EXPERT SYSTEMS

BASIS AND IMPLEMENTATION

蔡连成　滕健　张牧　编著

专家系统基础与实现

天津大学出版社

专家系统基础与实现

蔡连成　滕健　张牧　编著

天津大学出版社

内 容 提 要

专家系统是人工智能发展的主要分支之一，它在医学、化学、地质、教育、军事、生物等领域的应用已取得巨大的经济效益和社会影响.

本书是专家系统的入门书，内容包括：LISP 语言，专家系统原理，知识表示方法，不精确推理，面向对象的程序设计及黑板编程技术，专家系统工具及专家系统的开发方法及步骤.

LISP 语言部分主要介绍LISP 的基本数据结构、基本函数、递归、迭代及数据驱动程序设计等；专家系统原理及知识表示方法部分着重介绍专家系统的基本原理、结构、特点及几种常用的知识表示方法；不精确推理部分介绍几种常用的不精确推理技术；“面向对象的程序设计”和“黑板编程技术”为专家系统的整体设计提供了良好的方法；专家系统工具部分对专家系统工具进行了分类，简要介绍各类工具的功能及特点，并对目前较为流行的专家系统工具——PC Plus进行了详细介绍；最后以一个简单的例子介绍专家系统的开发过程及步骤. 概览全书，不仅使读者能够步入专家系统之门槛，而且能具有一定的自己动手开发专家系统的能力 .

全书介绍由浅入深，层次分明，实例充实. 可作为大专院校计算机或有关专业专家系统课程的教材及参考书，也可供从事专家系统设计及开发的工作人员自学参考.

专家系统基础与实现

蔡连成　滕健　张牧　编著

*

天津大学出版社出版

（天津大学内）

河北省永清县印刷厂印刷

新华书店天津发行所发行

*

开本：787×1092毫米 1/16 印张：13 字数：321千字

1990年12月第一版　1990年12月第一次印刷

印数：1—3000

ISBN 7-5618-0227-7

TP·29

·定价：2.65元

序

我从事了40年自动控制和10年系统工程方面的教学和科学研究工作，深知要研究一个系统，必须搞清系统内在的与外界的关系，也就是必须构成一个可以描述系统内在的和与外界的各种因素之间关系的模型。40年前的自动控制就要求对所研究的系统找出一个解析模——数学模型。这类数学模型的研究工作早在自动控制之前就成为自然科学界的一大研究课题。数学模型的构成方法或是借助于机理推导、或是利用实验室甚至现场试验、或是两者相结合。科学家们和工程师们以一个“正确”的模型为基础，可以对系统进行分析、综合和设计。在自动控制和系统工程中，系统的数学模型更是完成自动控制或系统分析和设计任务的先决条件。在以机理推演模型和以实验的数据（在系统工程中则着重以统计数据）推导的方法论中，最近20多年已积累了许多能应付各种特殊情况的有效方法。特别是在计算机“独霸”天下的今天，这些方法的使用是越来越方便了。“文革”后，在我们刚开展系统工程研究和推广其应用之际，曾大声疾呼人们作决策不要单凭“拍脑袋”、主观、感情和想当然，而要在调查研究、收集数据和定量分析之后再下决心。现在看起来，我们又面临另外一个分岔口：是完全凭定量分析，认为数学模型万能呢？还是要结合定性与定量分析，而且在某些场合，还要以定性方法为主来分析系统、设计方案和作出最恰当的决策呢？（在这里我指的定性方法与旧时代“拍脑袋”的定性方法有一定区别，指的是以符号、陈述性知识和规则为主的定性方法，应该说是发展了的定性方法。）世界上的事情是极为复杂的，尤其是有人参与的系统，单以数学关系定量描述这些复杂系统内在的和外部的关系是远远不够的。不要说当今的数学水平还不够，即使将来数学水平发展到更高层次时，仍然不足以描述世界上的一切事情。在这方面恰恰是人类本身具有独特的直觉和智慧，可以对付某些数学手段无法对付的问题。就拿自动控制系统来说，自动控制要做的事无非是：

诊断（在线，分析系统过去的行为）；

监督（在线，分析系统当前的行为）；

控制（在线，用外加信息影响系统性能）；

预测（在线，估计系统将来的行为）；

规划（离线，为将来的目标制定控制策略）；

设计（离线，拟定一个符合给定规范的系统）等。

这些工作单靠计算机按数学模型做数值计算是完成不了的，还要靠人的智慧和经验，原因是：（1）有些待解决的问题无法完全用数学描述，有关系统的动态和现状信息的测试手段有不确定性而生成偏置；（2）更重要的是为了政治的、经济的或伦理的原因，在执行计算机做出的决定之前，必须由人事先检查决定的后果。所以，目前正在发展的比较完善的自动控制系统的决策，必须在计算机和专家密切配合下解决。说到系统工程则更是如此，因为系统工程除了要处理大量物理系统外，更要处理许多包括人在内的事理系统，如社会经济系统，人口、生态、农业、地区经济发展等大系统问题，合作的和非合作的多人多水平多目标决策问题，人类世界中各种各样的预测问题……等等。为求解这些问题，一方面要依靠和发展已有的利用计算机对数据和模型做出的数值分析，同时还必须依靠有关领域专家们对这类问题的专门

知识（他们的专家知识、经验、技巧和直觉，也包括他们的学习方法和推理方法）。这也是为什么近20年来自然智能、人工智能和各种类型专家系统的研究和应用得到蓬勃发展的原因。

也许是由于我国工业基础和科学技术比较薄弱的原因，我国研究人工智能和专家系统比工业发达国家晚一段时间。这方面的书籍也比较少，这又反过来影响了我们对自然智能、人工智能和专家系统研究的进度。蔡连成等最近编写的《专家系统基础与实现》一书，填补了我国出版界在这方面的空白。我非常高兴有机会先看到这本书的原稿，作者写得比较精炼。全书九章，每章间扣得很紧，而且深入浅出。本书不仅提供给读者许多有关专家系统的一般知识，而且对于具有数理逻辑知识和一般计算机应用能力的读者，还是一本很好的自学教材。这本书可以教会读者如何开发一个用LISP语言的专家系统。为此，特作此序。

刘　豹

1989年5月17日于天津

编著者的话

专家系统是人工智能中既活跃又卓有成效的前沿分支，也是一门综合性较强的边缘学科。近年的发展使专家系统取得了可喜的成果，也使专家系统成了各行各业的热门课题。至今为止，专家系统已广泛地应用于工业、农业、医疗卫生、物理、化学、数学、军事、法律等各个领域。最近几年，我国也有不少科技工作者对专家系统表示了极大的兴趣。因此，我们编写了这本书，希望能对我国的专家系统的开发、应用起到一点微薄的作用。

编著者之一蔡连成在美国辛辛那提大学访问期间，曾有幸在该大学工学院人工智能和计算机视觉实验室从事专家系统及其工具的开发研究工作。在实验室主任、教授William.G. Wee博士的指导、帮助下，获取了大量的资料，积累了不少经验。而后，我们又在我院本科生和研究生的教学、科研中丰富了经验。本书正是在这样的基础上写成的。

本书介绍了专家系统的基础，还介绍了开发专家系统的专用语言LISP和专用工具PCPLUS，同时介绍了开发专家系统的一般步骤和两种较为先进的编程方法。为了在读者读完本书后，能开发自己的专家系统，书中在介绍专家系统原理、方法、手段的同时，还有大量的实例。本书可作高校计算机科学、自动化、机械、管理工程、系统工程、化工、农、医、生物工程、军事等专业高年级学生和研究生的专家系统教材或参考书，也可供从事专家系统开发研究的科技人员参考。

本书共分九章。第一章绪论。第二章介绍LISP语言的基础，一方面为阅读后续章节奠定基础，另一方面也为专家系统的实现提供可能性。第三章介绍专家系统的一般定义，与传统程序的差异、应用分类及一般原理。第四章集中介绍当前最常用的人工智能、专家系统中的知识表示方法。第五章介绍当前专家系统中的热门课题——不精确推理的几种常见方法。第六、七章是关于“面向对象”和“黑板”的编程技术。第八章在介绍专家系统工具分类后，重点介绍一个能在微机上使用的专家系统工具——PCPLUS的功能和使用方法。第九章叙述了专家系统开发的一般步骤。本书第一、三、七、九章及第四章第一节由蔡连成执笔；第二章由张牧、滕健共同完成；第五章及第四章第二、三节由张牧执笔；第六、八章及第四章第四节由滕健执笔。担任主编的蔡连成副教授负责确定了各章、各节的内容，并对全书作了审定。在整个编写过程中得到了吴翼平教授的热忱关怀和帮助，吴翼平教授还对全书作了认真的审阅。天津大学教授刘豹先生仔细地阅读了全书，提出了不少有益的建议及修改意见，并为本书作了序。在本书编写中，还得到邓学荣、刘文菊、张之华等同志的帮助。对此，一并致谢。

由于我们在专家系统方面的知识还很浅薄，再加之编著方面经验不足，书中难免有错误和不足之处，恳请专家、同仁及广大读者指教。

蔡连成　滕健　张牧

1990年于天津纺织工学院

目　录

第一章 绪 论

人工智能（AI）是50年代在美国首先兴起的一门综合性很强的边缘科学。它和“能源技术”、“空间技术”一起被誉为本世纪三大科学技术成就。人工智能已引起世界各国、多学科科学家的重视。许多不同学科的科学家正在人工智能的不同领域中发挥他们的才能。问题求解、定理证明、自然语言理解、语音识别、自动程序设计、机器人学、计算机视觉、图像处理、专家系统等各个人工智能的分支，也越来越完善，越来越实用。专家系统是人工智能领域中最广泛、最实用、最有成就的分支之一。迄今为止，专家系统已广泛应用于化学、医学、电子工程、农业、地质、信息管理、制造、数学、控制、气象、空间技术等各个方面。

数理逻辑、自动机理论、控制论、信息论、系统论及电子计算机的发明为人工智能的兴起提供了充分的准备。1956年夏，在美国的Dartmouth学院，由J.McCarthy、M.Minsky、N.Lochester、C.Shammon发起召开了历时两个月的讨论会，共同探讨了用机器模拟智能的问题。在这次会上第一次正式使用了人工智能（Artificial Intelligence）这一术语。这次会议的召开标志着人工智能这门学科的正式诞生。在60年代，科学家试图发现一些通用的办法来模仿复杂的人类思维过程，以解决各类问题。A.Newell、J.Shaw和H.Simon等人所建立的通用问题求解系统（General Problem Solver简作GPS）便是一个著名的系统。GPS可以解十余种不同类型的问题。但是，后来的工作证明这种思想在通用问题求解方面并无重大突破。对于庞大复杂的现实问题，这种“通用”办法往往难以实现，同时成果甚微。于是，人工智能的专家们开始寻求其它途径来模拟人类的思维、智能。70年代，人们开始致力于“知识表示”和“解的搜索”等问题。虽然，这种方法也取得了一些成果，但是仍然没有什么实质性的突破。

直至70年代末期，人工智能的科学家们才真正开始认识到知识的重要性。1977年，著名的人工智能学者、Stanford大学教授Feigenbaum在第五届人工智能国际联合会议上的特邀发言中指出，一个系统的问题求解能力是来源于它所具有知识的本身，而不是来自于它所采用的形式和推理方式。这在人工智能领域是一个概念性的突破。事实上，要想使一个系统具有智能、思维，必须提供大量的、有关该问题的特殊知识。这种思想的实现导致了某些有特殊目标的人工智能系统的出现。这些系统具有大量的特殊知识，在某些较为狭窄的领域里工作起来很有成效。人们称这些系统为专家系统或基于知识的系统。概念上的突破，再加上一些成功的专家系统的出现，促进了专家系统的蓬勃发展。

迄今为止，已有许多成功的专家系统出现。图1.1表示了80年代前一些著名专家系统的发展情况。

下面将图中出现的一些专家系统作一简单介绍。

DENDRAL系统是由Stanford大学Feigenbaum等人于1965年开始开发研究的。这个专家系统能够根据质谱仪所产生的数据，推断出可能存在的分子结构及原子结构。在DENDRAL的基础上，后来又开发了META-DENDRAL系统。第一个DENDRAL系统，又

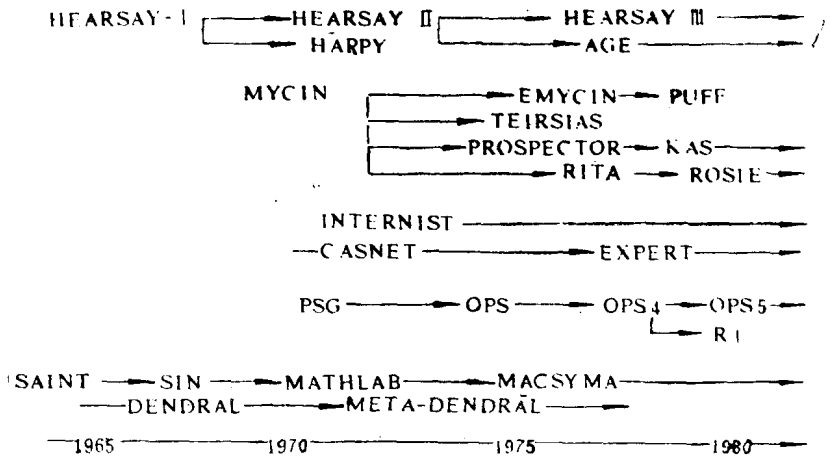

图1.1 一些著名专家系统的发展情况

称启发式DENDRAL，它采用了生成—测试的问题求解方法。这种生成器能列出各种可能的有机化学结构，而试验器用来鉴定这些结构，以决定是否接受或者否定这些结构。META-DENDRAL比DENDRAL，增加了对知识分析的功能，它接受已知的质谱—化学结构对作为输入，试图推出质谱学的专门知识。启发性 DENDRAL 是一个执行系统，而 META-DENDRAL 则是一个学习系统。DENDRAL 在它所完成的工作中已表现得比人类专家出色，并且已在化学研究中导致了人和机器作用的重新定义。

SAINT是由 MIT的 Slagle 等人从 1961 年起开发建立的一个符号数学的专家系统。MACSYMA是其最高峰。象DENDRAL一样，MACSYMA也胜过了绝大多数人类专家。它可以进行微积分的符号运算，并擅长于简化符号表达式。

Weiss和Kulikowski等人在70年代末期，由CASNET演变开发的 EXPERT 系统是一种建立专家系统的语言。CASNET是一个诊断、治疗青光眼的专家系统，EXPERT系统已用来建立眼科、内分泌病、风湿病的专家系统。

由Carnegie-Mellon大学Pople等人研制的CADUCEUS系统由一个表示内科疾病和症状之间关系的语义网络构成。这个庞大的系统约有100,000条关系，表示了全部相关知识的85%。美国国家卫生组织已对这种诊断的指导意义作出评价。

Stanford大学Shortliffe等研制的 MYCIN 则是早期很有影响的一个医疗诊断专家系统。它是用于诊断、治疗传染性血液病的，包括大约400条IF—THEN型的规则。MYCIN根据所提供的数据，以逆向推理的方式来获得问题可能的解答，并且有不精确推理和解释功能。MYCIN的解释功能也被认为是超过某些人类专家的。后来，将MYCIN中有关传染性血液病知识抽掉以后，形成了EMYCIN系统。EMYCIN 为以后建立医疗诊断专家系统提供了极大的方便，用EMYCIN建立的PUFF系统便是一个较为成功的例子。

Stanford 研究所 (SRI) 从 1974 年起研制开发的 PROSPECTOR是个较为大型的用于地质勘探的专家系统。Stanford研究所花了三十几个人年，直到 1983 年才建成今天的

PROSPECTOR系统，它包含12个模型1000多条规则。

ROSIE系统是RAND公司在RITA系统的基础上进化而来的，ROSIE除了有解释功能外，还具有多种知识表达方法以及用户和系统之间的通讯等功能。它是设计用来支持多种新的专家系统应用的第一个系统。

70年代初期Newell和Mc Dermott在Carnegie—Mellon大学对PSG产生式系统的工作导致了OPS系列（OPS4，OPS5，OPS83）等产生式语言的诞生。

R1又称XCON，是美国DEC公司用于根据用户的订货配置计算机系列的专家系统。它是OPS语言在建成专家系统中最成功的例子。开始使用时，R1仅有250条规则，以后不断扩充，使之功能日益增强。目前，已有规则3300条。它不仅可配置 VAX-11/780 ，而且还可以配置PDP-11/44、MicroVAX-Ⅰ、MicroPDP 11等DEC公司的各类计算机产品。

最后要介绍的是Carnegie-Mellon大学开发的 HEARSAY Ⅰ、Ⅱ、Ⅲ语言理解系统，HEARSAY-Ⅱ可在 1000 个词汇的范围内识别连续的语音。但语音识别仍是一个很困难的课题。

专家系统二十多年的发展历史，已充分显示了它的作用。不少系统在某些方面的功能已超过了人类专家的水平。一些实用专家系统的成功引起了人们对专家系统的极大兴趣。专家系统已成为当今世界的热门学科，预料在不久的将来，国内外将有更多的人从事专家系统的研制工作，大批的更有成效的专家系统将大量涌现。

本书共分九章。第二章介绍LISP语言的基本概念、基本函数，希望为读者阅读有关章节及今后学习开发专家系统打下一个基础。第三章讲什么是专家系统，介绍专家系统的特点、和传统程序的差异、专家系统适合于解决什么样的问题、建成专家系统需具备什么条件等问题，同时还介绍专家系统的应用分类和结构。第四章是知识表示，介绍在建立专家系统时，常用的一些知识表示方法。第五章为不精确推理，介绍了目前常用的四种不精确推理方法。第六、第七章分别介绍了较为常用的黑板结构和面向对象的编程技术。第八章讲专家系统工具，以一个具体工具为例，介绍了专家系统工具的功能，并对专家系统工具作了分类介绍。第九章是专家系统的设计，介绍了专家系统的一般设计步骤，并给出一个例子。

第二章 LISP 语 言

LISP语言是人工智能学科领域中应用最广泛的一种程序设计语言。它是由美国麻省理工学院的J.McCarthy等人于1960年提出的。二十多年来，LISP语言在人工智能的各个领域中发挥了非常重要的作用。迄今为止，世界上大多数的人工智能系统都是用LISP语言编写的。尤其在专家系统建立和专家系统工具的开发中，LISP语言更是必不可少的。本章主要介绍LISP语言的特点、基本数据类型、简单的输入、输出函数、基本控制结构、自定义函数、常用的超函数、宏函数及初步的数据驱动程序技术。目的是使读者对LISP语言有一初步的了解并能顺利地阅读后续章节中给出的用LISP语言编写的程序，进而更好地学习专家系统这门课程。

§2.1 LISP语言的特点及基本的数据类型

2.1.1 LISP语言的特点

美国著名的人工智能专家P.H.Winston说："在人工智能领域使用LISP语言就象其他学科使用微分方程或布尔代数一样，因为LISP语言是人工智能的数学。"LISP语言是人工智能程序设计及人工智能理论研究必不可少的工具，它之所以成为建立人工智能系统的优秀语言，是因为它具有一些异于其他语言的特点：

（1）LISP语言是一种擅长于对符号及由符号构成的表进行加工处理的程序设计语言实际上，LISP是表处理语言（List Processing Language）的缩写。

（2）LISP语言的唯一数据结构是符号表达式，也叫S-表达式。S-表达式可以方便地用来表示人工智能系统中的各类知识。

（3）LISP语言是函数型语言，用LISP语言编写的程序可粗略地认为是一函数序列。函数定义是通过定义函数的函数来实现的。LISP程序设计过程，可看作是函数定义和编排函数调用序列的过程，LISP程序执行过程就是函数调用的过程。

（4）LISP语言的函数和数据在形式上都是用S-表达式表示，可以象处理数据一样处理函数。换句话说，数据和函数在LISP语言中，有相同的形式和地位。毫无疑问，这将给程序设计带来极大的方便。

（5）递归是LISP语言中主要的控制结构，这种控制结构使程序设计简化、结构清晰、易读、易懂。

目前，世界上流行的LISP版本有很多种，不同的版本给软件的交流带来许多不便，LISP语言的标准化已势在必行。当前，人们趋于将COMMON LISP作为标准LISP的基础。GCLISP（GOLDEN COMMON LISP）语言是美国Gold Hill Computer公司于1984年底推出的一个在IBM-PC/XT、PC/AT及其兼容机上实现，在PC-DOS（或MS-DOS）2.0版本或更高的操作系统支持下运行的LISP版本。GCLISP是以GOMMON LISP为基础设计而成的，它和COMMON LISP的核心部分兼容。GCLISP是迄今为止和COMMON

LISP最为接近且能在IBM-PC/XT、PC/AT机上运行的LISP语言。

本书介绍的LISP均引自GCLISP。所介绍的程序也都是在IBM-PC/XT、PC/AT机上运行过的GCLISP程序。

2.1.2 LISP的基本数据类型

每一种程序设计语言由于应用领域不同，它所提供的基本数据类型也是各不相同的。LISP语言几乎包括了FORTRAN、PASCAL等常用程序设计语言所具有的各种数据类型。LISP语言的基本数据类型是符号表达式，即S-表达式。这里S是SYMBOLIC的缩写。S-表达式又可分成原子和表两大类。

（1）原子（ATOM）

原子是S-表达式的最简单的情况，它是由字母和数字组成的符号串。由字母和数字符号组成的原子称为符号原子，符号原子可用来表示变量、函数名，还可以用来描述事实、属性等（以数字开头的字母，数字串在GCLISP中也是合法的符号原子）。

ABC，Fred-Smith，A4，B5E7，1TWO 都是正确的符号原子。数字串称为数原子。如：3.147，-8都是数原子。数原子又分为整数、定点数和浮点数。

在GCLISP中一些特殊的符号如空格“ ”、括号“(”、“)”、反斜杠“\”等不能用来构造原子。如（A），\person都不是原子，B C不是一个原子，而是B和C两个原子。

（2）表（LIST）

表是LISP语言中最常用的数据类型，也是主要的处理对象。

表是由圆括号括起来的由空格分开的若干个元素的集合。这里元素可以是原子，也可以是表。元素之间是用空格分开的。

例1 （This is a list） 这是一个四个元素的表，元素均为符号原子。

（1 one 2 two 3 three） 这是一个六个元素的表，元素为符号原子和数原子。

表也可以没有元素，如（ ），称这个表为空表。（ ）与符号原子NIL是等同的。表还可以任意嵌套。

例2 （A B (C D) E)

这个表有四个元素，其中元素(C D)也是一个表，称这种在一张表中充当元素的表为子表。

特别要注意的是表一定是以左括号开始，以右括号结束，而且在表中左右括号一定是成对出现的。

概括起来，GCLISP语言中的基本数据类型及它们之间的关系可用图 2.1.1 表示。

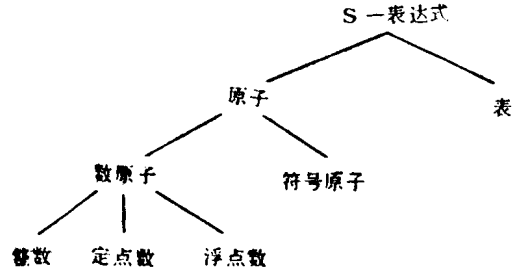

图 2.1.1 GCLISP 中基本数据类型及它们之间的关系

（3）变量

前面提到变量可由符号原子表示。在GCLISP中，变量可以取本语言提供的任何数据类型的值，即变量的值可以是表或原子，也可以是数原子或符号原子。在下一节中将介绍如何给一个变量赋值。

§2.2 基本函数

LISP 语言是函数型语言，其程序实际上是由一系列函数组成的。函数定义和函数调用是LISP程序设计的主要风格。一般LISP语言都提供一些基本函数。有了这些基本函数，通过复合、递归、迭代等手段就可以定义其它复杂函数，进而完成程序设计所要求的一些复杂的功能。下面介绍GCLISP语言提供的基本函数。

2.2.1 基本表处理函数

如前所述，LISP语言的突出优点是进行表处理，设计 LISP 程序时经常要对表进行操作。比如，取出表中的元素，加元素到表中，合并两个表等等。这些操作都是通过调用基本表处理函数实现的。下面介绍一些基本表处理函数。

（1）CAR函数

调用形式为：（CAR〈参数〉）；其中〈参数〉为一张表

CAR函数是取头函数，函数的参数是一张表。CAR的作用是取出表中的第一个元素。

例1 *(CAR '(BANANA APPLE ORANGE PEAR))

BANANA

“*”为GCLISP的提示符，跟在“*”后面的S-表达式为键入的内容，下面一行印出的表达式为GCLISP系统调用函数后回送的值。该值为函数调用的结果，通常称之为“返回值”。在后续章节中，将延用此约定。例中出现的“'”表示禁止求值，有关“'”的意义和求值及返回值的概念将在后面解释。

上例中，表（BANANA APPLE ORANGE PEAR）为CAR函数的参数。CAR的调用结果是将表中的第一个元素取出来。

例2 *(CAR '((A B) C))

(A B)

例2中，返回值为（A B）。注意：表（(A B) C D）中的第一个元素是表（A B），而不是原子A。

例3 *(CAR '( ))

NIL

例3表明：如果CAR的参数是空表（ ），则返回值是NIL。

（2）CDR函数

调用形式为：（CDR〈参数〉）

CDR是一个取尾函数，它的参数同CAR函数的参数一样也是一张表。CDR 函数调用的结果是原表中除了第一个元素以外的所有元素组成的一张新表。

例4 *(CDR '(BANANA APPLE ORANGE

(APPLE ORANG PEAR)

例5 *(CDR '(A B C))

(B C)

例6 *(CDR '( ))

NIL

例4中，CDR函数的参数是表(BANANA APPLE ORANGE PEAR)。而表(APPLE ORANGE PEAR)是返回值，它是表(BANANA APPLE ORANGE PEAR)去掉第一个元素BANANA后剩下的表。例6说明，如果对空表取尾，则返回值是NIL。由于NIL和空表是等同的，所以可以认为CDR函数的返回值总是一张表。

对照CAR函数可以看出，CDR函数是CAR函数的补函数。一个取头，一个取尾。在LISP程序设计中要经常使用CAR和CDR函数将一张表分解。

函数CAR和CDR的联合使用，可以获得更强的功能。正确地使用可以取到表中的任一元素。

例7
```
*(CAR (CDR (CDR  '(A  B C))))
C
*(CDR (CAR  '((A B)  C)))
(B)
```

但是，多次反复使用会很繁琐，GCLISP中提供一种简写形式，例如用CADDR和CDAR来代替例7中的使用。

例8
```
*(CADDR  '(A  B C))
C
*(CDAR  '((A B) C))
(B)
```

GCLISP中规定，最多复合三次。CADDR是合法的，但是CAADDR是非法的。

下面再介绍三个构造新表的函数CONS、APPEND和LIST，它们可用于构造、合成新表。

（3）CONS函数

调用形式为：(CONS〈参数1〉〈参数2〉)

其中：〈参数1〉为一S-表达式，〈参数2〉是一张表。

CONS函数可用来构造一张新表。它完成的操作是把函数的第一个参数作为一个元素加到第二个参数中去，并作为所构成新表中的第一元素。

例9
```
*(CONS 'DOG '(LION CAT TIGER))
(DOG LION CAT TIGER)
```

例10
```
*(CONS '(A B) '(C  D))
((A B) C D)
```

例11
```
*(CONS 'HELLO '( ))
(HELLO)
```

例9是把一个原子加入到表中的情况，例10则是把一张表作为一个元素加另一张表的情形，例11说明原子和空表通过CONS的作用构成了表。

（4）APPEND函数

调用形式为：(APPEND〈参数1〉……〈参数n〉)

其中，各参数均为表。

APPEND是个合并函数，它将若干个表中的元素合并成一个新表。

例12
```
*(APPEND '(A B) '(C D) '(E F))
(A B C D E F)
```

例13 *(APPEND '((A) B) '(C (D E)))

((A) B C (D E))

由上可见，APPEND函数调用的返回值是一张新表，新表中的元素是函数的参数——若干个表中的元素的依次排列。

（5）LIST函数

调用形式：（LIST 〈参数1〉 …… 〈参数n〉）

其中，参数可以是任意的S-表达式。

LIST函数的作用是把若干S-表达式作为元素括在一起构成一张新表。

例14 *(LIST 'A 'B 'C)

(A B C)

例15 *(LIST '(A B) 'C)

((A B) C)

例16 *(LIST '(A B) '(C D) 'E)

((A B) (C D) E)

例17 *(LIST 'A)

(A)

从例子可见LIST 函数的返回值是一张新表，新表中的元素与函数的参数一一对应依次排列。

2.2.2 赋值函数及求值函数

LISP语言中，变量由符号原子表示，变量的值是 S-表达式。调用赋值函数可给一个变量赋值。

（1）SET函数

调用形式为：（SET 〈参数1〉 〈参数2〉）

其中，〈参数1〉是符号原子，〈参数2〉为任一S-表达式。

SET是LISP中的赋值函数，它将一个表达式赋给一个符号原子。这时符号原子就成为一个有值的变量，它的值可以是一个符号原子，可以是一个表，也可以是一个数字原子。

例1 *A

ERROR ; 出错

UNBOUND VARIABLE: A ; A未被赋值

例2 * (SET 'A 'ATOM1)

ATOM1

*A

ATOMI

通过调用SET函数，原子A有了值ATOM1。

例3 *(SET 'B '(This is a list))

(THIS IS A LIST)

*B

(THIS IS A LIST)

例4 *(SET 'NUMBER 3.14)

3.14

*NUMBER

3.14

上述三个例子中，分别对变量A、B、NUMBER赋给了原子，数和表三种不同类型的S-表达式。

SET函数还有一种调用形式，可一次对几个变量同时赋值。其调用形式为：

（SET 〈参数A_1〉 〈参数S_1〉 〈参数A_2〉 〈参数S_2〉 …… 〈参数A_n〉 〈参数S_n〉）

其中，参数A_i（$1 \leqslant i \leqslant n$）均为符号原子，参数$S_i$（$1 \leqslant i \leqslant n$）均为S-表达式。

该函数的功能是将〈参数S_i〉赋给〈参数A_i〉（$1 \leqslant i \leqslant n$）。这样几个变量的赋值可一次完成。

例5 *(SET 'A 'ATOM2 'B '(THIS IS A LIST) 'NUMBER 56)

56

*A

ATOM2

*B

(THIS IS A LIST)

*C

56

例5中，一次调用SET函数完成了例1、例2、例3三次调用SET函数的工作。需注意的是例5中函数的返回值不是全部三个变量的值，只是最后一个变量的值56。在这里，返回值并不重要，重要的是“副作用”，即通过SET函数的调用，各变量都已具有相应的值。事实上，在例1、例2、例3和其它的函数调用中，副作用往往更重要。

（2）SETQ函数

调用形式：（SETQ 〈参数1〉 〈参数2〉）

其中，〈参数1〉是符号原子，〈参数2〉是任一S-表达式。

SETQ是GCLISP中的另一个赋值函数，它的作用是将表达式赋给符号原子。注意，使用SETQ时，符号原子前不加“'”。

例6 *(SETQ A '(A B C)) ；把表（A B C）赋给符号原子A

(A B C) ；返回（A B C）

*A

(A B C) ；变量A的值为表（A B C）

(SETQ A '(A B C))这一调用结果是把（A B C）赋给符号原子A。A成为具有值（A B C）的变量。

注意：只有符号原子能被赋值，数字原子的值就是该数字本身。

（3）求值、返回值及QUOTE函数

目前，在IBM-PC机上提供的GCLISP是一种解释性语言，它具有良好的交互性。每当你键入一个S-表达式后，LISP解释系统对这一S-表达式进行解释。这一解释过程在LISP中叫做求值，LISP系统随即将所求得的值回送到显示屏上，称所求得的值为返回值。整个LISP的执行过程即为“读入——求值——显示”的过程。