21世纪高等学校计算机科学与技术规划教材

编译原理与技术

刘春林　谭庆平　刘　越　编著

北京邮电大学出版社

21世纪高等学校计算机科学与技术规划教材

编译原理与技术

刘春林　谭庆平　刘　越　编著

北京邮电大学出版社
www.buptpress.com

内 容 简 介

本书介绍了编译程序设计的基本原理、技术和方法。主要内容包括：编译程序的概述，高级语言及其语法描述，词法分析，语法分析，属性文法和语法制导翻译，语义分析和中间代码产生，符号表的内容与组织，运行时存储空间组织，代码优化以及目标代码生成等。全书结构严谨、科学，内容由浅入深、循序渐进。每一章节都精选了必要的示例并配备了丰富的习题。

本书适合作为全日制普通高等学校编译原理课程的教材或教学参考书，还可供其他需学习或了解编译原理和技术的人员阅读。

图书在版编目(CIP)数据

编译原理与技术/刘春林，谭庆平，刘越编著. —北京：北京邮电大学出版社，2004

ISBN 7-5635-0841-4

Ⅰ.编… Ⅱ.①刘…②谭…③刘… Ⅲ.编译程序—程序设计 Ⅳ.TP314

中国版本图书馆 CIP 数据核字(2004)第 129751 号

书　　名	编译原理与技术
编　　著	刘春林　谭庆平　刘　越
责任编辑	陈露晓
出版发行	北京邮电大学出版社
社　　址	北京市海淀区西土城路 10 号(100876)
电话传真	010-62282185(发行部)　010-62283578(传真)
	E-mail　sanwen99@mail.edu.cn
经　　销	各地新华书店
印　　刷	国防科技大学印刷厂印刷
开　　本	787mm×960mm　1/16
印　　张	20
字　　数	341 千字
版　　次	2005 年 4 月第 1 版　2005 年 4 月第 1 次印刷

ISBN 7-5635-0841-4/TP·113　　定价　29.50 元

21世纪高等学校计算机科学与技术规划教材

编委会

序

自20世纪80年代以来，高等学校计算机教育发展迅速，计算机教育的内容不断扩展、程度不断加深。特别是近十年来，计算机向高度集成化、网络化和多媒体化发展的速度一日千里；社会信息化不断向纵深发展，各行各业的信息化进程不断加速；计算机应用技术与其他专业的教学、科研工作的结合更加紧密；各学科与以计算机技术为核心的信息技术的融合，促进了计算机学科的发展，各专业对学生的计算机应用能力也有更高和更加具体的要求。

基于近年来计算机学科的发展，以及国家教育部关于计算机基础教学改革的指导思路，我们确立了这套“21世纪高等学校计算机科学与技术规划教材”的编写思想与编写计划。教材是教学过程中的“一剧之本”，是高校计算机教学的首要问题。该套系列教材编写计划的制定凝聚了编委会和作者的心血，是大家多年来计算机学科教学和研究成果的体现，并得到了陈火旺院士的亲自指导与充分肯定。

这套系列教材由北京邮电大学出版社三文工作室精心策划和组织。编写过程中，充分考虑了计算机学科的发展与《计算机学科教学计划》中内容和模块的调整，使得整套教材更具科学性和实用性。整套系列教材体系结构按课程设置进行划分。每册教材均涵盖了相应课程教学大纲所要求的内容，既具备学科设置的合理性，又符合计算机学科发展的需要。从结构上遵循教学认知规律，基本上能够满足不同层次院校、不同教学计划的要求。

各册教材的作者均为多年来从事教学、研究的专家和学者，他们有丰富的教学实践经验，所编写的教材结构严谨、内容充实、层次清晰、概念准确、论理充分、理论联系实际、深入浅出、通俗易懂。

教材建设是一项长期艰巨的系统工程，尤其是计算机科学技术发展迅速、内容更新快，为使教材更新能跟上科学技术的发展，我们将密切关注计算机科学技术的发展新动向，以使我们的教材编写在内容上不断推陈出新、体系上不断发展完善，以适应高校计算机教学的需要。

21世纪高等学校计算机科学与技术规划教材编委会

2005年3月

前 言

编译原理是计算机专业的一门核心课程，在计算机专业教学体系中占有十分重要的地位。近年来，我国高等教育迅速发展，许多学校纷纷开设计算机本科专业，不少学校正在进行教学内容、教学方法和教学时数等方面的改革。为了适应教学发展的需求，作者结合多年来编译原理课程教学的经验，并依据中国计算机学会、全国高校计算机教育研究会制定的“计算机科学与技术教程(CCC2002)”对编译原理课程教学的基本要求编写了本书。

全书共分为十一章。第一章是编译程序的概述；第二章介绍高级语言及其语法描述；第三章介绍词法分析，主要是词法分析设计、正规式和有限自动机的内容；第四章介绍自上而下语法分析，主要是LL(1)分析以及递归下降分析法和预测分析法；第五章介绍自下而上语法分析，主要包括算符优先分析和LR分析；第六章介绍属性文法和语法制导翻译的基本概念与方法；第七章是语义分析和中间代码产生；第八章介绍符号表的内容与组织；第九章介绍运行时存储空间组织；第十章介绍代码优化；第十一章介绍目标代码生成。讲授全书需要60学时左右。

编译原理是一门兼有很强的理论性与实践性的课程。在本书的内容选取和编写过程中，作者力求合理组织、由浅入深、循序渐进。每一章节都精选了必要的示例，以帮助读者更好地理解其中的概念、原理和方法。在每章末尾给出了本章小结并配备了丰富的习题。

本书适合作为全日制普通高等学校编译原理课程的教材或教学参考书，还可供其他需学习或了解编译原理和技术的人员阅读。

本书由国防科技大学刘春林编写第一、二、四、五、六、七、十章以及第十一章的部分内容；谭庆平编写第三章；刘越编写第八、九章以及第十一章的部分内容；由刘春林最后统稿。由于编者水平有限，书中难免存在疏漏和错误，敬请批评指正。

编 者

2005年1月

目　录

第一章 引 论

1.1 编译程序概述

自20世纪40年代第一台电子计算机诞生以来，计算机应用得到了迅速的发展，已从根本上改变了人们的工作与生活。计算机之所以能得到迅速的发展，高级程序语言及其编译技术的发展起着极其重要的作用。可以说，没有高级语言，计算机就难以如此广泛地被应用；而没有编译程序，高级语言就无法使用。

程序语言在计算机科学中占有特殊的地位，它是人们描述计算过程的工具和手段。在计算机发展初期，人们只能用机器语言编写程序，这种用0和1表示的机器语言很不直观、难以理解、容易出错、可移植性很差，程序设计人员必须受过一定的训练并且熟悉计算机硬件，这就大大限制了计算机的推广应用。后来出现了汇编语言，它用直观的助记符代替操作指令，使得程序的可读性有一定的提高。但汇编语言仍然是依赖于机器的，使用起来还很不方便，并且程序设计的效率也很低。为进一步解决这些问题，20世纪50年代中期人们参照数学语言设计了第一个高级语言——Fortran语言，之后，又相继出现了许许多多应用于各个领域的高级语言，如Algol，COBOL，Pascal，C，Ada、Java等。这类高级语言完全摆脱了机器指令形式的约束，用它编写的程序更接近自然语言和习惯上对算法的描述。高级语言相对于机器语言和汇编语言这样的低级语言，大大提高了程序的可读性、可写性、可移植性和可维护性；缩短了程序的开发周期，降低了程序的开发和维护费用。正是由于高级语言的出现和发展，极大地推动了计算机应用的迅速发展。

但是，通常的情况下，用高级语言编写的程序无法直接在计算机上执行，在计算机上执行一个高级语言程序一般要分为两步：第一步，用一个编译程序把高级语言翻译成机器语言程序；第二步，运行所得的机器语言程序求得计算结果。

通常所说的翻译程序(Translator)是指这样的一个程序,它能够把某一种语言程序(称为源语言程序)转换成另一种语言程序(称为目标语言程序),而后者与前者在逻辑上是等价的。如果源语言是高级语言,而目标语言是低级语言,这样的一个翻译程序就称为编译程序(Compiler)。高级语言程序的编译和执行过程如图 1-1 所示。

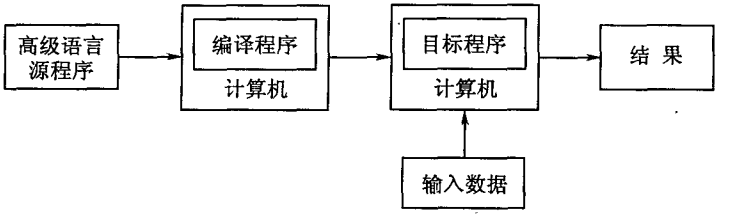

图 1-1 高级语言程序的编译和执行

高级语言程序除了像上面所说的先编译后执行外,有时也可"解释"执行。一个源语言的解释程序(Interpretor)是这样的程序,它以该语言写的源程序作为输入,但不产生目标程序,而是边解释边执行源程序本身。高级语言源程序的解释执行过程如图 1-2 所示。

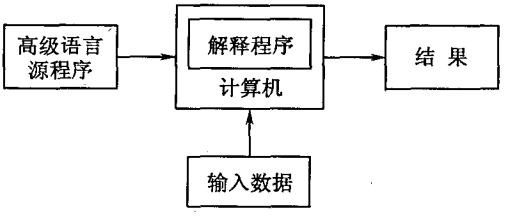

图 1-2 高级语言程序的解释执行

采用编译方式执行高级语言程序的优点是执行效率高,通过一次性的编译,产生高效的目标程序,之后便可反复运行而不必重新编译;不足是查错和调试较为困难。采用解释方式执行高级语言程序则相反,其优点是易于查错和调试,如果源程序中存在错误,解释程序可以确定出错误语句的位置,而且允许用户在程序执行过程中修改程序;它的不足是执行效率较低,因为每次执行,解释程序都对逐个语句进行分析和翻译。本书将不对解释程序作专门的讨论。实际上,许多编译程序的构造与实现技术同样适用于解释程序。

根据不同的用途和侧重,编译程序还可进一步分类。专门用于帮助程序开发和调试的编译程序称为诊断编译程序(Diagnostic Compiler),着重于提高目

标代码效率的编译程序叫优化编译程序(Optimizing Compiler)。现在很多编译程序同时提供了调试、优化等多种功能,用户可以通过"开关"进行选择。运行编译程序的计算机称宿主机,运行编译程序所产生目标代码的计算机称目标机。如果一个编译程序产生不同于其宿主机的机器代码,则称它为交叉编译程序(Cross Compiler)。如果不需重写编译程序中与机器无关的部分就能改变目标机,则称该编译程序为可变目标编译程序(Retargetable Compiler)。

世界上第一个编译程序——FORTRAN 语言编译程序是 20 世纪 50 年代中期研制成功的。当时,人们普遍认为设计和实现编译程序是一件十分困难、令人生畏的事情。经过数十年的努力,编译理论与技术得到迅速发展,现在已形成了一套比较成熟的、系统化的理论与方法,并且开发出了一些实用的编译程序的实现语言、环境与工具。因此,设计并实现一个编译程序不再是高不可攀的事情。

1.2 编译过程及编译程序结构

1.2.1 编译过程

编译程序的功能是把高级语言源程序翻译成等价的低级语言目标程序,从输入源程序开始到输出目标程序为止的整个过程,是非常复杂的。但就其过程而言,它与人们进行自然语言之间的翻译有许多相近之处。当我们把一种文字翻译为另一种文字,例如,把一段英文翻译为中文时,通常需经下列步骤:

(1) 识别出句子中的一个个单词;

(2) 分析句子的语法结构;

(3) 根据句子的含义进行初步翻译;

(4) 对译文进行修饰;

(5) 写出最后的译文。

类似地,编译程序的工作过程一般也可以划分为五个阶段:词法分析、语法分析、语义分析及中间代码生成、优化、目标代码生成。

1. 词法分析

词法分析的任务是:输入源程序,对构成源程序的字符串进行扫描和分解,识别出一个个的单词符号(或简称符号),如基本字、标识符、常数、算符和界符。

例如，对于 Pascal 的循环语句

while i<1 do

词法分析的结果是识别出如下的单词符号：

基本字	while
标识符	i
关系运算副符	<
整常数	1
基本字	do

这些单词是组成上述 Pascal 语句的基本符号。单词符号是语言的基本组成成分，是人们理解和编写程序的基本要素。识别和理解这些要素无疑也是翻译的基础。如同将英文翻译成中文的情形一样，如果你对英语单词不理解，那就谈不上进行正确的翻译。在词法分析阶段的工作中所依循的是语言的词法规则（或称构词规则）。描述词法规则的有效工具是正规式和有限自动机。

2. 语法分析

语法分析的任务是：在词法分析的基础上，根据语言的语法规则，把单词符号串分解成各类语法单位（语法范畴），如“表达式”、“语句”、“程序段”和“程序”等。通过语法分析，确定整个输入串是否构成语法上正确的“程序”。例如，在很多语言中，符号串

X:=3.14*R*R

代表一个“赋值语句”，而其中的 3.14*R*R 代表一个“算术表达式”。因而，语法分析的任务就是识别 3.14*R*R 为算术表达式，同时，识别上述整个符号串属于赋值语句这个范畴。

语法分析所依循的是语言的语法规则。语法规则通常用上下文无关文法描述。

3. 语义分析与中间代码产生

这一阶段的任务是：对语法分析所识别出的各类语法范畴，分析其含义，并进行初步翻译（产生中间代码）。这一阶段通常包括两个方面的工作。首先，对每种语法范畴进行静态语义检查，例如，变量是否定义、类型是否正确等等。如果语义正确，则进行另一方面工作，即进行中间代码的翻译。例如，下面语句

Y:=a*a-4*b*c

可被翻译为如下的所谓三地址代码：

T_1:=a*a

T_2:=4*b

$T_3:=T_2*b$

$Y:=T_1+T_2$

这一阶段所依循的是语言的语义规则。通常使用属性文法描述语义规则。

"翻译"仅仅在这里才开始涉及到。所谓"中间代码"是一种含义明确、便于处理的记号系统,它通常独立于具体的硬件。这种记号系统或者与现代计算机的指令形式有某种程度的接近,或者能够比较容易地把它变换成现代计算机的机器指令。常用的中间代码有四元式、三元式、间接三元式、逆波兰记号和树形表示等等。

4. 优化

优化的任务在于对前段产生的中间代码进行加工变换,以期在最后阶段能产生出更为高效(省时间和空间)的目标代码。优化的主要方面有:公共子表达式的提取、循环优化、删除无用代码等等。有时,为了便于"并行运算",还可以对代码进行并行化处理。优化所依循的原则是程序的等价变换规则。

5. 目标代码生成

这一阶段的任务是:把中间代码(或经优化处理之后)变换成特定机器上的低级语言代码。这阶段实现了最后的翻译,它的工作有赖于硬件系统结构和机器指令含义。这阶段工作非常复杂,涉及到硬件系统功能部件的运用,机器指令的选择,各种数据类型变量的存储空间分配,以及寄存器和后缓寄存器的调度,等等。如何产生出足以充分发挥硬件效率的目标代码是一件非常不容易的事情。

目标代码的形式可以是绝对指令代码或可重定位的指令代码或汇编指令代码。如目标代码是绝对指令代码,则这种目标代码可立即执行。如果目标代码是汇编指令代码,则需汇编器汇编之后才能运行。必须指出,现代多数实用编译程序所产生的目标代码都是一种可重定位的指令代码。这种目标代码在运行前必须借助于一个连接装配程序把各个目标模块(包括系统提供的库模块)连接在一起,确定程序变量(或常数)在主存中的位置,装入内存中指定的起始地址,使之成为一个可以运行的绝对指令代码程序。

上述编译过程的5个阶段是一种典型的分法。事实上,并非所有编译程序都分成这5个阶段。有些编译程序对优化没有什么要求,优化阶段就可省去。在某些情况下,为了加快编译速度,中间代码生成阶段也可以去掉。有些最简单的编译程序是在语法分析的同时产生目标代码。但是,多数实用编译程序的工作过程大致都像上面所说的那5个阶段。

上述编译过程的5个阶段是编译程序工作时的动态特征。编译程序的结构