中国高等学校计算机科学与技术专业（应用型）规划教材

丛书主编 陈明

编译原理

毛红梅 严云洋 主 编
程 勇 王廷蔚 副主编

清华大学出版社

中国高等学校计算机科学与技术专业（应用型）规划教材

丛书主编 陈明

编译原理

毛红梅 严云洋 主 编
程 勇 王廷蔚 副主编

清华大学出版社
北京

内 容 简 介

本书是根据高等院校计算机专业的培养目标和基本要求，结合作者多年的教学和应用实践经验编写而成的一本编译原理与技术的教材。本书系统全面地介绍了编译原理的相关知识点，共分为11章。主要内容包括编译简介、文法和语言、上下文无关语言、词法分析、语法分析——自顶向下分析、自底向上分析——优先分析法、自底向上分析——LR分析法、语法制导翻译和中间代码生成、符号表与错误处理、代码优化、目标代码生成等内容。

本书在每一章的知识点讲解中，尽量将难于理解的理论用浅显易懂的语言阐述，并配合大量实例，使读者加深对理论知识的理解。章节后均有“学习加油站”板块，将每章中的重要知识点用多个实例进行讲解，帮助读者将知识点应用到实际问题中，加深对理论知识的理解，并提高实践动手能力。

本书对编译原理的理论阐述详尽、易学易用，注重理论与实践结合。本书适合作为普通高等学校计算机科学与技术等相关计算机专业的教材，也适合软件开发人员及爱好者培训和自学使用。

本书配有电子教案，方便教学。

图书在版编目（CIP）数据

编译原理 / 毛红梅，严云洋主编．--北京：清华大学出版社，2011.8
（中国高等学校计算机科学与技术专业（应用型）规划教材）
ISBN 978-7-302-25303-7

Ⅰ．①编…　Ⅱ．①毛…②严…　Ⅲ．①编译程序－程序设计－高等学校－教材
Ⅳ．①TP314

中国版本图书馆CIP数据核字（2011）第065652号

责任编辑：谢　琛　李　晔
责任校对：白　蕾
责任印制：李红英

出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦A座
http://www.tup.com.cn　　**邮　　编**：100084
社　　总　　机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62795954，c-jsjjc@tup.tsinghua.edu.cn
质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：北京富博印刷有限公司
装 订 者：北京市密云县京文制本装订厂
经　　销：全国新华书店
开　　本：185×260　　**印　张**：14.25　　**字　数**：343千字
版　　次：2011年8月第1版　　**印　次**：2011年8月第1次印刷
印　　数：1～4000
定　　价：25.00元

产品编号：041034-01

编委会

序　言

应用是推动学科技术发展的原动力，计算机科学是实用科学，计算机科学技术广泛而深入地应用推动了计算机学科的飞速发展。应用型创新人才是科技人才的一种类型，应用型创新人才的重要特征是具有强大的系统开发能力和解决实际问题的能力。培养应用型人才的教学理念是教学过程中以培养学生的综合技术应用能力为主线，理论教学以够用为度，所选择的教学方法与手段要有利于培养学生的系统开发能力和解决实际问题的能力。

随着我国经济建设的发展，对计算机软件、计算机网络、信息系统、信息服务和计算机应用技术等专业技术方向的人才的需求日益增加，主要包括：软件设计师、软件评测师、网络工程师、信息系统监理师、信息系统管理工程师、数据库系统工程师、多媒体应用设计师、电子商务设计师、嵌入式系统设计师和计算机辅助设计师等。如何构建应用型人才培养的教学体系以及系统框架，是从事计算机教育工作者的责任。为此，中国计算机学会计算机教育专业委员会和清华大学出版社共同组织启动了《中国高等学校计算机科学与技术专业（应用型）学科教程》的项目研究。参加本项目的研究人员全部来自国内高校教学一线具有丰富实践经验的专家和骨干教师。项目组对计算机科学与技术专业应用型学科的培养目标、内容、方法和意义，以及教学大纲和课程体系等进行了较深入、系统的研究，并编写了《中国高等学校计算机科学与技术专业（应用型）学科教程》（简称《学科教程》）。《学科教程》在编写上注意区分应用性人才与其他人才在培养上的不同，注重体现应用型学科的特征。在课程设计中，《学科教程》在依托学科设计的同时，更注意面向行业产业的实际需求。为了更好地体现《学科教程》的思想与内容，我们组织编写了《中国高等学校计算机科学与技术专业（应用型）规划教材》，旨在能为计算机专业应用型教学的课程设置、课程内容以及教学实践起到一个示范作用。本系列教材的主要特点如下：

1. 完全按照《学科教程》的体系组织编写本系列教材，特别是注意在教材设置、教材定位和教材内容的衔接上与《学科教程》保持一致。

2. 每门课程的教材内容都按照《学科教程》中设置的大纲精心编写，尽量体现应用型教材的特点。

3. 由各学校精品课程建设的骨干教师组成作者队伍，以课程研究为基础，将教学的研究成果引入教材中。

4. 在教材建设上，重点突出对计算机应用能力和应用技术的培养，注重教材的实践性。

5. 注重系列教材的立体配套，包括教参、教辅以及配套的教学资源、电子课件等。

高等院校应培养能为社会服务的应用型人才，以满足社会发展的需要。在培养模式、教学大纲、课程体系结构和教材都应适应培养应用型人才的目标。教材体现了培养目标和育

人模式，是学科建设的结晶，也是教师水平的标志。本系列教材的作者均是多年从事计算机科学与技术专业教学的教师，在本领域的科学研究与教学中积累了丰富的经验，他们将教学研究和科学研究的成果融入教材中，增强了教材的先进性、实用性和实践性。

目前，我们对于应用型人才培养的模式还处于探索阶段，在教材组织与编写上还会有这样或那样的缺陷，我们将不断完善。同时，我们也希望广大应用型院校的教师给我们提出更好的建议。

《中国高等学校计算机科学与技术专业（应用型）规划教材》主编

陈明

2008 年 7 月

前　言

编译程序是计算机系统的基本组成部分之一，而且多数计算机都含有不止一个高级语言的编译程序，对有些高级语言甚至配置了几种不同性能的编译程序。编译程序出现在20世纪50年代早期，国外编译原理的技术比较早而且先进。为了研究学习编译原理的技术，本书结合了外国先进的编译原理技术和国内编译原理的研究现状。

本书按照编译器的编译过程来研究编译原理的相关知识，主要介绍了程序设计语言编译程序构造的一般原理、基本设计方法、主要分析技术等内容。全书分为11章，每一章结合详尽的例子讲解基础知识，然后在学习加油站中通过习题实例灵活运用本章的知识点，以加深读者对本章内容的理解。本书主要内容如下：

第1章简单地介绍编译原理的概念和分析过程。

第2章主要介绍文法和语言的相关概念，以及如何构造一种语言的相关文法。

第3章主要介绍上下文无关文法的基本概念，以及上下文无关文法的化简。

第4章主要介绍三种单词描述工具：正规文法、正规表达式和自动机，并介绍这三种描述工具之间的相互转换。

第5章主要介绍编译过程中的语法分析阶段，自顶向下分析的语法分析方法，以及LL(1)文法及其判别方法。

第6章主要介绍编译过程中的语法分析的自底向上分析中的优先分析法，包含简单优先分析法和算符优先分析法。

第7章主要介绍编译过程中的语法分析的自底向上分析中的LR分析法，包含LR(0)、SLR(1)、LR(1)、LALR(1)分析法。

第8章主要介绍语法制导翻译的基本思想及自下向上的语法制导翻译方法，以及几种常见的中间代码形式。

第9章主要介绍符号表的作用、组织，以及在符号表中的语法错误和语义错误的校正。

第10章主要介绍代码的优化技术，包括局部优化、循环优化以及窥孔优化。

第11章主要介绍代码生成的相关理论知识及现在研究现状。

本书具有以下特色：

1. 定位准确

本书主要针对普通高等学校计算机科学与技术等相关计算机专业的学生学习编译原理的需要，以在编译程序的工作过程中涉及的一系列知识和方法的运用组织内容，并配合相关的例子进行讲解，力求做到概念清晰、原理讲解浅显易懂，方便学生理解和掌握。

2. 体系结构和内容组织新颖

本书以“基础理论-浅显讲解-学习加油站”为主线组织编写。全书内容分得很细、很详

尽，对知识点概括得很清楚，讲解得浅显易懂。学生学习起来会比较轻松，利于理解。

3. 理论与实践相结合

本书在每个章节后都有一个“学习加油站”板块，设有一个或几个习题，涵盖本章的知识点，方便学生巩固和加深对知识点的理解和掌握。理论与实践的结合，便于提高读者的综合实践能力。

本书将难于理解的编译原理知识阐述得浅显易懂，注重理论和实践的结合。本书适合作为普通高等学校计算机科学与技术等相关专业的教材，也适合软件设计人员及爱好者培训和自学使用。

本书由毛红梅（南昌航空大学）、严云洋任主编，程勇、王廷蔚任副主编。在本书编写过程中，王静、何光明、王珊珊、吴涛涛、赵梨花、陈海燕、陈智、王程凌、云邈等同志提出了许多建设性意见，并在文字校对中做了大量工作，在此深表感谢！

本书配有电子教案，方便教学。

限于作者水平，书中难免存在不当之处，恳请广大读者批评指正。

作　者

2011 年 4 月

目　录

Chapter 1

第1章 编译简介

本章要点

- 理解编译器的概念
- 典型的语言处理系统
- 理解编译过程各阶段的作用
- 理解编译器的前端和后端

编译程序的原理和技术具有十分普遍的意义，以致在每一个计算机科学家的研究生涯中都会用到本书中涉及的相关原理和技术。编译器的编写会涉及程序设计语言、计算机体系结构、语言理论、算法和软件工程等学科。不过，有几种基本编译器编写技术已经被用于构建许多计算机的多种语言编译器。本章通过编译器的组成、编译器的工作环境以及简化编译器建造过程的软件工具来介绍编译。

1.1 编译器

编译器是现代计算机系统的基本组成部分之一，而且多数计算机系统都含有不止一个高级语言的编译器，对于某些高级语言甚至配置了几个不同性能的编译程序。从功能上看，一个编译器就是一个语言翻译程序，它读入用某种语言（源语言）编写的程序（源程序）并将其翻译成一个与之等价的以另一种语言（目标语言）编写的程序（如图 1.1 所示）。作为编译器的一个重要组成部分，编译器能够向用户报告被编译的源程序中出现的错误。比如，汇编程序是一个翻译程序，它把汇编语言程序翻译成机器语言程序。如果源语言是像 Fortran、C++ 、JAVA 等高级语言，目标语言是像汇编语言或机器语言那样的低级语言，则这种翻译程序称为编译程序。

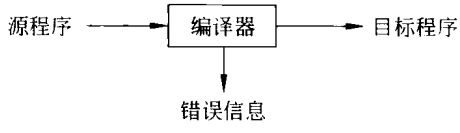

图 1.1 编译器

目前，世界上存在着数千种源语言，既有 Fortran 和 Pascal 这样的传统程序设计语言，也有各计算机应用领域中出现的专用语言。目标程序可以是另一种程序设计语言或者从微处理器到超级计算机的任何计算机的机器语言。不同语言需要不同的编译器。根据编译器的构造方法或者它们要实现的功能，编译器被分为一遍编译器、多遍编译器、装入并执行编译器、调试编译器、优化编译器等多种类别。从表面来看，编译器的种类似乎千变万化，多种多样。实质上，任何编译器所要完成的基本任务都是相同的。通过理解这些任务，我们可以

利用同样的基本技术为各种各样的源程序和目标机器构造编译器。

从20世纪50年代早期第一个编译器出现至今，我们所掌握的有关编译器的知识已经得到了长足的发展。很难说出第一个编译器出现的准确时间，因为最初的很多实验和实现是由不同的工作小组独立完成的。编译器的早期工作主要集中在如何把算术表达式翻译成机器代码。

整个20世纪50年代，编译器的编写一直被认为是一个极难的问题。比如Fortran的第一个编译器花了18年才得以实现(Backus al.[1957])。目前我们已经系统地掌握了处理编译期间发生的许多重要任务的技术。良好的实现语言、程序设计环境和软件工具也被陆续开发出来。

1.1.1 编译的分析-综合模型

编译由两部分组成：分析与综合。分析部分将源程序切分成一些基本块并形成源程序的中间表示，综合部分把源程序的中间表示转换为所需要的目标程序。在这两部分中，综合部分需要大量的专门化技术。

在分析期间，源程序所蕴含的操作将被确定下来并被表示成为一个称为语法树的分层结构。语法树的每个节点表示一个操作，该节点的子节点表示这个操作的参数。

许多操纵源程序的软件工具都首先要完成某种类型的分析。下面是这类软件工具的示例。

1. 结构编辑器

结构编辑器将一个命令序列作为输入来构造一个源程序。结构编辑器不仅实现普遍的文本编辑器的文本创建和修改功能，而且还对程序文本进行分析，为源程序构造恰当的层次结构。结构编辑器能够完成程序准备过程中所需要的功能。例如，它可以检查输入的格式是否正确，能自动地提供关键字(例如，当用户敲入关键字while的时候，编译器能够自动提供匹配的关键字do并提醒用户必须在两者之间插入一个条件体)，能够从begin或者左括号跳转到与之匹配的end或者右括号。这类结构编辑器的输出常常类似一个编译器的分析阶段的输出。

2. 智能打印机

智能打印机能够对程序进行分析，打印出结构清晰的程序。例如，注释一种特殊的字体打印；根据各个语句在程序的层次结构中的嵌套深度来缩排这些语句。

3. 静态检查器

静态检查器读入一个程序，分析这个程序，并在不运行这个程序的条件下试图发现程序的潜在错误。比如，静态检查器可以查出源程序中永远不能执行的语句，也可以查出变量在被定义以前被引用，还可以捕获诸如将实型变量用作指针这样的逻辑错误。

4. 解释器

解释器不是通过编译来产生目标程序，而是直接执行源程序中蕴含的操作。由于命令语句中执行的每个操作通常都是对编辑器或编译器一类复杂例程的调用，因此解释器经常用于执行命令语言。类似地，一些“非常高级”的语言，如APL，通常都是解释执行的，因为有许多关于数据的信息(如数组的大小和形状)不能在编译时得到。

按照传统的观念，编译器一般被看成是把使用Fortran等高级语言编写的源程序翻译

成汇编语言或某种计算机的机器语言的程序。然而，在很多与语言翻译毫不相关的场合，编译技术也常常被使用。下面举出的每一个例子中的分析部分都与传统观念中的编译器的分析部分相似。

(1) 文本格式器(text formatter)。文本格式器的输入是一个字符流。输入字符流中的多数字符串是需要排版输出的字符串，同时字符流中也包含一些用来说明字符流中的段落、图表或者上标和下标等数学结构的命令。

(2) 硅编译器(silicon compiler)。硅编译器的输入是一个源程序，这个源程序的程序设计语言类似于传统的程序设计语言。但是，该语言中的变量不是内存中的地址，而是开关电路中的逻辑符号(0 或 1)或符号组。硅编译器的输出是一个以适当语言书写的电路设计。

(3) 查询解释器(query interpreter)。查询解释器把含有关系和布尔运算的谓词翻译成数据库命令，在数据库中查询满足该谓词的记录。

1.1.2 编译器的前驱和后继

为了建立可执行的目标程序，除了编译器外，我们还需要几个其他的程序：源程序可能被分为模块存储在不同的文件中，而一个称为预处理器的程序会把存储在不同文件中的程序模块集成一个完整的源程序。预处理器也能够把程序中称为宏的缩写语句展开为原始语句加入到源程序中。

图 1.2 给出了一个典型的语言处理系统。从图 1.2 中可以清楚地看到一个语言处理的过程，下面简要概括一下该过程。

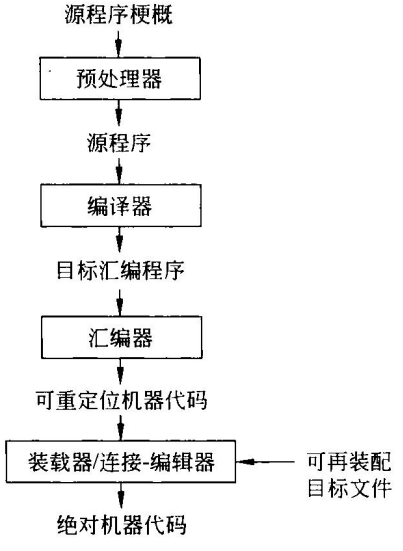

图 1.2 一个典型的语言处理系统

(1) 预处理器将可能位于不同文件中的几个模块的源程序梗概汇集在一起，形成一个源程序。预处理器可负责宏的展开，如 C 语言中的预处理器要完成文件的合并、宏展开等任务。

(2) 编译器将由预处理器处理过的源程序进行编译，生成目标汇编程序。值得注意的是，一个编译器的输入可能是由一个预处理器来产生的，也可能是由若干个预处理器来产生的。

(3) 汇编器将目标汇编程序翻译成可重新定位的机器代码。

(4) 装载器(连接-编辑器)负责将可重定位的机器代码和可再装配的目标文件进行处理，生成最后可被计算机识别的绝对的机器代码。

1.2 编译器的分析过程

编译器负责将源程序编译生成目标程序，从整体上说，一个编译器的整个工作过程是划分为不同的阶段进行的，每一个阶段将源程序的一种表示形式转化为另一种表示形式，各个

阶段进行的操作在逻辑上紧密连接在一起。图 1.3 给出了一个典型的划分方法。

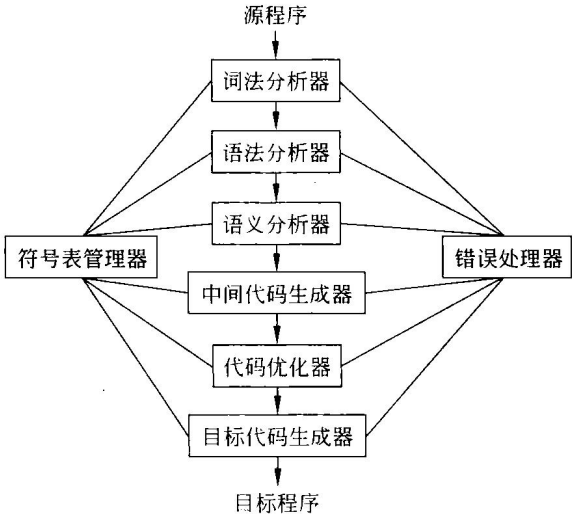

图 1.3　编译器各个阶段的划分

实际上，编译器的某些阶段是可以合并到一起的，因此如果某几个阶段可以合并到一起，这些阶段的中间表示就不需要明确地构造出来。图 1.3 中，将编译器处理语言的过程分为了六个阶段：词法分析器、语法分析器、语义分析器、中间代码生成器、代码优化器和目标代码生成器。其中，词法分析器、语法分析器和语义分析器这三个过程构成了编译器的分析部分。

符号表管理器和错误处理器是这六个阶段都需要涉及的两个部分，在编译过程中源程序中的各种信息都被保留在各种不同的符号表格中，编译器的各个部分的工作都要涉及构造、查找或更新有关的表格，因此需要有符号表的管理器；如果在编译过程中发现源程序有错误，编译程序应该能够报告出错误的性质和错误发生的地点，并且将错误所造成的影响限制在极小的范围内，使得源程序的其他部分能够继续被编译下去，有些编译器能够自动校正错误，这种工作称为出错处理。因此，将这两个过程称为“编译器”的过程。

下面将针对编译器的这几个过程进行简单的介绍。

1.2.1　词法分析阶段

词法分析，也称为线性分析或者扫描，是编译过程的第一个阶段，这个阶段中，从左到右地读取源程序的字符流，对字符流进行扫描并分解为多个单词，而这些单词就是具有整体含义的字符序列。比如，我们所熟悉的标识符必须是以字母字符开头，后跟字母、数字字符的字符序列组成的一种单词。另外，还有保留字(或关键字)、算术运算符、分界符等。

例如，在词法分析中，用 C++ 语言编写的某程序片段为：

```
int  price=3,num=60,sum=0; sum=price * num;
```

在词法分析阶段会将这段程序分组为以下的单词序列：