电子计算机应用系列教材

编译方法

迟忠先 等 编著

科学出版社

电子计算机应用系列教材

编 译 方 法

迟忠先等 编著

科学出版社

1992

(京)新登字 092 号

内 容 简 介

本书以 PASCAL 语言为背景，深入浅出地介绍了高级语言的编译方法，并力图介绍当前最新、最适用的方法和实现技巧. 全书共分十章，前七章以一个具体的 PASCAL 子集的编译程序为线索，分别阐述了词法分析、语法分析、符号表的组织、存贮分配、语义分析与目标代码生成等的实现方法以及与之有关的理论背景. 第八章包括各种中间语言形式及优化的基本概念和方法. 第九章专论错误处理. 最后一章讨论编译程序的开发方法和工具. 各章均附有习题. 为了帮助读者系统地理解本书的全部内容，特以一个编译程序实例贯穿全书.

本书可作为计算机软硬件人员的培训教材或自学用书，也可以供大专院校有关专业作为教材或教学参考书，亦可供从事实际工作的软件人员参考.

电子计算机应用系列教材

编 译 方 法

迟忠先等 编著

责任编辑 杨家福

科学出版社出版

北京东黄城根北街 16 号

邮政编码：100707

湖北省武汉市华中电子信息产业开发集团公司激光照排

天津市静一胶印厂印刷

新华书店北京发行所发行　　各地新华书店经售

1992 年 2 月第 一 版　　开本：787×1092　1/16

1992 年 2 月第一次印刷　　印张：15

印数：0001—5400　　字数：338 000

ISBN7-03-001348-4/TP・86

定价：9.70元

电子计算机应用系列教材主持、组织编著单位

主持编著单位：

国务院电子信息系统推广应用办公室

组织编著单位(以笔划为序)：

广东、广西、上海、山东、山西、天津、云南、内蒙古、
四川、辽宁、北京、江苏、甘肃、宁夏、江西、安徽、
河北、河南、贵州、浙江、湖北、湖南、黑龙江、福建、
新疆、广州、大连、宁波、西安、沈阳、武汉、青岛、
重庆、哈尔滨、南京等35省、市、自治区、计划单列市

电子振兴
计 算 机领导小组办公室
科技工作

电子计算机应用系列教材联合编审委员会名单

(以姓氏笔划为序)

主编审委员：

王长胤* 苏世生 何守才 陈有祺 陈莘萌* 邹海明* 郑天健
殷志鹤 童 颋 赖翔飞 (有"*"者为常务主编)

常务编审委员：

于占涛 王一良 冯锡祺 刘大昕 朱维华 陈火旺 陈洪陶 余 俊
李 祥 苏锦祥 佟震亚 张广华 张少润 张吉生 张志浩 张建荣
钟伯刚 胡秉光 高树森 徐洁盘 曹大铸 谢玉光 谢育先 韩兆轩
韩培尧 董继润 程慧霞

编审委员：

王升亮 王伦津 王树人 王振宇 王继青 王翰虎 毛培法 叶以丰
冯鉴生 刘开瑛 刘尚威 刘国靖 刘晓融 刘德镇 孙令举 孙其梅
孙耕田 朱泳岭 许震宇 何文兴 陈凤枝 陈兴业 陈启泉 陈时锦
邱玉辉 吴宇尧 吴意生 李克洪 李迪义 李忠民 迟忠先 沈林兴
肖金声 苏松基 杨润生 闵福德 张志弘 张银明 张 勤 张福源
张翼鹏 郑玉林 郑 重 郑桂林 孟昭光 林俊伯 林钧海 周俊林
赵振玉 赵惠溥 姚卿达 段银田 钟维明 袁玉馨 唐肖光 唐楷全
徐国平 徐拾义 康继昌 高登芳 黄友谦 黄 侃 程锦松 楼朝城
潘正运 潘庆荣

秘 书 组：

秘 书 长：胡茂生
副秘书长：何兴能 林茂荃 易 勤 黄雄才

序

当代新技术革命的蓬勃发展，带来社会生产力新的飞跃，引起整个社会的巨大变革．电子计算机技术是新技术革命中最活跃的核心技术，在工农业生产、流通领域、国防建设和科学研究方面得到越来越广泛的应用．

党的十一届三中全会以来，我国计算机应用事业的发展是相当迅速的．到目前为止，全国装机量已突破三十万台，十六位以下微型计算机开始形成产业和市场规模，全国从事计算机科研、开发、生产、应用、经营、服务和教学的科技人员已达十多万人，与1980年相比，增长了近八倍．他们在工业、农业、商业、城建、金融、科技、文教、卫生、公安等广阔的领域中积极开发应用计算机技术，取得了优异的成绩，创造了显著的经济效益和社会效益，为开拓计算机应用的新局面作出了重要贡献．实践证明，人才是计算机开发应用的中心环节．我们必须把计算机应用人才的开发与培养放在计算机应用事业的首位，要坚持不懈地抓住人才培养这个关键．

从目前来看，我国计算机应用人才队伍虽然有了很大的发展，但是这支队伍的数量和质量还远不适应计算机应用事业发展的客观需要，复合型人才的培养与教育还没有走上规范化、制度化轨道，教材建设仍显薄弱，培训质量不高．因此，在国务院电子信息系统推广应用办公室领导、支持下，全国三十五个省、市、自治区、计划单列市计算机应用主管部门共同组织118所大学和科研单位的400多位专家、教授编写了全国第一部《电子计算机应用人才培训大纲》以及与之配套使用的电子计算机应用系列教材，在人才培训和开发方面做了一件很有意义的工作，对实现培训工作规范化、制度化将起到很好的推动作用．

《电子计算机应用人才培训大纲》和电子计算机应用系列教材贯穿了从应用出发、为应用服务，大力培养高质量、多层次、复合型应用人才这样一条主线．大纲总结了近几年各地计算机技术培训正反两方面的经验，提出了计算机应用人才的层次结构、不同层次人才的素质要求和培养途径，制定了一套必须遵循的层次化培训办学规范，编制了适应办学规范的“课程教学大纲”．这部大纲为各地方、各部门、各单位制定人才培养规划和工作计划提供了原则依据，为科技人员、管理人员以及其他人员学习计算机技术指出了努力方向和步骤，为社会提供了考核计算机应用人才的客观尺度．“电子计算机应用系列教材”是培训大纲在教学内容上的展开与体现，是我国目前规模最大的一套计算机应用教材．教材的体系为树型结构，模块化与系统性、连贯性、完整性相兼容，教学内容注重实用性、工程性、科学性，并具有简明清晰、通俗易懂、方便教学、易于自学等特点，是一套很好的系列教材．

这部大纲和系列教材的诞生是各方面团结合作、群策群力的结果，它的公开出版和发行，对计算机应用人才的培训工作将起到积极的推动作用．希望全国各地区、各部门、各单位广泛运用这套系列教材，发挥它应有的作用，并在实践中检验、修改、补充和完善它．

通过培训教材的建设，把培训工作与贯彻国家既定的成人教育、函授教育、电视教育

和科技人员继续工程教育等制度相结合，逐步把计算机应用人才的培训工作引向规范化、制度化轨道，为培养和造就大批高素质、多层次、复合型计算机应用人才而努力奋斗，更好地推动计算机应用事业向深度和广度发展.

李祥林

1988年10月17日

前　言

编译方法是计算机科学中发展得比较成熟的分支，对软件工程学的发展有很大影响．为了促进我国软件工程水平的进一步提高，为培养更多软件技术人员提供一本有用的教材，我们特编写了本书．

本书主要内容包括词法分析、语法分析、语义分析、优化与目标代码生成等．作为一本介绍编译方法的教材，本书不仅注重理论对于实践的指导作用，力图说明各种方法的理论渊源，而且还特别注重方法的具体实现细节和编译程序的整体概念．

全书共十章．前七章以一个 PASCAL 子集的编译程序为线索，系统地介绍了高级语言编译程序的任务、结构及各个组成部分的实现方法．三、四两章除了介绍词法分析、语法分析所需的各种实用的方法及其理论背景外，还结合其中的递归子程序法具体地讨论了程序实现，以使所述的概念、方法落到实处．五、六、七章讨论了符号表、存贮分配、语义分析及目标代码生成，除介绍一般思想外，还针对具体编译程序实例加以展开．最后三章分别介绍优化、错误处理及编译程序的开发方法．

本教材的参考学时为 56 学时，适于作计算机应用人才的培训教材和自学用书，亦可作为大专院校有关专业的教材或教学参考书．

采用本书作教材时，根据学员的基础与学时的多寡，以及不同层次的教学内容和教学要求，可以对带星号的章节酌情进行删减．此外，因为上机实习是学习本课程的重要组成部分，所以许多章末都附有上机实习的习题．贯穿于全书的编译程序实例可用于说明程序实现方面的细节和加强编译程序的整体观念．这个实例的源码和可执行码已由大连理工大学计算机系录入软盘，可供上机实习之用（适用于 IBM-PC 系列机）．

本教材由迟忠先主编，王一良主审．参加编写工作的有迟忠先（一至五章和八至十章），杨元生（六至七章）．刘友红、李宪廷、许宏、刘川调试了编译程序实例并提供了部分初稿．参加审阅工作的还有张成学等同志．他们对本书的编写给予极大关怀和支持，提出了许多宝贵意，这里对他们表示诚挚的感谢．由于编著者水平有限，书中难免还存在一些缺点和错误，殷切希望广大读者批评指正．

目　录

第一章 引 论

1.1 编程语言

语言是交流思想的工具. 由于人类的思维遵循着共同的逻辑规律，所以不同民族之间的语言可以通过翻译来沟通. 现在，人们要利用计算机求解客观世界的各种问题，也必须把要解的问题以及如何求解用某种语言表达出来，告诉计算机. 这种语言通常称为**程序设计语言**或**编程语言**. 用编程语言表达计算机**解题步骤**(即**程序**)的活动称为**程序设计**或**编程序**. 因此，学习并使用编程语言乃是人们步入计算机世界的第一阶梯.

综观迄今为止的许许多多编程语言，它们大体上可以分为以下几类：

(1)低级语言或面向机器的语言：机器语言、汇编语言.

(2)高级语言或面向用户的语言：过程式语言，如 BASIC，FORTRAN，ALGOL，PASCAL，ADA 等；非过程式语言，又分函数型语言，如 LISP 和逻辑型语言，如 PROLOG.

(3)面向问题的语言.

机器语言 机器语言通常指的是机器的指令系统. 它是编程语言的最低形式. 程序中的每条指令都是用数值代码表示的，而编程的所有簿记性细节都要由程序员承担. 显而易见，要把客观世界的各种问题用这种呆板的数码表示出来是何等困难.

汇编语言 这种语言实际上是机器语言的符号表示. 如用 ADD 代表加法，用 MUL 表示乘法等等. 虽然在程序的可读性等方面，汇编语言有一定的改进，但仍然包含了许多依赖具体机器的特色.

高级语言 高级语言是一种独立于机器的编程语言. 它屏蔽了种种与机器有关的细节，增加了许多描述各种数据结构和控制结构的成份，使得问题的解更加接近它在问题领域中的自然表示，从而大大简化了编程工作.

面向问题的语言 这种语言能以问题领域中最自然的表示法来描述问题的计算机解法. 例如，用于数据库检索的 SEQUEL 和用于土木工程中的 COGO 等都属于这类语言.

虽然，计算机科学的最终目标是要提供一种能直接描述问题的语言，但目前距此目标还相差甚远. 高级语言仍然是编程的主要工具.

然而，高级语言最终必须翻译成机器语言才能在机器上运行. 虽然，目前有人在研究能直接执行高级语言的机器，但就现有的种种高级语言而言，它们都是通过一个**编译程序**自动翻译成机器语言的. 由于高级语言的重要性，它的实现技术——编译方法的研究构成了计算机科学的重要领域，而且已经取得了相当丰富的成果. 这些成果不仅可以直接用于系统软件的开发，而且可供其它领域借鉴和参考. 可以说，学习编译技术将会使计算机界的所有人获益. 本书将致力于介绍编译程序的设计方法及原理. 当涉及到具体的实现技术时，将以 PASCAL 为主要语言对象(间或辅以 FORTRAN 或 ALGOL)展开讨论. PASCAL 语言是当今世界上最为流行的编程语言之一，其结构具有广泛的代表性. 因而，PASCAL 语

言的编译方法也就有着普遍的可用性.

1.2 翻译程序

翻译程序 它是这样一个计算机程序,能把用高级语言或汇编语言写的程序翻译为等价的机器语言.被翻译的程序称为**源程序**,翻译出来的程序称为**目标程序**或**目标代码**.相应地,书写源程序的语言称为**源语言**,书写目标程序的语言称为**目标语言**.每种高级(或汇编)语言都有自己的翻译程序.通常,把汇编语言的翻译程序称为**汇编程序**,把高级语言的翻译程序称为**编译程序**.

编译程序的目标语言可以是机器语言,也可以是汇编语言.在后一种场合下,还需要再经过汇编程序的翻译才能得到用机器语言写的目标程序.于是,一个高级语言写的程序在机器上的执行过程可分为两个阶段(图 1.1):

(1)编译阶段:将源程序翻译为等价的目标程序.

(2)运行阶段:在机器上执行目标程序,以获得计算结果.

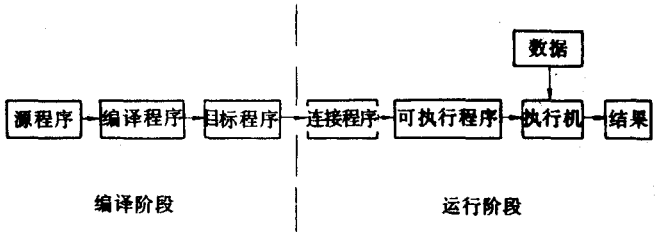

图 1.1 源程序的编译过程

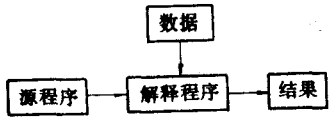

图 1.2 源程序的解释过程

现代编译程序只负责编译阶段的工作,运行阶段的工作由操作系统完成.通常是先由一个连接程序进行连接装配,得到一个可执行的程序,然后再运行这个可执行程序.

另外,有一类翻译程序称为**解释程序**.这种翻译程序集编译与运行为一体,同时处理源程序和数据.它采取边分析边执行的方式得到计算结果.图 1.2 给出了这种解释过程.

解释程序比起编译程序来具有许多优点,比如,解释程序很容易实现与用户的交互会话;局部程序的改动不需要重新编译整个程序等等.但是解释程序的运行速度通常是很慢的,使得人们不得不提出许多改进方案来提高其运行速度.一种方案是为高级语言提供两

种执行方式：编译方式与解释方式，各取所需. 例如，BASIC，PROLOG，LISP 等高级语言均有两种方式. 另一种方案是采取混合策略：把最经常执行的核心部分进行编译，其余部分进行解释. 当然，也可以先将源程序翻译为一种中间码，然后再解释执行中间码.

本书重点讨论编译程序的构造和实现方法.

1.3 编译程序的结构

编程语言之间的翻译与自然语言之间的翻译有许多共同之处. 将这两种翻译做一类比将有助于理解编译程序的工作原理. 通常，比如英文到中文的翻译大体上包括以下几个步骤：

(1)识别单词(词法分析).

(2)分析语法(语法分析).

(3)理解含义(语义分析).

(4)修饰(目标优化).

(5)写出译文(生成目标).

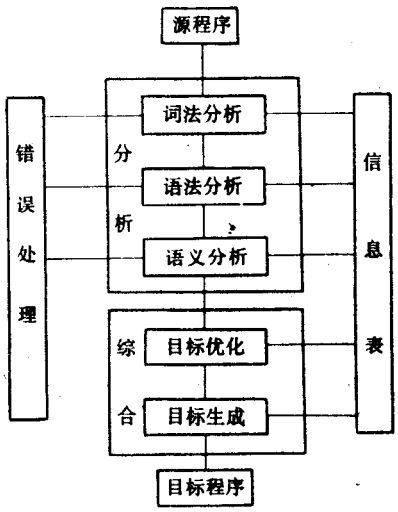

图 1.3 编译程序模型

与此相应地，编译程序中也应有相应部分来完成上述的任务(在括号内标出).

图 1.3 给出的是编译程序的基本模型. 虽然不同的编译程序都有自己特定的结构和工作方式，但是它们所应有的逻辑功能却基本相同，都不过是图 1.3 的不同翻版而已. 下面，我们来简要介绍一下图中各组成部分的基本功能，以便使读者对编译程序先有一个概括的了解.

编译程序必需完成两大任务：对源程序的分析和对目标程序的综合. 分析的任务是把源程序分解成它的基本成分；综合的任务则是利用这些基本成分建立它的等价的目标程序.

首先，源程序被看作是字符串输入到编译程序中去的，然后被下面各模块加工，最后便得到目标程序.

编译程序的第一个模块是**词法分析程序**(或称为**扫描器**).它的功能是把输入串划分成一系列的单词.**单词**是具有独立意义的最小语法单位.它也是一个字符串.词法分析的任务是把单词识别出来,从而把字符串形式的源程序转换为单词串的形式.单词串构成了下一个模块的输入.

第二个模块称为**语法分析程序**,又称为**分析器**.分析器具有两个功能,一是检查源程序(单词串形式)是否符合源语言的语法规则;一是按照语法结构重排源程序,通常采用的是树形结构(语法树),然后把这种树形表示的源程序输给它的下一个模块.

编译程序的下一个模块称为**语义分析程序**.它的功能是把树形表示的源程序转换为某种中间语言表示(如三元组、四元组等).这种表示在外表上很象机器语言,但还保留了语法树中有关的语法和语义信息.

目标优化的任务是改变源程序中的运算次序,以便产生更有效的目标程序.这里,有效的含义是指,或者目标程序短,或者目标程序的运行时间短.

最后一个模块是**目标生成程序**.它的任务是把(经过优化的)中间语言表示转换为目标语言程序.

上述各个模块都要与一个**表处理模块**打交道.表处理模块把分析阶段得到的有关信息填到相应的表中(如标识符表,常数表等),以便于生成目标时使用.

错误处理模块是任何软件系统都不能缺少的部分.我们不能期望语言的用户所写的源程序"绝对正确".好的编译程序应能检查出源程序中存在的错误,并把错误的性质、原因以及出错的位置报告给用户,进而对错误做出适当的处理,以便使编译过程得以继续进行.这就是错误处理模块的任务.

由图 1.3 可以看出,源程序在向目标程序转换的过程中需经过五道工序(词法分析、语法分析、语义分析、优化和目标生成).虽然有些工序可以结合进行,但它们的先后次序一般说来是不能改变的(优化除外,有些编译程序设有优化模块,有时优化可以在目标生成之后进行).有些编译程序是通过对源程序一次扫描就完成各道工序而直接生成目标程序的.这种编译程序称为是**单遍**的.而另一些编译程序则是通过多次扫描(每次扫描只完成一道或几道工序而产生中间语言),最后产生目标程序的.这种编译程序称为是**多遍**的.多遍扫描的优点是编译程序的结构清晰,便于优化,研制时也便于分工,缺点是可能要做一些重复的工作.

学习编译方法就是要弄清图 1.3 中各模块的工作原理及相互关系.我们将在三至七章中以一个单遍的编译程序为背景分别介绍词法分析、语法分析、语义分析及目标生成.多遍扫描、优化及错误处理将在八、九两章介绍,最后一章介绍编译程序的开发方法.

1.4 编译程序与其它软件工具的关系

编译程序是一个非常重要的软件开发工具,它与其它软件工具一起构成了所谓的**编程环境**.最理想的是建立一个软件开发环境,它不仅包含有丰富的软件工具,还包含程序设计方法学.它将全面支持软件的开发,从而大大提高软件生产率.但是,在现有的编程环境中,很少有完整的工具集.目前所拥有的大多数工具都与编译程序有着密切关系,它们都是直接或间接支持编程过程的.图 1.4 给出了一个简单的编程环境.现就其中与编译程

序有直接关系的几个工具作一简要介绍.

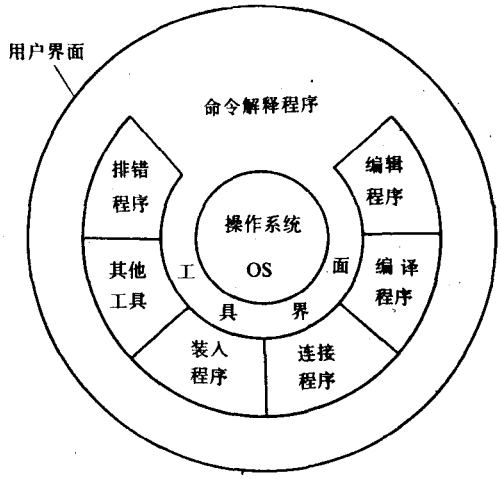

图 1.4

1.4.1 编辑程序

这一软件工具是用来建立新的源程序或对已有的源程序进行编辑加工的.编辑程序通常应具有下列功能:

(1)删除源程序中的某一行,或某一行中某些字符.

(2)在源程序中插入一行或在某一行中插入某些字符.

(3)搜索源程序中的某些行.

(4)用某些新行去替换源程序中的某些行.

(5)显示或打印源程序中的某些行.

现有的编辑程序大体上可以分为以下三类:

(1)行编辑:编辑工作是以行为单位进行的.

(2)屏幕编辑:编辑工作是以字符为单位进行的.

(3)语法制导编辑:这种编辑程序除了具有一般编辑功能外,最大的特点是源程序的编辑是在源语言的语法制导下进行的,因而所产生的源程序不再含有或很少含有语法错误了.

1.4.2 装入程序与连接程序

源程序经过编译程序的翻译变为等价的目标程序.但是,目标程序不能直接在机器上运行,还需要再做如下的处理才能变为真正可以运行的程序(可执行程序).

(1)装入:把目标程序装入到内存中去,以备执行.

(2)重定位:修改目标程序,使之被装入到与原先分配的地址不同的地方去.

(3)连接:把两个或更多个独立的目标程序连接到一起,使之能相互引用信息,一起运

行.

装入程序就是一个能完成装入功能的软件工具.有许多装入程序同时具有重定位和连接功能.有些系统使用一个称为连接程序(或连接编辑程序)完成连接功能,而重定位和装入则由单独一个装入程序完成.在许多情况下,同一系统中的所有编程语言的源程序都被翻译为相同格式的目标程序.于是,利用连接程序或装入程序可以把几个不同语言的源程序的目标程序连接到一起,构成一个可执行程序.

1.4.3 排错程序

这个工具能帮助排除源程序中的错误.在排错状态下,编译程序将在目标程序中生成若干便于用户查错的指令,如显示程序运行的线路,印出某些变量的值等等.排错程序还提供若干命令,这些命令可在目标程序运行时使用,如可在源程序中设置断点,使相应的目标程序运行到此处能暂停下来,以便使用其它的排错命令,可以显示或修改某些变量的值,也可以使源程序一个语句一个语句地执行.这些命令对于检查源程序的语义错误是非常有用的.

1.4.4 其它软件工具

例如,各种测试工具(静态测试工具,动态分析工具),漂亮的打印程序等.

习 题

1.1 您所熟悉的计算机都配置了哪些编程语言?试写出一种高级语言的上机操作过程.

1.2 试归纳几种高级语言(如 PASCAL,ALGOL,FORTRAN)在数据结构与控制结构上有哪些共同特点.

1.3 您熟悉哪几种机器上的汇编语言?试列出其中的算术运算指令、转移指令和比较指令.

1.4 您都使用过哪些软件工具?它们与编译程序有什么关系?

1.5 试论述高级语言及其编译程序在软件开发中的作用.

1.6 试比较解释式的 BASIC 与编译式的 BASIC 各有哪些优缺点.

1.7 试比较单遍编译程序与多遍编译程序的优缺点.

第二章　源语言的定义

编程语言的定义是语言设计者的任务.编译程序设计者的任务是在特定的计算机上实现编程语言.诚然,一语言的编译程序的最终实现也是该语言的一个定义,但这个定义必须与语言设计者所描述的定义相一致.因此,编写编译程序之前必须有一个关于它所要实现的源语言的精确定义.这个定义既是语言的用户用以编程的依据,又是语言实现者编写编译程序的依据.本章讨论源语言定义问题.我们先分别从生成和识别的观点来介绍语言的定义方法,而后把形式语言作为源语言的数学模型,把自动机作为编译程序的数学模型来展开讨论.

2.1　语言及其表示法

完整地定义一个语言时必须说明,什么样的符号是合法程序所允许的(定义终极符集或字母表),什么样的符号串是合法的程序(定义语言的语法)以及对合法程序赋予什么含义(定义语言的语义).这三项任务中只有定义终极符集一项比较容易实现.最困难的是定义语言的语义问题.就现有的编程语言来看,其语义多数都是利用自然语言描述的.这种非形式化的描述很不适于机械翻译.虽然近年来语义形式化的研究取得了很大进展,有些成果已用于 ADA 语言的实现中,但详细讨论形式语义已超出了本书范围.我们仍然采用自然语言来描述编程语言的语义.因此,本章所讨论的源语言定义问题实际上指的是语法定义问题.

形式地看,终极符集 V_T 上的一个语言是由 V_T 的元素所组成的串(包括空串)集合 V_T^* (V_T 的闭包)的一个子集.由 V_T^* 组成的语言没有多大用途,因为它太大了.通常把语言 L 定义为某个有穷字母表 V_T 上的一组串或句子,所以,$L \subset V_T^*$.

问题是怎样把一个语言表示出来.对于有穷语言(它的句子的数目是有穷的),可以通过枚举它的所有句子的办法来描述它.例如,假定字母表 V_T 是 26 个英文字母的集合,则 V_T 上的串集

{red, yellow, blue}

(在语法上)定义了一个只有三个句子的语言.然而,真正实用的语言大多都由无穷个句子组成.对于这种无穷语言,枚举的办法是行不通的.另一方面,定义语言的方法本身也必须是有穷的.满足这一需要的一种方法是使用一种称为**文法**的生成装置.文法是由所谓规则的一个有穷集合组成.这有穷个规则给语言的句子规定了一种结构,这就是语言的语法.利用这种生成装置可以系统地生成语言的所有句子,从而也就定义了这个语言.关于文法的研究构成了计算机科学的一个重要领域——形式语言理论.这一领域是在 50 年代中形成的.那时(1959),Noam Chomsky 在研究自然语言时给出了文法的数学模型.其后,文法被用来描述 ALGOL60 的语法,从而使文法这一概念变得越来越重要了,文法成为现今定义编程语言语法的强有力的工具.

定义语言的第二种方法是给出一种称为**识别机**或**接受机**的有穷装置.这种装置对于给定的串能识别它是否属于某一语言.关于接受机的研究也构成了计算机科学的一个重要领域——自动机理论.实际上,自动机可以作为编译程序的数学模型.我们将在词法分析和语法分析中进一步介绍与之有关的基本概念,并揭示文法和自动机之间的重要联系.本章将从生成观点出发,介绍语言的定义方法.

2.2 文法的基本概念

文法由有穷个规则组成.这有穷个规则给语言的句子规定了一种结构.自然语言中,这种结构是用主语、谓语、短语、名词等来描述的.对于程序来说,它的结构则是用过程、语句、表达式等来刻划的.通常把用来定义一种语言的语言称为**元语言**;把被定义的语言称为**对象语言**.在我国多数大学中,用来教英语的元语言是汉语.当然,英语也可以作为描述英语的元语言.在这里,我们主要讨论的是语言的语法(而不是语义)的描述.文法就是给出语言的语法定义的一种有效方法.

例如,无符号整数可以用如下的文法规则定义:

1.〈无符号整数〉→〈数字串〉

2.〈数字串〉→〈数字串〉〈数字〉|〈数字〉

3.〈数字〉→0|1|2|3|4|5|6|7|8|9

规则3定义〈数字〉是0到9之间的一个符号.规则2定义了组成〈数字串〉的方法:〈数字串〉可以仅由一个〈数字〉组成,也可以由原有的〈数字串〉后面再添加一个〈数字〉而构成.这是一种递归定义的方法,即在定义〈数字串〉时又用到了〈数字串〉本身.实际上,正是这种递归的手法才使我们得以用有穷的方法定义一个无穷语言.规则1则是简单地把〈无符号整数〉定义为〈数字串〉.于是,我们只用了三条规则就定义了〈无符号整数〉这个无穷语言.

当然,也可以使用别的规则.例如,下面的规则也产生全部无符号整数:

1.〈无符号整数〉→〈数字串〉

2.〈数字串〉→〈数字〉〈数字串〉|〈数字〉

3.〈数字〉→0|1|2|3|4|5|6|7|8|9

能产生相同语言的两个文法被认为是**等价**的.

在上述的定义方法中出现了两类符号:一类是组成〈无符号整数〉的基本符号,如0,1,…,9,它们是不需要再用规则加以定义的符号,称为**终极符**.全体终极符构成一个非空有穷集合,这称为**(终极)字母表**,用 V_T 表示.另一类符号,如→,|以及用"〈"与"〉"括起来的符号,它们是为了描述无符号整数的语法而引进的一些符号,称为**元语言符号**.用尖括号括起来的项是需要进一步加以定义的语法单位的名字,称为(元语言)**变量**或**非终极符**,它们的全体记为 V_N.通常,假定集合 V_N 与 V_T 是不相交的,并称集合 $V_N \cup V_T$ 为语言的**词汇表**.词汇表中的元素称为**文法符号**.以后我们约定大写字母 $A,B,\cdots,Z$ 表示非终极符;小写字母 $a,b,c,\cdots$ 表示终极符,$x,y,z,\cdots$ 表示终极符串;希腊字母 $\alpha,\beta,\cdots$ 表示词汇表上的符号串.串 α 的长度用 $|\alpha|$ 表示,空串(长度为0的串)用 ε 表示.这里,串的长度是指串中所含符号的个数.元符号"→"的含义是"定义为",而"|"的含义是"或者".$A \to \alpha|\beta$ 意为