新编高等院校计算机科学与技术规划教材

BIANYI YUANLI YU JISHU
LIANXI JIEDA YU SHIYAN ZHIDAO

编译原理与技术练习解答与实验指导

李劲华　赵　赟　编著

北京邮电大学出版社
www.buptpress.com

新编高等院校计算机科学与技术规划教材

编译原理与技术练习解答与实验指导

李劲华　赵　赟　编著

北京邮电大学出版社
·北京·

内 容 简 介

本书是《编译原理与技术》教材的配套参考书，其内容、知识点和题目都是根据相关课程的范围和难度组织和设计的。

全书共分为2个部分：第1部分按照课本《编译原理与技术》的章节，首先简要地总结每章的知识要点，然后分析典型题目的解题思路，并给出题解规范，最后对教材中每道练习给出参考答案与题解分析。第2部分是实验指导，包括对编译器中部分功能的手工编程实现，以及编译工具LEX和YACC的使用。

本书针对性强、选题范围广、难易适当，不仅可以作为计算机及相关专业编译课程的教学、学习和实验参考书，而且对编译课程的相关考试也具有参考价值。

图书在版编目(CIP)数据

编译原理与技术练习解答与实验指导/李劲华，赵赟编著. —北京：北京邮电大学出版社，2008
ISBN 978-7-5635-1837-1

Ⅰ. 编… Ⅱ. ①李…②赵… 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(2008)第139236号

书　　名：编译原理与技术练习解答与实验指导
作　　者：李劲华　赵　赟
责任编辑：艾莉莎
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(邮编：100876)
发 行 部：电话：010-62282185　传真：010-62283578
E-mail：publish@bupt.edu.cn
经　　销：各地新华书店
印　　刷：北京市梦宇印务有限公司
开　　本：787 mm×1 092 mm　1/16
印　　张：16.5
字　　数：409千字
印　　数：1—3 000册
版　　次：2008年10月第1版　2008年10月第1次印刷

ISBN 978-7-5635-1837-1　　定　价：28.00元

前　言

编译原理与技术是计算机专业的一门核心课程，在计算机的本科教育中占有十分重要的地位。该课程的特点是理论概念十分抽象，符号处理算法诸多。对学生的理解能力、抽象能力、分析能力、动手能力以及综合运用知识解决问题的能力这些方面都提出了较高的要求。对于这门专业课，大多数编译课本都在例题方面十分“吝啬”，而且，国内外编译教材中的例子基本雷同。这些都使得学生在学习这门课时普遍感到内容抽象，难以掌握，对练习题无从下手。

另外，目前大多数编译教材都没有配备相应的实验指导，仅提供了上机练习题目；有些教材提供的实验内容过于庞大复杂，更适合作为课程设计采用，难以作为实验题目使用。任课教师往往需要自己设计和编写实验指导材料，使得教与学都不方便。

编者认为，阅读模仿、动手练习和上机实践是学习和掌握计算机知识的重要途径。不同类型的练习有助于学生从不同的角度和层次理解概念、原理、算法和系统。为此，我们编写了本书，作为《编译原理与技术》课本的配套参考书。

全书共分为2个部分：第1部分按照课本《编译原理与技术》（本书中简称“教材”）的章节，首先简要列举了每章的知识点、难点和重点，然后对精选和自编的一些典型题目给出了不同的解题思路、详细步骤以及解答规范，最后，对教材中每章后面的练习给出了参考答案，还对其中较难的题目进行了解题分析。精选的例题包括近年来国内重点院校研究生入学考题。通过这些练习，有助于学生理解概念，掌握枯燥的理论和抽象的算法，开阔解题思路，进而掌握编译原理与技术的基础知识。

第2部分的实验指导包含5个实验题目，由简到难，题目即有编译功能的手工代码编写，也有编译工具的使用，还有阅读与理解有关的编译程序，以便满足不同层次的教学需求。任课教师可以根据实际需要和兴趣，选择或改编实验题

目。为了使读者能够顺利完成实验，本书不仅给出了阅读文献和实验指南，还有上机调试过的实验代码(请任课教师向作者免费索取)。

本书由李劲华执笔完成，赵赟编写了实验指导和部分上机题目，丁洁玉、逄瑞娟和朱梅霞解答了部分练习，在此表示感谢。

由于编者水平有限，书中难免存在一些疏误和不妥之处，恳请广大读者批评、指正。

编　者

2008 年 10 月

目　录

第1章 概 论

1.1 基本知识总结

本章概述编译程序与编译过程，主要知识点如下。

1. 编译程序的基本概念，

(1) 编译程序定义；

(2) 编译程序与解释程序；

(3) 编译程序的类型；

(4) 编译的前端、后端与“趟”的概念。

2. 编译的主要过程、编译程序的参考模型及各个模块的基本功能。

3. 编译程序的自展、移植和辅助工具开发技术。

4. 编译程序在计算机系统中的作用和关系。

重点：有关编译程序的概念、基本组成及编译过程。

难点：全面理解编译程序、“趟”的概念及移植技术 T 形图的应用。

1.2 典型例题解析

解释图 1-1 所示的 T 形转换图的含义。

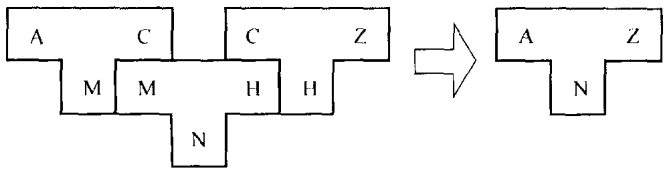

图 1-1 T 形图的例子

【解答】

在机器 M 上运行的编译器把源语言 A 翻译成语言 C，在机器 N 上运行的编译器把机器 M 的语言翻译成语言 H，在机器 H 上运行的编译器则把语言 C 翻译成语言 Z。这样，就可以在机器 N 运行编译器，把语言 A 翻译成语言 Z。

1.3 练习与参考答案

1. 为什么高级程序语言需要编译程序？

【解答】

编译程序是计算机系统经典、核心的系统软件。按照现在的计算机体系结构和组成原理以及软件开发的理论和实践，高级程序语言仍将是开发计算机应用系统的关键技术，与之不可分离的是高级程序语言的编译程序。用高级编程语言（如 Fortran，Pascal，Ada，Smalltalk，C，C＋＋，C＃和 Java）编写程序方便而且效率高，但是，计算机需要把高级编程语言的程序翻译成机器语言代码或汇编程序才能运行。而编译程序正好实现上述功能，所以高级程序语言需要编译程序。

2. 解释下列术语：源程序，目标程序，翻译程序，编译程序，解释程序。

【解答】

源程序：用高级语言编写的程序。

目标程序：机器语言或汇编语言的程序。

翻译程序：将源语言程序翻译成目标语言程序的程序。

编译程序：源程序是用高级语言编写的，经翻译后生成目标程序，使之可以在计算机上运行，这样的翻译程序称为编译程序。

解释程序：在翻译过程中不产生目标程序，而是一边翻译一边运行源程序，即解释程序的同时处理源程序和源程序要加工的数据。

3. 简单叙述编译程序的主要工作过程。

【解答】

把计算机高级编程语言翻译成计算机可以执行的代码的工作包括一系列的活动和任务，是一个复杂的完整过程。计算机程序的编译过程类似，一般划分为 5 个阶段：词法分析、语法分析、语义分析及中间代码生成、代码优化、目标代码生成。

词法分析的任务是逐步地扫描和分解构成源程序的字符串，识别出一个一个的单词符号。词法分析的工作主要包括识别出程序中的单词符号，在编译程序符号表中查找并登记单词符号及其信息，如单词符号的类型、内部表示、数值等。

语法分析的任务是在词法分析基础上，根据语言的语法规则把单词符号串分解成各类语法单元，例如“短语”、“子句”、“语句”、“程序段”、“函数”和“程序”等。

语义分析的任务是检查程序语义的正确性，解释程序结构的含义。语义分析完成之后，编译程序通常就依据语言的语义规则，利用语法制导技术把源程序翻译成某种中间代码。

代码优化的主要任务是对前一阶段产生的中间代码进行等价变换，以便产生速度快、空间小的目标代码。

目标代码生成的主要任务是把（经过优化处理的）中间代码翻译成特定的机器指令或汇编程序。

4. 编译程序的典型体系结构包括哪些构件，主要关系如何，请用辅助图示意。

【解答】

编译程序的典型体系结构包括词法分析器、语法分析器、语义分析与中间代码生成器、

优化器、目标代码生成器以及错误处理和符号管理模块。它们的关系如图 1-2 所示。

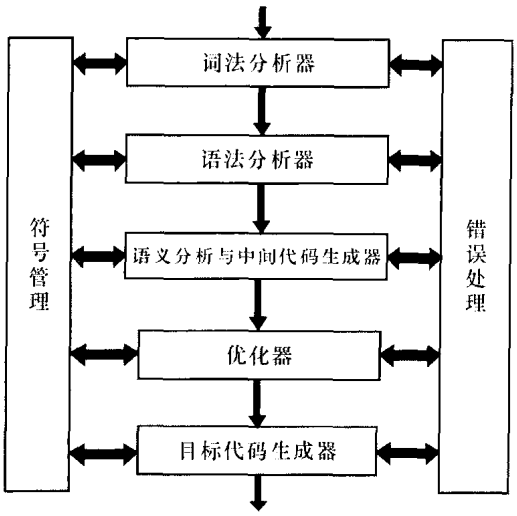

图 1-2　编译程序的典型体系结构

5. 编译程序的开发有哪些途径？了解你熟悉的高级编程语言编译程序的开发方式。

【解答】

除了手工从头到尾编写编译程序以外，还有下列 3 种常用的开发方法。

(1) 编译程序的自展技术

对于具有自编译性的高级编程语言，可以运用自展技术来构造编译程序。

(2) 编译程序的移植技术

编译程序可以采用移植技术产生，即用宿主计算机上的高级语言编写一个能在另外类型目标机上运行的编译程序。

(3) 编译工具

为了缩短编译程序的开发时间，保证编译程序的正确性，已经研究和开发了编译程序的自动生成工具。其中词法分析生成器和语法分析生成器最为成熟，获得了广泛的研究和应用。本书第 2 章介绍的 LEX 是一个通用的词法分析生成器，它输入的是描述单词结构的正规式，输出的就是词法分析程序。另外一个经典的编译工具是语法分析生成器 YACC(Yet Another Compiler Compiler)，它接受 LALR(1)语法，生成一个相应的 LALR(1)语法分析器，而且可以和 LEX 连接使用。

已经广泛使用的 Java 语言就有许多开发方法，例如，JavaCC(Java Compiler Compiler)是 Sun Microsystem 公司提供的一个 Java 语言的词法和语法分析器自动生成器，它产生的是递归下降分析器，是 100%纯 Java 代码，无须修改就可以在各种 Java 兼容的平台上运行。有关该工具的介绍和使用可以在下面网站中找到：https://javacc.dev.java.net/。

Jikespg 是 Jikes parser generator 的缩写，是 IBM 公司开发的一个 LALR 语法分析器的自动生成器，已经应用在 IBM 的 Java 编译器 Jikes 中，既可以产生 C++代码，也可以产生 Java 代码。Jikes 是一个开源的 Java 语言编译器，本身是用 C++编写实现。

6. 运用编译技术的软件开发和维护工具有许多类，简单叙述每一类的主要用途。

【解答】

编译技术和方法也已经应用在其他软件开发工具和环境当中，目前的集成化软件开发环境中都包括编译程序、面向语言的编辑程序、程序格式化输出等工具。下面是其他一些典型的工具。

(1) 语法制导编辑器

这类工具运用程序语言的语法知识，在用户编写程序的时候按照词法和语法分析的信息提供智能的帮助，包括自动地提供关键字及其匹配的关键字、左右括号的配对、对象的属性和操作等。例如，在C语言环境下用户输入了关键字do以后，语法制导编辑器就自动地输入do-while的结构，包括关键字while；在Java语言的环境下，用户引用对象student时，在输入了student之后系统就提供可以选择的属性或操作。这样就可以使编程人员专注于算法的设计和实现，不用记忆语言的细节。最典型的通用语法制导编辑器是自由软件EMACS，只要给它设置某个语言的语法结构，EMACS就可以作为该语言的智能正文编辑器。

(2) 程序调试工具

编译程序只可以发现静态的语法和语义错误，要进一步了解程序的动态错误，看程序的执行结果与编程人员的设想是否一致，程序的执行是否实现了预计的算法和功能，就需要程序调试工具。调试的目的是根据程序的异常，追踪和确定错误在程序中的具体位置，并且修改程序、消除错误。例如，调试工具可以根据语义分析后生成的中间代码，在虚拟机中一步一步(单步)地执行程序，观察程序的状态或程序中特定变量值的变化。

(3) 程序测试工具

程序测试是为了发现错误而执行程序的过程，基于编译技术的测试辅助工具可以分为静态分析器和动态测试工具。静态分析器就是采用编译中的全局控制流和数据流分析技术，无须运行源程序来进行分析，以发现诸如“对变量未赋值就引用”，或“赋值后没有引用”，或“多余的源代码”等错误。动态测试工具是在源程序的适当位置插入某些信息，用测试数据运行程序并记录程序运行时的实际路径；或者输入符号(而不是具体的数值)，执行符号运算，把运行结果与期望的结果进行比较，从而发现程序运行时的错误。

(4) 程序理解工具

在软件测试、软件维护以及软件的再向工程和逆向工程等工作中，需要人们理解和分析程序，得到需要的软件信息，这类工具称为程序理解工具。利用编译技术的语法分析、语义分析和流分析，可以得到程序中各类名字(例如变量、函数、类)的定义、使用以及交叉引用关系。程序切片技术可以根据感兴趣的一组程序变量，静态地分析程序，抽取并显示程序中与这些变量相关的语句，缩小了程序规模，方便了程序的阅读、理解和测试。

7. 了解一个真实编译系统的组成和基本功能。

【解答】

本题的目的是，要求读者对实际使用的编译系统从本课程的知识出发，在理论上加深认识，以便更好地运用和掌握编译系统及其理论。下面简单描述一个Java语言的编译系统GJC。

GJC 是 Sun Hotspot J2SE 中的 Java 编译器，可以在 Sun 公司的网站上通过授权获得 Hotspot 虚拟环境的全部代码，GJC 是 J2SE 中的一部分。图 1-3 示意了 GJC 的体系结构。

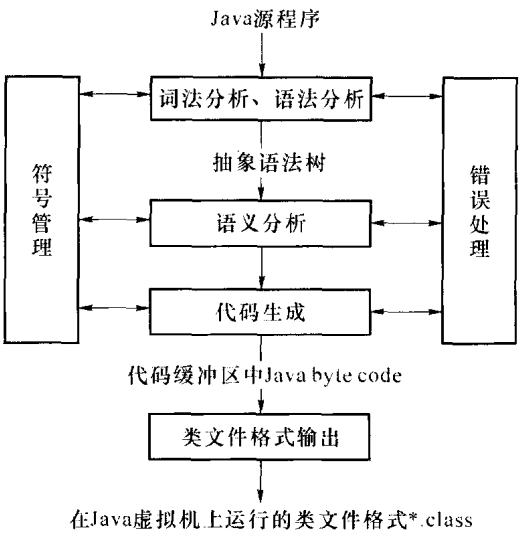

图 1-3　GJC 编译程序的典型构成

词法分析和语法分析在编译器的一遍扫描中完成。语法分析程序 Parser 调用词法分析程序 Scanner，得到一个单词符号序列，对其分析并建立抽象语法树作为输出的结果。GJC 采用了递归下降的 LL 分析方法。语义分析程序 Env 作为编译器中独立的一趟，在遍历抽象语法树的过程中执行语义分析，包括类的完整性检查、确定类的参数以及类中定义符号的范围。作为编译器中独立的一趟，代码生成器 Gen 遍历抽象语法树，为每一种语言结构产生并输出中间代码 Java byte code。符号表(Symtab)管理部分完成 Java 程序中各种类符号的维护，用在语义分析和代码生成的功能部分。错误处理程序主要是诊查和报告 Java 程序中语法和语义的错误，功能分布在 GJC 的各处。

8. 简单说明学习编译程序的意义和作用。

【解答】

编译程序构造的原理和技术一直属于最近公布的 ACM/IEEE Computing Curriculum 2004 的核心知识域，是计算机科学必备的专业基础知识，它所建立的理论、技术和方法值得深入研究和学习。

第一，编译构造正确地建立了研究的问题领域和研究方式：分析输入内容、构造一个语义表示并合成输出。对于不同的源语言，可以用一个单一的语义表示，产生出不同的目标语言，运行在不同的环境中。而且，编译构造可以划分成便于控制和管理的阶段，每个阶段的工作结果正好对应编译程序的子系统或模块。编译程序的这种分析-合成模式以及解释程序的解释模式已经成为软件开发领域最成功的设计模式和软件架构，在软件开发中获得了广泛的应用。

第二，针对编译程序构造的某些部分已经开发了标准的形式化技术，依据它们研制的编译程序生成工具，极大地减轻了编译程序的构造工作，使得编译程序成为计算机系统最可靠的基础软件之一。

第三，编译程序包含许多普遍使用的数据结构和算法，例如散列法（哈希算法）、栈机制、堆机制、垃圾收集、集合算法、表驱动算法、图算法等。尽管其中的每一种都可以独立地学习，但是，在诸如编译程序这样一个有意义的环境中学习将更有教育意义。

第四，编译程序的许多构造技术已经得到了广泛的应用。许多应用程序非常接近于编译程序，已经采用了编译构造的部分技术，例如读入格式化的数据、文件转换问题等。

第五，学习编译原理和技术还有助于理解程序设计语言，编写优秀的软件。

9. 如果机器 H 上有 2 个编译：一个把语言 A 翻译成语言 B，另一个把 B 翻译成 C，那么可以把第一个编译的输出作为第二个编译的输入，结果在同一类机器上得到从 A 到 C 的编译。请用 T 形图示意过程和结果。

【解答】

图 1-4 显示了在机器 H 上把源语言 A 翻译成目标语言 C 的编译程序的构造过程和结果。

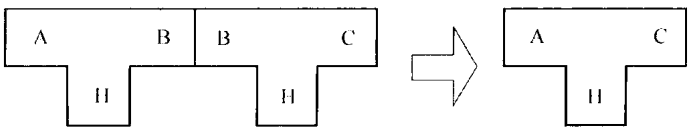

图 1-4　在机器 H 上从 A 到 C 的编译程序的构造过程

第2章 词法分析

2.1 基本知识总结

本章介绍词法分析的设计原理，主要知识点如下。

1. 设计词法分析器应考虑的一些问题，

(1) 词法分析器的功能、输入和输出，识别单词的表示；

(2) 词法分析器的两种实现模式；

(3) 词法分析器与符号表的关系；

(4) 输入的预处理；

(5) 超前搜索与最长匹配。

2. 基于状态转换图的词法分析器的手工实现。

3. 符号的基本概念：字母，字母表，符号串，闭包。

4. 正规表达式与语言的正规集。

5. 有限状态自动机 FA，

(1) 确定有限状态机 DFA 与非确定有限状态机 NFA 的定义，识别语言的含义；

(2) DFA 和 NFA 的等价性，ε-闭包的计算，最小子集法；

(3) 可区分状态，DFA 的最小化算法；

(4) 正规式向 FA 的转换。

6. 词法分析器的自动生成工具 LEX。

重点：词法分析器的功能，超前搜索与最长匹配的概念，符号串的运算，状态转换图的实现，正规表达式的概念和应用，有限状态机的概念和应用，ε-闭包运算，NFA 向 DFA 转换的算法，正规式向 FA 的转换规则，DFA 的最小化算法。

难点：正规表达式的等价转换，使用正规表达式描述语言，使用 FA 描述语言，ε-闭包运算，可识别状态的概念，正规式向 FA 的转换。

2.2 典型例题解析

1. 理解下列正规表达式，说明它们所表示的语言。

(1) $(aa|bb)^{+}$。

(2) $(1|01)^{*}(0|\varepsilon)$。

(3) $1^{*}01^{*}0(1|0)^{*}$。

(4) $a(aa)^{*}bb(bb)^{*}(cc)^{*}c$。

【分析】 这类题目的目的在于使读者理解正规表达式的含义，用自然语言或集合等数

学形式描述出来。不同的表达方式不尽相同，基本要求是表达出相应语言的主要特点。为了理解正规式，可以把它展开，从简单到复杂，归纳总结、精练表达。

【解答】

(1) 正则闭包的基础是一个或运算符号连接的2个子表达式，其中每个子正规式有2个相同的符号。所以，该正规表达式的语言是a和b都是成对出现的{a, b}上的符号串。或者该正规表达式的语言是{a, b}上由aa或bb组成的符号串。

(2) 该正规式可以分成2个连接的正规式$(1|01)^*$和$(0|\varepsilon)$，其中$(1|01)^*$表示的语言或者为空，或者长度至少是1、结尾是1，而且没有连续的0。而$(1|01)^*0$表示的语言长度至少是1、结尾是0、不含连续的0。所以，$(1|01)^*(0|\varepsilon)$表达的语言是长度或者为空(不含连续的0)，或者至少是1、不含连续0的0和1的串。简言之，该正规表达式的语言是不含连续0的{0,1}上的符号串。

(3) 如果$1^*01^*0(1|0)^*$中含闭包的3个子正规式都是空的话，那么该正规式的语言是00；如果这些子正规式不是空，那么，该正规式的语言就包含形如'…01…10…'的子串，其中两个0之间的1的个数是任意的。即不管哪个闭包取空值，$1^*01^*0(1|0)^*$都是至少有2个0。所以，该正规式的语言是：至少有2个0的{0,1}上的符号串。

(4) 逐个看由一种字母组成的子正规式的含义。$a(aa)^*=\{a, aaa, aaaaa, \cdots\}$，即含有奇数个a的串；$bb(bb)^*=\{bb, bbbb, bbbbbb, \cdots\}$是含有偶数个b的串，$(cc)^*c=\{c, ccc, ccccc, \cdots\}$是含有奇数个c的串。所以，该正规式的语言是奇数个a、偶数个b以及奇数个c连接而成的串。或者，该正规式所表示的语言是$\{a^{2n+1}b^{2k}c^{2m+1} \mid n\geqslant 0, k\geqslant 1, m\geqslant 0\}$。

2. 给出描述下列语言的正规表达式。

(1) $\{a^n b^m \mid n\geqslant 1, m\geqslant 1\}$。

(2) 在{0, 1}上不以0开头的、以11结尾的字符串集合。

(3) 最多只含2个a的{a, b}上的语言。

【解答】

(1) 该语言是任意个a后跟任意个b的符号串，产生该语言的正规式为a^+b^+。

(2) 根据题目分析，要求的正规式包含3个连接的子正规式：第一个正规式为1，最后一个正规式为11，它们之间可以是任意的0和1的子串，即$(1|0)^*$。所以，最终结果是$1(1|0)^*11$，再考虑一个特殊的符号串11，就得到最终的结果$11|1(1|0)^*11$。

(3) 分析1：本题可以分别构造不含a的子串、只含一个a的子串以及只含2个a的子串，然后把它们用"或"运算连接起来就得到最终的结果。不含a的子串是b^*，只含一个a的子串是b^*ab^*，只含2个a的子串是$b^*ab^*ab^*$。综合这3个子串，得到所要求的正规表达式为$b^*|b^*ab^*|b^*ab^*ab^*$。

分析2：含2个a的子串是$b^*ab^*ab^*$，若第一个或第二个a没有，则成为只含一个a的子串；若2个a都没有，则$b^*ab^*ab^*$就成为不含a的子串。有一个或没有a的正规式为$a|\varepsilon$。所以，本题要求的正规式就是$b^*(a|\varepsilon)b^*(a|\varepsilon)b^*$。它也可以从$b^*|b^*ab^*|b^*ab^*ab^*$化简得到。

3. 分别将下列非确定有限状态机确定化并最小化。

【分析】 非确定有限状态机确定化的基本方法是子集法，基本步骤是：首先确定起始状态及其闭包的状态子集作为新的起始状态，利用闭包运算和移动函数，不断生成新的状态子集和转换函数，并通过转换矩阵表示出来；对新的转换矩阵中的状态子集重新命名，可选地画出状态转换图。最小化就是合并无法区分的状态，并且合并相应的转换函数或转移，对新的转换矩阵中的状态重新命名，可选地画出状态转换图。

确定化和最小化是证明有限状态机的等价性的一个基本技术，也可以应用在正规表达式的等价性证明。

(1) 非确定有限状态机如图 2-1 所示。

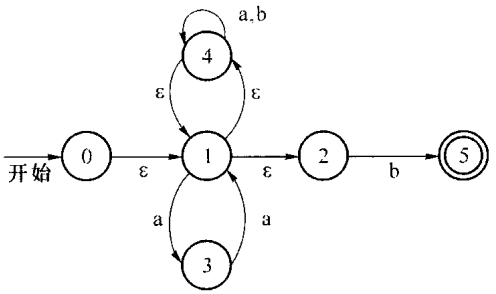

图 2-1 非确定有限状态机

【解答】

起始状态是 0，从它开始经过 ε 所能达到的状态集合——状态 0 的 ε 闭包是{0,1,2,4}，以它作为构造状态转换矩阵的起始状态。然后分别构造{0,1,2,4}经过 a 或 b 所能达到状态集合。{0,1,2,4}经过 a 到达的状态集合就是其中的每个状态在面临 a 时的转移状态，只有状态 1 经过 a 后到达状态 3，结果是{3}。然后对它计算 ε 闭包，得{3,1,2,4}。同样，对{0,1,2,4}找出经过 b 到达的状态集，然后再计算 ε 闭包，得{4,1,2,5}。

继续对{3,1,2,4}和{4,1,2,5}计算转移后的闭包。最终构造状态转换矩阵的过程如表 2-1 所示，其中包含状态 0 的状态子集是起始状态；含有状态 5 的状态子集是终结状态。重新命名后的转换矩阵如表 2-2 所示，其中状态 1 是起始状态；状态 3 是终结状态。

表 2-1 确定化的状态转换矩阵

	I_a	I_b
{0,1,2,4}	{3,4,1,2}	{4,1,2,5}
{3,4,1,2}	{3,4,1,2}	{4,1,2,5}
{4,1,2,5}	{3,4,1,2}	{4,1,2,5}

表 2-2 重新命名的状态转换矩阵

	I_a	I_b
1	2	3
2	2	3
3	2	3

对表 2-2 所示的 DFA 进行最小化：把状态划分成终结状态子集{3}和非终结状态子集{1,2}。由于 $\{1,2\}_a=\{2\}\subset\{1,2\}$，$\{1,2\}_b=\{3\}$，所以，状态 1 和状态 2 不可区分，故得到最小化的 DFA 如表 2-3 所示，其中 1 是起始状态；2 是终结状态。

表 2-3 最小化后的状态转换表

	I_a	I_b
1	1	2
2	1	2