

高等院校计算机专业教育改革推荐教材

编译方法 学习指导与实践

贺 汛 编著

本书是编译原理课程的辅导教材。书中对该课程的各部分内容进行了整理、归纳和总结,并指出其重点。同时,选取了一些具有一定难度和代表性的例题,进行了深入细致的分析和解答,为读者提供了解题思路和解题方法。书中还精选了一些与历届研究生入学试题相类似、难度相当的题目作为习题,并给出了参考答案。另外,本书还设计了一个简单的程序设计语言的编译程序,并将其作为实验内容,通过该实验可使读者加深对编译理论的理解,提高程序设计水平。

本书可作为编译原理课程的考研复习用书、教学辅导用书和实验指导书,也可作为计算机软件人员的参考资料。

图书在版编目(CIP)数据

编译方法学习指导与实践/贺汛编著. —北京:机械工业出版社, 2004.8

(高等院校计算机专业教育改革推荐教材)

ISBN 7-111-14821-5

I. 编... II. 贺... III. 编译程序—程序设计—高等学校—教学参考资料 IV. TP314

中国版本图书馆 CIP 数据核字(2004)第 063575 号

机械工业出版社(北京市百万庄大街 22 号 邮政编码 100037)

策 划: 胡毓坚

责任编辑: 陈振虹

责任印制: 石 冉

三河市宏达印刷有限公司印刷·新华书店北京发行所发行

2004 年 8 月第 1 版·第 1 次印刷

787mm×1092mm 1/16·15.5 印张·381 千字

0001—5000 册

定价: 22.00 元

凡购本图书,如有缺页、倒页、脱页,由本社发行部调换

本社购书热线电话(010) 68993821、88379646

封面无防伪标均为盗版

高等院校计算机专业教育改革推荐教材

编委会成员名单

主 编 刘大有

副主编 王元元

编 委 (按姓氏笔画排序)

刘晓明 李师贤 张桂芸 徐汀荣

耿亦兵 顾军华 黄国兴 薛永生

编者的话

计算机科学技术日新月异的飞速发展和计算机科学技术专业教育的相对滞后,已是不争的事实。

有两个发人深省的现象:一是,由于非计算机专业的学生既具有一门非计算机专业的专业知识,又具有越来越高的计算机应用技术水平,从而使计算机专业的学生感受到一种强烈的冲击和压力;二是,创建软件学院的工作已有近两年的历史,但软件学院的计算机专业教育的定位仍在探讨之中。

我们认为计算机科学与技术专业(以下简称计算机专业)教育的改革势在必行,正确认识和划分计算机专业教育的层次,对该专业的教育改革无疑是一个非常重要的问题。我国的计算机专业教育主要分三个层次。一般说来,这三个层次通常分布在以下三类高等院校:

第一层次主要以具有计算机一级学科博士学位授予权的教育部属重点高等院校为代表(包括具有两个博士点的大学)。这一类大学本科着重培养理论基础比较坚实、技术掌握熟练、有一定研究和开发能力的计算机专业学科型人才,其中部分学生(约本科生的10%)可攻读博士学位。

第二层次主要以具有一个计算机二级学科专业博士点的教育部属高等院校为代表。这一类高等院校本科着重培养有一定的理论基础、技术掌握比较熟练、有一定的研究或开发能力的计算机专业人才,其中一部分培养成学科型人才,另一部分培养成应用型人才,一小部分学生(约本科生的5%)可攻读博士学位。

第三层次主要以具有计算机二级学科专业硕士点的省属高等院校为代表。这一类高等院校本科面向企业应用,侧重培养对计算机技术或部分计算机技术掌握比较熟练,有一定的开发、应用能力的计算机专业应用型人才,其中很小一部分学生(约本科生的2.5%)可攻读博士学位。

国家教育部、计委批准的或省教育厅批准的示范性软件学院,就其培养目标和办学特色而言,分别与第二层次中应用型人才培养部分以及第三层次比较相近,但在如下方面有所不同:将软件工程课程作为专业教学重点;更加强调英语教学,更加重视实践能力培养,并对两者有更高的要求。

我们本着对高等院校的计算机专业状况的认识,主要面向与上述第二、第三两个层次对应的院校及与之相近的软件学院,总结多年的计算机专业的教改经验,在一定程度上溶入了ACM& IEEE CC2001和CCC2002(中国计算机科学与技术学科教程)的教改思路,组织我国一直投身于计算机教学和科研的教师,编写了这套“高等院校计算机专业教育改革推荐教材”(以下简称“推荐教材”)。自然,“推荐教材”中所贯穿的改革思路和做法,也是针对上述第二、第三两个层次对应院校的计算机专业学生。这些思路和做法可概括成以下三句话:

- 适度调整电子技术基础、计算机理论基础和系统软件的教学内容。
- 全面强化计算机工具软件、应用软件的教学要求。
- 以应用为目标大力展开软件工程的教学与实践。

电子技术基础、计算机理论基础、系统软件教学关系到学生的基本素质、发展潜力和日后的应变能力。“推荐教材”在调整它们的教学内容时的做法是:适度压缩电子线路、数字

电路和信号系统的教学内容,变三门课程为两门,并插入数字信号处理的基础内容;合并“计算机组成原理”、“微型计算机接口技术”和“汇编语言”为“计算机硬件技术基础”一门课程;注意适当放宽“离散数学”课程的知识面,使之与 CCC2002 的要求基本接轨,但适度降低其深度要求;更新系统软件课程的教学内容,以开放代码的 Linux 作为操作系统原理的讲授载体,更加关注系统软件的实践性和实用性。

为了提高计算机专业人才的计算机应用能力,全面强化计算机工具软件、实用软件的教学要求是十分重要的,这也是上述改革思路的核心。为此,“系列教材”的做法是:强化程序设计技术,强化人机接口技术,强化网络应用技术。

为强化程序设计技术,“推荐教材”支持在单片机环境、微机平台、网络平台的编程训练;支持运用程序设计语言、程序设计工具以及分布式对象技术的编程训练。大大加强面向对象程序设计课程的组合(设计了三门课程:面向对象的程序设计语言 C++,面向对象的程序设计语言 Java 和分布式对象技术),方便教师和读者的选择。

为强化人机接口技术,“推荐教材”设计了“人机交互教程”,“计算机图形学”和“多媒体应用技术”等可供选择的、有层次特色的课程组合。

为强化网络应用技术,“推荐教材”设计了“计算机网络技术”,“计算机网络程序设计”,“计算机网络实验教程”和“因特网技术及其应用”等可供选择的、新颖丰富的课程组合。

将软件工程课程作为专业教学重点,以应用为目标大力展开软件工程的教学与实践,是“推荐教材”改革思路的又一亮点。为改变以往软件工程课程纸上谈兵的老毛病,“推荐教材”从工程应用出发,理论联系实际,突出建模语言及其实现工具的运用,设计了“软件工程的方法与实践”,“统一建模语言 UML 导论”和“ROSE 对象建模方法与技术”等可供选择的、创新独特的软件工程课程组合。对于各类软件学院,“推荐教材”的这一特色无疑是很有吸引力的。

强调实践也是计算机学科永恒的主题,对计算机应用专业的学生来说更是如此。重应用和重实践是“推荐教材”的一个整体特点。这一特点,一方面有利于解决本文开始所指出的计算机专业学生较之非计算机专业学生,在应用开发工作中上手慢的问题;另一方面,使计算机专业的学生能在更大范围内、更高层面上掌握计算机应用技术。这一特点正是许多高等院校计算机专业教育改革追求的一个目标,也是国家教育部倡导软件学院的初衷之一。

“推荐教材”由基础知识、程序设计、应用技术、软件工程和实践环节等五个模块组成。各模块有其对应的培养目标与功能,从而构架出一个创新的、完整的计算机应用专业的课程体系。模块化的设计,使各学校可根据学生及学校的特点做自由的选择和组合,既能达到本专业的总体要求,又能体现具有特色的个性发展。整套教材的改革脉络清晰,结构特色鲜明,值得各高等院校在改革教学内容、编制教学计划、挑选教材书目时借鉴和参考。当然,很多书目也适合很多相关学科的计算机课程用作教材。

“推荐教材”的组成模块和书目详见封底。显然它不能说是完备的(实践环节模块更是如此),其改革的思路、改革的举措也可能有值得探讨的地方。我们衷心希望得到计算机教育界同仁和广大读者的批评指正。

高等院校计算机专业教育改革推荐教材
编委会

前 言

编译技术是计算机语言发展的支柱，也是计算机科学中发展最迅速、最成熟的一个分支。编译原理作为计算机专业的一门核心课程，在教学中占有十分重要的地位。由于其内容具有较强的理论性和实践性，学生在学习过程中普遍感到内容抽象、难于理解、题目难做。为此特编写本书，希望本书能够成为学生学习编译原理的好帮手，能使读者更全面、更透彻地理解和掌握编译的原理和方法，不再感到这门课程难学。

本书作为编译原理课程的学习辅导用书，内容包括了编译各阶段的基本理论与方法，以及在编译过程中涉及到的一些问题。全书共分 9 章：第 1 章简单介绍了编译程序的功能、编译阶段的划分及编译程序的结构；第 2 章对语言与它的描述工具——文法进行了叙述；第 3 章为词法分析，涉及的主要技术是正规式与有限自动机；第 4 章为语法分析，主要包括自顶向下分析法中的预测分析法（LL(1)分析法）和递归下降分析法、自底向上分析法中的算符优先分析法和 LR 分析法；第 5 章为语法制导翻译与中间代码生成的相关内容；第 6 章介绍了编译过程中大量使用的符号表，包括它的作用及组织方法和管理方式；第 7 章从编译的角度介绍了程序运行时如何分配、管理好存储空间；第 8 章为代码优化，主要介绍了局部优化、循环优化的方法；第 9 章为目标代码生成，主要介绍了基本块的优化算法。

书中每章的结构都相同，包括四部分：（1）主要内容。介绍这一章应掌握的概念、技术和相关算法，并点出这一章的重点；（2）例题解析。精选了一些典型题目，给出了解题思路、解题方法和要注意的问题，题目不多，但具有代表性；（3）习题。要学好编译的方法，就必须亲自动手做题目，这里给出了一些与例题相对应的作业；（4）习题参考答案。对习题中的每一道题目都给出了参考答案，大部分题目还给出了解题过程。

本书还给出了一个简单的程序设计语言（S 语言）的编译程序，将其作为实验。该实验将编译程序划分为三遍，分别实现词法分析、语法分析并生成中间代码、生成目标代码。书的最后附上了 C 语言程序清单。

本书实验程序由戴跃发、刘继臣、王继刚等同学编程、调试，在此表示衷心感谢。

由于作者水平有限，书中不妥之处，敬请读者批评指正。

编 者

目 录

编者的话

前言

第 1 章 编译概述	1
1.1 主要内容	1
1.1.1 计算机的语言层次	1
1.1.2 翻译程序	1
1.1.3 编译的阶段	1
1.1.4 编译的遍(趟)	2
1.1.5 本章重点	3
1.2 例题解析	3
1.3 习题	4
1.4 习题参考答案	5
第 2 章 文法和语言	6
2.1 主要内容	6
2.1.1 基础概念	6
2.1.2 文法的形式定义及相关概念	6
2.1.3 文法、语言的分类	8
2.1.4 本章重点	8
2.2 例题解析	9
2.3 习题	13
2.4 习题参考答案	14
第 3 章 词法分析	18
3.1 主要内容	18
3.1.1 词法分析的任务	18
3.1.2 单词的种类与机内表示方法	18
3.1.3 单词的描述	19
3.1.4 正规文法、正规式、NFA、DFA 之间的等价变换	20
3.1.5 词法分析器的设计	22
3.1.6 词法分析器的设计技术	23
3.1.7 本章重点	23
3.2 例题解析	23
3.3 习题	33
3.4 习题参考答案	36
第 4 章 语法分析	45
4.1 主要内容	45

4.1.1	语法分析的任务	45
4.1.2	自顶向下分析法	45
4.1.3	自底向上分析法	48
4.1.4	本章重点	58
4.2	例题解析	59
4.3	习题	73
4.4	习题参考答案	78
第 5 章	语法制导翻译技术	102
5.1	主要内容	102
5.1.1	语法制导翻译的任务	102
5.1.2	语法制导翻译的基本思想	102
5.1.3	中间代码的形式	102
5.1.4	属性文法	103
5.1.5	各种典型语法成分的翻译要点	104
5.1.6	本章重点	108
5.2	例题解析	108
5.3	习题	116
5.4	习题参考答案	117
第 6 章	符号表	122
6.1	主要内容	122
6.1.1	符号表的作用	122
6.1.2	符号表的建立	122
6.1.3	符号表的内容	122
6.1.4	符号表的组织	122
6.1.5	符号表的管理	125
6.1.6	本章重点	125
6.2	例题解析	125
6.3	习题	128
6.4	习题参考答案	129
第 7 章	运行时存储空间组织	131
7.1	主要内容	131
7.1.1	运行时存储器的划分	131
7.1.2	数据空间的分配策略	131
7.1.3	简单程序设计语言的栈式存储分配	132
7.1.4	嵌套过程语言的栈式实现	132
7.1.5	过程调用、进入、返回	133
7.1.6	本章重点	135
7.2	例题解析	135
7.3	习题	139

7.4	习题参考答案	141
第 8 章	代码优化	146
8.1	主要内容	146
8.1.1	局部优化	146
8.1.2	循环优化	149
8.1.3	本章重点	149
8.2	例题解析	149
8.3	习题	156
8.4	习题参考答案	158
第 9 章	目标代码生成	163
9.1	主要内容	163
9.1.1	目标代码	163
9.1.2	目标机器模型	163
9.1.3	一个简单的代码生成器	164
9.1.4	本章重点	166
9.2	例题解析	166
9.3	习题	168
9.4	习题参考答案	169
附录		172
附录 A	S 语言说明	172
A.1	字符集的定义	172
A.2	单词集的定义	172
A.3	数据类型定义	172
A.4	表达式定义	172
A.5	语句定义	173
A.6	程序定义	173
A.7	源程序书写格式的规定	173
附录 B	S 语言词法分析程序	174
B.1	任务	174
B.2	要求	174
B.3	数据结构	175
B.4	程序参考结构及模块说明	176
附录 C	S 语言语法/语义分析程序	179
C.1	任务	179
C.2	要求	179
C.3	数据结构	180
C.4	程序参考结构及模块说明	180
附录 D	S 语言目标代码生成程序	192
D.1	任务	192

D.2 要求	192
D.3 数据结构	193
D.4 程序参考结构	193
D.5 寄存器分配策略	194
D.6 代码生成器的模块结构及说明	194
附录 E S 语言编译程序清单	196
参考文献	236

第1章 编译概述

1.1 主要内容

1.1.1 计算机的语言层次

计算机的语言分三个层次：机器语言、汇编语言、高级语言。

机器语言不需要翻译就可直接执行，汇编语言和高级语言都必须经过翻译才能被计算机识别，这一过程由翻译程序完成。

1.1.2 翻译程序

将源语言程序转换成另一种语言程序的程序通称为翻译程序。

从汇编语言到机器语言的翻译程序称汇编程序，逆向的转换程序称为反汇编程序。

将一种高级语言程序翻译成另一种高级语言程序的程序称为转换程序。

从高级语言到汇编语言或机器语言的翻译程序称为编译程序或解释程序。

1.1.3 编译的阶段

编译程序的工作一般分为以下五个阶段：

1. 词法分析

此阶段的主要任务是从输入的源程序进行扫描，识别出一个个单词符号，如标识符、各类常数、关键字、运算符等，并把它们表示成机内单词形式。

词法分析是根据程序设计语言的构词规则进行分析的。

完成词法分析的程序称为词法分析程序，也称扫描器。

2. 语法分析

此阶段的主要任务是“组词成句”，即根据词法分析得到的单词串分析出源程序的各种语法单位，如表达式、各类语句乃至程序，并指出其中的语法错误。

语法分析是根据语言的语法规则进行分析的。

完成语法分析的程序称为语法分析程序或语法分析器。

3. 语义分析与中间代码生成

此阶段的任务是对语法分析得到的各种语法单位分析出其含义，并进行初步的翻译，形成中间代码程序。

翻译的依据是语言的语义规则。

中间代码独立于具体硬件，将其转换成机器指令较为容易。中间代码的形式有三元式、四元式、逆波兰式等。

完成此阶段任务的程序称为中间代码生成程序或中间代码生成器。

4. 代码优化

此阶段的任务是对中间代码进行等价的加工变换，以提高运行效率，节省存储空间。

代码优化的依据是程序的等价变换原则。

代码优化的方法有局部优化、循环优化和全局优化等。

完成此阶段任务的程序称为代码优化程序或代码优化器。

5. 目标代码生成

此阶段的任务是把中间代码程序变换成特定机器的绝对指令程序或可重新定位的指令代码或汇编指令代码，从而实现最终翻译任务。

完成此阶段任务的程序称为代码生成程序或代码生成器。

另外，在以上各阶段中都要涉及到表格管理和错误处理。表格用于保留编译过程中的各种信息。一个编译程序在编译过程中应尽量找出源程序中的错误，并向用户提供更多、更准确的错误信息。

在这五个阶段中，词法、语法、语义分析是对源程序的分析，而代码优化和目标代码生成则是对它的综合。

值得注意的是，以上五个阶段的划分是典型的编译程序的结构，不同的编译程序其结构不一定相同，也不一定五个阶段都有，如中间代码的生成及代码优化阶段就可以没有，五个阶段的顺序也仅为编译程序的逻辑顺序，并不代表时间上的执行顺序。编译模型如图 1-1 所示。

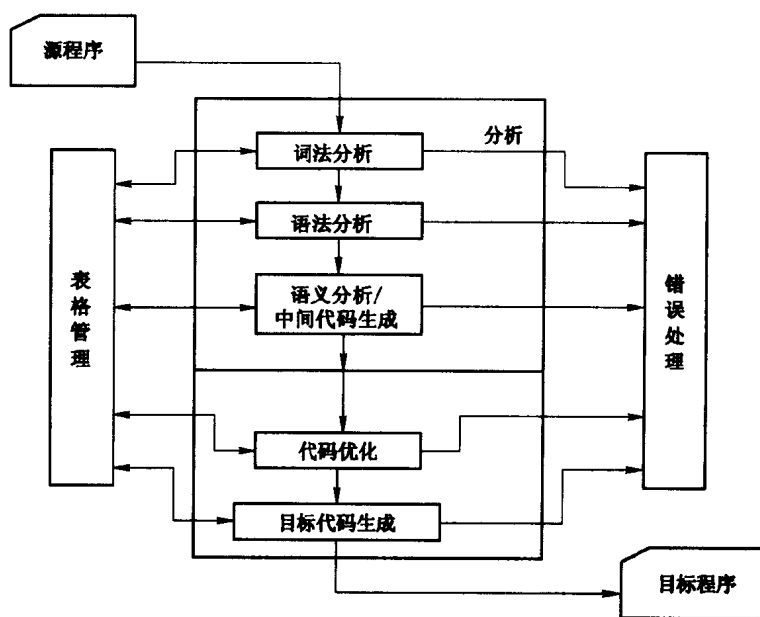


图 1-1 编译模型

1.1.4 编译的遍（趟）

“一遍”或“一趟”是指编译程序在编译时把源程序或源程序的等价物（如中间代码程序）从头到尾扫描一遍并转换成另一种等价物的全过程。

编译程序分单遍扫描的编译程序和多遍扫描的编译程序。在多遍扫描的编译程序中，每遍完成编译的一个阶段或几个阶段的工作。

1.1.5 本章重点

- 编译程序与解释程序的区别。
- 编译的阶段划分及各阶段的任务。
- “遍（趟）”的概念。

1.2 例题解析

例1 编译程序与解释程序的主要区别是什么？

答：编译程序与解释程序的主要区别是：

编译程序对全部源程序进行翻译，生成一个完整的目标代码程序并予以保存，然后再执行目标代码程序，即先翻译后执行，而且目标代码程序可以反复执行。

解释程序则对源程序进行逐句翻译执行，即读入一条高级语言程序中的语句，就将其翻译成一段机器代码指令并立即执行，然后再读入下一条语句，这样边翻译边执行，不生成完整的目标代码程序，翻译出的机器指令也不保存，故若要重新执行，就必须重新翻译源程序。

就速度而言，在编译方式下源程序的执行比解释方式下快，但解释方式更便于程序的调试。

例2 编译的五个阶段是否都是必须的？

答：编译的五个阶段中，词法分析、语法分析、语义分析和代码生成是必须完成的，而中间代码生成是为了简化目标代码生成、便于代码优化和代码移植，代码优化则是为了提高目标代码的质量，这两个阶段都不是必须的。

例3 编译程序采用多遍扫描的方式比采用单遍扫描的方式要好吗？

答：此说法不正确。多遍扫描常可节省内存空间、提高目标程序质量、加快编译程序的生成速度，但其各遍之间的输入输出工作则会延长编译时间、降低编译效率，所以不能说多遍扫描一定优于单遍扫描。具体的某个编译程序采用多遍还是单遍扫描，应根据被编译语言的特性、机器的状况、对编译程序生成速度的要求以及对目标程序的运行效率等因素来定。

例如，被编译的程序设计语言允许标识符先使用后说明，则其编译程序就必须多遍扫描。

例4 什么叫“自编译”？

答：编译程序可以用“自编译”的方式产生。方法是：先对语言的核心部分构造一个很小的编译程序，该编译程序可用低级语言实现，然后再用它来构造能编译语言更多成分的编译程序，如此不断扩展下去，直到形成整个语言的编译程序。这种“自编译”技术也叫“自展”技术。

例5 编译程序的“前端”与“后端”是如何划分的？这样划分有什么好处？

答：编译程序常构造“前端+后端”的形式，“前端”主要由编译程序中与源语言有关而与目标机器无关的部分构成，包括词法分析、语法分析、语义分析与中间代码生成，以及部分与目标机器无关的代码优化工作。“后端”则由编译程序中与目标机器有关的部分构成，包括与目标机器有关的代码优化、目标代码生成等，“后端”不依赖于源语言而仅与中间语

言有关。

为使一种语言在不同的机器上运行就需要为每个机器编写不同的编译程序,采用前、后端方式,可使编程工作量大大减少:选用同一种中间语言,编写一个前端程序,再为每个机器编写不同的后端程序,这样,同样的前端结合不同的后端即形成该语言的多个编译程序。

同样,不同的程序设计语言在同一机器上运行需要为每种语言编写编译程序,采用前、后端方式也可使工作量大大减少:选用同一种中间语言,对这种语言编写该机器的后端程序,再对每一种语言编写不同的前端程序,这样,不同的前端程序结合同样的后端程序即形成不同语言的编译程序。

1.3 习题

1. 填空

- (1) 编译程序是指能将 a) 程序翻译成 b) 程序的程序。
- (2) 若编译程序生成的目标程序是机器代码程序,则源程序的执行分两个阶段: a) 和 b) ; 若编译程序生成的目标程序是汇编语言程序,则源程序的执行分三个阶段 c) 、 d) 和 e) 。
- (3) 要为某种语言构造一个编译程序,必须弄清下述三方面的内容: a) 、 b) 和 c) 。
- (4) “用高级语言书写的源程序都必须经过编译,产生目标代码后才能投入运行。”这种说法正确吗? a) 。
- (5) 编译方式与解释方式的根本区别在于 a) 。
- (6) 一个编译程序中,不仅包括 a) 、 b) 、 c) 、 d) 、 e) 等五个部分,还应包括 f) 和 g) 。
- (7) 在编译的五个部分中, a) 和 b) 不是每个编译程序都必须的。

2. 选择

- (1) 编译程序是一种常用的_____软件。
a) 应用 b) 系统 c) 实时系统 d) 分布式系统
- (2) 编译程序的大多数时间花在_____上。
a) 词法分析 b) 语法分析 c) 出错处理 d) 管理表格
- (3) 将编译程序分成若干“遍”将_____。
a) 提高编译程序的执行效率。
b) 使编译程序的结构更加清晰。
c) 充分利用内存空间,提高机器的执行效率。
- (4) 编译程序各阶段都涉及到的工作有_____。
a) 词法分析 b) 语法分析 c) 表格管理 d) 语义分析
- (5) 编译程序生成的目标代码程序_____是可执行程序。
a) 一定 b) 不一定
- (6) 词法分析的主要功能是_____。
a) 识别字符串 b) 识别语句 c) 识别单词 d) 识别标识符

1.4 习题参考答案

1. 填空

- (1) a) 源语言 b) 目标语言
- (2) a) 编译阶段 b) 运行阶段 c) 编译阶段 d) 汇编阶段 e) 运行阶段
- (3) a) 源语言 b) 目标语言 c) 编译方法
- (4) a) 不正确
- (5) a) 是否生成目标代码
- (6) a) 词法分析 b) 语法分析 c) 语义分析和中间代码生成
d) 代码优化 e) 目标代码生成 f) 表格管理
g) 出错处理
- (7) a) 中间代码生成 b) 代码优化

2. 选择

- (1) b)
- (2) d)
- (3) b)
- (4) c)
- (5) b)
- (6) c)

第2章 文法和语言

2.1 主要内容

2.1.1 基础概念

字母表：有穷非空字符集，常用字母 V 、 Σ 或其他大写字母表示。

字符串：字母表中的符号组成的有穷序列，简称串、行。不含任何符号的串称空串，用 ϵ 表示。

串的连接：串 x 、 y 的连接为 xy 。

字符串集合 A 、 B 的积： $A \cdot B = \{xy \mid x \in A \text{ 且 } y \in B\}$

字符串集合 A 的闭包： $A^* = A^0 \cup A^1 \cup A^2 \cup \dots \cup A^n \dots$

$$A^0 = \{\epsilon\} \quad A^i = A^{i-1} \cdot A$$

字符串集合 A 的正闭包： $A^+ = A^1 \cup A^2 \cup \dots \cup A^n \dots = A^* - \{\epsilon\}$

（形式）语言：一个字母表上按某种规则构成的所有串的集合。

递归可枚举的语言：如果一个语言中的所有句子能由某个过程产生或识别，则该语言是递归可枚举的。

递归语言：如果一个语言中的所有句子能由某个算法产生或识别，则该语言是递归的。

形式语言理论：对形式语言的表示法、结构及特性进行研究的理论。

2.1.2 文法的形式定义及相关概念

1. 文法的形式定义

文法是描述语言的工具，其定义为：

文法为一个四元组 $G = (V_N, V_T, P, S)$ ，

其中， V_N ——非终结符集；

V_T ——终结符集；

P ——产生式集；

S ——开始符号。

2. 直接推导、推导和最左（最右）推导

如 $\alpha \rightarrow \beta$ 是文法 $G = (V_N, V_T, P, S)$ 的规则（产生式）， γ 、 δ 是 V^* 中的任意符号，若有符号串 v 、 w 满足

$$v = \gamma\alpha\delta, \quad w = \gamma\beta\delta$$

则说 v （应用规则 $\alpha \rightarrow \beta$ ）直接产生 w ，或说 w 是 v 的直接推导（即用产生式的右部替换该产生式的左部）。记作 $v \Rightarrow w$ 。

若存在直接推导序列

$$v = w_0 \Rightarrow w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow w_n \Rightarrow w \quad (n > 0)$$

则称 w 是 v 的推导, 推导长度为 n , 记作: $v \xRightarrow{*} w$ 。

如果有 $v \xRightarrow{*} w$ 或 $v = w$, 则记作 $v \xRightarrow{*} w$ 。

若在推导的任何一步 $v \Rightarrow w$ 中, 都是对 v 中最左边(右边)的非终结符进行替换, 这样的推导称最左(右)推导。

最右推导也称规范推导。

3. 归约

推导的逆过程叫作归约。规范推导的逆过程叫作规范归约。

4. 句型 and 句子

由文法的开始符号 S 推导出的符号串称为文法的句型。由规范推导推出的句型称为规范句型。

若一个句型仅含终结符, 则称该句型为句子。

5. 短语、直接短语和句柄

设 S 是文法 G 的开始符号, $x\beta y$ 是 G 的句型, 即 $S \xRightarrow{*} x\beta y$, 如果有

$$S \xRightarrow{*} xAy \text{ 且 } A \xRightarrow{*} \beta$$

则称 β 是句型 $x\beta y$ 相对于非终结符 A 的短语。

若 β 是句型 $x\beta y$ 相对于非终结符 A 的短语, 且 $A \rightarrow \beta$ 是文法 G 的产生式, 则称 β 是句型 $x\beta y$ 相对于非终结符 A 的直接短语。

一个句型的最左直接短语叫作该句型的句柄。

6. 语言

一个文法 G 的所有句子的集合叫作该文法所描述的语言 $L(G)$, 即

$$L(G) = \{w \mid w \in V_T^*, \text{ 且 } S \xRightarrow{*} w\}$$

7. 语法树

语法树是一张表示句型推导的图, 它是一棵倒立的树, 根在上, 叶在下, 结点表示一个文法符号(终结符或非终结符), 根结点为文法的开始符号。结点间的父子关系表示一种推导关系。

从语法树上看, 以下说法成立:

1) 文法的一个句型必有它对应的语法树, 或者说, 一棵语法树的叶结点从左到右排列成的符号串就是文法的一个句型。

2) 语法树的一棵子树的叶结点从左到右排列成的符号串, 就是句型中相对于该子树根结点的短语。

3) 语法树中有且只有两层的子树的叶结点从左到右排列成的符号串就是句型的直接短语。

4) 语法树中最左边、最下面的有且只有两层的子树的叶结点, 从左到右排列成的符号串, 就是句型的句柄(即最左直接短语)。