4.1 PROCESSORS

4.12 Compilers and Generators

CUNIN, P. Y.; GRIFFITHS, M. (IUCAL, Vandoeuvre, France) ; AND VOIRON, J. (IMAG, Grenoble, France) 38,780

Comprende la compilation. (*Understanding compilation*) Springer-Verlag New York, Inc., New York, 1980, 267 pp., $29, ISBN 0-387-10327-9.

Textbooks about compilation are not numerous, and probably none is above criticism. Gries' work [1], although ten years old, is still popular, because of its simple and pragmatic approach, but its programming style and theoretical foundations are now obsolete. The second book by Aho and Ullman [2], although less theoretically oriented than their first one [3], is much more up to date, but it gives too much emphasis to parsing, and especially to LR methods, while almost forgetting meta-compilers. Cocke and Schwartz [4] provide useful information about global optimization, and we think it regrettable that their book has not been published. However, they stop short of object code production, like Lewis, Rosenkrantz, and Stearns [5]. And for the French-speaking teacher or student, there exists almost nothing usable in his native language. Consequently, the book by Cunin, Griffiths, and Voiron may fill an important gap about an important matter.

The book comprises 22 chapters distributed into six parts:

1) Introduction (21 pages): structure and properties of algorithmic languages, and fundamental concepts of compilation.

2) Analysis (59 pages): scanning and parsing, with the necessary notions of language theory. The relative brevity of this part is significant of the authors' will to devote to this aspect no more than is required by its relative importance in the compilation process.

3) Program Meaning (36 pages): semantics, name association, meaning of sentences.

4) Evaluation Environment (52 pages): run-time problems, data handling, sub-program management.

5) Code Generation (40 pages): some notions about optimization.

6) Relations between Languages and Compilers (20 pages): particular problems in PL/I, ALGOL 68, and "new languages."

Two appendices describe the fictitious target computer used in the examples, and give solutions to some exercises. There is no index, and references are sketchy, when not completely unserviceable.

Style is lucid, and treatment is practical, concrete, fit for use. All three authors are university teachers, and they prefer to be understood rather than to be exhaustive. They do not claim their short book to be a compendium, but only a primer, and they warn the reader that specialization in the matter should be acquired in more detailed books, and above all in practical experience.

The following matters seem to us especially well dealt with:

- the usefulness of understanding the compilation process, even for apparently unrelated applications, like payroll production, or reading a card deck (Chapter 4);
- the really convincing explanation of deterministic top-down parsing, and of all LL(1) techniques (Chapter 7);
- the whole fourth part, on run-time handling, which very lucidly presents matters too often hastily discussed.

On the other hand, we regret the superficiality of the discussion of "new languages" (Chapter 22), and of the presentation of optimization techniques (Chapter 20). Above all, our main criticism is the somewhat dated conception of the book. References to PASCAL are almost completely lacking, the authors evidently preferring the ALGOL 68 spirit—which is perfectly legitimate, of course.

Much more important, however, the overall structure of the book is similar to that of Gries', although in ten years much has progressed in the knowledge and practice of compilation. For example, meta-compilers are almost ignored in the book, which seems to force the reader into beginning from scratch when starting the construction of a compiler, although many useful and usable tools provide for simplification and standardization of many tasks, especially in scanning and in parsing, but also in the subsequent phases of compilation. In fact, it is realistic—and not uncommon—to build a compilation course upon the systematic use of one of the available compiler writing systems. The authors should at least have emphasized the current industrialization of compilers, by describing several modern methods, rather than only the craft one. Software engineering applies to compilation too.

With these reservations, we must note that this book fills a yawning chasm in the French-speaking publishing trade (although the publisher is German!). Moreover, its brevity, simplicity, clarity, and practicality should also appeal to anybody with at least a slight knowledge of technical French, since we know of no English textbook with the same qualities.

O. Lecarme, Nice, France; and
B. Meyer, Clamart, France

REFERENCES

[1] GRIES, D. *Compiler construction for digital computers*, John Wiley & Sons, New York, 1971.

[2] AHO, A. V.; AND ULLMAN, J. D. *Principles of compiler design*, Addison-Wesley, Reading, Mass., 1977; see *CR* **19**, 6 (June 1978), Rev. 33,085.

[3] AHO, A. V.; AND ULLMAN, J. D. *The theory of parsing, translation, and compiling*, Prentice-Hall, Englewood Cliffs, N. J., 1972; see *CR* **14**, 5 (May 1973), Rev. 24,964.

[4] COCKE, J.; AND SCHWARTZ, J. T. *Programming languages and their compilers*, Courant Institute of Mathematical Sciences, New York Univ., 1970.

[5] LEWIS, P. M.; ROSENKRANTZ, D. J.; AND STEARNS, R. E. *Compiler design theory*, Addison-Wesley, Reading, Mass., 1976; see *CR* **18**, 9 (Sept. 1977), Rev. 31,918.

4.19 Miscellaneous

See: 38,787 [Cat. 6.29].

4.2 PROGRAMMING LANGUAGES

4.22 Procedure- and Problem-Oriented Languages

See also: 38,780 [Cat. 4.12].