编 译 技 术

高仲仪　蒋立源　编

西北工业大学出版社

编 译 技 术

北京航空学院　高仲仪
西北工业大学　蒋立源　编

西北工业大学出版社

内 容 简 介

本书为航空高等院校计算机专业统编教材。全书内容分为两部分：第一部分介绍编译的基本原理与技术，包括形式语言理论基础、词法分析、语法分析、符号表、存贮组织与分配、目标代码结构、语义分析和代码生成、出错处理及编译程序生成方法；第二部分介绍一个教学用的编译系统模型。各章之后附有习题，在第一部分之后还附有上机实习题。

在全书的组织方面，注意了理论与实践相结合、少而精、内容由浅入深及循序渐进的原则。

本书可作为工科院校计算机专业编译技术课的教材，也可供软件工程技术人员参考和作为自学用书。

编 译 技 术

高仲仪　蒋立源　编

责任编辑　刘　忠

西北工业大学出版社出版

陕西省新华书店发行

西北工业大学印刷厂印刷

开本 787×1092 毫米　1/16　印张 18　字数 449 千字

1985 年 9 月第一版　1985 年 9 月第一次印刷　印数 0001—2400 册

统一书号：15433·007　定价：3.48 元

前　言

本书系航空高等院校计算机科学与工程专业统编教材之一。

全书共分两部分：第一部分介绍编译的基本原理与技术，包括形式语言理论基础、词法分析、语法分析、符号表、存贮空间的组织与分配、语义分析和代码生成、出错处理及编译程序生成方法等；第二部分介绍一个教学用的编译系统模型。

第一部分供课堂讲授之用，约需70～80学时；第二部分供在教师指导下由学生自己阅读和分析，需20学时左右。

本书主要介绍经典的直到目前仍在广泛使用的基本编译技术，如递归子程序法和算符优先分析法等，同时对于七十年代以来，在编译技术领域所取得的新成果，如 LL 分析和 LR 分析等也作了介绍。

为了加强对学生能力的培养，以及使读者在学习了本门课程之后能对编译程序的整体结构及实现方法形成较完整的概念，本书在介绍了编译的基本理论及各种编译技术之后，还提供了一个教学用的编译系统模型，以便读者通过对该系统的学习和分析，一方面能进一步加深对前一部分内容的理解，另一方面也为具体设计和实现一个简单语言的编译程序起一定的引路和示范作用。

《编译技术》是一门实践性很强的课程，所以，最好的学习方法是进行实践，即在掌握一般原理和方法的基础上，自己动手构制编译程序。本书在第十章之后给出了一个简单语言的文法，作为学生上机实习题。该项作业可以分散在整个学期中进行，若条件许可，最好能安排在学完课程的第一部分内容之后集中时间进行，并建议用允许递归过程的高级语言（例如 PASCAL）作为编写程序的工具。

在各章节之后附有一定数量的习题。其中，不打星号的是必做题，打星号的是选做题。

本书是在编者多年教学实践的基础上，根据原有的讲义和讲稿整理而成。因此，在内容上遵循由浅入深的原则，在一些基本概念的阐述上力求通俗详尽，以便适于读者自学。

本书的一至四章和七至十六章由北京航空学院高仲仪编写，五和六章以及第四章中第一节的五和第二节的四由西北工业大学蒋立源编写，主审稿人是南京航空学院郁靖同志。

在本书的编写过程中，我们曾得到北京航空学院软件教研室李昭原、唐发根和李鼐超等同志以及西北工业大学软件教研室的帮助和支持；南京航空学院郁靖同志，沈阳航空工业学院黄正文同志以及三机部教材编审组杨心灿等同志参加了教材编写大纲的讨论；特别是北京航空学院金茂忠同志审阅了本书的初稿，提出了许多宝贵的意见，在此一并表示感谢。

由于我们学识浅薄，编写时间仓促，故错误和不妥之处在所难免，请读者批评和指正。

编　者

1983年10月

目　　录

第一部分　编译的基本原理和技术

第二部分　编译系统模型（ACOM）

第一部分　编译的基本原理和技术

第一章　绪　　言

第一节　程序设计语言的发展和编译程序的功能

众所周知，在计算机发展的初期，人们直接使用机器语言（机器指令）编写程序（称为手编程序）。但这是件极其繁琐的工作，需要耗费大量的人力和时间，其中很大一部分是机械的、重复性的工作，并且机器语言很不直观、难写、难读、容易出错，并且出错以后难于检查，查出错误也难以修改。程序设计和检查错误所费时间往往比机器解题所需的时间多出百倍，甚至数千倍以上，而且机器指令因机器不同而异，所以，程序设计要由受过一定训练并熟悉机器的程序员来做，这就大大限制了计算机的推广与使用。

为了解决这一问题，开始出现了符号语言，即用较直观的符号来代替用纯碎数字表示的机器指令代码，这样便于记忆，也便于检查程序。在符号语言的基础上，又进一步发展成为汇编语言。用汇编语言书写程序，除用直观的记忆码代替操作命字以对应一条条机器指令之外，还增加了若干宏指令，这些宏指令构成指令码的扩展。宏指令对应一组机器指令，完成一些特定的功能。

但是汇编语言仍然是依赖于具体机器的，不懂机器的人使用起来仍不方便。为了较彻底地解决这一问题，后来人们参照数学语言设计了一种算法语言（即高级程序设计语言）。由于这种语言完全摆脱了机器指令形式的约束，用它所编出的程序更接近于自然语言和习惯上对算法的描述，故称为不依赖于具体机器和面向问题的语言。这种语言便于人们学习和掌握，所编制的程序也便于阅读和修改，并能大大提高编制程序的效率。

算法语言的出现及在计算机上的应用，大大提高了计算机的功能，促进了计算机的普及使用，这在计算机特别是在软件发展史上是一个很重要的标志。

自 1956～1958 年第一个高级语言 FORTRAN 问世以来，各种语言如雨后春笋般地发展起来，如相继出现了 Algol，COBOL，PL/1，PASCAL，Jovial，Ada 等高级语言。现在世界上高级语言大约已有几百种之多，其中大多数是专用语言，比较通用的也有几十种。

应该指出，用高级语言编写的程序机器是不能立即执行的(因机器只能执行机器指令)，必须通过一个翻译程序的加工，使之转变为与其等价的机器语言程序，机器才能执行。这种翻译程序，我们称它为“编译程序”。某种高级语言的编译程序加上一些相应的子程序就构成该语言的编译系统。编译系统是计算机的重要系统软件之一。所以，要用高级程序设计语

言编程上机，需要相应的系统软件支持。

本课程的任务就在于介绍这种系统软件的设计原理及其实现的技术。

第二节 基 本 术 语

一、源程序、目标程序、翻译程序

（一）源程序　用源语言写的程序称为“源程序”。源语言可以是汇编语言，也可以是高级程序设计语言，所以用汇编语言或是高级程序设计语言写出的程序称为源程序。

（二）目标程序　也称为”结果程序”，是源程序经翻译程序加工以后所生成的程序。目标程序可以用机器语言表示（因此也称为目标代码或结果代码）；也可以用汇编语言或其他中间语言表示。

（三）翻译程序　所谓翻译程序是指一个把源程序翻译成等价的目标程序的程序。翻译程序本身在翻译时执行，源程序是它的输入，而目标程序则是其输出。

二、汇编程序、编译程序、解释程序

（一）汇编程序　汇编程序是这样一种程序：若源程序是用汇编语言所写，经翻译程序加工所生成的目标程序为机器语言程序，那么，该翻译程序就称为“汇编程序”。所以，用汇编语言编写的源程序先要经过汇编程序的加工，使其转变为与其等价的以机器语言表示的目标程序，才能上机执行。

（二）编译程序　若源程序是用高级程序设计语言所写，经翻译程序加工生成目标程序，那么，该翻译程序就称为“编译程序”。所以，用高级程序设计语言写的源程序要上机执行，则首先要经编译程序加工成为机器语言表示的目标程序。若目标程序是用汇编语言表示的，则还要经过一次汇编程序的加工，才能上机执行。

汇编程序和编译程序的区别是加工语言对象不同，前者是汇编语言写的源程序，而后者则是高级语言写的源程序。如前所述，用这两种语言写的源程序上机运行要分两个阶段进行：

1.编译或汇编阶段　将高级语言或汇编语言写的源程序经编译程序或汇编程序加工，转变成为与源程序等价的目标程序，见图 1.1 。

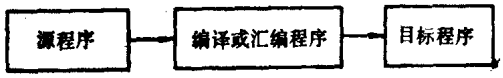

图 1.1　源程序的编译或汇编阶段

2.运行阶段　源程序经编译或汇编以后，生成机器语言表示的目标程序，就可装入机器运行，见图 1.2 。

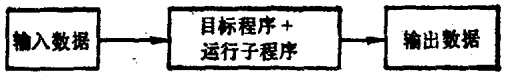

图 1.2　源程序运行阶段

其中运行子程序（RUN PROGRAM）是编译系统的组成部分。目标程序运行时，还要有一些子程序陪同目标程序运行，即目标程序运行时，要调用这些子程序，如数组动态存贮

分配子程序，下标变量地址计算子程序，下标出界检查子程序等等。这些子程序构成所谓运行系统，编译程序和运行系统合起来称为”编译系统”。

（三）解释程序　也是一种翻译程序，其功能是首先将用高级语言写的源程序加工成为一种中间代码（INTERMEDIATE CODE），然后直接解释执行该中间代码程序，这样一种程序称为“解释程序”。也有一种纯粹的解释程序，它没有生成中间代码这一步，而是直接解释执行源程序。解释程序对源程序进行解释执行过程中，先生成中间代码的目的是为了缩短执行每个语句所需的“译码”或分析的时间。

所以，用高级语言写的源程序要上机运行，可采用两种方式：一种是通过编译程序加工成为机器语言表示的目标程序，然后上机执行；另一种是由解释程序直接解释执行。由于解释程序对源程序中要重复执行的语句(例如，循环体中的语句)需要重复地解释执行，因此，较之编译方式要多花费执行时间，因而效率较低。但解释执行方式很便于实现人机对话，当程序员要根据前面执行的情况，随时调整后面的工作，随时更改后面的程序时，采用解释程序是很适合的。一些所谓“会话语言”（例如BASIC）的翻译往往就由解释程序来实现的。

本课程主要介绍编译程序的设计原理与技术。但是，很多技术也同样适用于解释程序，一个熟悉编译程序构造的人是不难了解和掌握解释程序的设计原理及其实现方法的。要进一步了解解释程序的读者，请参看有关参考书。

第三节　编译过程与编译程序结构

一、编译过程

如前所述，编译程序的功能是将高级语言表示的源程序翻译为等价的目标程序。它的工作过程与外文翻译过程有很多类似之处。一篇外文文章，是由字母、空格字符和各种标点符号所组成的字符串。若要把它翻译成为汉语，其翻译的大致过程是：首先从字符串中识别出一个个单词，然后根据文法规则，分析和识别出由这些单词所能组成的短语，继而识别出由短语所组成的句子。在分析和识别各种短语和句子的过程中，同时进行语法检查；接着再进行从单词、短语到句子的语义分析，这就完成了一个句子的翻译过程。然后，再翻译下一个句子，即重复上述过程；待全篇文章翻译完以后，再根据上下文对词句作必要的修饰工作，最后翻译成文，翻译过程宣告结束。既然编译程序也是做一种翻译工作，是将一种语言形式转变为另一种语言形式，因此，从某种意义上讲它也应具有上述这些类似的功能，所以，和普通翻译外文相仿，较典型的编译过程也大致要经过下列几个步骤：

1. 词法分析；
2. 语法分析；
3. 语义分析和中间代码生成；
4. 代码优化；
5. 目标程序生成。

下面以一个翻译简单的Algol源程序（省略了输入输出部分）为例子来说明这些步骤。

例　计算圆柱体表面积的Algol程序为

```
begin
  real R, H, S;
  S:=2*3.1416*R*(H+R)
end
```

程序中 begin、end 和 real 是语言的定义符，也称保留字，其意义是开始、结束和实型；R是存放圆柱体半径的单元，H是存放圆柱体高度的单元，S是存放圆柱体表面积的结果单元。R、H、S被说明为实型量。程序中一个可执行语句为赋值语句。对此源程序其翻译过程为（仿照外文翻译过程）：

（一）词法分析　源程序经穿孔（卡片或纸带）或直接通过键盘输入到机器的存贮器中，此时，在内存就得到一个连续存放的共有 38 个字符组成的字符串：

begin real R, H, S; S:=2*3.1416*R*(H+R)*end

其中 begin 和 real 之间，real 和 R 之间有一个空格字符。

像翻译外文资料那样，编译程序首先要从左到右扫描该字符串，并根据词法规则（即单词的组成规则）将该字符串分解为一个个具有独立语法意义的单词符号（也称为“单词”或“符号”），如保留字(begin, real, end)、标识符(R, H, S)、常数(2, 3.1416)、分界符(,, ;, :=, +, (,), *)等。这些单词是组成语句的基本符号。上述源程序经过这一步加工以后，就转变成由上述 22 个单词符号所组成的单词符号串了。

有的编译过程在进行词法分析时，除将源程序字符串分解为单词符号外，就造各种表（统称“符号表”），记录有关信息，以备后面进行语法语义分析及生成目标程序时使用，如对识别出的用户定义的标识符可造名字特性表，对常数可造常数表，故对于上述源程序，经词法分析以后可造出下列两张表，见图 1.3 。

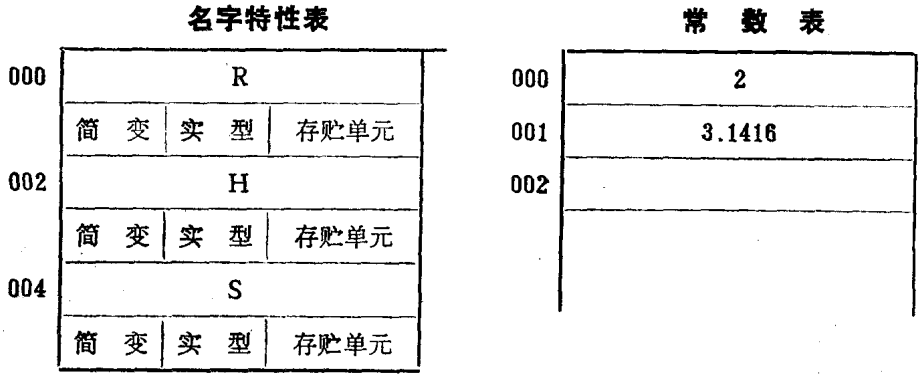

图 1.3　名字特性表和常数表

图中表格外的数字表示该信息单元在表区的相对地址。这里假定每个名字在名字特性表中占两个地址单元，第一个单元放名字本身，第二个单元放名字的特性信息；常数占一个地址单元，放常数的值。

词法分析时所得到的单词符号一般用一个二元式表示，见图 1.4 。其中单词类别表示单

图 1.4　词法分析程序输出的单词符号

词的种类，一般用整数编码表示。我们假定：标识符、无符号实数、无符号整数各作为一类，保留字及分界符统归为一类，并分别用整数编码1、2、3、0表示之。单词的值可用单词本身的编码来表示；也可以用一种内部编码表示；有时候，还可以是指示字（如果在词法分析阶段就造符号表的话）。例如，对标识符其单词的值就可用该标识符在名字特性表中的相对地址，对常数就用该常数在常数中的相对地址。

我们假定：单词符号用4位8进制数表示。第1位8进制数表示单词类别，后三位数表示单词值，对于保留字和分界符，其单词值用内部编码表示；对于标识符和无符号数，其单词值分别用相应名字特性表和常数表的相对地址表示。例如，我们采用如下的内部编码来表示保留字和分界符的单词值：

符号	内部编码
begin	001
end	002
real	003
:=	004
*	005
+	006
,	007
(	010
)	011
;	012

那么，上述源程序经词法分析以后，将加工成为如下的符号串（以内部编码表示的源程序符号串）：

0001	0003	1000	0007	1002	0007	1004	0012
begin	real	R	,	H	,	S	;
1004	0004	3000	0005	3001	0005	1000	0005
S	:=	2	*	3.1416	*	R	*
0010	1002	0006	1000	0011	0002		
(	H	+	R	)	end		

（二） 语法分析 经过词法分析以后，源程序符号串已经加工成为以内部编码表示的符号串了。和翻译外文一样，翻译的下一步任务是进行语法分析，即根据语法规则，将单词符号组合成（识别出）各种语法成份（如表达式、说明、语句、分程序等）。在组合各种语法成份时，同时进行语法检查，即检查各种语法成份在语法结构上的正确性。

例如，对于上例的源程序，通过语法分析，可以识别出“begin”是保留字，是组成分程序或复合语句的开始符号；“real”也是保留字，是属于语法成份〈类型说明符〉；“R，H，S”可构成语法成份〈标识符表〉，并由〈类型说明符〉〈标识符表〉构成新的语法成份〈类型说明〉，由〈类型说明〉构成〈说明〉，由〈说明〉构成〈说明串〉；下面可依次识别出“；”为语句结束符，“S”是〈变量〉，“:=”为赋值号，“2*3.1416*R*(R+H)”可组合为〈表

达式〉，并由〈变量〉:=〈表达式〉构成〈赋值语句〉，由〈赋值语句〉构成〈语句〉，由〈语句〉构成〈语句串〉；“end”是保留字，是分程序或复合语句的结束符；最后可识别出“begin〈说明串〉;〈语句串〉end”是一个分程序。可以看到，组合和识别各类语法成份的过程，也就是进行语法检查的过程。通过语法分析，可以建立起一棵倒立的树（称为语法树）。例如，对于上面的例子，通过语法分析可以建立如图 1.5 所示的语法树。

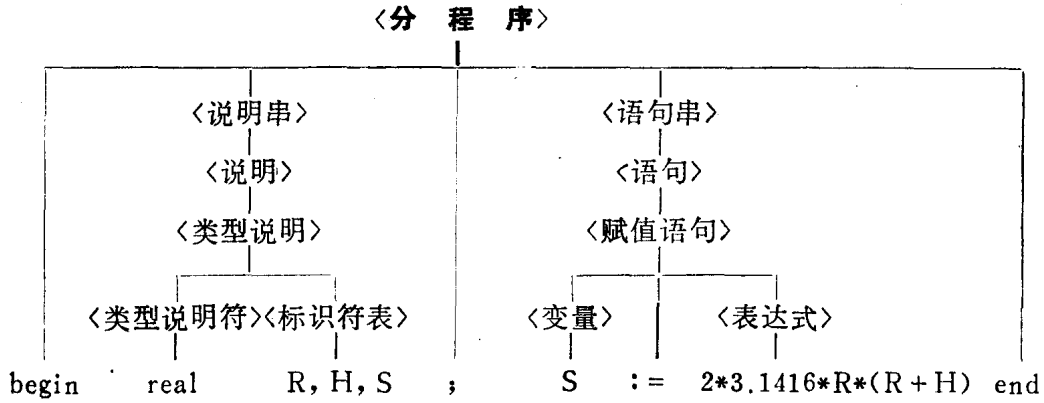

图 1.5 语法树

算法语言有一整套必须严格遵守的语法规则，语法分析就是按照所给定的语法规则进行的。因此，这种分析过程可以机械地进行，即可编出程序让计算机去做。在第二、三、四章，我们将详细讨论进行语法分析的原理和介绍各种语法分析的方法。

通过语法分析，可以证明上述源程序在语法结构上是正确的（留待后面去证明）。

（三）语义分析和中间代码生成　在语法分析的基础上，就可进行语义分析并生成中间代码。众所周知，定义一种语言，除要求定义其语法，即对语言的构成形式进行具体规定外，还要定义其语义，即对语言的各种语法成份赋予具体的意义。例如，对上述的类型说明语句，其语义为：R、H、S为实型简变；上述赋值语句的语义为：计算赋值号右边的表达式的值，并把它送到赋值号左边的变量所确定的内存单元中。所以，在计算机中，语言的语义就表现为要求机器完成哪些操作。

在进行语言的翻译时，首先要进行语法分析，识别出各类语法成份，并作语法检查。在此基础上，再作语义分析，根据所确定的语法成份的语义，设法用另一种语言形式（比源程序语言更接近于目标语言的语言或直接用目标语言）来表述这种语义，这就是语言的翻译过程。所以，语法和语义分析是翻译过程中的关键性步骤。

翻译程序在语义方面的主要工作，一是进行语义正确性检查。例如，通过语法分析识别出一个说明性语句时，语义分析就要检查那些被说明的标识符是否在同一程序中被说明了两次，若说明两次，常么就发生语义上的错误。如有分程序：

```
begin
    real A;
    integer A;
     ⋮
end
```

在该分程序中，变量A有二个语法上均是正确的说明语句，其中第一个说明语句说明A是实

型的，而第二个则说明A是整型的，A究竟是整型还是实型呢？这就发生了语义上的错误。再如，对某些语言来说，要求赋值语句赋值号左右两边的变量和表达式是同一类型的。这样，当语法分析识别出赋值语句时，语义分析就要去检查这种一致性；第二是要进行语义处理。语义处理的任务随语法成份的不同而不同，若是不生成目标程序的说明性语句（如Algol中的类型说明，常界数组说明），则将说明性语句中所定义的名字及其有关信息送进符号表并分配存贮单元等；若是表达式或是要生成目标程序的语句（如Algol中的开关说明，变界数组说明和赋值语句，条件语句等），则要根据其语义生成相应的中间代码。

所谓中间代码，是介于源语言和目标语言之间的一种中间语言形式。在编译过程中生成中间代码的目的是为了便于对目标程序作较细的优化处理，或者是为实现翻译程序的移植创造条件，或者便于解释执行。所以，并不是所有翻译过程都需要生成中间代码的。对于那些对目标程序质量要求不是很高和不考虑移植的编译程序，可不必生成中间代码，而可以直接生成目标代码。

中间代码的形式常因各种不同的要求而有很大的不同。比较常用的有三元式（二地址指令），四元式（三地址指令）和逆波兰表示等。例如，对于上例中的赋值语句

$$S:=2*3.1416*R*(H+R)$$

可被翻译成如下形式的一串四元式：

运算符	左运算对象	右运算对象	工作单元
*	2	3.1416	T_1
*	T_1	R	T_2
+	H	R	T_3
*	T_2	T_3	T_4
:=	S	T_4	

表中T_1、T_2、T_3、T_4是编译程序引进的工作单元，存放每一条指令的运算结果。我们定义上述每一个四元式所表示的语义从上到下依次为：

$$2*3.1416 \Rightarrow T_1;$$
$$T_1*R \Rightarrow T_2;$$
$$H+R \Rightarrow T_3;$$
$$T_2*T_3 \Rightarrow T_4;$$
$$T_4 \Rightarrow S\ 。$$

这样，我们就将一种Algol语言形式翻译为以四元式表示的另一种语言形式，这两种语言，在结构形式上是不同的，但在语义上是等价的。

上述赋值语句也可生成以三元式表示的中间代码：

运算符	左运算对象	右运算对象
*	2	3.1416
*	S_1	R
+	H	R
*	S_2	S_3
:=	S	S_4