structure of denotational semantics definitions and syntax-oriented compiler descriptions has been recognized. Following [30], in [7], [16], and [20] the use of abstract data types has been suggested. In particular the latter paper was concerned with structuring compiler definitions hierarchically, using the specification language OBJ [19]. In addition, many authors of denotational descriptions have tried to impose structure on their descriptions. In particular [29] and [30] proposed general language independent combinators, abbreviating pieces of \( \lambda \)-notation. Algebraic descriptions of compilers in the context of automatic compiler generation are considered in [16] and [8]. Modularization of compiler descriptions is not investigated in these papers.

* This work was in part supported by the Sonderforschungsbereich 49, Programmiertechnik, at the Technical University of Munich.