<<编译器设计之路>>

图书基本信息

书名：<<编译器设计之路>>

13位ISBN编号：9787111321644

10位ISBN编号：7111321642

出版时间：2011-1

出版时间：机械工业出版社

作者：裘巍

页数：449

前言

与其他自然科学相比，计算机科学的发展历史并不久远，是较新的学科体系，尚有许多未知的领域有待探索。
因此，本专业学生或工程技术人员仅仅满足于学习或应用几门程序设计语言是远远不够的。
一些看似抽象的课程才是提高专业人员“内功”修为的秘技，例如数据结构、操作系统、编译原理、计算机系统结构、计算机网络等。
不过，经典课程的学习并不是一蹴而就的，如何学习与理解课程的精髓是值得关注的。
本书将从编译器设计的角度，为读者揭示编译原理的精髓。
学习编译技术的意义有人认为，编译技术似乎已经相当成熟了，继续深入研究是没有任何意义的。
实际上，任何科学技术都是发展变化的。
表面上看，编译器设计的高层问题似乎已经形成了完美的体系，但当我们深入其内核就会发现事实并非如此。
现代编译器设计面临的挑战是来自目标计算机系统结构、新颖程序设计语言及本身的计算资源等多方面的。
其中，任何一方面的因素都足以颠覆某些传统理论与算法。
例如，在现代编译器设计中，计算资源的增加允许设计者采用更耗费时间、空间的算法，而不必过分关注其代价。
对于优化算法设计者而言，这是令人兴奋的。
为了追求目标代码的更优，即使设计一些时空复杂度稍高的算法也是可以接受的。
当然，从更高的层次上讲，学习编译器设计的目的还不仅仅局限于其本身的理论与技术。
作为一个系统软件的设计学科，其解决问题的思路与方法更是值得读者细细品味的。
这可能是一个漫长而艰辛的历程，不过，这才是经典学科的魅力所在。
以品味经典为目的来学习与研究操作系统、数据库技术、计算机网络、编译技术等学科是笔者多年来的努力方向。
从20世纪50年代中期以来，编译器设计就一直是计算机科学界的一个重要研究领域。
Fortran语言之父John Backus认为，除非编译器生成的代码与手工编写的机器代码的性能非常接近，否则程序员就不会放弃汇编语言程序设计的思想与方法，这就是编译技术研究的源动力。
从学术研究的角度讲，众所周知，图灵奖被誉为“计算机界的诺贝尔奖”，自1966年设立至今，54位获奖者中就有16位是由于程序设计语言及编译技术的研究成果而获此殊荣的。
编译技术在计算机科学领域的地位由此可见一斑。

<<编译器设计之路>>

内容概要

本书系统地介绍了一个实际的Pascal编译器Neo Pascal的设计与实现。
结合Neo Pascal的源代码，详细讲述了LL(1)语法分析器、符号表系统、中间表示、类型系统、优化技术、运行时刻的存储管理、代码生成器等编译器设计的核心话题。
各章都附有少量以实践应用为主的练习题，既可作为阅读思考题，也可作为课程设计选题。
　　与国内其他介绍编译技术的图书相比，本书更关注的是编译器的实现细节，而不仅仅局限于理论阐述。
本书可供从事编译器设计相关工作的工程人员阅读，也可作为高等院校计算机专业的编译原理课程参考书。
　　读者可在http://neopascal.sourceforge.net获得Neo Pascal的源代码及相关文档。

<<编译器设计之路>>

书籍目录

出版说明前言第1章 概述 1.1 编译技术概述 1.1.1 程序设计语言基础 1.1.2 程序设计语言的翻译机制 1.1.3 编译器的基本结构 1.2 Pascal语言基础 1.2.1 Pascal语言简介 1.2.2 Pascal程基本组成 1.2.3 Pascal的声明部分 1.2.4 Pascal的类型 1.2.5 Pascal的运算符 1.2.6 Pascal的语句 1.3 开发环境与Delphi基础 1.3.1 开发环境与文件列表 1.3.2 Delphi基础 1.4 深入学习 1.5 实践与思考 1.6 大师风采--Niklaus Wirth 第2章 词法分析 2.1 词法分析概述 2.1.1 词法分析的任务 2.1.2 单词的分类 2.2 词法分析器的设计 2.2.1 识别单词 2.2.2 转换图构造词法分析器 2.3 词法分析器的实现 2.3.1 词法定义 2.3.2 构造转换图与转换表 2.3.3 关数据结构 2.3.4 源代码实现 2.4 深入学习 2.5 实践与思考 2.6 大师风采--Dennis M. Ritchie 章 语法分析 3.1 程序设计语言的语法描述 3.1.1 上下文无关文法 3.1.2 推导 3.1.3 语法树 3.1.4 归约简介 3.2 语法分析概述 3.2.1 语法分析的任务 3.2.2 自上而下的语法分析法 3.2.3 构造语法分析器 3.3 语法分析器的实现 ……第4章 符号表系统 第5章 中间表示 第6章 表达式语义 第7章 优化技术 第8章 运行时刻的存储管理 第9章 目标代码生成 第10章 GCC内核与现代编 参考文献

章节摘录

插图：迄今为止，程序设计语言仍是人类与计算机交流的主要途径，它的应用领域也仅限于操纵与控制计算机。
从第一台计算机诞生之日起，人类就始终在探索一种有效的方式与计算机进行对话交流，使之能为人类服务。
虽然时隔数十年，计算机能识别与处理的语言仍然是二进制形式的机器语言描述的源程序。
当然，不可否认二进制机器语言的优点非常多，但机器语言的易用性差也是不可回避的。
即使是计算机专家想直接使用机器语言与计算机进行交流也是非常困难的。
在20世纪50年代，计算机科学家们就已经意识到必须解决这一棘手的问题，否则计算机将无法得到普及。
经过多年努力，汇编语言、C、Pascal等程序设计语言终于横空出世。
根据语言的形式与特点，习惯上，将机器语言与汇编语言（一种比较接近机器语言的程序设计语言）称为低级语言，将其余的C、Pascal之类的语言称为高级语言。
读者必须注意，低级语言与高级语言之分并不是说明语言本身的优劣，仅仅是说明语言的形式与机器语言的相似程度。
所谓低级语言指的是与机器语言比较类似的语言，而高级语言指的是与机器语言差别较大而与自然语言比较类似的语言。
由于这些非机器语言的诞生，也就出现了将非机器语言等价翻译成机器语言的需求。
显然，将非机器语言翻译成机器语言的工作不能由手工完成，否则，非机器语言的产生就没有任何意义了。
因此，人们试图借助于一个程序工具自动完成翻译工作。
根据语言不同，翻译工具的复杂程度也不尽相同。
比如，汇编语言比较接近机器语言，所以其翻译工具较易实现。
而高级语言与机器语言差别较大，所以其翻译工具的实现也较为复杂。
习惯上，将前者称为汇编器，而将后者称为编译器。
当然，有些书上对于汇编器与编译器并没有严格区分，都将其称为编译器，反正这只是一个名词而已，读者不必深究。
编译器的源语言是一种较为高级的程序设计语言，而目标语言可以是汇编语言、机器语言或者另一种高级语言。
笔者必须澄清～点，人们普遍认为编译器的目标语言就是低级语言，这个观点的确没有错，但并不完整。

编辑推荐

《编译器设计之路》：完整介绍一个实际编译器Neo Pascal的设计与实现，详细讲述LL(1)分析器、符号表系统、优化等核心话题。
阐述系统软件的设计观点探讨现代编译技术热点提供编译器全部源代码

版权说明