<<编译原理>>

图书基本信息

书名：<<编译原理>>

13位ISBN编号：9787111326748

10位ISBN编号：7111326741

出版时间：2011-1

出版时间：机械工业出版社

作者：Alfred V. Aho,Monica S. Lam,Ravi Sethi,Jeffrey D. Ullman

页数：1009

前言

In the time since the 1986 edition of this book, the world of compiler designhas changed significantly. Programming languages have evolved to present newcompilation problems. Computer architectures offer a variety of resources ofwhich the compiler designer must take advantage. Perhaps most interestingly,the venerable technology of code optimization has found use outside compilers.It is now used in tools that find bugs in software, and most importantly, findsecurity holes in existing code. And much of the "front-end" technology ——grammars, regular expressions, parsers, and syntax-directed translators —— arestill in wide use.Thus, our philosophy from previous versions of the book has not changed.We recognize that few readers will build, or even maintain, a compiler for amajor programming language. Yet the models, theory, and algorithms associ-ated with a compiler can be applied to a wide range of problems in softwaredesign and software development. We therefore emphasize problems that aremost commonly encountered in designing a language processor, regardless ofthe source language or target machine.Use of the BookIt takes at least two quarters or even two semesters to cover all or most of thematerial in this book. It is common to cover the first half in an undergraduatecourse and the second half of the book —— stressing code optimization —— ina second course at the graduate or mezzanine level. Here is an outline of thechapters:Chapter 1 contains motivational material and also presents some backgroundissues in computer architecture and programming-language principles.Chapter 2 develops a miniature compiler and introduces many of the impor-taut concepts, which are then developed in later chapters. The compiler itselfappears in the appendix.Chapter 3 covers lexical analysis, regular expressions, finite-state machines, andscanner-generator tools. This material is fundamental to text-processing of allsorts.

<<编译原理>>

内容概要

本书是编译领域无可替代的经典著作，被广大计算机专业人士誉为“龙书”。
本书上一版自1986年出版以来，被世界各地的著名高等院校和研究机构（包括美国哥伦比亚大学、斯坦福大学、哈佛大学、普林斯顿大学、贝尔实验室）作为本科生和研究生的编译原理课程的教材。
该书对我国高等计算机教育领域也产生了重大影响。

第2版对每一章都进行了全面的修订，以反映自上一版出版20多年来软件工程。
程序设计语言和计算机体系结构方面的发展对编译技术的影响。
本书全面介绍了编译器的设计，并强调编译技术在软件设计和开发中的广泛应用。
每章中都包含大量的习题和丰富的参考文献。

本书适合作为高等院校计算机专业本科生和研究生的编译原理与技术课程的教材，也可供广大计算机技术人员参考。

cd中包含如下内容：
· comptia project+考试模拟题。

· 作者的项目管理培训视频。

· 项目管理工作表和模板。

作者简介

Alfred
V．Aho，美国歌伦比亚大学教授，美国国家工程院院士，ACM和IEEE会士，曾获得IEEE的冯·诺伊曼奖。
著有多部算法、数据结构、编译器、数据库系统及计算机科学基础方面的著作。

<<编译原理>>

书籍目录

章节摘录

插图：Languagel, are used to search databases. Database queries consist of predicatescontaining relational and boolean operators. They can be interpreted or com-piled into commands to search a database for records satisfying that predicate.Compiled SimulationSimulation is a general technique used in many scientific and engineering disci-plines to understand a phenomenon or to validate a design. Inputs to a simula-tor usually include the description of the design and specific input parametersfor that particular simulation run. Simulations can be very expensive. We typi-cally need to simulate many possible design alternatives on many different inputsets, and each experiment may take days to complete on a high-performancemachine. Instead of writing a simulator that interprets the design, it is fasterto compile the design to produce machine code that simulates that particulardesign natively. Compiled simulation can run orders of magnitude faster thanan interpreter-based approach. Compiled simulation is used in many state-of-the-art tools that simulate designs written in Verilog or VHDL.1.5.5 Software Productivity ToolsPrograms are arguably the most complicated engineering artifacts ever pro-duced; they consist of many many details, every one of which must be correctbefore the program will work completely. As a result, errors are rampant inprograms; errors may crash a system, produce wrong results, render a systemvulnerable to security attacks, or even lead to catastrophic failures in criticalsystems. Testing is the primary technique for locating errors in programs.An interesting and promising complementary approach is to use data-flowanalysis to locate errors statically （that is, before the program is run）. Data-flow analysis can find errors along all the possible execution paths, and notjust those exercised by the input data sets, as in the case of program testing.Many of the data-flow-analysis techniques, originally developed for compileroptimizations, can be used to create tools that assist programmers in theirsoftware engineering tasks.The problem of finding all program'errors is undecidable. A data-flow anal-ysis may be designed to warn the programmers of all possible statements witha particular category of errors. But if most of these warnings are false alarms,users will not use the tool. Thus, practical error detectors are often neithersound nor complete. That is, they may not find all the errors in the program,and not all errors reported are guaranteed to be real errors. Nonetheless, var-ious static analyses have been developed and shown to be effective in findingerrors, such as dereferencing null or freed pointers, in real programs. The factthat error detectors may be unsound makes them significantly different fromcompiler optimizations. Optimizers must be conservative and cannot alter thesemantics of the program under any circumstances.

<<编译原理>>

编辑推荐

《编译原理(英文版·第2版)》：经典原版书库

<<编译原理>>

版权说明