浙江省高等学校重点教材

人工智能导论（第3版）

Introduction of Artificial Intelligence *Third Edition*

王万良 编著

高等教育出版社
HIGHER EDUCATION PRESS

浙江省高等学校重点教材

人工智能导论

Rengong Zhineng Daolun

（第3版）

王万良　编著

高等教育出版社·北京
HIGHER EDUCATION PRESS　BEIJING

内容提要

本书是一本内容基础性强、可读性好、适合讲授的人工智能教材。读者通过学习本书，能够掌握人工智能的基本知识，并能了解人工智能研究的一些前沿内容，为进一步学习人工智能理论与应用奠定基础。

全书共11章。第1章绪论；第2章知识表示；第3章确定性推理方法；第4章不确定性推理方法；第5章搜索求解策略；第6章遗传算法及其应用；第7章专家系统与机器学习；第8章人工神经网络及其应用；第9章智能体与多智能体系统；第10章自然语言处理及其应用；第11章人工智能在游戏设计中的应用。附录中给出了本书部分习题的简要解答。

本书可作为电气信息类、机械类、电子信息科学类以及其他专业的本科生学习人工智能课程的教材。由于书中几大部分内容相对独立，教师可以根据课程计划灵活选择相关内容。

图书在版编目(CIP)数据

人工智能导论／王万良编著．—3版．—北京：高等教育出版社，2011.2

ISBN 978-7-04-031503-5

Ⅰ.①人… Ⅱ.①王… Ⅲ.①人工智能-高等学校-教材 Ⅳ.①TP18

中国版本图书馆CIP数据核字(2010)第230582号

策划编辑 张 龙 **责任编辑** 李善亮 **封面设计** 于文燕 **责任绘图** 尹 莉
版式设计 范晓红 **责任校对** 陈旭颖 **责任印制** 韩 刚

出版发行	高等教育出版社	**购书热线**	010-58581118
社　　址	北京市西城区德外大街4号	**咨询电话**	400-810-0598
邮政编码	100120	**网　　址**	http://www.hep.edu.cn
			http://www.hep.com.cn
经　　销	蓝色畅想图书发行有限公司	**网上订购**	http://www.landraco.com
印　　刷	北京鑫丰华彩印有限公司		http://www.landraco.com.cn
		畅想教育	http://www.widedu.com
		版　　次	2005年1月第1版
开　　本	787×1092　1/16		2011年2月第3版
印　　张	16.5	**印　　次**	2011年2月第1次印刷
字　　数	400 000	**定　　价**	26.10元

本书如有缺页、倒页、脱页等质量问题，请到所购图书销售部门联系调换。

物料号　31503-00

前　言

人工智能是目前迅速发展的新兴学科，已经成为许多高新技术产品中的核心技术，也是计算机游戏等数字媒体产品中的重要设计技术。人工智能在 Internet 时代获得了前所未有的发展机遇，Web 环境下智能信息处理技术成为推动人工智能在网络环境中发展的一大动力。由于人工智能是模拟人类智能解决问题的方法，几乎在所有领域都具有非常广泛的应用，所以，目前不仅许多专业的研究生开设人工智能课程，而且许多专业的本科生，特别是计算机与自动化等电气信息类以及机械类、电子信息科学类等专业的本科生，都开设了人工智能课程。

1. 本书的形成

本书作者于 1989 年开始从事人工智能及其应用方面的研究，从 1993 年开始从事人工智能方面的教学。为控制科学与工程、计算机科学与技术、机械工程等专业研究生讲授“人工智能原理与应用”课程，为计算机、自动化等专业本科生讲授“人工智能导论”课程，还为全校工学、理学、经济学、管理学、哲学、文学、法学等专业学生开设“人工智能及其应用”公选课。在多年的教学实践中，深感需要编著一本基础性强、可读性好、适合讲授的人工智能教材。本书作者在自己多年来的讲稿基础上，于 2005 年编写出版了《人工智能及其应用》，2008 年修订出版了《人工智能及其应用》(第二版)，作为研究生和本科生的人工智能教材，被许多高校选用。

随着人工智能技术的发展，越来越多的本科专业开设了人工智能课程。因此，面向本科生的人工智能课程的教材建设具有重要意义。作者在《人工智能及其应用》(第二版)的基础上，选择基础、实用的内容，并充实了一些应用性内容，编写了这本用于本科教学的《人工智能导论》，其目的是使学生学习和掌握人工智能的基本概念和基本原理，了解人工智能的一些前沿内容，拓宽知识面，启发思路，为今后在相关领域应用人工智能技术奠定基础。

2. 主要内容

全书共 11 章。第 1 章除了介绍人工智能的基本概念、发展简史，着重介绍目前人工智能的主要研究内容与各种应用，以开阔读者的视野，引导读者进入人工智能各个研究领域。第 2 章介绍一阶谓词逻辑、产生式、框架等基本的知识表示方法。第 3 章介绍基于谓词逻辑的确定性推理方法。详细介绍了在自动定理证明中具有重要地位的鲁宾逊归结原理。通过多个典型例题清楚地介绍了将谓词公式化为子句集的步骤，详细讲述了利用归结原理证明定理和求解问题的方法。第 4 章介绍不确定性推理方法。主要介绍了比较实用的可信度方法、证据理论、模糊推理方法。第 5 章介绍了基于搜索的问题求解策略。第 6 章介绍了以遗传算法为代表的适用于大规模优化的随机搜索算法。第 7 章介绍了专家系统与机器学习的概念、工作原理、建立方法，简单介绍了几个典型的专家系统实例以及开发工具。第 8 章介绍了人工神经网络的基本理论与方法，着重介绍了应用广泛的 BP 神经网络及其学习算法和 Hopfield 神经网络及其在联想记忆与优化求解中的应用。介绍了神经网络在软测量与求解旅行商问题、生产调度等优化问题中的应用，启发读者如何应用神经网络解决工程实际问题。第 9 章介绍了广泛应用的多智能体技术。简要介绍了多智能体系统的通信、协调、协作与协商方法。第 10 章介绍了应用日益广泛的自然语言处理技术。最后，针对游戏、动漫等新型产业发展需要，简要介绍了人工智能在游戏设计中的应用。附

录中给出了本书部分习题的简要解答。

3. 编写特色

① 语言简明，可读性好。本书尽量用通俗的语言深入浅出地讲解，语言流畅，避免像专著和研究生教材那样追求理论严谨，语言艰涩。学生能够有兴趣、有耐心阅读本教材，领略人工智能的思想与基本方法。

② 内容实用，注重应用。由于人工智能正处于迅速发展阶段，内容非常庞杂。本书在内容上精选一些基本理论与实用方法，而不是像专著和研究生教材那样追求前沿理论方法。学生学习本书后，能够很容易阅读其他专门的书籍，掌握更广、更深的知识。书中尽可能介绍了一些符合这些内容、能够让本科生理解的应用实例，引导学生学习应用新理论解决工程问题的方法。

③ 精选例题和习题，引导学生解答。本书精选了一些例题，有助于人工智能理论与方法的理解。此外，精选了习题，并在书末给出了答案，对有些难题给出了比较详细的解答，也弥补了现有许多人工智能教材中习题解答和人工智能习题集少的缺陷。

④ 编排醒目，有利于学习。每章开始设置了导读，使读者在学习该章之前就知道为什么要学习该章内容，以及该章主要介绍哪些内容。每章最后扼要总结了该章的重要概念、公式、定理与方法。本书将重要的概念、公式、定理与方法明显地标注出来，引起学生注意。

4. 教师如何使用本教材

本书主要作为电气信息类、机械类以及其他各类专业本科生学习人工智能课程的教材，特别是作为计算机、数字媒体等专业本科生学习人工智能课程的教材。本书也可供希望运用人工智能技术的研究人员与工程技术人员参考。

① 教学内容与时间安排。本书用于“人工智能导论”以及同类课程的教学。由于书中几大部分内容相对独立，教师可以根据课程计划和专业需要选择部分内容，仍可保持课程体系结构的完整性。本书课堂教学时数一般为 32 ~ 56 学时。对于数字媒体技术、计算机、自动化等需要深入了解人工智能的专业，可以安排 8 ~ 16 学时左右的人工智能实验。

② 如何讲授本书。本书内容比较广泛，着重介绍人工智能的基本理论与实用方法，教师在教学过程中可以增加一些相关的比较深入的内容。特别是结合自己的研究与实践增加一些工程实例，或者选择自己比较熟悉、符合所教专业的一些人工智能实例，作为补充内容进行介绍，从而丰富教学内容，增加学生兴趣，培养学生创新能力。

③ 如何获取该课程教学资源。使用本书作为教材的教师可向作者或通过中国高校计算机课程网 http://computer.cncourse.com 免费获取电子教案、习题详细解答、实验指导书等教学资源。

5. 作者致谢

本书为浙江省高等学校重点教材（ZJB2009054）。感谢赵燕伟教授、杨旭华副教授、管秋副教授、徐新黎副教授、蒋一波博士、郑建炜博士等参与了本书部分编写工作，提出了许多建设性意见。衷心感谢高等教育出版社为本书出版付出的辛勤劳动与许多有益的修改建议。

6. 联系作者

本书内容虽然经过多年使用和修改，但仍会存在缺点，欢迎使用本书的教师和读者提出宝贵意见。E-mail：wwl@zjut.edu.cn。

王万良

2010 年 7 月于杭州

目　录

第 1 章 绪论

人工智能是在计算机科学、控制论、信息论、神经心理学、哲学、语言学等多学科研究的基础上发展起来的综合性很强的交叉学科，是一门新思想、新观念、新理论、新技术不断出现的新兴学科，也是正在迅速发展的前沿学科。自 1956 年正式提出人工智能这个术语并把它作为一门新兴学科的名称以来，人工智能获得了迅速的发展，并取得了惊人的成就，引起了人们的高度重视，受到了很高的评价。它与空间技术、原子能技术一起被誉为 20 世纪三大科学技术成就。有人称它为继三次工业革命后的又一次革命，认为前三次工业革命主要是扩展了人手的功能，把人类从繁重的体力劳动中解放出来，而人工智能则是扩展了人脑的功能，实现脑力劳动的自动化。

本章将首先介绍人工智能的基本概念以及人工智能的发展简史，然后简要介绍当前人工智能的主要研究内容及主要研究领域，以开阔读者的视野，使读者对人工智能极其广阔的研究与应用领域有总体的了解。

1.1 人工智能的基本概念

1.1.1 智能的概念

人工智能(Artificial Intelligence, AI)的目标是用机器实现人类的部分智能。因此，下面首先讨论人类的智能行为。

智能及智能的本质是古今中外许多哲学家、脑科学家一直在努力探索和研究的问题，但至今仍然没有完全了解。智能的发生与物质的本质、宇宙的起源、生命的本质一起被列为自然界四大奥秘。

近年来，随着脑科学、神经心理学等研究的进展，人们对人脑的结构和功能有了初步认识，但对整个神经系统的内部结构和作用机制，特别是脑的功能原理还没有认识清楚，有待进一步的探索。因此，很难给出智能的确切定义。

目前，根据对人脑已有的认识，结合智能的外在表现，从不同的角度、不同的侧面，用不同的方法对智能进行研究，人们提出了几种不同的观点。其中影响较大的观点有思维理论、知识阈值理论及进化理论等。

（1）思维理论

认为智能的核心是思维，人的一切智能都来自大脑的思维活动，人类的一切知识都是人类思维的产物，因而通过对思维规律与方法的研究可望揭示智能的本质。

（2）知识阈值理论

认为智能行为取决于知识的数量及其一般化的程度，一个系统之所以有智能是因为它具有可运用的知识。因此，知识阈值理论把智能定义为：智能就是在巨大的搜索空间中迅速找到一个满意解的能力。这一理论在人工智能的发展史中有着重要的影响，知识工程、专家系统等都是在这一理论的影响下发展起来的。

（3）进化理论

认为人的本质能力是在动态环境中的行走能力、对外界事物的感知能力、维持生命和繁衍生息的能力。正是这些能力对智能的发展提供了基础，因此智能是某种复杂系统所浮现的性质，是由许多部件交互作用产生的，智能仅仅由系统总的行为以及行为与环境的联系所决定，它可以在没有明显的可操作的内部表达的情况下产生，也可以在没有明显的推理系统出现的情况下产生。该理论的核心是用控制取代表示，从而取消概念、模型及显式表示的知识，否定抽象对于智能及智能模拟的必要性，强调分层结构对于智能进化的可能性与必要性。这是由美国麻省理工学院（MIT）的布鲁克（R. A. Brook）教授提出来的。1991 年他提出了“没有表达的智能”，1992 年又提出了“没有推理的智能”，这些是他根据对人造机器动物的研究与实践提出的与众不同的观点。目前这些观点尚未形成完整的理论体系，有待进一步的研究，但由于它与人们的传统看法完全不同，因而引起了人工智能界的注意。

综合上述各种观点，可以认为：智能是知识与智力的总和。其中，知识是一切智能行为的基础，而智力是获取知识并应用知识求解问题的能力。

1.1.2　智能的特征

1. 具有感知能力

感知能力是指通过视觉、听觉、触觉、嗅觉等感觉器官感知外部世界的能力。

感知是人类获取外部信息的基本途径，人类的大部分知识都是通过感知获取有关信息，然后经过大脑加工获得的。如果没有感知，人们就不可能获得知识，也不可能引发各种智能活动。因此，感知是产生智能活动的前提。

根据有关研究，视觉与听觉在人类感知中占有主导地位，80% 以上的外界信息是通过视觉得到的，10% 是通过听觉得到的。因此，在人工智能的机器感知研究方面，主要研究机器视觉及机器听觉。

2. 具有记忆与思维能力

记忆与思维是人脑最重要的功能，是人有智能的根本原因。记忆用于存储由感知器官感知到的外部信息以及由思维所产生的知识；思维用于对记忆的信息进行处理，即利用已有的知识对信息进行分析、计算、比较、判断、推理、联想及决策等。思维是一个动态过程，是获取知识以及运用知识求解问题的根本途径。

思维可分为逻辑思维、形象思维以及顿悟思维等。

(1) 逻辑思维

逻辑思维又称为抽象思维。它是一种根据逻辑规则对信息进行处理的理性思维方式。人们首先通过感觉器官获得外部事物的感性认识,将它们存储于大脑中,然后,通过匹配选出相应的逻辑规则,并且作用于已经表示成一定形式的已知信息,进行相应的逻辑推理。这种推理一般都比较复杂,一般不是用一条规则作一次推理就能够解决问题,而是要对第一次推出的结果再运用新的规则进行新一轮的推理。推理是否成功取决于两个因素,一是用于推理的规则是否完备,二是已知的信息是否完善、可靠。如果推理规则是完备的,由感性认识获得的初始信息是完善、可靠的,则通过逻辑思维可以得到合理、可靠的结论。

逻辑思维具有如下特点:

① 依靠逻辑进行思维。

② 思维过程是串行的,表现为一个线性过程。

③ 容易形式化,其思维过程可以用符号串表达出来。

④ 思维过程具有严密性、可靠性,能对事物未来的发展给出逻辑上合理的预测,可使人们对事物的认识不断深化。

(2) 形象思维

形象思维又称为直感思维。它是一种以客观现象为思维对象、以感性形象认识为思维材料、以意象为主要思维工具、以指导创造物化形象的实践为主要目的的思维活动。思维过程有两次飞跃:第一次飞跃是从感性形象认识到理性形象认识的飞跃,即把对事物的感觉组合起来,形成反映事物多方面属性的整体性认识(即知觉),再在知觉的基础上形成具有一定概括性的感觉反映形式(即表象),然后经形象分析、形象比较、形象概括及组合形成对事物的理性形象认识;第二次飞跃是从理性形象认识到实践的飞跃,即对理性形象认识进行联想、想象等加工,在大脑中形成新的意象,然后回到实践中,接受实践的检验。这个过程不断循环,就构成了形象思维从低级到高级的运动发展。

形象思维具有如下特点:

① 主要是依据直觉,即感性形象进行思维。

② 思维过程是并行协同式的,表现为一个非线性过程。

③ 形式化困难,没有统一的形象联系规则,对象不同、场合不同,形象的联系规则亦不相同,不能直接套用。

④ 在信息变形或缺少的情况下仍有可能得到比较满意的结果。

由于逻辑思维与形象思维分别具有不同的特点,因而可分别用于不同的场合。当要求迅速作出决策而不要求十分精确时,可用形象思维,但当要求进行严格的论证时,就必须用逻辑思维;当要对一个问题进行假设、猜想时,需用形象思维,而当要对这些假设或猜想进行论证时,则要用逻辑思维。人们在求解问题时,通常把这两种思维方式结合起来使用,首先用形象思维给出假设,然后用逻辑思维进行论证。

(3) 顿悟思维

顿悟思维又称为灵感思维。它是一种显意识与潜意识相互作用的思维方式。当人们遇到一个无法解决的问题时,会“苦思冥想”。这时,大脑处于一种极为活跃的思维状态,会从不同的角度用不同的方法去寻求解决问题的方法。有时一个“想法”突然从脑中涌现出来,使人“茅塞顿

开”,问题便迎刃而解。像这样用于沟通有关知识或信息的“想法”通常被称为灵感。灵感也是一种信息,可能是与问题直接有关的一个重要信息,也可能是一个与问题并不直接相关并且不起眼的信息,只是由于它的到来使解决问题的智慧被启动起来了。顿悟思维比形象思维更复杂,至今人们还不能确切地描述灵感的机理。1830 年奥斯特在指导学生实验时,看见电流能使磁针偏转,从而发现了电磁关系。虽然很偶然,但也是在他 10 年探索的基础上发现的。

顿悟思维具有如下特点:

① 具有不定期的突发性。

② 具有非线性的独创性及模糊性。

③ 它穿插于形象思维与逻辑思维之中,起着突破、创新及升华的作用。

应该指出,人的记忆与思维是不可分的,总是相随相伴的。它们的物质基础都是由神经元组成的大脑皮质,通过相关神经元此起彼伏地兴奋与抑制实现记忆与思维活动。

3. 具有学习能力

学习是人的本能。人人都在通过与环境的相互作用,不断地学习,从而积累知识,适应环境的变化。学习既可能是自觉的、有意识的,也可能是不自觉的、无意识的;既可以是有教师指导的,也可以是通过自己实践进行的。

4. 具有行为能力

人们通常用语言或者某个表情、眼神及形体动作来对外界的刺激作出反应,传达某个信息,这些称为行为能力或表达能力。如果把人们的感知能力看作是信息的输入,则行为能力就可以看作是信息的输出,它们都受到神经系统的控制。

1.1.3　人工智能

所谓人工智能就是用人工的方法在机器(计算机)上实现的智能,也称为机器智能(machine intelligence)。

关于“人工智能”的含义,早在它正式提出之前,就由英国数学家图灵(A. M. Turing)提出了。1950 年他发表了题为《计算机与智能(Computing Machinery and Intelligence)》的论文,文章以“机器能思维吗?”开始,论述并提出了著名的“图灵测试”,形象地指出了什么是人工智能以及机器应该达到的智能标准。图灵在这篇论文中指出不要问机器是否能思维,而是要看它能否通过如下测试:分别让人与机器位于两个房间里,二者之间可以通话,但彼此都看不到对方,如果通过对话,人的一方不能分辨对方是人还是机器,那么就可以认为对方的那台机器达到了人类智能的水平。为了进行这个测试,图灵还设计了一个很有趣且智能性很强的对话内容,称为“图灵的梦想”。

现在许多人仍把图灵测试作为衡量机器智能的准则。但也有许多人认为图灵测试仅仅反映了结果,没有涉及思维过程。即使机器通过了图灵测试,也不能认为机器就有智能。针对图灵测试,哲学家约翰·塞尔勒在 1980 年设计了“中文屋思想实验”以说明这一观点。在中文屋思想实验中,一个完全不懂中文的人在一间密闭的屋子里,有一本中文处理规则的书。他不必理解中文就可以使用这些规则。屋外的测试者不断通过门缝给他写一些有中文语句的纸条。他在书中查找处理这些中文语句的规则,根据规则将一些中文字符抄在纸条上作为对相应语句的回答,并将纸条递出房间。这样,从屋外的测试者看来,仿佛屋里的人是一个以中文为母语的人,但他实际

上并不理解他所处理的中文，也不会在此过程中提高自己对中文的理解。用计算机模拟这个系统，可以通过图灵测试。这说明一个按照规则执行的计算机程序不能真正理解其输入、输出的意义。许多人对塞尔勒的中文屋思想实验进行了反驳，但还没有人能够彻底将其驳倒。

实际上，要使机器达到人类智能的水平，是非常困难的。但是，人工智能的研究正朝着这个方向前进，图灵的梦想总有一天会变成现实。特别是在专业领域内，人工智能能够充分利用计算机的特点，具有显著的优越性。

人工智能是一门研究如何构造智能机器（智能计算机）或智能系统，使它能模拟、延伸、扩展人类智能的学科。通俗地说，人工智能就是要研究如何使机器具有能听、会说、能看、会写、能思维、会学习、能适应环境变化、能解决面临的各种实际问题等功能的一门学科。

1.2 人工智能的发展简史

人工智能的发展历史可归结为孕育、形成和发展三个阶段。

1.2.1 孕育

这个阶段主要是指 1956 年以前。自古以来，人们就一直试图用各种机器来代替人的部分脑力劳动，以提高人们征服自然的能力，其中对人工智能的产生、发展有重大影响的主要研究成果包括：

① 早在公元前 384—公元前 322 年，伟大的哲学家亚里士多德（Aristotle）就在他的名著《工具论》中提出了形式逻辑的一些主要定律，他提出的三段论至今仍是演绎推理的基本依据。

② 英国哲学家培根（F. Bacon）曾系统地提出了归纳法，还提出了“知识就是力量”的警句。这对于研究人类的思维过程，以及自 20 世纪 70 年代人工智能转向以知识为中心的研究都产生了重要影响。

③ 德国数学家和哲学家莱布尼茨（G. W. Leibniz）提出了万能符号和推理计算的思想，他认为可以建立一种通用的符号语言以及在此符号语言上进行推理的演算。这一思想不仅为数理逻辑的产生和发展奠定了基础，而且是现代机器思维设计思想的萌芽。

④ 英国逻辑学家布尔（G. Boole）致力于使思维规律形式化和实现机械化，并创立了布尔代数。他在《思维法则》一书中首次用符号语言描述了思维活动的基本推理法则。

⑤ 英国数学家图灵（A. M. Turing）在 1936 年提出了一种理想计算机的数学模型，即图灵机，为后来电子数字计算机的问世奠定了理论基础。

⑥ 美国神经生理学家麦克洛奇（W. McCulloch）与匹兹（W. Pitts）在 1943 年建成了第一个神经网络模型（M－P 模型），开创了微观人工智能的研究领域，为后来人工神经网络的研究奠定了基础。

⑦ 美国爱荷华州立大学的阿塔纳索夫（Atanasoff）教授和他的研究生贝瑞（Berry）在 1937 年至 1941 年间开发的世界上第一台电子计算机“阿塔纳索夫－贝瑞计算机（Atanasoff-Berry Computer，ABC）”为人工智能的研究奠定了物质基础。需要说明的是：世界上第一台计算机不是许多书上所说的由美国的莫克利和埃柯特在 1946 年发明。这是美国历史上一桩著名的公案。

由上面的发展过程可以看出，人工智能的产生和发展绝不是偶然的，它是科学技术发展的必

然产物。

1.2.2　形成

这个阶段主要是指 1956—1969 年。1956 年夏季，由当时达特茅斯（Dartmouth）大学的年轻数学助教、现任斯坦福大学教授麦卡锡（J. McCarthy）联合哈佛大学年轻数学和神经学家、麻省理工学院教授明斯基（M. L. Minsky），IBM 公司信息研究中心负责人洛切斯特（N. Rochester），贝尔实验室信息部数学研究员香农（C. E. Shannon）共同发起，邀请普林斯顿大学的莫尔（T. Moore）和 IBM 公司的塞缪尔（A. L. Samuel）、麻省理工学院的塞尔夫里奇（O. Selfridge）和索罗莫夫（R. Solomonff）以及兰德（RAND）公司和卡内基梅隆大学的纽厄尔（A. Newell）、西蒙（H. A. Simon）等在美国达特茅斯大学召开了一次为时两个月的学术研讨会，讨论关于机器智能的问题。会上经麦卡锡提议正式采用了“人工智能”这一术语。麦卡锡因而被称为人工智能之父。这是一次具有历史意义的重要会议，它标志着人工智能作为一门新兴学科正式诞生了。此后，美国形成了多个人工智能研究组织，如纽厄尔和西蒙的 Carnegie RAND 协作组，明斯基和麦卡锡的 MIT 研究组，塞缪尔的 IBM 工程研究组等。

自这次会议之后的 10 多年间，人工智能的研究在机器学习、定理证明、模式识别、问题求解、专家系统及人工智能语言等方面都取得了许多引人注目的成就，例如：

① 在机器学习方面，1957 年 Rosenblatt 研制成功了感知机。这是一种将神经元用于识别的系统，它的学习功能引起了广泛的兴趣，推动了连接机制的研究，但人们很快发现了感知机的局限性。

② 在定理证明方面，美籍华人数理逻辑学家王浩于 1958 年在 IBM－704 机器上用 3～5 min 证明了《数学原理》中有关命题演算的全部定理（220 条），并且还证明了谓词演算中 150 条定理的 85%；1965 年鲁宾逊（J. A. Robinson）提出了归结原理，为定理的机器证明作出了突破性的贡献。

③ 在模式识别方面，1959 年塞尔夫里奇推出了一个模式识别程序；1965 年罗伯特（Roberts）编制出了可分辨积木构造的程序。

④ 在问题求解方面，1960 年纽厄尔等人通过心理学试验总结出了人们求解问题的思维规律，编制了通用问题求解程序 GPS，可以用来求解 11 种不同类型的问题。

⑤ 在专家系统方面，美国斯坦福大学的费根鲍姆（E. A. Feigenbaum）领导的研究小组自 1965 年开始专家系统 DENDRAL 的研究，1968 年完成并投入使用。该专家系统能根据质谱仪的实验，通过分析推理决定化合物的分子结构，其分析能力已接近，甚至超过有关化学专家的水平，在美、英等国得到了实际的应用。该专家系统的研制成功不仅为人们提供了一个实用的专家系统，而且对知识表示、存储、获取、推理及利用等技术是一次非常有益的探索，为以后专家系统的建造树立了榜样，对人工智能的发展产生了深刻的影响，其意义远远超过了系统本身在实用上所创造的价值。

⑥ 在人工智能语言方面，1960 年麦卡锡研制出了人工智能语言 LISP，成为建造专家系统的重要工具。

1969 年成立的国际人工智能联合会议（International Joint Conferences On Artificial Intelligence，IJCAI）是人工智能发展史上一个重要的里程碑，它标志着人工智能这门新兴学科已

经得到了世界的肯定和认可。1970 年创刊的国际性人工智能杂志《Artificial Intelligence》对推动人工智能的发展,促进研究者们的交流起到了重要的作用。

1.2.3 发展

这个阶段主要是指 1970 年以后。进入 20 世纪 70 年代,许多国家都开展了人工智能的研究,涌现了大量的研究成果。例如,1972 年法国马赛大学的科麦瑞尔(A. Comerauer)提出并实现了逻辑程序设计语言 PROLOG;斯坦福大学的肖特利夫(E. H. Shortliffe)等人从 1972 年开始研制用于诊断和治疗感染性疾病的专家系统 MYCIN。

但是,和其他新兴学科的发展一样,人工智能的发展道路也不是平坦的。例如,机器翻译的研究没有像人们最初想象的那么容易。当时人们总以为只要一部双向词典及一些词法知识就可以实现两种语言文字间的互译。后来发现机器翻译远非这么简单。实际上,由机器翻译出来的文字有时会出现十分荒谬的错误。例如,当把“眼不见,心不烦”的英语句子“Out of sight, out of mind.”翻译成俄语变成“又瞎又疯”;当把“心有余而力不足”的英语句子“The spirit is willing but the flesh is weak”翻译成俄语,然后再翻译回来时竟变成了“The wine is good but the meat is spoiled”,即“酒是好的,但肉变质了”;当把“光阴似箭”的英语句子“Time flies like an arrow”翻译成日语,然后再翻译回来的时候,竟变成了“苍蝇喜欢箭”。由于机器翻译出现的这些问题,1960 年美国政府顾问委员会的一份报告裁定:“还不存在通用的科学文本机器翻译,也没有很近的实现前景。”因此,英国、美国当时中断了对大部分机器翻译项目的资助。在其他方面,如问题求解、神经网络、机器学习等,也都遇到了困难,使人工智能的研究一时陷入了困境。

人工智能研究的先驱者们认真反思,总结前一段研究的经验和教训。1977 年费根鲍姆在第五届国际人工智能联合会议上提出了“知识工程”的概念,对以知识为基础的智能系统的研究与建造起到了重要的作用。大多数人接受了费根鲍姆关于以知识为中心展开人工智能研究的观点。从此,人工智能的研究又迎来了蓬勃发展的以知识为中心的新时期。

这个时期中,专家系统的研究在多种领域中取得了重大突破,各种不同功能、不同类型的专家系统如雨后春笋般地建立起来,产生了巨大的经济效益及社会效益。例如,地矿勘探专家系统 PROSPECTOR 拥有 15 种矿藏知识,能根据岩石标本及地质勘探数据对矿藏资源进行估计和预测,能对矿床分布、储藏量、品位及开采价值进行推断,制定合理的开采方案。应用该系统成功地找到了超亿美元的钼矿。专家系统 MYCIN 能识别 51 种病菌,正确地处理 23 种抗菌素,可协助医生诊断、治疗细菌感染性血液病,为患者提供最佳处方。该系统成功地处理了数百个病例,并通过了严格的测试,显示出了较高的医疗水平。美国 DEC 公司的专家系统 XCON 能根据用户要求确定计算机的配置。由专家做这项工作一般需要 3 小时,而该系统只需要 0.5 分钟,速度提高了 360 倍。DEC 公司还建立了另外一些专家系统,由此产生的净收益每年超过 4000 万美元。信用卡认证辅助决策专家系统 American Express 能够防止不应有的损失,据说每年可节省 2700 万美元左右。

专家系统的成功,使人们越来越清楚地认识到知识是智能的基础,对人工智能的研究必须以知识为中心来进行。对知识的表示、利用及获取等的研究取得了较大的进展,特别是对不确定性知识的表示与推理取得了突破,建立了主观 Bayes 理论、确定性理论、证据理论等,对人工智能中模式识别、自然语言理解等领域的发展提供了支持,解决了许多理论及技术上的问题。

人工智能在博弈中的成功应用也举世瞩目。人们对博弈的研究一直抱有极大的兴趣，早在1956 年人工智能刚刚作为一门学科问世时，塞缪尔就研制出了跳棋程序。这个程序能从棋谱中学习，也能从下棋实践中提高棋艺。1959 年它击败了塞缪尔本人，1962 年又击败了一个州的冠军。1991 年 8 月在悉尼举行的第 12 届国际人工智能联合会议上，IBM 公司研制的“深思”(Deep Thought)计算机系统就与澳大利亚象棋冠军约翰森(D. Johansen)举行了一场人机对抗赛，结果以 1:1 平局告终。1957 年西蒙曾预测 10 年内计算机可以击败人类的世界冠军。虽然在 10 年内没有实现，但 40 年后深蓝计算机击败国际象棋棋王卡斯帕罗夫(Kasparov)，仅仅比预测迟了 30 年。

1996 年 2 月 10 日至 17 日，为了纪念世界上第一台电子计算机诞生 50 周年，美国 IBM 公司出巨资邀请国际象棋棋王卡斯帕罗夫与 IBM 公司的深蓝计算机系统进行了六局的“人机大战”。这场比赛被人们称为“人脑与电脑的世界决战”。参赛的双方分别代表了人脑和电脑的世界最高水平。当时的深蓝是一台运算速度达每秒 1 亿次的超级计算机。第一盘，深蓝就给卡斯帕罗夫一个下马威，赢了这位世界冠军，给世界棋坛以极大的震动。但卡斯帕罗夫总结经验，稳扎稳打，在剩下的五盘中赢三盘，平两盘，最后以总比分 4:2 获胜。一年后，即 1997 年 5 月 3 日至 11 日，深蓝再次挑战卡斯帕罗夫。这时，深蓝是一台拥有 32 个处理器和强大并行计算能力的 RS/6000 SP/2 的超级计算机，运算速度达每秒 2 亿次。计算机里存储了百余年来世界顶尖棋手的棋局。5 月 3 日棋王卡斯帕罗夫首战击败深蓝，5 月 4 日深蓝扳回一盘，之后双方战平三局。双方的决胜局于 5 月 11 日拉开了帷幕，卡斯帕罗夫在这盘比赛中仅仅走了 19 步便放弃了抵抗，比赛用时只有 1 小时多一点。这样，深蓝最终以 3.5:2.5 的总比分赢得这场举世瞩目的“人机大战”的胜利。深蓝的胜利表明了人工智能所达到的成就。尽管它的棋路还远非真正地对人类思维方式的模拟，但它已经向世人说明，电脑能够以人类远远不能企及的速度和准确性，实现属于人类思维的大量任务。深蓝精湛的残局战略使观战的国际象棋专家们大为惊讶。卡斯帕罗夫也表示：“这场比赛中有许多新的发现，其中之一就是计算机有时也可以走出人性化的棋步。在一定程度上，我不能不赞扬这台机器，因为它对盘势因素有着深刻的理解，我认为这是一项杰出的科学成就。”因为这场胜利，IBM 的股票升值为 180 亿美元。

我国自 1978 年开始也把“智能模拟”作为国家科学技术发展规划的主要研究课题之一，并在1981 年成立了中国人工智能学会(CAAI)，目前在专家系统、模式识别、机器人学及汉语的机器理解等方面都取得了很多研究成果。

1.3　人工智能研究的基本内容

1. 知识表示

世界上的每一个国家或民族几乎都有自己的语言和文字。它是人们表达思想、交流信息的工具。它促进了人类的文明及社会的进步。人类语言和文字是人类知识表示的最优秀、最通用的方法，但人类语言和文字的知识表示方法并不适合于计算机处理。

人工智能研究的目的是要建立一个能模拟人类智能行为的系统，但知识是一切智能行为的基础，因此首先要研究知识表示方法。只有这样才能把知识存储到计算机中去，供求解现实问题使用。

对于知识表示方法的研究,离不开对知识的研究与认识。由于目前对人类知识的结构及机制还没有完全搞清楚,因此关于知识表示的理论及规范尚未建立起来。尽管如此,人们在对智能系统的研究及建立过程中,还是结合具体研究提出了一些知识表示方法。

知识表示方法可分为如下两大类:符号表示法和连接机制表示法。

符号表示法是用各种包含具体含义的符号,以各种不同的方式和顺序组合起来表示知识的一类方法。它主要用来表示逻辑性知识,本书第 2 章中将要介绍的各种知识表示方法都属于这一类。目前用得较多的知识表示方法有:一阶谓词逻辑表示法、产生式表示法、框架表示法、语义网络表示法、状态空间表示法、神经网络表示法、脚本表示法、过程表示法、Petri 网络表示法及面向对象表示法等。

连接机制表示法是用神经网络表示知识的一种方法。它把各种物理对象以不同的方式及顺序连接起来,并在其间互相传递及加工各种包含具体意义的信息,以此来表示相关的概念及知识。相对于符号表示法而言,连接机制表示法是一种隐式的知识表示方法。在这里,知识并不像在产生式系统中表示为若干条规则,而是将某个问题的若干知识在同一个网络中表示。因此,特别适用于表示各种形象性的知识。

2. 机器感知

所谓机器感知就是使机器(计算机)具有类似于人的感知能力,其中以机器视觉(machine vision)与机器听觉为主。机器视觉是让机器能够识别并理解文字、图像、物景等;机器听觉是让机器能识别并理解语言、声响等。

机器感知是机器获取外部信息的基本途径,是使机器具有智能不可缺少的组成部分,正如人的智能离不开感知一样,为了使机器具有感知能力,就需要为它配置上能“听”、会“看”的感觉器官。对此,人工智能中已经形成了两个专门的研究领域,即模式识别与自然语言理解。

3. 机器思维

所谓机器思维是指对通过感知得来的外部信息及机器内部的各种工作信息进行有目的的处理。正如人的智能是来自大脑的思维活动一样,机器智能也主要是通过机器思维实现的。因此,机器思维是人工智能研究中最重要、最关键的部分。它使机器能模拟人类的思维活动,能像人那样既可以进行逻辑思维,又可以进行形象思维。

4. 机器学习

知识是智能的基础,要使计算机有智能,就必须使它有知识。人们可以把有关知识归纳、整理在一起,并用计算机可接受、处理的方式输入到计算机中去,使计算机具有知识。显然,这种方法不能及时地更新知识,特别是计算机不能适应环境的变化。为了使计算机具有真正的智能,必须使计算机像人类那样,具有获得新知识、学习新技巧并在实践中不断完善、改进的能力,实现自我完善。

机器学习(machine learning)就是研究如何使计算机具有类似于人的学习能力,使它能通过学习自动地获取知识。计算机可以直接向书本学习,通过与人谈话学习,能通过对环境的观察学习,并在实践中实现自我完善。

机器学习是一个难度较大的研究领域,它与脑科学、神经心理学、计算机视觉、计算机听觉等都有密切联系,依赖于这些学科的共同发展。因此,经过近些年的研究,虽然已经取得了很大的进展,提出了很多学习方法,但并未从根本上解决问题。

5. 机器行为

与人的行为能力相对应，机器行为主要是指计算机的表达能力，即“说”、“写”、“画”等能力。对于智能机器人，它还应具有人的四肢功能，即能走路、能取物、能操作等。

1.4　人工智能的主要研究领域

目前，随着智能科学与技术的发展和计算机网络技术的广泛应用，人工智能技术应用到越来越多的领域。下面简要介绍几个主要研究领域。

1. 自动定理证明

自动定理证明是人工智能中最先进行研究并得到成功应用的一个研究领域，同时它也为人工智能的发展起到了重要的推动作用。实际上，除了数学定理证明以外，医疗诊断、信息检索、问题求解等许多非数学领域问题，都可以转化为定理证明问题。

定理证明的实质是证明由前提 P 得到结论 Q 的永真性。但是，要直接证明 $P \rightarrow Q$ 的永真性一般来说是很困难的。通常采用的方法是反证法。在这方面海伯伦（Herbrand）与鲁宾逊（Robinson）先后进行了卓有成效的研究，提出了相应的理论及方法，为自动定理证明奠定了理论基础。尤其是鲁宾逊提出的归结原理使定理证明得以在计算机上实现，对机器推理作出了重要贡献。我国吴文俊院士提出并实现的几何定理机器证明“吴氏方法”，是机器定理证明领域的一项标志性成果。

2. 博弈

诸如下棋、打牌、战争等一类竞争性的智能活动称为博弈（game playing）。下棋是一个斗智的过程，不仅要求参赛者具有超凡的记忆能力、丰富的下棋经验，而且要求有很强的思维能力，能对瞬息万变的随机情况迅速地作出反应，及时采取有效的措施。对于人类来说，博弈是一种智能性很强的竞争活动。

人工智能研究博弈的目的并不是为了让计算机与人进行下棋、打牌之类的游戏，而是通过对博弈的研究来检验某些人工智能技术是否能实现对人类智慧的模拟，促进人工智能技术的深入研究。正如俄罗斯人工智能学者亚历山大·克隆罗德所说“象棋是人工智能中的果蝇”，将象棋在人工智能研究中的作用类比于果蝇在生物遗传研究中作为实验对象所起的作用。

3. 模式识别

模式识别（pattern recognition）是一门研究对象描述和分类方法的学科。分析和识别的模式可以是信号、图像或者普通数据。

模式是对一个物体或者某些其他感兴趣实体定量的或者结构的描述，而模式类是指具有某些共同属性的模式集合。用机器进行模式识别的主要内容是研究一种自动技术，依靠这种技术，机器可以自动地或者尽可能少需要人工干预地把模式分配到它们各自的模式类中去。

传统的模式识别方法有统计模式识别和结构模式识别等类型。近年来迅速发展的模糊数学及人工神经网络技术已经应用到模式识别中，形成模糊模式识别、神经网络模式识别等方法，展示了巨大的发展潜力。

4. 机器视觉

机器视觉（machine vision）或者计算机视觉（computer vision）是用机器代替人眼进行测量和

判断,是模式识别研究的一个重要方面。计算机视觉通常分为低层视觉与高层视觉两类。低层视觉主要执行预处理功能,如边缘检测、移动目标检测、纹理分析,以及立体造型、曲面色彩等。主要目的是使得看见的对象更突出。这时还不是理解阶段。高层视觉主要是理解对象,需要掌握与对象相关的知识。机器视觉的前沿课题包括:实时图像的并行处理,实时图像的压缩、传输与复原,三维景物的建模识别,动态和时变视觉等。

机器视觉系统是指通过图像摄取装置将被摄取的目标转换成图像信号,传送给专用的图像处理系统,根据像素分布和宽度、颜色等信息,转换成数字信号,图像系统对这些信号进行各种运算,抽取目标的特征,进而根据判别的结果来控制现场的设备动作。机器视觉的主要研究目标是使计算机具有通过二维图像认知三维环境信息的能力,能够感知与处理三维环境中物体的形状、位置、姿态、运动等几何信息。

机器视觉与模式识别存在很大程度的交叉性,两者的主要区别是机器视觉更注重三维视觉信息的处理,而模式识别仅仅关心模式的类别。此外,模式识别还包括听觉等非视觉信息。

在国外,机器视觉的应用相当普及,主要集中在电子、汽车、冶金、食品饮料、零配件装配及制造等行业。机器视觉系统在质量检测的各个方面已经得到广泛的应用。在国内,由于近年来机器视觉产品刚刚起步,目前主要集中在制药、印刷、包装、食品饮料等行业。但随着国内制造业的快速发展,对于产品检测和质量的要求不断提高,各行各业对图像和机器视觉技术的工业自动化需求将越来越大,因此机器视觉在未来制造业中将会有很大的发展空间。

5. 自然语言理解

目前人们使用计算机时,大多是用计算机的高级语言(如 C、Java 等语言)编制程序来告诉计算机“做什么”以及“怎么做”的。这对计算机的利用带来了诸多不便,严重阻碍了计算机应用的进一步推广。如果能让计算机“听懂”、“看懂”人类语言(如汉语、英语等),那将使计算机具有更广泛的用途,特别是大大推进机器人技术的发展。自然语言理解(natural language understanding)就是研究如何让计算机理解人类自然语言,是人工智能中十分重要的一个研究领域。它是研究能够实现人与计算机之间用自然语言进行通信的理论与方法。具体地说,它要达到如下三个目标:

① 计算机能正确理解人们用自然语言输入的信息,并能正确回答输入信息中的有关问题。

② 对输入的自然语言信息,计算机能够产生相应的摘要,能用不同词语复述输入信息的内容。

③ 计算机能把用某一种自然语言表示的信息自动翻译为用另一种自然语言表示的相同信息。

关于自然语言理解的研究可以追溯到 20 世纪 50 年代初期。当时由于通用计算机的出现,人们开始考虑用计算机把一种语言翻译成另一种语言的可能性,在此之后的 10 多年中,机器翻译一直是自然语言理解中的主要研究课题。起初,主要是进行“词对词”的翻译,当时人们认为翻译工作只要进行“查词典”及简单的“语法分析”就可以了,即对一篇要翻译的文章,首先通过查词典找出两种语言间的对应词,然后经过简单的语法分析调整词序就可以实现翻译。出于这一认识,人们把主要精力用于在计算机内构造不同语言对照关系的词典上。但是这种方法并未达到预期的效果,以致闹出了一些阴差阳错、颠三倒四的笑话,正像本书在前面列举的一些例子那样。

进入 20 世纪 70 年代后，一批采用语法 - 语义分析技术的自然语言理解系统脱颖而出，在语言分析的深度和难度方面都比早期的系统有了长足的进步。这期间，有代表性的系统主要有维诺格拉德（T. Winograd）于 1972 年研制的 SHRDLU，伍德（W. Woods）于 1972 年研制的 LUNAR，夏克（R. Schank）于 1973 年研制的 MARGIE 等。其中，SHRDLU 是一个在“积木世界”中进行英语对话的自然语言理解系统，系统模拟一个能操作桌子上一些玩具积木的机器人手臂，用户通过与计算机对话命令机器人操作积木块，如让它拿起、放下某个积木等。LUNAR 是一个用来协助地质学家查找、比较和评价阿波罗 - 11 飞船带回来的月球岩石和土壤标本化学分析数据的系统，是第一个实现了用普通英语与计算机对话的人机接口系统。MARGIE 是夏克根据概念依赖理论建成的一个心理学模型，目的是研究自然语言理解的过程。

进入 20 世纪 80 年代后，更强调知识在自然语言理解中的重要作用，1990 年 8 月在赫尔辛基召开的第 13 届国际计算机语言学大会上，首次提出了处理大规模真实文本的战略目标，并组织了“大型语料库在建造自然语言系统中的作用”、“词典知识的获取与表示”等专题讲座，预示着语言信息处理的一个新时期的到来。

近 10 年来，在自然语言理解的研究中，一个值得注意的事件是语料库语言学（Corpus Linguistics）的崛起，它认为语言学知识来自于语料，人们只有从大规模语料库中获取理解语言的知识，才能真正实现对语言的理解。目前，基于语料库的自然语言理解方法还不成熟，正处于研究之中，但它是一个应引起重视的研究方向。

6. 智能信息检索

数据库系统是存储大量信息的计算机系统。随着计算机应用的发展，存储的信息量越来越庞大，研究智能信息检索系统具有重要的理论意义和实际应用价值。

智能信息检索系统应具有下述功能：

① 能理解自然语言。允许用户使用自然语言提出检索要求和询问。

② 具有推理能力。能根据数据库存储的事实，推理产生用户要求和询问的答案。

③ 系统拥有一定的常识性知识。系统根据这些常识性知识和专业知识能演绎推理出专业知识中没有包含的答案。例如，某单位的人事档案数据库中有下列事实：“张强是采购部工作人员”、“李明是采购部经理”。如果系统具有“部门经理是该部门工作人员的领导”这一常识性知识，就可以对询问“谁是张强的领导”演绎推理出答案“李明”。

7. 数据挖掘与知识发现

随着计算机网络的飞速发展，计算机处理的信息量越来越大。数据库中包含的大量信息无法得到充分的利用，造成信息浪费，甚至变成大量的数据垃圾。因此，人们开始考虑以数据库作为新的知识源。数据挖掘（data mining）和知识发现（knowledge discovery）是 20 世纪 90 年代初期崛起的一个活跃的研究领域。

知识发现系统通过各种学习方法，自动处理数据库中大量的原始数据，提炼出具有必然性的、有意义的知识，从而揭示出蕴涵在这些数据背后的内在联系和本质规律，实现知识的自动获取。知识发现是从数据库中发现知识的全过程，而数据挖掘则是这个全过程的一个特定的、关键的步骤。

数据挖掘的目的是从数据库中找出有意义的模式。这些模式可以是一组规则、聚类、决策树、依赖网络或以其他方式表示的知识。一个典型的数据挖掘过程可以分成 4 个阶段，即数据预

处理、建模、模型评估及模型应用。数据预处理阶段主要包括数据的理解、属性选择、连续属性离散化、数据中噪声及丢失值处理、实例选择等。建模包括学习算法的选择、算法参数的确定等。模型评估是进行模型训练和测试,对得到的模型进行评价。在得到满意的模型后,就可以运用此模型对新数据进行解释。

知识获取是人工智能的关键问题之一。因此,知识发现和数据挖掘成为当前人工智能的一个研究热点。

8. 专家系统

专家系统(expert system)是目前人工智能中最活跃、最有成效的一个研究领域。自费根鲍姆等研制出第一个专家系统 DENDRAL 以来,它已获得了迅速的发展,广泛地应用于医疗诊断、地质勘探、石油化工、教学及军事等各个方面,产生了巨大的社会效益和经济效益。

专家系统是一个智能的计算机程序,运用知识和推理步骤来解决只有专家才能解决的疑难问题。因此,可以这样来定义:专家系统是一种具有特定领域内大量知识与经验的程序系统,它应用人工智能技术模拟人类专家求解问题的思维过程来求解领域内的各种问题,其水平可以达到甚至超过人类专家的水平。

9. 自动程序设计

自动程序设计是将自然语言描述的程序自动转换成可执行程序的技术。自动程序设计与一般的编译程序不同,编译程序只能把用高级程序设计语言编写的源程序翻译成目标程序,而不能处理自然语言类的高级形式语言。

自动程序设计包括程序综合与程序正确性验证两个方面的内容。程序综合用于实现自动编程,即用户只需要告诉计算机要“做什么”,无须说明“怎么做”,计算机就可自动实现程序的设计。程序正确性验证是要研究出一套理论和方法,通过运用这套理论和方法就可以证明程序的正确性。目前常用的验证方法是用一组已知其结果的数据对程序进行测试,如果程序的运行结果与已知结果一致,就认为程序是正确的。这种方法对于简单程序来说尚可,但对于一个复杂系统来说就很难行得通。因为复杂程序总存在着纵横交错的复杂关系,形成难以计数的通路,用于测试的数据即使很多,也难以保证对每一条通路都能进行测试,这就不能保证程序的正确性。程序正确性验证至今仍是一个比较困难的课题,有待进一步开展研究。自动程序设计是人工智能与软件工程相结合的课题。

10. 机器人

机器人是指可模拟人类行为的机器。人工智能的所有技术几乎都可以在它身上得到应用,因此,它可作为人工智能理论、方法、技术的实验场地。反过来,对机器人的研究又可大大地推动人工智能研究的发展。

自 20 世纪 60 年代初研制出尤尼梅特和沃莎特兰这两种机器人以来,机器人的研究已经从低级到高级经历了三代的发展历程。

程序控制机器人(第一代):第一代机器人是程序控制机器人,它完全按照事先装入到机器人存储器中的程序安排的步骤进行工作。程序的生成及装入有两种方式,一种是由人根据工作流程编制程序并将它输入到机器人的存储器中;另一种是“示教 - 再现”方式,所谓“示教”是指在机器人第一次执行任务之前,由人引导机器人去执行操作,即教机器人去做应做的工作,机器人将其所有动作一步步地记录下来,并将每一步表示为一条指令,示教结束后机器人通过执行这

些指令以同样的方式和步骤完成同样的工作(即再现)。如果任务或环境发生了变化,则要重新进行程序设计。这一代机器人能成功地模拟人的运动功能,它们会拿取和安放、会拆卸和安装、会翻转和抖动,能尽心尽职地看管机床、熔炉、焊机、生产线等,能有效地从事安装、搬运、包装、机械加工等工作。目前国际上商品化、实用化的机器人大都属于这一类。这一代机器人的最大缺点是它只能刻板地完成程序规定的动作,不能适应变化了的情况,环境情况略有变化(如装配线上的物品略有倾斜),就会出现问题。更糟糕的是它会对现场的人员造成危害,由于它没有感觉功能,有时会出现机器人伤人的情况。日本就曾经出现机器人把现场的一个工人抓起来塞到刀具下面的情况。

自适应机器人(第二代):第二代机器人的主要标志是自身配备有相应的感觉传感器,如视觉传感器、触觉传感器、听觉传感器等,并用计算机对其进行控制。这种机器人通过传感器获取作业环境、操作对象的简单信息,然后由计算机对获得的信息进行分析、处理、控制机器人的动作。由于它能随着环境的变化而改变自己的行为,故称为自适应机器人。目前,这一代机器人也已进入商品化阶段,主要从事焊接、装配、搬运等工作。第二代机器人虽然具有一些初级的智能,但还没有达到完全"自治"的程度,有时也称这类机器人为人 - 眼协调型机器人。

智能机器人(第三代):这是指具有类似于人的智能的机器人,即它具有感知环境的能力,配备有视觉、听觉、触觉、嗅觉等感觉器官,能从外部环境中获取有关信息;具有思维能力,能对感知到的信息进行处理,以控制自己的行为;具有作用于环境的行为能力,能通过传动机构使自己的"手"、"脚"等肢体行动起来,正确、灵巧地执行思维机构下达的命令。目前研制的机器人大都只具有部分智能,真正的智能机器人还处于研究之中。

11. 组合优化问题

有许多实际问题属于组合优化问题。例如,旅行商问题、生产计划与调度、通信路由调度等都是属于这一类问题。

组合优化问题一般是 NP 完全问题。NP 完全问题是指:用目前知道的最好的方法求解,问题求解需要花费的时间(或称为问题求解的复杂性)是随问题规模增大以指数关系增长。至今还不知道对 NP 完全问题是否有花费时间较少的求解方法。例如,可使求解时间随问题规模按多项式关系增长。

随着求解问题规模的增大,问题求解程序的复杂性(用于求解程序运行所需的时间和空间或求解步数)可随问题规模按线性关系、多项式关系或指数关系增长,大多数组合优化问题求解程序都面临着组合爆炸问题。因此,经典的优化方法难以求解大规模组合优化问题,需要研究人工智能求解方法。目前已经提出了许多有效的方法,特别是遗传算法、神经网络方法等。

组合优化问题的求解方法已经应用于生产计划与调度、通信路由调度、交通运输调度、列车组编、空中交通管制和军事指挥自动化等系统。

12. 人工神经网络

人工神经网络是一个用大量简单处理单元经广泛连接而组成的人工网络,用来模拟大脑神经系统的结构和功能。早在 1943 年,神经和解剖学家麦克洛奇(W. S. McCulloch)和数学家匹兹(W. Pitts)就提出了神经元的数学模型(M - P 模型),从此开创了神经科学理论研究的时代。20 世纪 60 年代至 70 年代,由于神经网络研究自身的局限性,致使其研究陷入了低潮。特别是著名人工智能学者明斯基(Minsky)等人在 1969 年以批评的观点编写的很有影响的《感知器》一

书,直接导致了神经网络的研究进入萧条时期。具有讽刺意味的是 BP 学习算法已经在 1969 年由 Bryson 和 Ho 首次提出。到 20 世纪 80 年代,对神经网络的研究取得突破性进展,特别是鲁梅尔哈特(Rumelhart)等人提出多层前向神经网络的 BP 学习算法,霍普菲尔德(J. J. Hopfield)提出霍普菲尔德神经网络模型,有力地推动了神经网络的研究,由此人工神经网络的研究进入了一个新的发展时期,取得了许多研究成果。具有讽刺意味的是实际上 Bryson 和 Ho 在 1969 年就已经提出了类似算法。

现在,神经网络已经成为人工智能中一个极其重要的研究领域。对神经网络模型、算法、理论分析和硬件实现的大量研究,为神经计算机走向应用提供了物质基础。神经网络已经在模式识别、图像处理、组合优化、自动控制、信息处理、机器人学等领域获得日益广泛的应用。

13. 分布式人工智能与多智能体

分布式人工智能(Distributed Artificial Intelligence, DAI)是分布式计算与人工智能结合的结果。分布式人工智能系统以鲁棒性作为控制系统质量的标准,并具有互操作性,即不同的异构系统在快速变化的环境中,具有交换信息和协同工作的能力。

分布式人工智能的研究目标是要创建一种描述自然系统和社会系统的模型。DAI 中的智能并非独立存在,只能在团体协作中实现,因而其主要研究问题是各智能体(Agent)之间的合作与对话,包括分布式问题求解和多智能体系统(multi-agent system, MAS)两个领域。分布式问题求解把一个具体的求解问题划分为多个相互合作和知识共享的模块或者结点。多智能体系统则研究各智能体之间行为的协调。这两个研究领域都要研究知识、资源和控制的划分问题。但分布式问题求解往往含有一个全局的概念模型、问题和成功标准。而 MAS 则含有多个局部的概念模型、问题和成功标准。多智能体系统更能够体现人类的社会智能,具有更大的灵活性和适应性,更适合开放和动态的世界环境,成为人工智能领域的研究热点。

14. 智能控制

智能控制就是把人工智能技术引入控制领域,建立智能控制系统。自从国际知名美籍华裔科学家傅京孙(K. S. Fu)在 1965 年首先提出把人工智能的启发式推理规则用于学习控制系统以来,国内外众多的研究者投身于智能控制系统,并取得一些成果。经过 20 年努力,到 20 世纪 80 年代中期,智能控制新学科的形成条件已经逐渐成熟。1985 年 8 月,IEEE 在美国纽约召开了新一届智能控制学术讨论会。会上集中讨论了智能控制原理和智能控制系统的结构。1987 年 1 月,在美国费城 IEEE 控制系统学会和计算机学会联合召开了智能控制国际学术讨论会。会议显示出智能控制的长足进展,也说明了高新技术的发展要求重新考虑自动控制科学及其相关领域。这次会议表明,智能控制已作为一门新学科,出现在国际科学舞台上。

智能控制具有两个显著的特点:

① 智能控制是同时具有知识表示的非数学广义世界模型和传统数学模型混合表示的控制过程,也往往是含有复杂性、不完全性、模糊性或不确定性以及不存在已知算法的过程,并以知识进行推理,以启发来引导求解过程。

② 智能控制的核心在高层控制,即组织级控制,其任务在于实际环境或过程进行组织,即决策与规划,以实现广义问题求解。

智能控制系统的智能可归纳为以下几方面:

① 先验智能。有关控制对象及干扰的先验知识,可以从一开始就考虑到控制系统的设

计中。

② 反应性智能。在实时监控、辨识及诊断的基础上，对系统及环境变化的正确反应能力。

③ 优化智能。包括对系统性能的先验性优化及反应性优化。

④ 组织与协调智能。表现为对并行耦合任务或子系统之间的有效管理与协调。

智能控制的开发，目前认为有以下途径：

① 基于专家系统的专家智能控制。

② 基于模糊推理和计算的模糊控制。

③ 基于人工神经网络的神经网络控制。

④ 综合以上三种方法的综合型智能控制。

15. 智能仿真

智能仿真就是将人工智能技术引入仿真领域，建立智能仿真系统，人们知道，仿真是对动态模型的实验，即行为产生器在规定的实验条件下驱动模型，从而产生模型行为。具体地说，仿真是在三种类型知识——描述性知识、目的性知识及处理知识的基础上产生另一种形式的知识——结论性知识。因此可以将仿真看作是一个特殊的知识变换器，从这个意义上讲，人工智能与仿真有着密切的关系。

利用人工智能技术能对整个仿真过程（包括建模、实验运行及结果分析）进行指导，能改善仿真模型的描述能力，在仿真模型中引进知识表示将为研究面向目标的建模语言打下基础，提高仿真工具面向用户、面向问题的能力。从另一方面讲，仿真与人工智能相结合可使仿真更有效地用于决策，更好地用于分析、设计及评价知识库系统，从而推动人工智能技术的发展。正是基于这些方面，近年来，将人工智能特别是专家系统与仿真相结合，已成为仿真领域中一个十分重要的研究方向，引起了大批仿真专家的关注。

16. 智能 CAD

智能 CAD（ICAD）就是把人工智能技术引入计算机辅助设计领域，建立智能 CAD 系统。事实上，AI 几乎可以应用到 CAD 技术的各个方面。从目前发展的趋势来看，至少有下述四个方面：

① 设计自动化。

② 智能交互。

③ 智能图形学。

④ 自动数据采集。

从具体技术来看，ICAD 技术大致可以分为如下几种方法：

① 规则生成法。

② 约束满足方法。

③ 搜索法。

④ 知识工程方法。

⑤ 形象思维方法。

17. 智能 CAI

智能 CAI 就是把人工智能技术引入计算机辅助教学领域，建立智能 CAI 系统即 ICAI。ICAI 的特点是能对学生因材施教地进行指导。为此，ICAI 应具备下列智能特征：

① 自动生成各种问题与练习。

② 根据学生的水平和学习情况自动选择与调整教学内容与进度。

③ 在理解教学内容的基础上自动解决问题生成解答。

④ 具有自然语言生成和理解能力。

⑤ 对教学内容有理解咨询能力。

⑥ 能诊断学生错误,分析原因并采取纠正措施。

⑦ 能评价学生的学习行为。

⑧ 能不断地在教学中改善教学策略。

为了实现上述 ICAI 系统,一般把整个系统分成专门知识、教导策略和学生模型等三个基本模块和一个自然语言的智能接口。

总之,ICAI 已是人工智能的一个重要应用领域和研究方向,引起了人工智能界和教育界的极大关注和共同兴趣。特别是 20 世纪 80 年代以来,由于知识工程、专家系统技术的发展,使得 ICAI 与专家系统的关系日益密切。近几届美国与国际人工智能会议都把 ICAI 的研究列入议程,甚至还召开了专门的智能教学系统会议。

18. 智能管理与智能决策

智能管理是现代管理科学技术发展的新动向。智能管理是人工智能与管理科学、系统工程、计算机技术及通信技术等多学科互相结合、互相渗透而产生的一门新学科。

智能管理就是把人工智能技术引入管理领域,建立智能管理系统,研究如何提高计算机管理系统的智能水平,以及智能管理系统的设计理论、方法与实现技术。

智能管理系统是在管理信息系统、办公自动化系统、决策支持系统的功能集成和技术集成的基础上,应用人工智能专家系统、知识工程、模式识别、人工神经网络等方法和技术,进行智能化、集成化、协调化,进而设计和实现的新一代的计算机管理系统。

智能决策就是把人工智能技术引入决策过程,建立智能决策支持系统。智能决策支持系统是在 20 世纪 80 年代初提出来的。它是决策支持系统与人工智能(特别是专家系统中知识及知识处理的特长)的结合,既可以进行定量分析,又可以进行定性分析,能有效地解决半结构化和非结构化的问题。从而,扩大了决策支持系统的范围,提高了决策支持系统的能力。

智能决策支持系统是在传统决策支持系统的基础上发展起来的,传统决策支持系统再加上相应的智能部件就构成了智能决策支持系统。智能部件可以有多种模式,如专家系统模式、知识库模式等。专家系统模式是把专家系统作为智能部件,这是目前比较流行的一种模式。该模式适合于以知识处理为主的问题,但它与决策支持系统的接口比较困难。知识库系统模式是以知识库作为智能部件。在这种情况下,决策支持系统就是由模型库、方法库、知识库、数据库组成的四库系统。这种模式接口比较容易实现,其整体性能也较好。

一般来说,智能部件中可以包含如下一些知识:

① 建立决策模型和评价模型的知识。

② 如何形成候选方案的知识。

③ 建立评价标准的知识。

④ 如何修正候选方案,从而得到更好候选方案的知识。

⑤ 完善数据库,改进对它的操作及维护知识。

19. 智能多媒体系统

多媒体技术是当前计算机最为热门的研究领域之一。多媒体计算机系统就是能综合处理文字、图形、图像和声音等多种媒体信息的计算机系统。智能多媒体就是将人工智能技术引入多媒体系统,使其功能和性能得到进一步发展和提高。事实上,多媒体技术与人工智能所研究的机器感知、机器理解等技术也不谋而合。所以,智能多媒体实际上是人工智能与多媒体技术的有机结合。人工智能的计算机视听觉、语音识别与理解、语音对译、信息智能压缩等技术运用于多媒体系统,将会使现在的多媒体系统产生质的飞跃。目前,基于视频的动画技术、对环境感知的动画生成、虚拟中文打字机等都成为热点研究课题。

20. 智能操作系统

智能操作系统就是将人工智能技术引入计算机的操作系统之中,从质上提高操作系统的性能和效率。

智能操作系统的基本模型是以智能机为基础,并能支撑外层的 AI 应用程序,实现多用户的知识处理和并行推理。智能操作系统主要有三大特点:并行性,分布性和智能性。并行性是指能够支持多用户、多进程,同时进行逻辑推理和知识处理。分布性是指把计算机的硬件和软件资源分散而又有联系地组织起来,能支持局域网和远程网处理。智能性又体现于三个方面:一是操作系统所处理的对象是知识对象,具有并行推理和知识操作功能,支持智能应用程序的运行;二是操作系统本身的绝大部分程序也将使用 AI 程序(规则和事实)编制,充分利用硬件并行推理功能;三是其系统管理应具有较高智能程序的自动管理维护功能,如故障的监控分析等,以帮助系统维护人员作出必要的决策。

21. 智能计算机系统

智能计算机系统就是人们正在研制的新一代计算机系统。这种计算机系统从基本元件到体系结构,从处理对象到编程语言,从使用方法到应用范围,同当前的冯·诺依曼计算机相比,都有质的飞跃,它将全面支持智能应用开发,且自身就具有智能。

22. 智能通信

智能通信就是把人工智能技术引入通信领域,建立智能通信系统。智能通信就是在通信系统的各个层次和环节上实现智能化。例如,在通信网的构建、网管与网控、转接、信息传输与转换等环节,都可实现智能化。这样,网络就可运行在最佳状态,具有自适应、自组织、自学习、自修复等功能。

23. 智能网络系统

智能网络系统就是将人工智能技术引入计算机网络系统。如在网络构建、网络管理与控制、信息检索与转换、人机接口等环节,运用 AI 的技术与成果。研究表明,AI 的专家系统、模糊技术和神经网络技术可用于网络的连接接纳控制、业务量管制、业务量预测、资源动态分配、业务流量控制、动态路由选择、动态缓冲资源调度等许多方面。

24. 人工生命

1987 年,计算机科学家克里斯·兰顿(Christopher Langton)博士在美国洛斯阿拉莫斯国家实验室(Los Alamos National Laboratory)召开的“生成以及模拟生命系统的国际会议”上首先提出人工生命(artificial life,AL)的概念。

人工生命是以计算机为研究工具,模拟自然界的生命现象,生成表现自然生命系统行为特点

的仿真系统。该领域主要研究天体生物学、宇宙生物学、自催化系统、分子自装配系统、分子信息处理等的生命自组织和自复制，研究多细胞发育、基因调节网络、自然和人工的形态形成理论、生命系统的复杂性，研究进化的模式和方式、人工仿生学、进化博弈、分子进化、免疫系统进化、学习等，研究具有自治性、智能性、反应性、预动性和社会性的智能主体的形式化模型、通信方式、协作策略，研究生物感悟的机器人、自治和自适应机器人、进化机器人、人工脑。

1.5 小结

人类智能是自然界四大奥秘之一，很难给出确切的定义，目前有思维理论、知识阈值理论、进化理论等学派。简单地说，智能是知识与智力的总和。知识是一切智能行为的基础，智力是获取知识并应用知识求解问题的能力。

智能具有感知能力、记忆与思维能力、学习能力、行为能力等显著特征。

人工智能是用人工的方法在机器（计算机）上实现的智能。人工智能的发展历史，可归结为孕育、形成和发展三个阶段。

人工智能研究的基本内容为知识表示、机器感知、机器思维、机器学习、机器行为等几个方面。

思考题

1.1 什么是人类智能？它有哪些特点？

1.2 什么是人工智能？它的发展过程经历了哪些阶段？

1.3 人工智能研究的基本内容有哪些？

1.4 人工智能有哪些主要的研究领域？

第2章 知识表示

人类的智能活动主要是获得并运用知识。知识是智能的基础。为了使计算机具有智能，能模拟人类的智能行为，就必须使它具有知识。但知识需要用适当的模式表示出来才能存储到计算机中去，因此，知识的表示成为人工智能中一个十分重要的研究课题。

本章将首先介绍知识与知识表示的概念，然后介绍一阶谓词逻辑、产生式、框架、语义网络等当前人工智能中应用比较广泛的知识表示方法，为后面介绍推理方法、专家系统等奠定基础。

2.1 知识与知识表示的概念

2.1.1 知识的概念

知识是人们在长期的生活及社会实践中、在科学研究及实验中积累起来的对客观世界的认识与经验。人们把实践中获得的信息关联在一起，就形成了知识。一般来说，把有关信息关联在一起所形成的信息结构称为知识。信息之间有多种关联形式，其中用得最多的一种是用“如果……，则……”表示的关联形式。它反映了信息间的某种因果关系。例如，在我国北方，人们经过多年的观察发现，每当冬天要来临的时候，就会看到一群群的大雁向南方飞去，于是把“大雁向南飞”与“冬天就要来临了”这两个信息关联在一起，就得到了如下知识：如果大雁向南飞，则冬天就要来临了。

知识反映了客观世界中事物之间的关系，不同事物或者相同事物间的不同关系形成了不同的知识。例如，“雪是白色的”是一条知识，它反映了“雪”与“白色”之间的一种关系。又如“如果头痛且流涕，则有可能患了感冒”是一条知识，它反映了“头痛且流涕”与“可能患了感冒”之间的一种因果关系。在人工智能中，把前一种知识称为“事实”，而把后一种知识，即用“如果……，则……”关联起来所形成的知识称为“规则”。在下面将对它们作进一步介绍。

2.1.2 知识的特性

1. 相对正确性

知识是人类对客观世界认识的结晶，并且受到长期实践的检验。因此，在一定的条件及环境下，知识一般是正确的。这里，“一定的条件及环境”是必不可少的，它是知识正确性的前提。因

为任何知识都是在一定的条件及环境下产生的，因而也就只有在这种条件及环境下才是正确的。例如，牛顿力学在一定的条件下才是正确的。再如，1 + 1 = 2，这是一条妇孺皆知的正确知识，但它也只是在十进制的前提下才是正确的，如果是二进制，它就不正确了。

在人工智能中，知识的相对正确性更加突出。除了人类知识本身的相对正确性外，在建造专家系统时，为了减少知识库的规模，通常将知识限制在所求解问题的范围内。也就是说，只要这些知识对所求解的问题是正确的就行。例如，在动物识别系统中，如果仅仅识别虎、金钱豹、斑马、长颈鹿、企鹅、鸵鸟、信天翁等七种动物，那么，知识"IF 该动物是鸟 AND 善飞，THEN 该动物是信天翁"就是正确的。

2. 不确定性

由于现实世界的复杂性，信息可能是精确的，也可能是不精确的、模糊的；关联可能是确定的，也可能是不确定的。这就使得知识并不总是只有"真"与"假"这两种状态，而是在"真"与"假"之间还存在许多中间状态，即存在"真"的程度问题。知识的这一特性称为不确定性。

造成知识具有不确定性的原因主要有以下几个方面。

① 由随机性引起的不确定性。由随机事件所形成的知识不能简单地用"真"或"假"来刻画，它是不确定的。就以前面所说的"如果头痛且流涕，则有可能患了感冒"这一条知识来说，其中的"有可能"实际上就是反映了"头痛且流涕"与"患了感冒"之间的一种不确定的因果关系，因为具有"头痛且流涕"的人不一定都是"患了感冒"。因此它是一条具有不确定性的知识。

② 由模糊性引起的不确定性。由于某些事物客观上存在的模糊性，使得人们无法把两个类似的事物严格地区分开来，不能明确地判定一个对象是否符合一个模糊概念；又由于某些事物之间存在着模糊关系，使得人们不能准确地判定它们之间的关系究竟是"真"还是"假"。像这样由模糊概念、模糊关系所形成的知识显然是不确定的。

③ 由经验引起的不确定性。知识一般是由领域专家提供的，这种知识大都是领域专家在长期的实践及研究中积累起来的经验性知识。尽管领域专家能够得心应手地运用这些知识，正确地解决领域内的有关问题，但若让他们精确地表述出来却是相当困难的。这是引起知识不确定性的一个原因。另外，由于经验性自身就蕴涵着不精确性及模糊性，这就形成了知识的不确定性。因此，在专家系统中大部分知识都具有不确定性这一特性。

④ 由不完全性引起的不确定性。人们对客观世界的认识是逐步提高的，只有在积累了大量的感性认识后才能升华到理性认识的高度，形成某种知识。因此，知识有一个逐步完善的过程。在此过程中，或者由于客观事物表露得不够充分，致使人们对它的认识不够全面；或者对充分表露的事物一时抓不住本质，使人们对它的认识不够准确。这种认识上的不完全、不准确必然导致相应的知识是不精确、不确定的。因而不完全性是使知识具有不确定性的一个重要原因。

3. 可表示性与可利用性

知识的可表示性是指知识可以用适当形式表示出来，如用语言、文字、图形、神经网络等，这样才能被存储、传播。知识的可利用性是指知识可以被利用。这是不言而喻的，人们每天都在利用自己掌握的知识解决所面临的各种问题。

2.1.3 知识的表示

知识表示（knowledge representation）就是将人类知识形式化或者模型化。实际上就是对知识

的一种描述，或者说是一组约定，一种计算机可以接受的用于描述知识的数据结构。

目前已经提出了许多知识表示方法，如下面要着重介绍的一阶谓词逻辑、产生式、框架、状态空间、人工神经网络、遗传编码等。实际上还有许多知识表示方法没有介绍。已有知识表示方法大都是在进行某项具体研究时提出来的，有一定的针对性和局限性，应用时需根据实际情况作适当的改变，有时还需要把几种表示模式结合起来。在建立一个具体的智能系统时，究竟采用哪种表示模式，目前还没有统一的标准，也不存在一个万能的知识表示模式。

2.2　一阶谓词逻辑表示法

人工智能中用到的逻辑可划分为两大类。一类是经典命题逻辑和一阶谓词逻辑，其特点是任何一个命题的真值或者为“真”，或者为“假”，二者必居其一。因为它只有两个真值，因此又称为二值逻辑。另一类是泛指经典逻辑外的那些逻辑，主要包括三值逻辑、多值逻辑、模糊逻辑等，统称为非经典逻辑。

命题逻辑与谓词逻辑是最先应用于人工智能的两种逻辑，在知识的形式化表示方面，特别是定理的自动证明方面，发挥了重要作用，在人工智能的发展史中占有重要地位。

2.2.1　命题

谓词逻辑是在命题逻辑基础上发展起来的，命题逻辑可看作是谓词逻辑的一种特殊形式。下面首先讨论命题的概念。

定义 2.1　命题(proposition)是一个非真即假的陈述句。

判断一个句子是否为命题，首先应该判断它是否为陈述句，再判断它是否有唯一的真值。没有真假意义的语句(如感叹句、疑问句等)不是命题。

若命题的意义为真，称它的真值为真，记作 $T(True)$；若命题的意义为假，称它的真值为假，记作 $F(False)$。例如，“北京是中华人民共和国的首都”，“$3<5$”都是真值为 T 的命题；“太阳从西边升起”，“煤球是白色的”都是真值为 F 的命题。

一个命题不能同时既为真又为假，但可以在一种条件下为真，在另一种条件下为假。例如，“$1+1=10$”在二进制情况下是真值为 T 的命题，但在十进制情况下是真值为 F 的命题。同样，对于命题“今天是晴天”，也要看当天的实际情况才能决定其真值。

在命题逻辑中，命题通常用大写的英文字母表示，例如，可用英文字母 P 表示“西安是个古老的城市”这个命题。

英文字母表示的命题既可以是一个特定的命题，称为命题常量，也可以是一个抽象的命题，称为命题变元。对于命题变元而言，只有把确定的命题代入后，它才可能有明确的真值。

简单陈述句表达的命题称为简单命题或原子命题。引入否定、合取、析取、条件、双条件等连接词，可以将原子命题构成复合命题。可以定义命题的推理规则和蕴涵式，从而进行简单的逻辑证明。这些内容和谓词逻辑类似，可以参看有关书籍。

命题逻辑表示法有较大的局限性，它无法把它所描述的事物的结构及逻辑特征反映出来，也不能把不同事物间的共同特征表述出来。例如，对于“老李是小李的父亲”这一命题，若用英文字母表示，例如，用字母 P，则无论如何也看不出老李与小李的父子关系。又如对于“李白是诗

人”、“杜甫也是诗人”这两个命题，用命题逻辑表示时，也无法把两者的共同特征（都是诗人）形式化地表示出来。由于这些原因，在命题逻辑的基础上发展起了谓词逻辑。

2.2.2 谓词

谓词（predicate）逻辑是基于命题中谓词分析的一种逻辑。一个谓词可分为谓词名与个体两个部分。个体表示某个独立存在的事物或者某个抽象的概念；谓词名用于刻画个体的性质、状态或个体间的关系。

谓词的一般形式是：

$$P(x_1, x_2, \cdots, x_n)$$

其中，P 是谓词名，$x_1, x_2, \cdots, x_n$ 是个体。

谓词中包含的个体数目称为谓词的元数。$P(x)$ 是一元谓词，$P(x, y)$ 是二元谓词，$P(x_1, x_2, \cdots, x_n)$ 是 n 元谓词。

谓词名是由使用者根据需要人为定义的，一般用具有相应意义的英文单词表示，或者用大写的英文字母表示，也可以用其他符号，甚至中文表示。个体通常用小写的英文字母表示。例如对于谓词 $S(x)$，既可以定义它表示“x 是一个学生”，也可以定义它表示“x 是一只船”。

在谓词中，个体可以是常量，也可以是变元，还可以是一个函数。个体常量、个体变元、函数统称为“项”。

个体是常量，表示一个或者一组指定的个体。例如，“老张是一个教师”这个命题，可表示为一元谓词 *Teacher*(*Zhang*)。其中，*Teacher* 是谓词名，*Zhang* 是个体，*Teacher* 刻画了 *Zhang* 的职业是教师这一特征。

“5 > 3”这个不等式命题，可表示为二元谓词 *Greater*(5,3)。其中，*Greater* 是谓词名，5 和 3 是个体，*Greater* 刻画了 5 与 3 之间的“大于”关系。

“*SMITH* 作为一个工程师为 *IBM* 工作”这个命题，可表示为三元谓词 *Works*(*SMITH*, *IBM*, *Engineer*)。

一个命题的谓词表示也不是唯一的。例如，“老张是一个教师”这个命题，也可表示为二元谓词 *Is-a*(*Zhang*, *Teacher*)。

个体是变元，表示没有指定的一个或者一组个体。例如，“$x < 5$”这个命题，可表示为 *Less*(x, 5)。其中，x 是变元。

当变量用一个具体的个体的名字代替时，则变量被常量化。当谓词中的变元都用特定的个体取代时，谓词就具有一个确定的真值：T 或 F。

个体变元的取值范围称为个体域。个体域可以是有限的，也可以是无限的。例如，若用 $I(x)$ 表示“x 是整数”，则个体域是所有整数，它是无限的。

个体是函数，表示一个个体到另一个个体的映射。例如，“小李的父亲是教师”，可表示为一元谓词 *Teacher*(*father*(*LI*))；“小李的妹妹与小张的哥哥结婚”，可表示为二元谓词 *Married*(*sister*(*LI*), *brother*(*Zhang*))。其中，*sister*(*LI*), *brother*(*Zhang*) 是函数。

函数可以递归调用。例如，“小李的祖父”可以表示为 *father*(*father*(*LI*))。

函数与谓词表面上很相似，容易混淆，其实这是两个完全不同的概念。谓词的真值是“真”或“假”，而函数的值是个体域中的某个个体，函数无真值可言，它只是在个体域中从一个个体到

另一个个体的映射。

在谓词 $P(x_1, x_2, \cdots, x_n)$ 中，若 $x_i(i=1, \cdots, n)$ 都是个体常量、变元或函数，称它为一阶谓词。如果某个 x_i 本身又是一个一阶谓词，则称它为二阶谓词，余者可以此类推。例如，“*SMITH* 作为一个工程师为 *IBM* 工作”这个命题，可表示为二阶谓词 *Works*(*Engineer*(*SMITH*), *IBM*)，因为其中个体 *Engineer*(*SMITH*)也是一个一阶谓词。本书讨论的都是一阶谓词。

2.2.3　谓词公式

无论是命题逻辑还是谓词逻辑，均可用下列连接词把一些简单命题连接起来构成一个复合命题，以表示一个比较复杂的含义。

1. 连接词(连词)

① ¬：称为“否定”(Negation) 或者“非”。它表示否定位于它后面的命题。当命题 P 为真时，$\neg P$ 为假；当 P 为假时，$\neg P$ 为真。

例如，“机器人不在 2 号房间内”，表示为

$$\neg\ INROOM(Robot, R2)。$$

② ∨：称为“析取”(Disjunction)。它表示被它连接的两个命题具有“或”关系。

例如，“李明打篮球或踢足球”，表示为

$$Plays(LiMing, Basketball) \vee Plays(LiMing, Football)。$$

③ ∧：称为“合取”(Conjunction)。它表示它连接的两个命题具有“与”关系。

例如，“我喜爱音乐和绘画”，表示为

$$Like(I, Music) \wedge Like(I, Painting)。$$

某些较简单的句子也可以用 ∧ 构成复合形式，如：“李住在一幢黄色的房子里”，表示为

$$LIVES(LI, HOUSE\text{-}1) \wedge COLOR(HOUSE\text{-}1, YELLOW)$$

④ →：称为“蕴涵”(Implication)或者“条件”(Condition)。$P \rightarrow Q$ 表示“P 蕴涵 Q”，即表示“如果 P，则 Q”。其中，P 称为条件的前件，Q 称为条件的后件。

例如，“如果刘华跑得最快，那么他取得冠军”表示为

$$RUNS(LIUHUA, FASTEST) \rightarrow WINS(LIUHUA, CHAMPION)$$

“如果该书是李明的，那么它是蓝色的”表示为

$$OWNS(LIMING, BOOK\text{-}1) \rightarrow COLOR(BOOK\text{-}1, BLUE)$$

“如果 *Jones* 制造了一个传感器，且这个传感器不能用，那么他或者在晚上进行修理，或者第二天把它交给工程师”表示为

$$Produces(Jones, Sensor) \wedge \neg\ Works(Sensor) \rightarrow Fix(Jones, Sensor, Evening)$$
$$\vee\ Give(Sensor, Engineer, Next\text{-}day)$$

如果后项取值 T(不管其前项的值如何)，或者前项取值 F(不管后项的值如何)，则蕴涵取值 T，否则蕴涵取值 F。注意，只有前项为真，后项为假时，蕴涵才为假，其余均为真，如表 2.1 所示。

“蕴涵”与汉语中的“如果……则……”有区别，汉语中前后要有联系，而命题中可以毫无关系。例如，如果“太阳从西边出来”，则“雪是白的”，是一个真值为 T 的命题。

⑤ ↔：称为“等价”(Equivalence)或“双条件”(Bicondition)。$P \leftrightarrow Q$ 表示“P 当且仅当 Q”。

以上连词的真值由表 2.1 给出。

表 2.1 谓词逻辑真值表

P	Q	$\neg P$	$P \vee Q$	$P \wedge Q$	$P \rightarrow Q$	$P \leftrightarrow Q$
T	T	F	T	T	T	T
T	F	F	T	F	F	F
F	T	T	T	F	T	F
F	F	T	F	F	T	T

2. 量词(quantifier)

为刻画谓词与个体间的关系,在谓词逻辑中引入了两个量词:全称量词和存在量词。

① 全称量词(universal quantifier)($\forall x$):表示"对个体域中的所有(或任一个)个体 x"。例如:

"所有的机器人都是灰色的"可表示为

$$(\forall x)[ROBOT(x) \rightarrow COLOR(x, GRAY)]$$

"所有的车工都操作车床",可表示为

$$(\forall x)[Turner(x) \rightarrow Operates(x, Lathe)]$$

② 存在量词(existential quantifier)($\exists x$):表示"在个体域中存在个体 x"。例如:

"1 号房间有个物体"可表示为:

$$(\exists x) INROOM(x, r1)$$

"某个工程师操作车床"可表示为:

$$(\exists x)[Engineer(x) \rightarrow Operates(x, Lathe)]$$

全称量词和存在量词可以出现在同一个命题中。例如,设谓词 $F(x,y)$ 表示 x 与 y 是朋友,则:

$(\forall x)(\exists y)F(x, y)$ 表示对于个体域中的任何个体 x 都存在个体 y,x 与 y 是朋友。

$(\exists x)(\forall y)F(x, y)$ 表示在个体域中存在个体 x,与个体域中的任何个体 y 都是朋友。

$(\exists x)(\exists y)F(x, y)$ 表示在个体域中存在个体 x 与个体 y,x 与 y 是朋友。

$(\forall x)(\forall y)F(x, y)$ 表示对于个体域中的任何两个个体 x 和 y,x 与 y 都是朋友。

当全称量词和存在量词出现在同一个命题中时,这时量词的次序将影响命题的意思。例如:

$(\forall x)(\exists y)(Employee(x) \rightarrow Manager(y, x))$ 表示"每个雇员都有一个经理";

而

$(\exists y)(\forall x)(Employee(x) \rightarrow Manager(y, x))$ 表示"有一个人是所有雇员的经理"。

又如:

$(\forall x)(\exists y)\ Love(x, y)$ 表示"每个人都有喜欢的人";

而

$(\exists y)(\forall x)\ Love(x, y)$ 表示"有的人大家都喜欢他"。

3. 谓词公式

定义 2.2 可按下述规则得到谓词公式:

① 单个谓词是谓词公式，称为原子谓词公式。

② 若 A 是谓词公式，则 $\neg A$ 也是谓词公式。

③ 若 A,B 都是谓词公式，则 $A \wedge B, A \vee B, A \rightarrow B, A \leftrightarrow B$ 也都是谓词公式。

④ 若 A 是谓词公式，则 $(\forall x)A$，$\exists xA$ 也都是谓词公式。

⑤ 有限步应用①～④生成的公式也是谓词公式。

谓词公式的概念：由谓词符号、常量符号、变量符号、函数符号以及括号、逗号等按一定语法规则组成的字符串的表达式。

在谓词公式中，连接词的优先级别从高到低排列是：

$$\neg, \wedge, \vee, \rightarrow, \leftrightarrow$$

4. 量词的辖域

位于量词后面的单个谓词或者用括弧括起来的谓词公式称为量词的辖域，辖域内与量词中同名的变元称为约束变元，不受约束的变元称为自由变元。

例如：

$$\exists x(P(x, y) \rightarrow Q(x, y)) \vee R(x, y)$$

其中，$(P(x, y) \rightarrow Q(x, y))$ 是 $(\exists x)$ 的辖域，辖域内的变元 x 是受 $(\exists x)$ 约束的变元，而 $R(x, y)$ 中的 x 是自由变元。公式中的所有 y 都是自由变元。

在谓词公式中，变元的名字是无关紧要的，可以把一个名字换成另一个名字。但必须注意，当对量词辖域内的约束变元更名时，必须把同名的约束变元都统一改成相同的名字，且不能与辖域内的自由变元同名；当对辖域内的自由变元改名时，不能改成与约束变元相同的名字。例如，对于公式 $(\forall x)P(x, y)$，可改名为 $(\forall z)P(z, t)$，这里把约束变元 x 改成了 z，把自由变元 y 改成了 t。

2.2.4　谓词公式的性质

1. 谓词公式的解释

在命题逻辑中，对命题公式中各个命题变元的一次真值指派称为命题公式的一个解释。一旦命题确定后，根据各连接词的定义就可以求出命题公式的真值（T 或 F）。

在谓词逻辑中，由于公式中可能有个体变元以及函数，因此不能像命题公式那样直接通过真值指派给出解释，必须首先考虑个体变元和函数在个体域中的取值，然后才能针对变元与函数的具体取值为谓词分别指派真值。由于存在多种组合情况，所以一个谓词公式的解释可能有很多个。对于每一个解释，谓词公式都可求出一个真值（T 或 F）。

2. 谓词公式的永真性、可满足性、不可满足性

定义 2.3　如果谓词公式 P 对个体域 D 上的任何一个解释都取得真值 T，则称 P 在 D 上是永真的；如果 P 在每个非空个体域上均永真，则称 P 永真。

定义 2.4　如果谓词公式 P 对个体域 D 上的任何一个解释都取得真值 F，则称 P 在 D 上是永假的；如果 P 在每个非空个体域上均永假，则称 P 永假。

可见，为了判定某个公式永真，必须对每个个体域上的所有解释逐个判定。当解释的个数为无限时，公式的永真性就很难判定了。

定义 2.5　对于谓词公式 P，如果至少存在一个解释使得公式 P 在此解释下的真值为 T，则

称公式 P 是可满足的，否则，则称公式 P 是不可满足的。

3. 谓词公式的等价性

定义 2.6 设 P 与 Q 是两个谓词公式，D 是它们共同的个体域，若对 D 上的任何一个解释，P 与 Q 都有相同的真值，则称公式 P 和 Q 在 D 上是等价的。如果 D 是任意个体域，则称 P 和 Q 是等价的，记作 $P \Leftrightarrow Q$。

下面列出今后要用到的一些主要等价式：

(1) 交换律

$$P \vee Q \Leftrightarrow Q \vee P$$
$$P \wedge Q \Leftrightarrow Q \wedge P$$

(2) 结合律

$$(P \vee Q) \vee R \Leftrightarrow P \vee (Q \vee R)$$
$$(P \wedge Q) \wedge R \Leftrightarrow P \wedge (Q \wedge R)$$

(3) 分配律

$$P \vee (Q \wedge R) \Leftrightarrow (P \vee Q) \wedge (P \vee R)$$
$$P \wedge (Q \vee R) \Leftrightarrow (P \wedge Q) \vee (P \wedge R)$$

(4) 德摩根律(De Morgen)

$$\neg (P \vee Q) \Leftrightarrow \neg P \wedge \neg Q$$
$$\neg (P \wedge Q) \Leftrightarrow \neg P \vee \neg Q$$

(5) 双重否定律(对合律)

$$\neg \neg P \Leftrightarrow P$$

(6) 吸收律

$$P \vee (P \wedge Q) \Leftrightarrow P$$
$$P \wedge (P \vee Q) \Leftrightarrow P$$

(7) 补余律(否定律)

$$P \vee \neg P \Leftrightarrow T$$
$$P \wedge \neg P \Leftrightarrow F$$

(8) 连接词化归律

$$P \rightarrow Q \Leftrightarrow \neg P \vee Q$$

(9) 逆否律

$$P \rightarrow Q \Leftrightarrow \neg Q \rightarrow \neg P$$

(10) 量词转换律

$$\neg (\exists x) P \Leftrightarrow (\forall x)(\neg P)$$
$$\neg (\forall x) P \Leftrightarrow (\exists x)(\neg P)$$

(11) 量词分配律

$$(\forall x)(P \wedge Q) \Leftrightarrow (\forall x) P \wedge (\forall x) Q$$
$$(\exists x)(P \vee Q) \Leftrightarrow (\exists x) P \vee (\exists x) Q$$

4. 谓词公式的永真蕴涵

定义 2.7 对于谓词公式 P 与 Q，如果 $P \rightarrow Q$ 永真，则称公式 P 永真蕴涵 Q，记作 $P \Rightarrow Q$，且称

Q 为 P 的逻辑结论，P 为 Q 的前提。

下面列出今后要用到的一些主要永真蕴涵式：

(1) 假言推理

$$P, P \rightarrow Q \Rightarrow Q$$

即由 P 为真及 $P \rightarrow Q$ 为真，可推出 Q 为真。

(2) 拒取式推理

$$\neg Q, P \rightarrow Q \Rightarrow \neg P$$

即由 Q 为假及 $P \rightarrow Q$ 为真，可推出 P 为假。

(3) 假言三段论

$$P \rightarrow Q, Q \rightarrow R \Rightarrow P \rightarrow R$$

即由 $P \rightarrow Q, Q \rightarrow R$ 为真，可推出 $P \rightarrow R$ 为真。

(4) 全称固化

$$(\forall x) P(x) \Rightarrow P(y)$$

其中，y 是个体域中的任一个体，利用此永真蕴涵式可消去公式中的全称量词。

(5) 存在固化

$$(\exists x) P(x) \Rightarrow P(y)$$

其中，y 是个体域中某一个可使 $P(y)$ 为真的个体。利用此永真蕴涵式可消去公式中的存在量词。

(6) 反证法

定理 2.1　Q 为 $P_1, P_2, \cdots, P_n$ 的逻辑结论，当且仅当 $(P_1 \wedge P_2 \wedge \cdots \wedge P_n) \wedge \neg Q$ 是不可满足的。

该定理是归结反演的理论依据。

上面列出的等价式及永真蕴涵式是进行演绎推理的重要依据，因此这些公式又称为推理规则。

2.2.5　一阶谓词逻辑知识表示方法

从前面介绍的谓词逻辑的例子可见，用谓词公式表示知识的一般步骤为：

① 定义谓词及个体，确定每个谓词及个体的确切定义。

② 根据要表达的事物或概念，为谓词中的变元赋以特定的值。

③ 根据语义用适当的连接符号将各个谓词连接起来，形成谓词公式。

例 2.1　用一阶谓词逻辑表示“每个储蓄钱的人都得到利息”。

解　定义谓词：$save(x)$ 表示 x 储蓄钱，$interest(x)$ 表示 x 获得利息。

则“每个储蓄钱的人都得到利息”可以表示为：

$$(\forall x)(save(x) \rightarrow interest(x))$$

一阶谓词逻辑表示并不是唯一的。例如，例 2.1 也可以按如下方法表示。

定义谓词：$save(x, y)$ 表示 x 储蓄 y，$money(y)$ 表示 y 是钱，$interest(y)$ 表示 y 是利息，$obtain(x, y)$ 表示 x 获得 y。则“每个储蓄钱的人都得到利息”可以表示为：

$$(\forall x)((\exists y)(money(y) \wedge save(x,y)) \rightarrow (\exists u)(interest(u) \wedge obtain(x,u)))$$

实际上,关系数据库也可以用一阶谓词表达。例如,用一阶谓词逻辑表示下列关系数据库。

住户	房间	电话号码	房间
Zhang	201	491	201
Li	201	492	201
Wang	202	451	202
Zhao	203	451	203

表中有两个关系:

OCCUPANT(给定用户和房间的居住关系)

TELEPHONE(给定电话号码和房间的电话关系)

用一阶谓词表示为:

OCCUPANT(*Zhang*, 201),*OCCUPANT*(*Li*, 201),…

TELEPHONE(491,201),*TELEPHONE*(492,201),…

2.2.6 一阶谓词逻辑表示法的特点

1. 一阶谓词逻辑表示法的优点

(1) 自然性

谓词逻辑是一种接近自然语言的形式语言,用它表示的知识比较容易理解。

(2) 精确性

谓词逻辑是二值逻辑,其谓词公式的真值只有"真"与"假",因此可用它表示精确的知识,并可保证演绎推理所得结论的精确性。

(3) 严密性

谓词逻辑具有严格的形式定义及推理规则,利用这些推理规则及有关定理证明技术可从已知事实推出新的事实,或证明所作的假设。

(4) 容易实现

用谓词逻辑表示的知识可以比较容易地转换为计算机的内部形式,易于模块化,便于对知识进行增加、删除及修改。

2. 一阶谓词逻辑表示法的局限性

(1) 不能表示不确定的知识

谓词逻辑只能表示精确性的知识,不能表示不精确、模糊性的知识,但人类的知识不同程度地具有不确定性,这就使得它表示知识的范围受到了限制。

(2) 组合爆炸

在其推理过程中,随着事实数目的增大及盲目地使用推理规则,有可能形成组合爆炸。目前人们在这一方面做了大量的研究工作,出现了一些比较有效的方法,如定义一个过程或启发式控制策略来选取合适的规则等。

(3) 效率低

用谓词逻辑表示知识时,其推理是根据形式逻辑进行的,把推理与知识的语义割裂开来,这就使得推理过程冗长,降低了系统的效率。

尽管谓词逻辑表示法有以上一些局限性,但它仍是一种重要的表示方法,许多专家系统的知识表达都采用谓词逻辑表示,如格林等人研制的用于求解化学方面问题的 QA3 系统,菲克斯等人研制的 STRIPS 机器人行动规划系统,菲尔曼等人研制的 FOL 机器证明系统。

2.3　产生式表示法

产生式表示法又称为产生式规则(production rule)表示法。

"产生式"这一术语是由美国数学家波斯特(E. Post)在 1943 年首先提出来的。他根据串替代规则提出了一种称为波斯特机的计算模型,模型中的每一条规则称为一个产生式。在此之后,几经修改与充实,如今已被用到多个领域中。例如,用它来描述形式语言的语法,表示人类心理活动的认知过程等。1972 年纽厄尔和西蒙在研究人类的认知模型中开发了基于规则的产生式系统。目前它已成为人工智能中应用最多的一种知识表示模型,许多成功的专家系统都用它来表示知识,如费根鲍姆等人研制的化学分子结构专家系统 DENDRAL、肖特里菲等人研制的诊断感染性疾病的专家系统 MYCIN 等。

2.3.1　产生式

产生式通常用于表示事实、规则以及它们的不确定性度量,适合于表示事实性知识和规则性知识。

1. 确定性规则知识的产生式表示

确定性规则知识的产生式表示的基本形式如下:

$$\text{IF}\quad P\quad \text{THEN}\quad Q$$

或者

$$P \rightarrow Q$$

其中,P 是产生式的前提,用于指出该产生式是否可用的条件;Q 是一组结论或操作,用于指出当前提 P 所指示的条件满足时,应该得出的结论或应该执行的操作。整个产生式的含义是:如果前提 P 被满足,则可 Q 或执行 Q 所规定的操作。例如:

r_4:IF　动物会飞　AND　会下蛋　THEN　该动物是鸟

这是一个产生式。其中,r_4 是该产生式的编号;"动物会飞 AND 会下蛋"是前提 P;"该动物是鸟"是结论 Q。

2. 不确定性规则知识的产生式表示

不确定性规则知识的产生式表示的基本形式如下:

$$\text{IF}\quad P\quad \text{THEN}\quad Q\ (\text{置信度})$$

或者

$$P \rightarrow Q\ (\text{置信度})$$

例如,在专家系统 MYCIN 中有这样一条产生式:

IF 本微生物的染色斑是革兰氏阴性
本微生物的形状呈杆状
病人是中间宿主
THEN 该微生物是绿脓杆菌,置信度为0.6

它表示当前提示中列出的各个条件都得到满足时,结论"该微生物是绿脓杆菌"可以相信的程度为0.6。这里,用0.6指出了知识的强度。

3. 确定性事实性知识的产生式表示

确定性事实一般用三元组表示:

(对象,属性,值)

或者

(关系,对象1,对象2)

例如,老李年龄是40岁,表示为(*Li*,*Age*,40)。老李和老王是朋友,表示为(*Friend*,*Li*,*Wang*)。

4. 不确定性事实性知识的产生式表示

不确定性事实一般用四元组表示:

(对象,属性,值,置信度)

或者

(关系,对象1,对象2,置信度)

例如,老李年龄很可能是40岁,表示为(*Li*,*Age*,40,0.8)。老李和老王不大可能是朋友,表示为(*Friend*,*Li*,*Wang*,0.1)。

产生式又称为规则或产生式规则;产生式的"前提"有时又称为"条件"、"前提条件"、"前件"、"左部"等;其"结论"部分有时称为"后件"或"右部"等。今后将不加区分地使用这些术语,不再作单独说明。

产生式与谓词逻辑中的蕴涵式的基本形式相同,但蕴涵式只是产生式的一种特殊情况,理由有如下两点:

① 除逻辑蕴涵外,产生式还包括各种操作、规则、变换、算子、函数等。例如,"如果炉温超过上限,则立即关闭风门"是一个产生式,但不是一个蕴涵式。产生式描述了事物之间的一种对应关系(包括因果关系和蕴涵关系),其外延十分广泛。逻辑中的逻辑蕴涵式和等价式,程序设计语言中的文法规则,数学中的微分和积分公式,化学中分子结构式的分解变换规则,甚至体育比赛中的规则,国家的法律条文,单位的规章制度等,都可以用产生式表示。

② 蕴涵式只能表示确定性知识,其真值或者为真,或者为假,而产生式不仅可以表示确定性的知识,而且还可以表示不确定性知识。决定一条知识是否可用,需要检查当前是否有已知事实可与前提中所规定的条件匹配。对谓词逻辑的蕴涵式来说,其匹配总要求是精确的。在产生式表示知识的系统中,匹配可以是精确的,也可以是不精确的,只要按某种算法求出的相似度落在预先指定的范围内就认为是可匹配的。

由于产生式与蕴涵式存在这些区别,导致它们在处理方法及应用等方面都有较大的差别。

为了严格地描述产生式,下面用巴克斯范式(BNF)给出它的形式进行描述:

<产生式> ::= <前提> → <结论>
<前 提> ::= <简单条件> | <复合条件>
<结 论> ::= <事实> | <操作>
<复合条件> ::= <简单条件> AND <简单条件> [AND <简单条件> …]
| <简单条件> OR <简单条件> [OR <简单条件> …]
<操 作> ::= <操作名> [(<变元> ,…)]

其中,符号“::=”表示“定义为”,符号“|”表示“或者是”,符号“[ ]”表示“可缺省”。

2.3.2　产生式系统

把一组产生式放在一起,让它们互相配合,协同作用,一个产生式生成的结论可以供另一个产生式作为已知事实使用,以求得问题的解,这样的系统称为产生式系统。

一般来说,一个产生式系统由规则库、控制系统(推理机)、综合数据库三部分组成。它们之间的关系如图 2.1 所示。

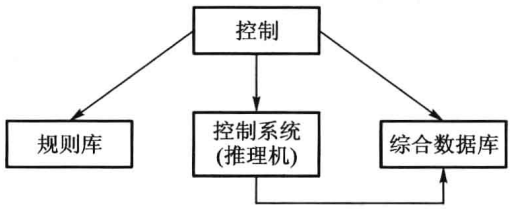


图 2.1　产生式系统的基本结构关系

1. 规则库

用于描述相应领域内知识的产生式集合称为规则库。

显然,规则库是产生式系统求解问题的基础,其知识是否完整、一致,表达是否准确、灵活,对知识的组织是否合理等,将直接影响到系统的性能。因此,需要对规则库中的知识进行合理的组织和管理,检测并排除冗余及矛盾的知识,保持知识的一致性。采用合理的结构形式,可使推理避免访问那些与求解当前问题无关的知识,从而提高求解问题的效率。

2. 综合数据库

综合数据库又称为事实库、上下文、黑板等。它是一个用于存放问题求解过程中各种当前信息的数据结构,如问题的初始状态、原始证据、推理中得到的中间结论及最终结论。当规则库中某条产生式的前提可与综合数据库的某些已知事实相匹配时,该产生式就被激活,并把它推出的结论放入综合数据库中,作为后面推理的已知事实。显然,综合数据库的内容是在不断变化的。

3. 控制系统

控制系统又称为推理机。它由一组程序组成,负责整个产生式系统的运行,实现对问题的求解。粗略地说,推理机要做以下几项工作:

① 按一定的策略从规则库中选择与综合数据库中的已知事实进行匹配。所谓匹配是指把规则的前提条件与综合数据库中的已知事实进行比较,如果两者一致,或者近似一致且满足预先规定的条件,则称匹配成功,相应的规则可被使用,否则称为匹配不成功。

② 冲突消解。匹配成功的规则可能不止一条，这称为发生了冲突。此时，推理机构必须调用相应的解决冲突策略进行消解，以便从匹配成功的规则中选出一条执行。

③ 执行规则。如果某一规则的右部是一个或多个结论，则把这些结论加入到综合数据库中；如果规则的右部是一个或多个操作，则执行这些操作。对于不确定性知识，在执行每一条规则时还要按一定的算法计算结论的不确定性。

④ 检查推理终止条件。检查综合数据库中是否包含了最终结论，决定是否停止系统的运行。

2.3.3 产生式系统的例子——动物识别系统

下面以一个动物识别系统为例，介绍产生式系统求解问题的过程。这个动物识别系统是识别虎、金钱豹、斑马、长颈鹿、企鹅、鸵鸟、信天翁等七种动物的产生式系统。

首先根据这些动物识别的专家知识，建立如下规则库。

r_1： IF 该动物有毛发 THEN 该动物是哺乳动物

r_2： IF 该动物有奶 THEN 该动物是哺乳动物

r_3： IF 该动物有羽毛 THEN 该动物是鸟

r_4： IF 该动物会飞 AND 会下蛋 THEN 该动物是鸟

r_5： IF 该动物吃肉 THEN 该动物是食肉动物

r_6： IF 该动物有犬齿 AND 有爪 AND 眼盯前方
THEN 该动物是食肉动物

r_7： IF 该动物是哺乳动物 AND 有蹄
THEN 该动物是有蹄类动物

r_8： IF 该动物是哺乳动物 AND 是反刍动物
THEN 该动物是有蹄类动物

r_9： IF 该动物是哺乳动物 AND 是食肉动物
AND 是黄褐色
AND 身上有暗斑点
THEN 该动物是金钱豹

r_{10}： IF 该动物是哺乳动物 AND 是食肉动物
AND 是黄褐色
AND 身上有黑色条纹
THEN 该动物是虎

r_{11}： IF 该动物是有蹄类动物 AND 有长脖子
AND 有长腿
AND 身上有暗斑点
THEN 该动物是长颈鹿

r_{12}： IF 该动物有蹄类动物 AND 身上有黑色条纹
THEN 该动物是斑马

r_{13}： IF 该动物是鸟 AND 有长脖子 AND 有长腿 AND 不会飞

AND　有黑白二色
THEN　该动物是鸵鸟

r_{14}：IF　该动物是鸟　AND　会游泳　AND　不会飞　AND　有黑白二色
THEN　该动物是企鹅

r_{15}：IF　该动物是鸟　AND　善飞
THEN　该动物是信天翁

由上述产生式规则可以看出，虽然系统是用来识别七种动物的，但它并不是简单地只设计 7 条规则，而是设计了 15 条。其基本想法是：首先根据一些比较简单的条件，如“有毛发”、“有羽毛”、“会飞”等对动物进行比较粗的分类，如“哺乳动物”、“鸟”等，然后随着条件的增加，逐步缩小分类范围，最后给出识别七种动物的规则。这样做至少有两个好处：一是当已知的事实不完全时，虽不能推出最终结论，但可以得到分类结果；二是当需要增加对其他动物（如牛、马等）的识别时，规则库中只需增加关于这些动物个性方面的知识，如 r_9 至 r_{15} 那样，而对 r_1 至 r_8 可直接利用，这样增加的规则就不会太多。$r_1, r_2, \cdots, r_{15}$ 分别是对各产生式规则所做的编号，以便于对它们的引用。

设在综合数据库中存放有下列已知事实：

该动物特征有暗斑点，长脖子，长腿，奶，蹄

并假设综合数据库中的已知事实与规则库中的知识是从第一条（即 r_1）开始逐条进行匹配的，则当推理开始时，推理机构的工作过程是：

① 从规则库中取出第一条规则 r_1，检查其前提是否可与综合数据库中的已知事实匹配成功。由于综合数据库中没有“该动物有毛发”这一事实，所以匹配不成功，r_1 不能被用于推理。然后取第二条规则 r_2 进行同样的工作。显然，r_2 的前提“该动物有奶”可与综合数据库中的已知事实“该动物有奶”匹配。再检查 r_3 至 r_{15}，结果均不能匹配。因为只有 r_2 一条规则被匹配，所以 r_2 被执行，并将其结论部分“该动物是哺乳动物”加入到综合数据库中。并且将 r_2 标注已经被选用过的记号，避免下次再被匹配。

此时综合数据库的内容变为：

该动物特征有暗斑点，长脖子，长腿，奶，蹄，哺乳动物

检查综合数据库中的内容，没有发现要识别的任何一种动物，所以要继续进行推理。

② 分别用 r_1, r_3, r_4, r_5, r_6 与综合数据库中的已知事实进行匹配，均不成功。但当用 r_7 与之匹配时，获得了成功。再检查 r_8 至 r_{15} 均不能匹配。因为只有 r_7 一条规则被匹配，所以执行 r_7 并将其结论部分“该动物是有蹄类动物”加入到综合数据库中，并且将 r_7 标注已经被选用过的记号，避免下次再被匹配。

此时综合数据库的内容变为：

该动物特征有暗斑点，长脖子，长腿，奶，蹄，哺乳动物，有蹄类动物

检查综合数据库中的内容，没有发现要识别的任何一种动物，所以还要继续进行推理。

③ 在此之后，除已经匹配过的 r_2, r_7 外，只有 r_{11} 可与综合数据库中的已知事实匹配成功，所以将 r_{11} 的结论加入综合数据库，此时综合数据库的内容变为：

该动物特征有暗斑点，长脖子，长腿，奶，蹄，哺乳动物，有蹄类动物，长颈鹿

检查综合数据库中的内容，发现要识别的动物长颈鹿包含在了综合数据库中，所以推出了

“该动物是长颈鹿”这一最终结论。至此,问题的求解过程就结束了。

上述问题的求解过程是一个不断地从规则库中选择可用规则与综合数据库中的已知事实进行匹配的过程,规则的每一次成功匹配都使综合数据库增加了新的内容,并朝着问题的解决方向前进了一步。这一过程称为推理,是专家系统中的核心内容。当然,上述过程只是一个简单的推理过程,后面将对推理的有关问题展开全面的介绍。

可以使用普通编程语言(如 C、C++)中的 if 语句实现产生式系统,但当产生式规则较多时会产生新的问题。例如,检查哪条规则被匹配需要很长时间遍历所有规则,因此,采用快速算法(如 RETE)匹配规则触发条件的专用产生式系统已经被开发出来。这种系统内嵌了消解多个冲突的算法。近年来,开发了专门用于计算机游戏开发的 RC++,它是 C++ 语言的超集,加入了控制角色行为的产生式规则,提供了反应式控制器的专用子集。

2.3.4 产生式表示法的特点

1. 产生式表示法的主要优点

(1) 自然性

产生式表示法用“如果……,则……”的形式表示知识,这是人们常用的一种表达因果关系的知识表示形式,既直观、自然,又便于进行推理。正是由于这一原因,才使得产生式表示法成为人工智能中最重要且应用最多的一种知识表示方法。

(2) 模块性

产生式是规则库中最基本的知识单元,它们同推理机构相对独立,而且每条规则都具有相同的形式。这就便于对其进行模块化处理,为知识的增、删、改带来了方便,为规则库的建立和扩展提供了可管理性。

(3) 有效性

产生式表示法既可表示确定性知识,又可表示不确定性知识;既有利于表示启发式知识,又可方便地表示过程性知识。目前已建造成功的专家系统大部分是用产生式来表达其过程性知识的。

(4) 清晰性

产生式有固定的格式。每一条产生式规则都由前提与结论(操作)这两部分组成,而且每一部分所含的知识量都比较少。这既便于对规则进行设计,又易于对规则库中知识的一致性及完整性进行检测。

2. 产生式表示法的主要缺点

(1) 效率不高

在产生式系统求解问题的过程中,首先要用产生式的前提部分与综合数据库中的已知事实进行匹配,从规则库中选出可用的规则,此时选出的规则可能不止一个,这就需要按一定的策略进行“冲突消解”,然后把选中的规则启动执行。因此,产生式系统求解问题的过程是一个反复进行“匹配—冲突消解—执行”的过程。鉴于规则库一般都比较庞大,而匹配又是一件十分费时的工作,因此其工作效率不高,而且大量的产生式规则容易引起组合爆炸。

(2) 不能表达具有结构性的知识

产生式适合于表达具有因果关系的过程性知识,但对具有结构关系的知识却无能为力,它不

能把具有结构关系的事物间的区别与联系表示出来。后面介绍的框架表示法可以解决这方面的问题。因此，产生式表示法除了可以独立作为一种知识表示模式外，还经常与其他表示法结合起来表示特定领域的知识。例如，在专家系统 PROSPECTOR 中用产生式与语义网络相结合，在 Alkins 中把产生式与框架表示法结合起来，等等。

3. 产生式表示法适合表示的知识

由上述关于产生式表示法的特点，可以看出产生式表示法适合于表示具有下列特点的领域知识。

① 由许多相对独立的知识元组成的领域知识，彼此间关系不密切，不存在结构关系，如化学反应方面的知识。

② 具有经验性及不确定性的知识，而且相关领域中对这些知识没有严格、统一的理论，如医疗诊断、故障诊断等方面的知识。

③ 领域问题的求解过程可被表示为一系列相对独立的操作，而且每个操作可被表示为一条或多条产生式规则。

2.4　框架表示法

1975 年美国著名的人工智能学者明斯基提出了框架理论。该理论认为人们对现实世界中各种事物的认识都是以一种类似于框架的结构存储在记忆中的。当面临一个新事物时，就从记忆中找出一个合适的框架，并根据实际情况对其细节加以修改、补充，从而形成对当前事物的认识。例如，一个人走进一个教室之前就能依据以往对“教室”的认识，想象到这个教室一定有四面墙，有门、窗，有天花板和地板，有课桌、凳子、讲台、黑板等。尽管他对这个教室的大小、门窗的个数、桌凳的数量、颜色等细节还不清楚，但对教室的基本结构是可以预见到的。因为他通过以往看到的教室，已经在记忆中建立了关于教室的框架。该框架不仅指出了相应事物的名称（教室），而且还指出了事物各有关方面的属性（如有四面墙，有课桌，有黑板，……）。通过对该框架的查找就很容易得到教室的各个特征。在他进入教室后，经观察得到了教室的大小、门窗的个数、桌凳的数量、颜色等细节，把它们填入到教室框架中，就得到了教室框架的一个具体事例。这是他关于这个具体教室的视觉形象，称为事例框架。

框架表示法是一种结构化的知识表示方法，现已在多种系统中得到应用。

2.4.1　框架的一般结构

框架（frame）是一种描述所论对象（一个事物、事件或概念）属性的数据结构。

一个框架由若干个被称为“槽”（slot）的结构组成，每一个槽又可根据实际情况划分为若干个“侧面”（facet）。一个槽用于描述所论对象某一方面的属性。一个侧面用于描述相应属性的一个方面。槽和侧面所具有的属性值分别被称为槽值和侧面值。在一个用框架表示知识的系统中一般都含有多个框架，一个框架一般都含有多个不同槽、不同侧面，分别用不同的框架名、槽名及侧面名表示。无论是对框架、槽或侧面，都可以为其附加上一些说明性的信息，一般是一些约束条件，用于指出什么样的值才能填入到槽和侧面中去。

下面给出框架的一般表示形式：

〈框架名〉		
槽名 1：	侧面名$_{11}$	侧面值$_{111}$，侧面值$_{112}$，…，侧面值$_{11p_1}$
	侧面名$_{12}$	侧面值$_{121}$，侧面值$_{122}$，…，侧面值$_{12p_2}$
	⋮	
	侧面名$_{1m}$	侧面值$_{1m1}$，侧面值$_{1m2}$，…，侧面值$_{1mp_m}$
槽名 2：	侧面名$_{21}$	侧面值$_{211}$，侧面值$_{212}$，…，侧面值$_{21p_1}$
	侧面名$_{22}$	侧面值$_{221}$，侧面值$_{222}$，…，侧面值$_{22p_2}$
	⋮	
	侧面名$_{2m}$	侧面值$_{2m1}$，侧面值$_{2m2}$，…，侧面值$_{2mp_m}$
⋮		
槽名 n：	侧面名$_{n1}$	侧面值$_{n11}$，侧面值$_{n12}$，…，侧面值$_{n1p_1}$
	侧面名$_{n2}$	侧面值$_{n21}$，侧面值$_{n22}$，…，侧面值$_{n2p_2}$
	⋮	
	侧面名$_{nm}$	侧面值$_{nm1}$，侧面值$_{nm2}$，…，侧面值$_{nmp_m}$
约束：	约束条件$_1$	
	约束条件$_2$	
	⋮	
	约束条件$_n$	

由上述表示形式可以看出，一个框架可以有任意有限数目的槽，一个槽可以有任意有限数目的侧面，一个侧面可以有任意有限数目的侧面值。槽值或侧面值既可以是数值、字符串、布尔值也可以是一个满足某个给定条件时要执行的动作或过程，还可以是另一个框架的名字，从而实现一个框架对另一个框架的调用，表示出框架之间的横向联系。

2.4.2 用框架表示知识的例子

下面举一些例子，说明框架的建立方法。

例 2.2 教师框架

框架名：〈教师〉
姓名：单位（姓、名）
年龄：单位（岁）
性别：范围（男、女）
缺省：男
职称：范围（教授，副教授，讲师，助教）

续表

缺省:讲师
部门:单位(系,教研室)
住址:〈住址框架〉
工资:〈工资框架〉
开始工作时间:单位(年、月)
截止时间:单位(年、月)
缺省:现在

该框架共有 9 个槽,分别描述了“教师”9 个方面的情况,或者说关于“教师”的 9 个属性。在每个槽里都指出了一些说明性的信息,用于对槽的填值给出某些限制。“范围”指出槽的值只能在指定的范围内挑选,例如,对“职称”槽,其槽值只能是“教授”、“副教授”、“讲师”、“助教”中的某一个,不能是别的,如“工程师”等;“缺省”表示当相应槽不填入槽值时,就以缺省值作为槽值,这样可以节省一些填槽的工作。例如,对“性别”槽,当不填入“男”或“女”时,就默认是“男”,这样对男性教师就可以不填这个槽的槽值。

对于上述这个框架,当把具体的信息填入槽或侧面后,就得到了相应框架的一个事例框架。例如,把某教师的一组信息填入“教师”框架的各个槽,就可得到:

框架名:〈教师 - 1〉
姓名:夏冰
年龄:36
性别:女
职称:副教授
部门:计算机系软件教研室
住址:〈*adr* - 1〉
工资:〈*sal* - 1〉
开始工作时间:1988,9
截止时间:1996,7

例 2.3　教室框架

框架名:〈教室〉
墙数:
窗数:

续表

门数：
座位数：
前墙：〈墙框架〉
后墙：〈墙框架〉
左墙：〈墙框架〉
右墙：〈墙框架〉
门：〈门框架〉
窗：〈窗框架〉
黑板：〈黑板框架〉
天花板：〈天花板框架〉
讲台：〈讲台框架〉

该框架共有 13 个槽，分别描述了“教室”的 13 个方面的情况或者属性。

例 2.4 关于自然灾害的新闻报道中所涉及的事实经常是可以预见的，这些可预见的事实就可以作为代表所报道的新闻中的属性。例如，将下列一则地震消息用框架表示：“某年某月某日，某地发生 6.0 级地震，若以膨胀注水孕震模式为标准，则三项地震前兆中的波速比为 0.45，水氡含量为 0.43，地形改变为 0.60。”

解 地震消息用框架如图 2.2 所示。“地震框架”也可以是“自然灾害框架”的子框架，“地震框架”中的值也可以是一个子框架，如图中的“地形改变”就是一个子框架。

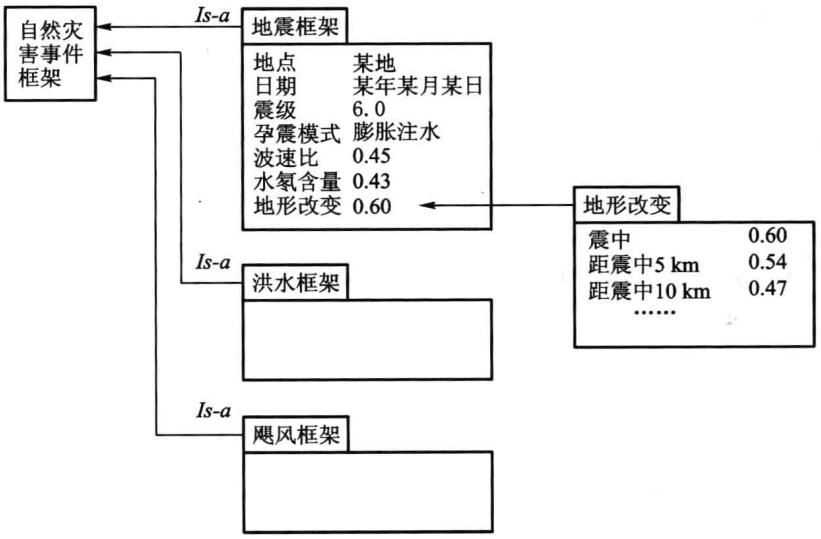


图 2.2 自然灾害框架

产生式规则也可以用框架表示。例如，产生式“如果头痛且发烧，则患感冒”，用框架表示为

框架名：〈诊断 1〉	
前提：条件 1	头痛
条件 2	发烧
结论：感冒	

2.4.3　框架表示法的特点

1. 结构性

框架表示法最突出的特点是便于表达结构性知识，能够将知识的内部结构关系及知识间的联系表示出来，因此它是一种结构化的知识表达方法。这是产生式不具备的，产生式系统中的知识单位是产生式规则，这种知识单位太小而难于处理复杂问题，也不能将知识间的结构关系表示出来。产生式规则只能表示因果关系，而框架表示法不仅可以通过 Infer 槽或者 Possible-reason 槽表示因果关系，还可以通过其他槽表示更复杂的关系。

2. 继承性

框架表示法通过使槽值为另一个框架的名字实现框架间的联系，建立表示复杂知识的框架网络。在框架网络中，下层框架可以继承上层框架的槽值，也可以进行补充和修改，这样不仅减少了知识的冗余，而且较好地保证了知识的一致性。

3. 自然性

框架表示法与人在观察事物时的思维活动是一致的，比较自然。

2.5　小结

1. 知识的概念

把有关信息关联在一起所形成的信息结构称为知识。

知识主要具有相对正确性、不确定性、可表示性与可利用性等特性。

造成知识具有不确定性的原因主要有：随机性、模糊性、经验性、不完全性。

2. 命题与一阶谓词公式

命题是一个非真即假的陈述句。

谓词的一般形式是：$P(x_1, x_2, \cdots, x_n)$，其中，P 是谓词名，$x_1, x_2, \cdots, x_n$ 是个体。个体可以是常量、变元、函数。

用否定、析取、合取、蕴涵、等价等连接词以及全称量词、存在量词把一些简单命题连接起来构成一个复合命题，以表示一个比较复杂的含义。

位于量词后面的单个谓词或者用括弧括起来的谓词公式称为量词的辖域，辖域内与量词中同名的变元称为约束变元，不受约束的变元称为自由变元。

对于谓词公式 P，如果至少存在一个解释使得公式 P 在此解释下的真值为 T，则称公式 P 是

可满足的，否则，则称公式 P 是不可满足的。

当且仅当 $(P_1 \wedge P_2 \wedge \cdots \wedge P_n) \wedge \neg Q$ 是不可满足的，则 Q 为 $P_1, P_2, \cdots, P_n$ 的逻辑结论。

一阶谓词逻辑表示法具有自然、精确、严密、容易实现等优点，但有不能表示不确定的知识、组合爆炸、效率低等缺点。

3. 产生式表示法

产生式表示法是目前应用最多的一种知识表示模型，许多成功的专家系统都用它来表示知识。

产生式通常用于表示事实、规则以及它们的不确定性度量。谓词逻辑中的蕴涵式只是产生式的一种特殊情况。

产生式不仅可以表示确定性规则，还可以表示各种操作、规则、变换、算子、函数等。产生式不仅可以表示确定性知识，而且还可以表示不确定性知识。

产生式表示法具有自然性、模块性、有效性、清晰性等优点，但存在效率不高、不能表达具有结构性的知识等缺点，适合表示由许多相对独立的知识元组成的领域知识、具有经验性及不确定性的知识，也可以表示为一系列相对独立的求解问题的操作。

一个产生式系统由规则库、综合数据库、控制系统（推理机）三部分组成。产生式系统求解问题的过程是一个不断地从规则库中选择可用规则与综合数据库中的已知事实进行匹配的过程，规则的每一次成功匹配都使综合数据库增加了新的内容，并朝着问题的解决方向前进了一步。这一过程称为推理，是专家系统中的核心内容。

4. 框架表示法

框架是一种描述所讨论对象（一个事物、事件或概念）属性的数据结构。

一个框架由若干个被称为“槽”（slot）的结构组成，每一个槽又可根据实际情况划分为若干个“侧面”（facet）。一个槽用于描述所讨论对象某一方面的属性。一个侧面用于描述相应属性的一个方面。槽和侧面所具有的属性值分别被称为槽值和侧面值。

框架表示法具有结构性、继承性、自然性的特点。

思考题

2.1　什么是知识？它有哪些特性？有哪几种分类方法？

2.2　什么是知识表示？如何选择知识表示方法？

2.3　什么是命题？请写出三个真值为 T 及真值为 F 的命题。

2.4　什么是谓词？什么是谓词个体及个体域？函数与谓词的区别是什么？

2.5　谓词逻辑和命题逻辑的关系如何？有何异同？

2.6　什么是谓词的项？什么是谓词的阶？请写出谓词的一般形式。

2.7　什么是谓词公式？什么是谓词公式的解释？

2.8　一阶谓词逻辑表示法适合于表示哪种类型的知识？它有哪些特点？

2.9　请写出用一阶谓词逻辑表示法表示知识的步骤。

2.10　产生式的基本形式是什么？它与谓词逻辑中蕴涵式有什么共同处和不同处？

2.11　产生式系统由哪几部分组成？

2.12　试述产生式系统求解问题的一般步骤。

2.13　产生式系统中，推理机的推理方式有哪几种？在产生式推理过程中，如果发生策略冲突，如何解决？

2.14　试述产生式表示法的特点。

2.15　框架的一般表示形式是什么？

2.16　框架表示法有何特点？请叙述用框架表示法表示知识的步骤。

2.17　试构造一个描述读者的办公室或卧室的框架系统。

2.18　试构造一个描述计算机主机的框架系统。

习题

2.1　设有下列语句，请用相应的谓词公式把它们表示出来：

(1) 有的人喜欢梅花，有的人喜欢菊花，有的人既喜欢梅花又喜欢菊花。

(2) 他每天下午都去玩足球。

(3) 所有人都有饭吃。

(4) 喜欢玩篮球的人必喜欢玩排球。

(5) 要想出国留学，必须通过外语考试。

2.2　分别指出下列谓词公式中各量词的辖域，并指出哪些是约束变元，哪些是自由变元。

(1) $(\forall x)(P(x,y) \vee (\exists y)(Q(x,y) \wedge R(x,y)))$

(2) $(\exists z)(\forall y)(P(z,y) \vee Q(z,x)) \vee R(u,v)$

(3) $(\forall x)(\neg P(x,f(x)) \vee (\exists z)(Q(x,z) \wedge \neg R(z,y)))$

(4) $(\forall z)((\exists y)((\exists t)(P(z,t) \vee Q(y,t))) \wedge R(z,y))$

2.3　设 $D=\{1,2\}$，试给出谓词公式 $(\exists x)(\forall y)(P(x,y) \rightarrow Q(x,y))$ 的一个解释，并且指出该谓词公式的真值。

2.4　试用谓词逻辑表达描述下列推理：

(1) 如果张三比李四大，那么李四比张三小。

(2) 甲和乙结婚了，则或者甲为男，乙为女；或者甲为女，乙为男。

(3) 如果一个人是老实人，他就不会说谎；张三说谎了，所以张三不是一个老实人。

2.5　将下面一则消息用框架表示："今天，一次强度为里氏 8.5 级的强烈地震袭击了下斯洛文尼亚(Low Slabovia)地区，造成 25 人死亡和 5 亿美元的财产损失。下斯洛文尼亚地区的主席说：多年来，靠近萨迪壕金斯(Sadie Haw Kins)断层的重灾区一直是一个危险地区。这是本地区发生的第 3 号地震。"

第3章 确定性推理方法

前面讨论了知识表示方法。这样就可以把知识用某种模式表示出来存储到计算机中去。但是,为使计算机具有智能,仅仅使计算机拥有知识是不够的,还必须使它具有思维能力,即能运用知识求解问题。推理是求解问题的一种重要方法。因此,推理方法成为人工智能的一个重要研究课题。目前,人们已经对推理方法进行了比较多的研究,提出了多种可在计算机上实现的推理方法。

下面首先讨论关于推理的基本概念,然后着重介绍鲁宾逊归结原理及其在机器定理证明和问题求解中的应用。其基本思想是先将要证明的定理表示为谓词公式,并化为子句集,然后再进行归结,如果归结出空子句,则定理得证。鲁宾逊归结原理使定理证明能够在计算机上实现。

3.1 推理的基本概念

3.1.1 推理的定义

人们在对各种事物进行分析、综合并最后作出决策时,通常是从已知的事实出发,通过运用已掌握的知识,找出其中蕴涵的事实,或归纳出新的事实。这一过程通常称为推理,即从初始证据出发,按某种策略不断运用知识库中的已知知识,逐步推出结论的过程称为推理。

在人工智能系统中,推理是由程序实现的,称为推理机。已知事实和知识是构成推理的两个基本要素。已知事实又称为证据,用以指出推理的出发点及推理时应该使用的知识;而知识是使推理得以向前推进,并逐步达到最终目标的依据。例如,在医疗诊断专家系统中,专家的经验及医学常识以某种表示形式存储于知识库中。为病人诊治疾病时,推理机就是从存储在综合数据库中的病人症状及化验结果等初始证据出发,按某种搜索策略在知识库中搜寻可与之匹配的知识,推出某些中间结论,然后再以这些中间结论为证据,在知识库中搜索与之匹配的知识,推出进一步的中间结论,如此反复进行,直到最终推出结论,即病人的病因与治疗方案为止。

3.1.2 推理方式及其分类

人类的智能活动有多种思维方式。人工智能作为对人类智能的模拟,相应地也有多种推理方式。下面分别从不同的角度对它们进行分类。

1. 演绎推理、归纳推理、默认推理

若从推出结论的途径来划分，推理可分为演绎推理、归纳推理和默认推理。

演绎推理(deductive reasoning)是从全称判断推导出单称判断的过程，即由一般性知识推出适合于某一具体情况的结论。这是一种从一般到个别的推理。

演绎推理是人工智能中的一种重要的推理方式。许多智能系统中采用了演绎推理。演绎推理有多种形式，经常用的是三段论式。它包括以下几项。

① 大前提：已知的一般性知识或假设。

② 小前提：关于所研究的具体情况或个别事实的判断。

③ 结论：由大前提推出的适合于小前提所示情况的新判断。

下面是一个三段论推理的例子。

① 大前提：足球运动员的身体都是强壮的。

② 小前提：高波是一名足球运动员。

③ 结论：高波的身体是强壮的。

归纳推理(inductive reasoning)是从足够多的事例中归纳出一般性结论的推理过程，是一种从个别到一般的推理。

若从归纳时所选的事例的广泛性来划分，归纳推理又可分为完全归纳推理和不完全归纳推理两种。

所谓完全归纳推理是指在进行归纳时考察了相应事物的全部对象，并根据这些对象是否都具有某种属性，从而推出这个事物是否具有这个属性。例如，某厂进行产品质量检查，如果对每一件产品都进行了严格检查，并且都是合格的，则推导出结论“该厂生产的产品是合格的”。

所谓不完全归纳推理是指考察了相应事物的部分对象，就得出了结论。例如，检查产品质量时，只是随机地抽查了部分产品，只要它们都合格，就得出了“该厂生产的产品是合格”的结论。

不完全归纳推理推出的结论不具有必然性，属于非必然性推理，而完全归纳推理是必然性推理。但由于要考察事物的所有对象通常都比较困难，因而大多数归纳推理都是不完全归纳推理。归纳推理是人类思维活动中最基本、最常用的一种推理形式。人们在由个别到一般的思维过程中经常要用到它。

默认推理又称为缺省推理(default reasoning)，是在知识不完全的情况下假设某些条件已经具备所进行的推理。

例如，在条件 A 已成立的情况下，如果没有足够的证据能证明条件 B 不成立，则默认 B 是成立的，并在此默认的前提下进行推理，推导出某个结论。例如，要设计一种鸟笼，但不知道要放的鸟是否会飞，则默认这只鸟会飞，因此，推出这个鸟笼要有盖子的结论。

由于这种推理允许默认某些条件是成立的，所以在知识不完全的情况下也能进行。在默认推理的过程中，如果到某一时刻发现原先所作的默认不正确，则要撤销所作的默认以及由此默认推出的所有结论，重新按新情况进行推理。

2. 确定性推理、不确定性推理

若按推理时所用知识的确定性来划分，推理可分为确定性推理与不确定性推理。

所谓确定性推理是指推理时所用的知识与证据都是确定的，推出的结论也是确定的，其真值或者为真或者为假，没有第三种情况出现。

本章将讨论的经典逻辑推理就属于这一类。经典逻辑推理是最先提出的一类推理方法，是根据经典逻辑（命题逻辑及一阶谓词逻辑）的逻辑规则进行的一种推理，主要有自然演绎推理、归结演绎推理及与/或形演绎推理等。由于这种推理是基于经典逻辑的，其真值只有“真”和“假”两种，因此它是一种确定性推理。

所谓不确定性推理是指推理时所用的知识与证据不都是确定的，推出的结论也是不确定的。

现实世界中的事物和现象大都是不确定的，或者模糊的，很难用精确的数学模型来表示与处理。不确定性推理又分为似然推理与近似推理或模糊推理，前者是基于概率论的推理，后者是基于模糊逻辑的推理。人们经常在知识不完全、不精确的情况下进行推理，因此，要使计算机能模拟人类的思维活动，就必须使它具有不确定性推理的能力。

3. 单调推理、非单调推理

若按推理过程中推出的结论是否越来越接近最终目标来划分，推理又分为单调推理与非单调推理。

单调推理是在推理过程中随着推理向前推进及新知识的加入，推出的结论越来越接近最终目标。

单调推理的推理过程中不会出现反复的情况，即不会由于新知识的加入否定了前面推出的结论，从而使推理又退回到前面的某一步。本章将要介绍的基于经典逻辑的演绎推理属于单调性推理。

非单调推理是在推理过程中由于新知识的加入，不仅没有加强已推出的结论，反而要否定它，使推理退回到前面的某一步，然后重新开始。

非单调推理一般是在知识不完全的情况下发生的。由于知识不完全，为使推理进行下去，就要先作某些假设，并在假设的基础上进行推理。当以后由于新知识的加入发现原先的假设不正确时，就需要推翻该假设以及由此假设推出的所有结论，再用新知识重新进行推理。显然，默认推理是一种非单调推理。

在人们的日常生活及社会实践中，很多情况下进行的推理都是非单调推理。明斯基举了一个非单调推理的例子：当知道 X 是一只鸟时，一般认为 X 会飞，但之后又知道 X 是企鹅，而企鹅是不会飞的，则取消先前加入的 X 能飞的结论，而加入 X 是不会飞的结论。

4. 启发式推理、非启发式推理

若按推理中是否运用与推理有关的启发性知识来划分，推理可分为启发式推理（heuristic inference）与非启发式推理。

如果推理过程中运用与推理有关的启发性知识，则称为启发式推理，否则称为非启发式推理。

所谓启发性知识是指与问题有关且能加快推理过程、求得问题最优解的知识。例如，推理的目标是要在脑膜炎、肺炎、流感这三种疾病中选择一个，又设有 $r1$、$r2$、$r3$ 这三条产生式规则可供使用，其中 $r1$ 推出的是脑膜炎，$r2$ 推出的是肺炎，$r3$ 推出的是流感。如果希望尽早地排除脑膜炎这一危险疾病，应该先选用 $r1$；如果本地区目前正在盛行流感，则应考虑首先选择 $r3$。这里，“脑膜炎危险”及“目前正在盛行流感”是与问题求解有关的启发性信息。

3.1.3　推理的方向

推理过程是求解问题的过程。问题求解的质量与效率不仅依赖于所采用的求解方法(如匹配方法、不确定性的传递算法等),而且还依赖于求解问题的策略,即推理的控制策略。

推理的控制策略主要包括推理方向、搜索策略、冲突消解策略、求解策略及限制策略等。推理方向分为正向推理、逆向推理、混合推理及双向推理四种。

1. 正向推理

正向推理是以已知事实作为出发点的一种推理。

正向推理的基本思想:从用户提供的初始已知事实出发,在知识库 KB 中找出当前可适用的知识,构成可适用知识集 KS,然后按某种冲突消解策略从 KS 中选出一条知识进行推理,并将推出的新事实加入到数据库中作为下一步推理的已知事实,此后再在知识库中选取可适用知识进行推理,如此重复这一过程,直到求得了问题的解或者知识库中再无可适用的知识为止。

正向推理的推理过程可用如下算法描述:

① 将用户提供的初始已知事实送入数据库 DB。

② 检查数据库 DB 是否已经包含了问题的解,若有,则求解结束,并成功退出;否则,执行下一步。

③ 根据数据库 DB 中的已知事实,扫描知识库 KB,检查 KB 中是否有可适用(即可与 DB 中已知事实匹配)的知识,若有,则转向④,否则转向⑥。

④ 把 KB 中所有的适用知识都选出来,构成可适用知识集 KS。

⑤ 若 KS 不空,则按某种冲突消解策略从中选出一条知识进行推理,并将推出的新事实加入 DB 中,然后转向②;若 KS 空,则转向⑥。

⑥ 询问用户是否可进一步补充新的事实,若可补充,则将补充的新事实加入 DB 中,然后转向③;否则表示求不出解,失败退出。

以上算法如图 3.1 所示。

为了实现正向推理,有许多具体问题需要解决。例如,要从知识库中选出可适用的知识,就要用知识库中的知识与数据库中已知事实进行匹配,为此就需要确定匹配的方法。匹配通常难以做到完全一致,因此还需要解决怎样才算是匹配成功的问题。

2. 逆向推理

逆向推理是以某个假设目标作为出发点的一种推理。

逆向推理的基本思想是:首先选定一个假设目标,然后寻找支持该假设的证据,若所需的证据都能找到,则说明原假设是成立的;若无论如何都找不到所需要的证据,则说明原假设不成立的,为此需要另作新的假设。

逆向推理过程可用如下算法描述:

① 提出要求证的目标(假设)。

② 检查该目标是否已在数据库中,若在,则该目标成立,退出推理或者对下一个假设目标进行验证;否则,转下一步。

③ 判断该目标是否是证据,即它是否为应由用户证实的原始事实,若是,则询问用户;否则,转下一步。

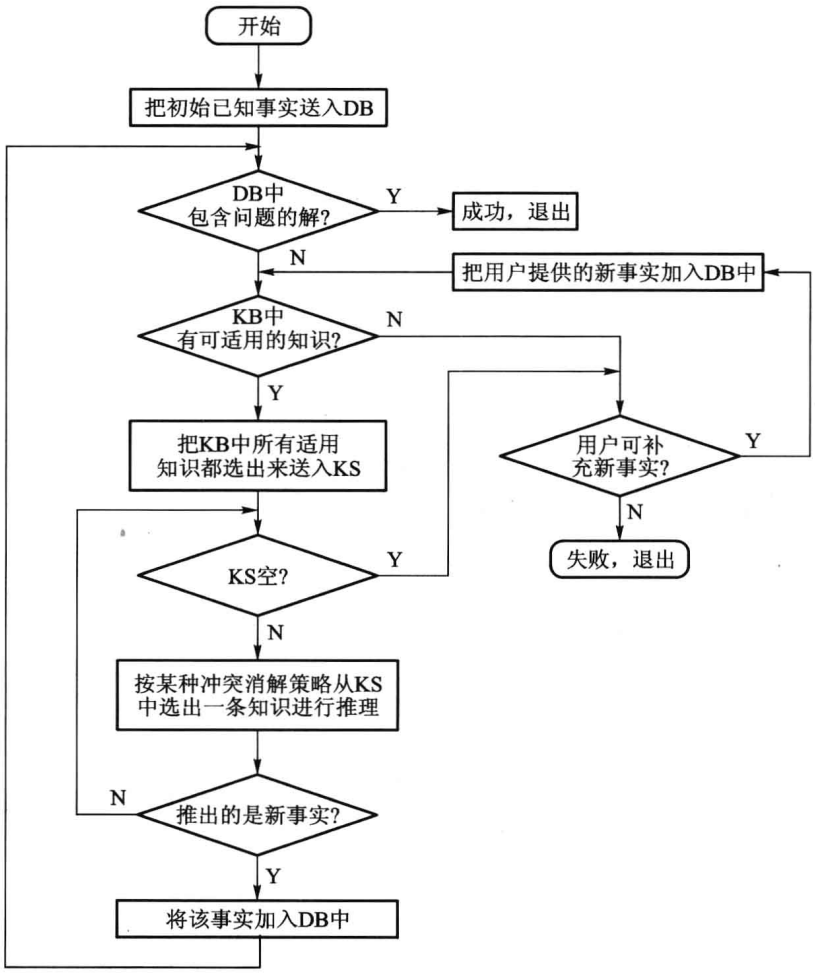


图 3.1 正向推理示意图

④ 在知识库中找出所有能导出该目标的知识，形成适用的知识集 KS，然后转下一步。

⑤ 从 KS 中选出一条知识，并将该知识的运用条件作为新的假设目标，然后转向②。

该算法可用图 3.2 示意。

与正向推理相比，逆向推理更复杂一些，上述算法只是描述了它的大致过程，许多细节没有反映出来。例如，如何判断一个假设是否是证据？当导出假设的知识有多条时，如何确定先选哪一条？另外，一条知识的运用条件一般有多个，当其中的一个经过验证成立后，如何自动地换为对另一个的验证？其次，在验证一个运用条件时，需要把它当作新的假设，并查找可导出该假设的知识，这样就又会产生一组新的运用条件，形成一个树状结构，当到达叶结点（即数据库中有相应的事实或者用户可肯定相应事实存在等）时，又需逐层向上返回，返回过程中有可能又要下到下一层，这样上上下下重复多次，才会导出原假设是否成立的结论。这是一个比较复杂的推理过程。

逆向推理的主要优点是不必使用与目标无关的知识，目的性强，同时它还有利于向用户提供

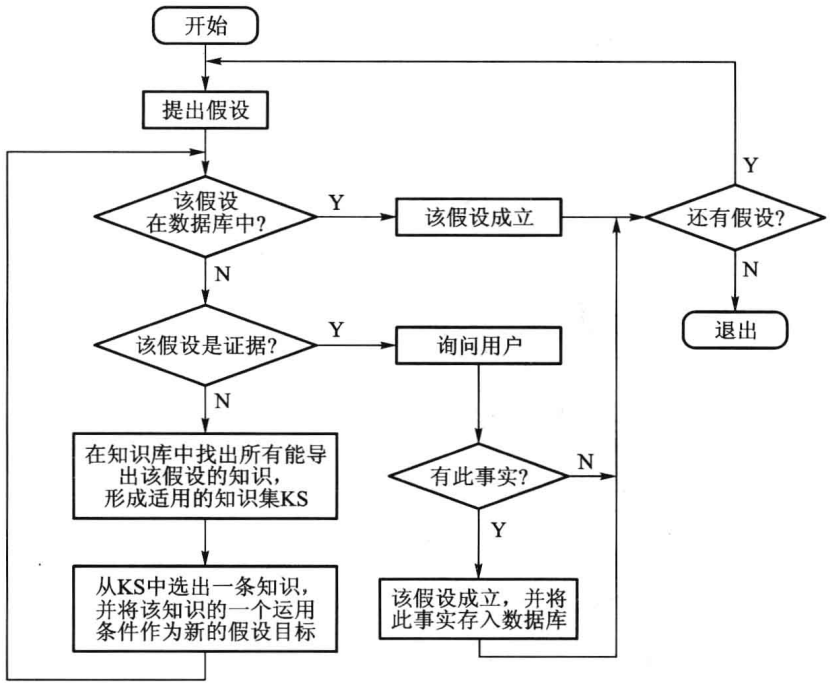


图 3.2　逆向推理示意图

解释。其主要缺点是起始目标的选择有盲目性,若不符合实际,就要多次提出假设,影响到系统的效率。

3. 混合推理

正向推理具有盲目、效率低等缺点,推理过程中可能会推出许多与问题无关的子目标。逆向推理中,若提出的假设目标不符合实际,也会降低系统的效率。为解决这些问题,可把正向推理与逆向推理结合起来,使其各自发挥自己的优势,取长补短。这种既有正向又有逆向的推理称为混合推理。另外,在下述几种情况下,通常也需要进行混合推理。

(1) 已知的事实不充分

当数据库中的已知事实不够充分时,若用这些事实与知识的运用条件相匹配进行正向推理,可能连一条适用知识都选不出来,这就使推理无法进行下去。此时,可通过正向推理先把其运用条件不能完全匹配的知识都找出来,并把这些知识可导出的结论作为假设,然后分别对这些假设进行逆向推理。由于在逆向推理中可以向用户询问有关证据,这就有可能使推理进行下去。

(2) 正向推理推出的结论可信度不高

用正向推理进行推理时,虽然推出了结论,但可信度可能不高,达不到预定的要求。因此为了得到一个可信度符合要求的结论,可用这些结论作为假设,然后进行逆向推理,通过向用户询问进一步的信息,有可能得到一个可信度较高的结论。

(3) 希望得到更多的结论

在逆向推理过程中,由于要与用户进行对话,有针对性地向用户提出询问,这就有可能获得一些原来不掌握的有用信息。这些信息不仅可用于证实要证明的假设,同时还有助于推出一些

其他结论。因此,在用逆向推理证实了某个假设之后,可以再用正向推理推出另外一些结论。例如,在医疗诊断系统中,先用逆向推理证实某病人患有某种病,然后再利用逆向推理过程中获得的信息进行正向推理,就有可能推出该病人还患有别的什么病。

由以上讨论可以看出,混合推理分为两种情况:一种是先进行正向推理,帮助选择某个目标,即从已知事实演绎出部分结果,然后再用逆向推理证实该目标或提高其可信度;另一种情况是先假设一个目标进行逆向推理,然后再利用逆向推理中得到的信息进行正向推理,以推出更多的结论。

先正向后逆向的推理过程如图 3.3 所示。

先逆向后正向的推理过程如图 3.4 所示。

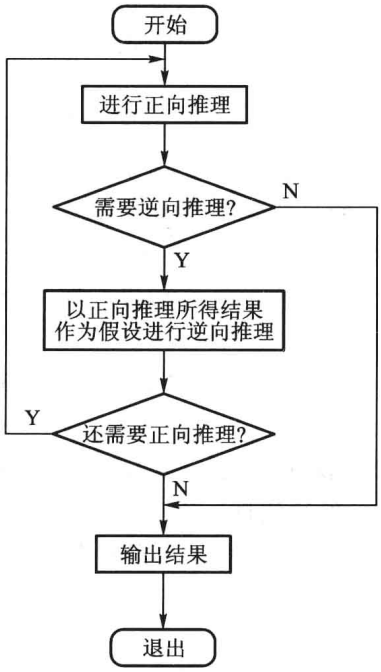


图 3.3 先正向后逆向混合推理示意图

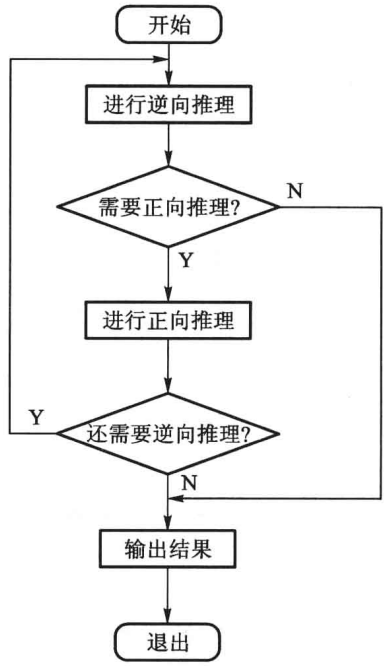


图 3.4 先逆向后正向混合推理示意图

4. 双向推理

在定理的机器证明等问题中,经常采用双向推理。所谓双向推理是指正向推理与逆向推理同时进行,且在推理过程中的某一步骤上“碰头”的一种推理。其基本思想是:一方面根据已知事实进行正向推理,但并不推到最终目标;另一方面从某假设目标出发进行逆向推理,但并不推至原始事实,而是让它们在中途相遇,即由正向推理所得到的中间结论恰好是逆向推理此时所要求的证据,这时推理就可结束,逆向推理时所做的假设就是推理的最终结论。

双向推理的困难在于“碰头”判断。另外,如何权衡正向推理与逆向推理的比重,即如何确定“碰头”的时机也是一个困难问题。

3.1.4 冲突消解策略

在推理过程中,系统要不断地用当前已知的事实与知识库中的知识进行匹配。此时,可能发

生如下三种情况：

① 已知事实恰好只与知识库中的一个知识匹配成功。

② 已知事实不能与知识库中的任何知识匹配成功。

③ 已知事实可与知识库中的多个知识匹配成功。或者多个(组) 已知事实都可与知识库中的某一个知识匹配成功;或者有多个(组)已知事实可与知识库中的多个知识匹配成功。

这里已知事实与知识库中的知识匹配成功的含义,对正向推理而言,是指产生式规则的前件和已知事实匹配成功;对逆向推理而言,是指产生式规则的后件和假设匹配成功。

对于第一种情况,由于匹配成功的知识只有一个,所以它就是可应用的知识,可直接把它应用于当前的推理。

当第二种情况发生时,由于找不到可与当前已知事实匹配成功的知识,使得推理无法继续进行下去。这或者是由于知识库中缺少某些必要的知识,或者由于要求解的问题超出了系统功能范围等,此时可根据当前的实际情况作相应的处理。

第三种情况刚好与第二种情况相反,推理过程中不仅有知识匹配成功,而且有多个知识匹配成功,称为发生了冲突。按一定的策略从匹配成功的多个知识中挑出一个知识用于当前的推理的过程称为冲突消解(conflict resolution)。解决冲突时所用的策略称为冲突消解策略。对正向推理而言,它将决定选择哪一组已知事实来激活哪一条产生式规则,使它用于当前的推理,产生其后件指出的结论或执行相应的操作。对逆向推理而言,它将决定哪一个假设与哪一个产生式规则的后件进行匹配,从而推出相应的前件,作为新的假设。

目前已有多种消解冲突的策略,其基本思想都是对知识进行排序。常用的有以下几种。

1. 按针对性排序

本策略是优先选用针对性较强的产生式规则。如果 $r2$ 中除了包括 $r1$ 要求的全部条件外,还包括其他条件,则称 $r2$ 比 $r1$ 有更大的针对性,$r1$ 比 $r2$ 有更大的通用性。因此,当 $r2$ 与 $r1$ 发生冲突时,优先选用 $r2$。因为它要求的条件较多,其结论一般更接近于目标,一旦得到满足,可缩短推理过程。

2. 按已知事实的新鲜性排序

在产生式系统的推理过程中,每应用一条产生式规则就会得到一个或多个结论或者执行某个操作,数据库就会增加新的事实。另外,在推理时还会向用户询问有关的信息,也使数据库的内容发生变化。一般把数据库中后生成的事实称为新鲜的事实,即后生成的事实比先生成的事实具有较大的新鲜性。若一条规则被应用后生成了多个结论,则既可以认为这些结论有相同的新鲜性,也可以认为排在前面(或后面)的结论有较大的新鲜性,根据情况决定。

设规则 $r1$ 可与事实组 A 匹配成功,规则 $r2$ 可与事实组 B 匹配成功,则 A 与 B 中哪一组新鲜,与它匹配的产生式规则就先被应用。

如何衡量 A 与 B 中哪一组事实更新鲜呢? 常用的方法有以下三种:

① 把 A 与 B 中的事实逐个比较其新鲜性,若 A 中包含的更新鲜的事实比 B 多,就认为 A 比 B 新鲜。例如,设 A 与 B 中各有五个事实,而 A 中有三个事实比 B 中的事实更新鲜,则认为 A 比 B 新鲜。

② 以 A 中最新鲜的事实与 B 中最新鲜的事实相比较,哪一个更新鲜,就认为相应的事实组更新鲜。

③ 以 A 中最不新鲜的事实与 B 中最不新鲜的事实相比较，哪一个更不新鲜，就认为相应的事实组有较小的新鲜性。

3. 按匹配度排序

在不确定性推理中，需要计算已知事实与知识的匹配度，当其匹配度达到某个预先规定的值时，就认为它们是可匹配的。若产生式规则 $r1$ 与 $r2$ 都可匹配成功，则优先选用匹配度较大的产生式规则。

4. 按条件个数排序

如果有多条产生式规则生成的结论相同，则优先应用条件少的产生式规则，因为条件少的规则匹配时花费的时间较少。

在具体应用时，可对上述几种策略进行组合，尽量减少冲突的发生，使推理有较快的速度和较高的效率。

3.2 自然演绎推理

从一组已知为真的事实出发，直接运用经典逻辑的推理规则推出结论的过程称为自然演绎推理。其中，基本的推理是 P 规则、T 规则、假言推理、拒取式推理等。

假言推理的一般形式是

$$P,\ P\rightarrow Q\Rightarrow Q$$

它表示：由 $P\rightarrow Q$ 及 P 为真，可推出 Q 为真。

例如，由“如果 x 是金属，则 x 能导电”及“铜是金属”可推出“铜能导电”的结论。

拒取式推理的一般形式是

$$P\rightarrow Q,\ \neg Q\Rightarrow\neg P$$

它表示：由 $P\rightarrow Q$ 为真及 Q 为假，可推出 P 为假。

例如，由“如果下雨，则地上就湿”及“地上不湿”可推出“没有下雨”的结论。

这里，应该注意避免如下两类错误：一种是肯定后件（Q）的错误，另一种是否定前件（P）的错误。

所谓肯定后件是指，当 $P\rightarrow Q$ 为真时，希望通过肯定后件 Q 为真来推出前件 P 为真，这是不允许的。

例如，伽利略在论证哥白尼的日心说时，曾使用了如下推理：

① 如果行星系统是以太阳为中心的，则金星会显示出位相变化。

② 金星显示出位相变化（肯定后件）。

③ 所以，行星系统是以太阳为中心。

因为这里使用了肯定后件的推理，违反了经典逻辑规则，他为此遭到非难。

所谓否定前件是指，当 $P\rightarrow Q$ 为真时，希望通过否定前件 P 来推出后件 Q 为假，这也是不允许的。例如，下面的推理就是使用了否定前件的推理，从而违反了逻辑规则：

① 如果下雨，则地上是湿的。

② 没有下雨（否定前件）。

③ 所以，地上不湿。

这显然是不正确的。因为当地上洒水时，地上也会湿。事实上，只要仔细分析蕴涵 $P \rightarrow Q$ 的定义，就会发现当 $P \rightarrow Q$ 为真时，肯定后件或否定前件所得的结论既可能为真，也可能为假，不能确定。

下面举例说明自然演绎推理方法。

例 3.1　设已知如下事实：

① 凡是容易的课程小王（*Wang*）都喜欢。

② *C* 班的课程都是容易的。

③ *ds* 是 *C* 班的一门课程。

求证：小王喜欢 *ds* 这门课程。

证明　首先定义谓词：

$EASY(x)$：x 是容易的；

$LIKE(x,y)$：x 喜欢 y；

$C(x)$：x 是 *C* 班的一门课程。

把上述已知事实及待求证的问题用谓词公式表示出来：

$(\forall x)(EASY(x) \rightarrow LIKE(Wang,x))$	凡是容易的课程小王都是喜欢的；
$(\forall x)(C(x) \rightarrow EASY(x))$	*C* 班的课程都是容易的；
$C(ds)$	*ds* 是 *C* 班的课程；
$LIKE(Wang,ds)$	小王喜欢的 *ds* 这门课程，这是待求证的问题。

应用推理规则进行推理：

因为

$$(\forall x)(EASY(x) \rightarrow LIKE(Wang,x))$$

所以由全称固化得

$$EASY(z) \rightarrow LIKE(Wang, z)$$

因为

$$(\forall x)\ (C(x) \rightarrow EASY(x))$$

所以由全称固化得

$$C(y) \rightarrow EASY(y)$$

由 *P* 规则及假言推理得

$$C(ds), C(y) \rightarrow EASY(y) \Rightarrow EASY(ds)$$

$$EASY(ds), EASY(z) \rightarrow LIKE(Wang,z)$$

由 *T* 规则及假言推理得

$$LIKE(Wang,ds)$$

即小王喜欢 *ds* 这门课程。

一般来说，由已知事实推出的结论可能有多个，只要其中包括了待证明的结论，就认为问题得到了解决。

自然演绎推理的优点是表达定理证明过程自然，容易理解，而且它拥有丰富的推理规则，推理过程灵活，便于在它的推理规则中嵌入领域启发式知识。其缺点是容易产生组合爆炸，推理过程得到的中间结论一般呈指数形式递增，这对于一个大的推理问题来说是十分不利的。

3.3 谓词公式化为子句集的方法

在谓词逻辑中,有下述定义:

原子(atom) 谓词公式是一个不能再分解的命题。

原子谓词公式及其否定,统称为文字(literal)。P 称为正文字,$\neg P$ 称为负文字。P 与 $\neg P$ 为互补文字。

任何文字的析取式称为子句(clause)。任何文字本身也是子句。

由子句构成的集合称为子句集。

不包含任何文字的子句称为空子句,表示为 *NIL*。

由于空子句不含有文字,它不能被任何解释满足,所以,空子句是永假的、不可满足的。

在谓词逻辑中,任何一个谓词公式都可以通过应用等价关系及推理规则化成相应的子句集,从而能够比较容易地判定谓词公式的不可满足性。下面结合一个具体的例子,说明把谓词公式化为子句集的步骤。

例 3.2 将下列谓词公式化为子句集:

$$(\forall x)((\forall y)P(x,y) \rightarrow \neg(\forall y)(Q(x,y) \rightarrow R(x,y)))$$

解 (1) 消去谓词公式中的"→"和"↔"符号

利用谓词公式的等价关系:

$$P \rightarrow Q \Leftrightarrow \neg P \vee Q$$

$$P \leftrightarrow Q \Leftrightarrow (P \wedge Q) \vee (\neg P \wedge \neg Q)$$

上例等价变换为

$$(\forall x)(\neg(\forall y)P(x,y) \vee \neg(\forall y)(\neg Q(x,y) \vee R(x,y)))$$

(2) 把否定符号移到紧靠谓词的位置上

利用谓词公式的等价关系:

双重否定律 $\neg(\neg P) \Leftrightarrow P$

德摩根律 $\neg(P \wedge Q) \Leftrightarrow \neg P \vee \neg Q$

$\neg(P \vee Q) \Leftrightarrow \neg P \wedge \neg Q$

量词转换律 $\neg(\forall x)P \Leftrightarrow (\exists x)\neg P$

$\neg(\exists x)P \Leftrightarrow (\forall x)\neg P$

把否定符号移到紧靠谓词的位置上,减少了否定符号的辖域。

上例等价变换为

$$(\forall x)((\exists y)\neg P(x,y) \vee (\exists y)(Q(x,y) \wedge \neg R(x,y)))$$

(3) 变量标准化

所谓变量标准化就是重新命名变元,使每个量词采用不同的变元,从而使不同量词的约束变元有不同的名字。这是因为在任一量词辖域内,受到该量词约束的变元为一哑元(虚构变量),它可以在该辖域内被另一个没有出现过的任意变元统一代替,而不改变谓词公式的值。

$$(\forall x)P(x) \equiv (\forall y)P(y)$$

$$(\exists x)P(x) \equiv (\exists y)P(y)$$

上例等价变换为

$$(\forall x)((\exists y)\neg P(x,y)\vee(\exists z)(Q(x,z)\wedge\neg R(x,z)))$$

(4) 消去存在量词

分两种情况：

一种情况是存在量词不出现在全称量词的辖域内。此时只要用一个新的个体常量替换受该存在量词约束的变元，就可以消去存在量词。因为如原谓词公式为真，则总能够找到一个个体常量，替换后仍然使谓词公式为真。这里的个体常量就是不含变量的 Skolem 函数。

另一种情况是存在量词出现在一个或者多个全称量词的辖域内。此时要用 Skolem 函数替换受该存在量词约束的变元，从而消去存在量词。这里认为所存在的 y 依赖于 x 值，它们的依赖关系由 Skolem 函数所定义。

对于一般情况

$$(\forall x_1)(\forall x_2)\cdots(\forall x_n)(\exists y)P(x_1,x_2,\cdots,x_n,y)$$

存在量词 y 的 Skolem 函数记为

$$y=f(x_1,x_2,\cdots,x_n)$$

可见，Skolem 函数把每个 $x_1,x_2,\cdots,x_n$ 值，映射到存在的那个 y。

用 Skolem 函数代替每个存在量词量化的变量的过程称为 Skolem 化。Skolem 函数所使用的函数符号必须是新的。

对于上面的例子，存在量词 $(\exists y)$ 及 $(\exists z)$ 都位于全称量词 $(\forall x)$ 的辖域内，所以都需要用 Skolem 函数代替。设 y 和 z 的 Skolem 函数分别记为 $f(x)$ 和 $g(x)$，则替换后得到

$$(\forall x)(\neg P(x,f(x))\vee(Q(x,g(x))\wedge\neg R(x,g(x))))$$

(5) 化为前束形

所谓前束形，就是把所有的全称量词都移到公式的前面，使每个量词的辖域都包括公式后的整个部分，即

前束形 =（前缀）{母式}

其中，（前缀）是全称量词串，{母式} 是不含量词的谓词公式。

对于上面的例子，因为只有一个全称量词，而且已经位于公式的最左边，所以，这一步不需要做任何工作。

(6) 化为 Skolem 标准形

Skolem 标准形的一般形式是：

$$(\forall x_1)(\forall x_2)\cdots(\forall x_n)M$$

其中，M 是子句的合取式，称为 Skolem 标准形的母式。

一般利用

$$P\vee(Q\wedge R)\Leftrightarrow(P\vee Q)\wedge(P\vee R)$$

或

$$P\wedge(Q\vee R)\Leftrightarrow(P\wedge Q)\vee(P\wedge R)$$

把谓词公式化为 Skolem 标准形。

对于上面的例子，有

$$(\forall x)((\neg P(x,f(x))\vee Q(x,g(x)))\wedge(\neg P(x,f(x))\vee\neg R(x,g(x))))$$

(7) 略去全称量词

由于公式中所有变量都是全称量词量化的变量,因此,可以省略全称量词。母式中的变量仍然认为是全称量词量化的变量。

对于上面的例子,有

$$(\neg P(x,f(x)) \vee Q(x,g(x))) \wedge (\neg P(x,f(x)) \vee \neg R(x,g(x)))$$

(8) 消去合取词,把母式用子句集表示

对于上面的例子,有

$$\{\neg P(x,f(x)) \vee Q(x,g(x)),\ \neg P(x,f(x)) \vee \neg R(x,g(x))\}$$

(9) 子句变量标准化,即使每个子句中的变量符号不同

谓词公式的性质有

$$(\forall x)[P(x) \wedge Q(x)] \equiv (\forall x)P(x) \wedge (\forall y)Q(y)$$

对于上面的例子,有

$$\{\neg P(x,f(x)) \vee Q(x,g(x)),\ \neg P(y,f(y)) \vee \neg R(y,g(y))\}$$

显然,在子句集中各子句之间是合取关系。

上面介绍了将谓词公式化为子句集的步骤。下面再举几个例子进一步说明。

例 3.3 将下列谓词公式化为子句集:

$$(\forall x)\{[\neg P(x) \vee \neg Q(x)] \rightarrow (\exists y)[S(x,y) \wedge Q(x)]\} \wedge (\forall x)[P(x) \vee B(x)]$$

解 (1) 消去蕴涵符号

$$(\forall x)\{\neg [\neg P(x) \vee \neg Q(x)] \vee (\exists y)[S(x,y) \wedge Q(x)]\} \wedge (\forall x)[P(x) \vee B(x)]$$

(2) 把否定符号移到每个谓词前面

$$(\forall x)\{[P(x) \wedge Q(x)] \vee (\exists y)[S(x,y) \wedge Q(x)]\} \wedge (\forall x)[P(x) \vee B(x)]$$

(3) 变量标准化

$$(\forall x)\{[P(x) \wedge Q(x)] \vee (\exists y)[S(x,y) \wedge Q(x)]\} \wedge (\forall w)[P(w) \vee B(w)]$$

(4) 消去存在量词

设 y 的 Skolem 函数是 $f(x)$,则

$$(\forall x)\{[P(x) \wedge Q(x)] \vee [S(x,f(x)) \wedge Q(x)]\} \wedge (\forall w)[P(w) \vee B(w)]$$

(5) 化为前束形

$$(\forall x)(\forall w)\{\{[P(x) \wedge Q(x)] \vee [S(x,f(x)) \wedge Q(x)]\} \wedge [P(w) \vee B(w)]\}$$

(6) 化为 Skolem 标准形

根据

$$P \wedge (Q \vee R) \Leftrightarrow (P \wedge Q) \vee (P \wedge R)$$

或者

$$(P \wedge Q) \vee (P \wedge R) \Leftrightarrow P \wedge (Q \vee R)$$

可以得到

$$(\forall x)(\forall w)\{\{[Q(x) \wedge P(x)] \vee [Q(x) \wedge S(x,f(x))]\} \wedge [P(w) \vee B(w)]\}$$

$$(\forall x)(\forall w)\{Q(x) \wedge [P(x) \vee S(x,f(x))] \wedge [P(w) \vee B(w)]\}$$

(7) 略去全称量词

$$Q(x) \wedge [P(x) \vee S(x,f(x))] \wedge [P(w) \vee B(w)]$$

(8) 消去合取词,把母式用子句集表示

$$\{Q(x),P(x) \vee S(x,f(x)),P(w) \vee B(w)\}$$

(9) 子句变量标准化,即使每个子句中的变量符号不同

$$\{Q(x),P(y) \vee S(y,f(y)),P(w) \vee B(w)\}$$

例 3.4　将下列谓词公式化为子句集:

$$(\forall x)\{P(x) \rightarrow \{(\forall y)[P(y) \rightarrow P(f(x,y))] \wedge \neg (\forall y)[Q(x,y) \rightarrow P(y)]\}\}$$

解　(1) 消去蕴涵符号

$$(\forall x)\{\neg P(x) \vee \{(\forall y)[\neg P(y) \vee P(f(x,y))] \wedge \neg (\forall y)[\neg Q(x,y) \vee P(y)]\}\}$$

(2) 把否定符号移到每个谓词前面

$$(\forall x)\{\neg P(x) \vee \{(\forall y)[\neg P(y) \vee P(f(x,y))] \wedge (\exists y)\{\neg [\neg Q(x,y) \vee P(y)]\}\}\}$$

$$(\forall x)\{\neg P(x) \vee \{(\forall y)[\neg P(y) \vee P(f(x,y))] \wedge (\exists y)[Q(x,y) \wedge \neg P(y)]\}\}$$

(3) 变量标准化

$$(\forall x)\{\neg P(x) \vee \{(\forall y)[\neg P(y) \vee P(f(x,y))] \wedge (\exists w)[Q(x,w) \wedge \neg P(w)]\}\}$$

(4) 消去存在量词

设 w 的 Skolem 函数是 $g(x)$,则

$$(\forall x)\{\neg P(x) \vee \{(\forall y)[\neg P(y) \vee P(f(x,y))] \wedge [Q(x,g(x)) \wedge \neg P(g(x))]\}\}$$

(5) 化为前束形

$$(\forall x)(\forall y)\{\neg P(x) \vee \{[\neg P(y) \vee P(f(x,y))] \wedge [Q(x,g(x)) \wedge \neg P(g(x))]\}\}$$

(6) 化为 Skolem 标准形

$$(\forall x)(\forall y)\{[\neg P(x) \vee \neg P(y) \vee P(f(x,y))] \wedge$$
$$[\neg P(x) \vee Q(x,g(x))] \wedge [\neg P(x) \vee \neg P(g(x))]\}$$

(7) 略去全称量词

$$\{[\neg P(x) \vee \neg P(y) \vee P(f(x,y))] \wedge [\neg P(x) \vee Q(x,g(x))] \wedge [\neg P(x) \vee \neg P(g(x))]\}$$

(8) 消去合取词,把母式用子句集表示

$$\{\neg P(x) \vee \neg P(y) \vee P(f(x,y)),\neg P(x) \vee Q(x,g(x)),\neg P(x) \vee \neg P(g(x))\}$$

(9) 子句变量标准化,即使每个子句中的变量符号不同

$$\{\neg P(x_1) \vee \neg P(y) \vee P(f(x_1,y)),\neg P(x_2) \vee Q(x_2,g(x_2)),\neg P(x_3) \vee \neg P(g(x_3))\}$$

例 3.5　将下列谓词公式化为不含存在量词的前束形:

$$(\exists x)(\forall y)((\forall z)(P(z) \wedge \neg Q(x,z)) \rightarrow R(x,y,f(a)))$$

解　消去存在量词,得

$$(\forall y)((\forall z)(P(z) \wedge \neg Q(b,z)) \rightarrow R(b,y,f(a)))$$

消去蕴涵符号,得

$$(\forall y)(\neg (\forall z)(P(z) \wedge \neg Q(b,z)) \vee R(b,y,f(a)))$$

$$(\forall y)((\exists z)(\neg P(z) \vee Q(b,z)) \vee R(b,y,f(a)))$$

设 z 的 Skolem 函数是 $g(y)$,则有

$$(\forall y)(\neg P(g(y)) \vee Q(b,g(y)) \vee R(b,y,f(a)))$$

上面把谓词公式化成了相应的子句集，下面的定理表明两者的不可满足性是等价的。

定理 3.1 谓词公式不可满足的充要条件是其子句集不可满足。

由此定理可知，要证明一个谓词公式是不可满足的，只要证明相应的子句集是不可满足的就可以了。如何证明一个子句集是不可满足的呢？下面介绍一下鲁宾逊归结原理。

3.4 鲁宾逊归结原理

从前面的分析可以看出，谓词公式的不可满足性分析可以转化为子句集中子句的不可满足性分析。为了判定子句集的不可满足性，就需要对子句集中的子句进行判定。而为了判定一个子句的不可满足性，需要对个体域上的一切解释逐个地进行判定，只有当子句对任何非空个体域上的任何一个解释都是不可满足的时候，才能判定该子句是不可满足的，这是一件非常困难的工作，要在计算机上实现其证明过程是很困难的。1965 年鲁宾逊提出了归结原理，使机器定理证明变为现实。

鲁宾逊归结原理（Robinson resolution principle）又称为消解原理，是鲁宾逊提出的一种证明子句集不可满足性，从而实现定理证明的一种理论及方法。它是机器定理证明的基础。

由谓词公式转化为子句集的过程可以看出，在子句集中子句之间是合取关系，其中只要有一个子句不可满足，则子句集就不可满足。由于空子句是不可满足的，所以，若一个子句集中包含空子句，则这个子句集一定是不可满足的。鲁宾逊归结原理就是基于这个思想提出来的。其基本方法是：检查子句集 S 中是否包含空子句，若包含，则 S 不可满足；若不包含，就在子句集中选择合适的子句进行归结，一旦通过归结得到空子句，就说明子句集 S 是不可满足的。

下面对命题逻辑及谓词逻辑分别给出归结的定义。

1. 命题逻辑中的归结原理

定义 3.1 设 C_1 与 C_2 是子句集中的任意两个子句，如果 C_1 中的文字 L_1 与 C_2 中的文字 L_2 互补，那么从 C_1 和 C_2 中分别消去 L_1 和 L_2，并将二个子句中余下的部分析取，构成一个新子句 C_{12}，这一过程称为归结。C_{12}称为 C_1 和 C_2 的归结式，C_1 和 C_2 称为 C_{12}的亲本子句。

下面举例说明具体的归结方法。

例如，在子句集中取两个子句 $C_1 = P, C_2 = \neg P$，可见，C_1 与 C_2 是互补文字，则通过归结可得归结式 $C_{12} = NIL$ 。这里 NIL 代表空子句。

又如，设 $C_1 = \neg P \vee Q \vee R, C_2 = \neg Q \vee S$，可见，这里 $L_1 = Q, L_2 = \neg Q$，通过归结可得归结式 $C_{12} = \neg P \vee R \vee S$

例如，设 $C_1 = \neg P \vee Q, C_2 = \neg Q \vee R, C_3 = P$。

首先对 C_1 和 C_2 进行归结，得到

$$C_{12} = \neg P \vee R$$

然后再用 C_{12}与 C_3 进行归结，得到

$$C_{123} = R$$

如果首先对 C_1 和 C_3 进行归结，然后再把其归结式与 C_2 进行归结，将得到相同的结果。归结过程可用树形图直观地表示出来。上面的归结过程可用图 3.5 表示。

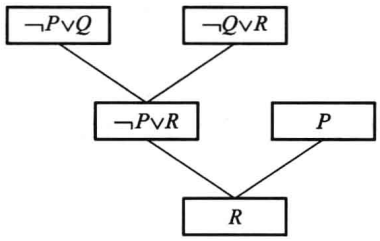


图 3.5　归结过程的树形表示

定理 3.2　归结式 C_{12} 是其亲本子句 C_1 与 C_2 的逻辑结论。即如果 C_1 与 C_2 为真，则 C_{12} 为真。

证明　设 $C_1 = L \vee C_1'$，$C_2 = \neg L \vee C_2'$，

通过归结可以得到 C_1 和 C_2 的归结式 $C_{12} = C_1' \vee C_2'$。

因为
$$C_1' \vee L \Leftrightarrow \neg C_1' \rightarrow L$$
$$\neg L \vee C_2' \Leftrightarrow L \rightarrow C_2'$$
所以
$$C_1 \wedge C_2 = (\neg C_1' \rightarrow L) \wedge (L \rightarrow C_2')$$
根据假言三段论得到
$$(\neg C_1' \rightarrow L) \wedge (L \rightarrow C_2') \Rightarrow \neg C_1' \rightarrow C_2'$$
因为
$$\neg C_1' \rightarrow C_2' \Leftrightarrow C_1' \vee C_2' = C_{12}$$
所以
$$C_1 \wedge C_2 \Rightarrow C_{12}$$

由逻辑结论的定义即由 $C_1 \wedge C_2$ 的不可满足性可推出 C_{12} 的不可满足性，可知 C_{12} 是其亲本子句 C_1 和 C_2 的逻辑结论。

（证毕）

这个定理是归结原理中的一个很重要的定理。由它可得到如下两个重要的推论。

推论 1　设 C_1 与 C_2 是子句集 S 中的两个子句，C_{12} 是它们的归结式，若用 C_{12} 代替 C_1 和 C_2 后得到新子句集 S_1，则由 S_1 不可满足性可推出原子句集 S 的不可满足性，即

$$S_1 \text{ 的不可满足性} \Rightarrow S \text{ 的不可满足性}$$

推论 2　设 C_1 与 C_2 是子句集 S 中的两个子句，C_{12} 是它们的归结式，若把 C_{12} 加入原子句集 S 中，得到新子句集 S_2，则 S 与 S_2 在不可满足的意义上是等价的，即

$$S_2 \text{ 的不可满足性} \Leftrightarrow S \text{ 的不可满足性}$$

这两个推论说明：为要证明子句集 S 的不可满足性，只要对其中可进行归结的子句进行归结，并把归结式加入子句集 S，或者用归结式替换它的亲本子句，然后对新子句集（S_1 或 S_2）证明不可满足性就可以了。注意到空子句是不可满足的，因此，如果经过归结能得到空子句，则立即可得到原子句集 S 是不可满足的结论。这就是用归结原理证明子句集不可满足性的基本思想。

2. 谓词逻辑中的归结原理

在谓词逻辑中，由于子句中含有变元，所以不像命题逻辑那样可直接消去互补文字，而需要先用最一般合一对变元进行代换，然后才能进行归结。

例如，设有如下两个子句

$$C_1 = P(x) \vee Q(x)$$
$$C_2 = \neg P(a) \vee R(y)$$

由于 $P(x)$ 与 $P(a)$ 不同，所以 C_1 与 C_2 不能直接进行归结，但若用最一般合一

$$\sigma = \{a/x\}$$

对两个子句分别进行代换

$$C_1\sigma = P(a) \vee Q(a)$$
$$C_2\sigma = \neg P(a) \vee R(y)$$

就可对它们进行直接归结，消去 $P(a)$ 与 $\neg P(a)$，得到如下归结式

$$Q(a) \vee R(y)$$

下面给出谓词逻辑中关于归结的定义。

定义 3.2 设 C_1 与 C_2 是两个没有相同变元的子句，L_1 和 L_2 分别是 C_1 和 C_2 中的文字，若 σ 是 L_1 和 $\neg L_2$ 的最一般合一，则称

$$C_{12} = (C_1\sigma - \{L_1\sigma\}) \vee (C_2\sigma - \{L_2\sigma\})$$

为 C_1 和 C_2 的二元归结式。

例 3.6 设 $C_1 = P(a) \vee \neg Q(x) \vee R(x)$，$C_2 = \neg P(y) \vee Q(b)$，求其二元归结式。

解 若选 $L_1 = P(a)$，$L_2 = \neg P(y)$，则 $\sigma = \{a/y\}$ 是 L_1 与 $\neg L_2$ 的最一般合一。因此，

$$C_1\sigma = P(a) \vee \neg Q(x) \vee R(x)$$
$$C_2\sigma = \neg P(a) \vee Q(b)$$

根据定义可得

$$\begin{aligned}
C_{12} &= (C_1\sigma - \{L_1\sigma\}) \vee (C_2\sigma - \{L_2\sigma\}) \\
&= (\{P(a), \neg Q(x), R(x)\} - \{P(a)\}) \vee (\{\neg P(a), Q(b)\} - \{\neg P(a)\}) \\
&= (\{\neg Q(x), R(x)\}) \vee (\{Q(b)\}) \\
&= \{\neg Q(x), R(x), Q(b)\} \\
&= \neg Q(x) \vee R(x) \vee Q(b)
\end{aligned}$$

若选 $L_1 = \neg Q(x)$，$L_2 = Q(b)$，$\sigma = \{b/x\}$，则可得

$$\begin{aligned}
C_{12} &= (\{P(a), \neg Q(b), R(b)\} - \{\neg Q(b)\}) \vee (\{\neg P(y), Q(b)\} - \{Q(b)\}) \\
&= (\{P(a), R(b)\}) \vee (\{\neg P(y)\}) \\
&= \{P(a), R(b), \neg P(y)\} \\
&= P(a) \vee R(b) \vee \neg P(y)
\end{aligned}$$

例 3.7 设 $C_1 = P(x) \vee Q(a)$，$C_2 = \neg P(b) \vee R(x)$，求其二元归结式。

解 由于 C_1 与 C_2 有相同的变元，不符合定义的要求。为了进行归结，需修改 C_2 中的变元的名字，令 $C_2 = \neg P(b) \vee R(y)$。此时，$L_1 = P(x)$，$L_2 = \neg P(b)$。

L_1 与 $\neg L_2$ 的最一般合一 $\sigma = \{b/x\}$。则

$$\begin{aligned}
C_{12} &= (\{P(b), Q(a)\} - \{P(b)\}) \vee (\{\neg P(b), R(y)\} - \{\neg P(b)\}) \\
&= \{Q(a), R(y)\} \\
&= Q(a) \vee R(y)
\end{aligned}$$

如果在参加归结的子句内部含有可合一的文字，则在归结之前应对这些文字先进行合一。

例 3.8　设有如下两个子句 $C_1 = P(x) \vee P(f(a)) \vee Q(x)$，$C_2 = \neg P(y) \vee R(b)$，求其二元归结式。

解　在 C_1 中有可合一的文字 $P(x)$ 与 $P(f(a))$，若用它们的最一般合一 $\theta = \{f(a)/x\}$ 进行代换，得到 $C_1\theta = P(f(a)) \vee Q(f(a))$。此时可对 $C_1\theta$ 和 C_2 进行归结，从而得到 C_1 与 C_2 的二元归结式。

对 $C_1\theta$ 和 C_2 分别选 $L_1 = P(f(a))$，$L_2 = \neg P(y)$。L_1 和 $\neg L_2$ 的最一般合一是 $\sigma = \{f(a)/y\}$，则 $C_{12} = R(b) \vee Q(f(a))$。

在上例中，把 $C_1\theta$ 称为 C_1 的因子。一般来说，若子句 C 中有两个或两个以上的文字具有最一般的合一 σ，则称 $C\sigma$ 为子句 C 的因子。如果 $C\sigma$ 是一个单文字，则称它为 C 的单元因子。

应用因子的概念，可对谓词逻辑中的归结原理给出如下定义。

定义 3.3　子句 C_1 和 C_2 的归结式是下列二元归结式之一：

① C_1 与 C_2 的二元归结式。

② C_1 与 C_2 的因子 $C_2\sigma_2$ 的二元归结式。

③ C_1 的因子 $C_1\sigma_1$ 与 C_2 的二元归结式。

④ C_1 的因子 $C_1\sigma_1$ 与 C_2 的因子 $C_2\sigma_2$ 的二元归结式。

与命题逻辑中的归结原理相同，对于谓词逻辑，归结式是其亲本子句的逻辑结论。用归结式取代它在子句集 S 中的亲本子句所得到的新子句集仍然保持着原子句集 S 的不可满足性。

另外，对于一阶谓词逻辑，从不可满足的意义上说，归结原理也是完备的。即若子句集是不可满足的，则必存在一个从该子句集到空子句的归结演绎；若从子句集存在一个到空子句的演绎，则该子句集是不可满足的。关于归结原理的完备性可用海伯伦的有关理论进行证明，这里不再讨论。

需要指出的是，如果没有归结出空子句，则既不能说 S 不可满足，也不能说 S 是可满足的。因为，有可能 S 是可满足的，而归结不出空子句，也可能是没有找到合适的归结演绎步骤，而归结不出空子句。但是，如果确定不存在任何方法归结出空子句，则可以确定 S 是可满足的。

3.5　归结反演

归结原理给出了证明子句集不可满足性的方法。根据第 2 章的知识可知，如欲证明 Q 为 $P_1, P_2, \cdots, P_n$ 的逻辑结论，只需证明

$$(P_1 \wedge P_2 \wedge \cdots \wedge P_n) \wedge \neg Q$$

是不可满足的。再根据定理 3.1 可知，在不可满足的意义上，谓词公式的不可满足性与其子句集的不可满足性是等价的。因此，可用归结原理进行定理的自动证明。

应用归结原理证明定理的过程称为归结反演。归结反演的一般步骤是：

① 将已知前提表示为谓词公式 F。

② 将待证明的结论表示为谓词公式 Q，并否定得到 $\neg Q$。

③ 把谓词公式集 $\{F, \neg Q\}$ 化为子句集 S。

④ 应用归结原理对子句集 S 中的子句进行归结，并把每次归结得到的归结式都并入到 S 中。如此反复进行，若出现了空子句，则停止归结，此时就证明了 Q 为真。

例 3.9 某公司招聘工作人员，A，B，C 三人应试，经面试后公司表示如下想法：

① 三人中至少录取一人。

② 如果录取 A 而不录取 B，则一定录取 C。

③ 如果录取 B，则一定录取 C。

求证：公司一定录取 C。

证明 设用谓词 $P(x)$ 表示录取 x，则把公司的想法用谓词公式表示如下：

(1) $P(A) \vee P(B) \vee P(C)$

(2) $P(A) \wedge \neg P(B) \rightarrow P(C)$

(3) $P(B) \rightarrow P(C)$

把要求证的结论用谓词公式表示出来并否定，得

(4) $\neg P(C)$

把上述公式化成子句集

(1) $P(A) \vee P(B) \vee P(C)$

(2) $\neg P(A) \vee P(B) \vee P(C)$

(3) $\neg P(B) \vee P(C)$

(4) $\neg P(C)$

应用归结原理进行归结

(5) $P(B) \vee P(C)$ (1)与(2)归结

(6) $P(C)$ (3)与(5)归结

(7) NIL (4)与(6)归结

所以公司一定录取 C。

上述归结过程可用图 3.6 的归结树表示。

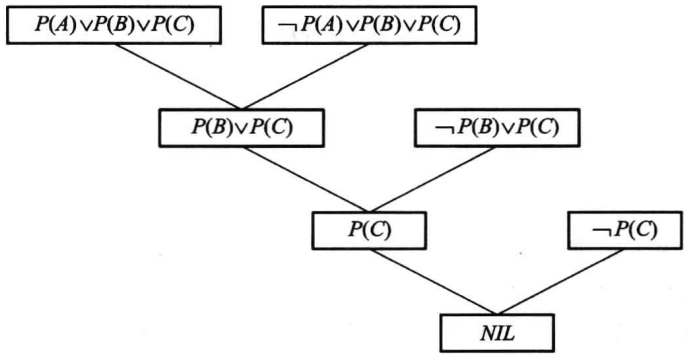


图 3.6 例 3.9 的归结树

例 3.10 已知如下信息。

规则 1：任何人的兄弟不是女性。

规则 2：任何人的姐妹必是女性。

事实：Mary 是 Bill 的姐妹。

求证：Mary 不是 Tom 的兄弟。

解　定义谓词：

$brother(x,y)$：x 是 y 的兄弟

$sister(x,y)$：x 是 y 的姐妹

$woman(x)$：x 是女性

把已知规则与事实表示成谓词公式，得

规则 1：$\forall x \forall y(brother(x,y) \rightarrow \neg woman(x))$

规则 2：$\forall x \forall y(sister(x,y) \rightarrow woman(x))$

事实：$sister(Mary,Bill)$

把要求证的结论表示成谓词公式，得

求证：$\neg brother(Mary,Tom)$

化规则 1 为子句

$$\forall x \forall y(\neg brother(x,y) \vee \neg woman(x))$$

$$C_1 = \neg brother(x,y) \vee \neg woman(x)$$

化规则 2 为子句

$$\forall x \forall y(\neg sister(x,y) \vee woman(x))$$

$$C_2 = \neg sister(u,v) \vee woman(u)$$

事实原来就是子句形式

$$C_3 = sister(Mary,Bill)$$

C_2 与 C_3 归结为

$$C_{23} = woman(Mary)$$

C_{23} 与 C_1 归结为

$$C_{123} = \neg brother(Mary,y)$$

设 $C_4 = brother(Mary,Tom)$，则

$$C_{1234} = NIL$$

所以，得证。

3.6　应用归结原理求解问题

归结原理除了可用于定理证明外，还可用来求取问题的答案，其思想与定理证明类似。下面给出应用归结原理求解问题的步骤：

① 把已知前提用谓词公式表示出来，并且化为相应的子句集，设该子句集的名字为 S。

② 把待求解的问题也用谓词公式表示出来，然后把它否定并与答案谓词 *ANSWER* 构成析取式，*ANSWER* 是一个为了求解问题而专设的谓词，其变元必须与问题公式的变元完全一致。

③ 把②中得到的析取式化为子句集，并把该子句集并入到子句集 S 中，得到子句集 S'。

④ 对 S' 应用归结原理进行归结。

⑤ 若得到归结式 *ANSWER*，则答案就在 *ANSWER* 中。

例 3.11　已知如下信息。

F_1：王（*Wang*）先生是小李（*Li*）的老师。

F_2:小李与小张($Zhang$)是同班同学。

F_3:如果 x 与 y 是同班同学,则 x 的老师也是 y 的老师。

求小张的老师是谁?

解 定义谓词。

$T(x,y)$:x 是 y 的老师。

$C(x,y)$:x 与 y 是同班同学。

把已知前提及待求解的问题表示成谓词公式,得

F_1:$T(Wang,Li)$

F_2:$C(Li,Zhang)$

F_3:$(\forall x)(\forall y)(\forall z)(C(x,y)\wedge T(z,x)\rightarrow T(z,y))$

把待求解的问题表示成谓词公式,并把它否定后与谓词 $ANSWER(x)$ 析取,得

G:$\neg(\exists x)T(x,Zhang)\vee ANSWER(x)$

把上述谓词公式化为子句集。

(1) $T(Wang,Li)$

(2) $C(Li,Zhang)$

(3) $\neg C(x,y)\vee\neg T(z,x)\vee T(z,y)$

(4) $\neg T(u,Zhang)\vee ANSWER(u)$

应用归结原理进行归结。

(5) $\neg C(Li,y)\vee T(Wang,y)$ (1)与(3)归结

(6) $\neg C(Li,Zhang)\vee ANSWER(Wang)$ (4)与(5)归结

(7) $ANSWER(Wang)$ (2)与(6)归结

由 $ANSWER(Wang)$ 得知小张的老师是王先生。

上述归结过程可用图 3.7 的归结树表示。

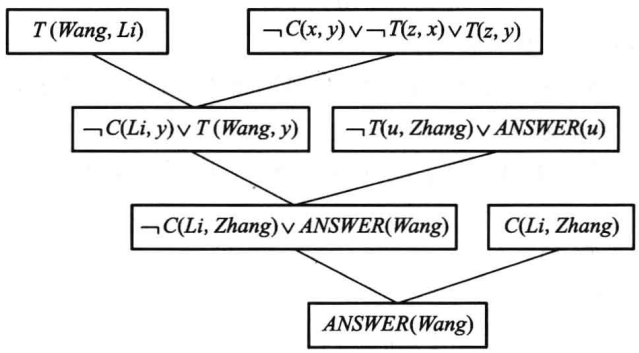


图 3.7 例 3.11 的归结树

例 3.12 设 A,B,C 三人中有人从不说真话,也有人从不说假话,某人向这三人分别提出同一个问题:谁是说谎者?A 答:"B 和 C 都是说谎者";B 答:"A 和 C 都是说谎者";C 答:"A 和 B 中至少有一个是说谎者"。求谁是老实人,谁是说谎者?

解 设用 $T(x)$ 表示 x 说真话。

如果 A 说真话，则有

$$T(A) \rightarrow \neg T(B) \wedge \neg T(C)$$

如果 A 说的是假话，则有

$$\neg T(A) \rightarrow T(B) \vee T(C)$$

对 B 和 C 说的话作相同的处理，可得

$$T(B) \rightarrow \neg T(A) \wedge \neg T(C)$$
$$\neg T(B) \rightarrow T(A) \vee T(C)$$
$$T(C) \rightarrow \neg T(A) \vee \neg T(B)$$
$$\neg T(C) \rightarrow T(A) \wedge T(B)$$

把上面这些公式化成子句集，得到 S。

(1) $\neg T(A) \vee \neg T(B)$

(2) $\neg T(A) \vee \neg T(C)$

(3) $T(A) \vee T(B) \vee T(C)$

(4) $\neg T(B) \vee \neg T(C)$

(5) $\neg T(C) \vee \neg T(A) \vee \neg T(B)$

(6) $T(C) \vee T(A)$

(7) $T(C) \vee T(B)$

下面首先求谁是老实人。把 $\neg T(x) \vee ANSWER(x)$ 并入 S 得到 S_1。即 S_1 比 S 多如下一个子句。

(8) $\neg T(x) \vee ANSWER(x)$

应用归结原理对 S_1 进行归结。

(9) $\neg T(A) \vee T(C)$　　(1)和(7)归结

(10) $T(C)$　　(6)和(9)归结

(11) $ANSWER(C)$　　(8)和(10)归结

所以 C 是老实人，即 C 从不说假话。

事实上，无论如何对 S_1 进行归结，都推不出 $ANSWER(B)$ 与 $ANSWER(A)$。

下面来证明 A 和 B 不是老实人。

设 A 不是老实人，则有 $\neg T(A)$，把它否定并入 S 中，得到子句集 S_2，即 S_2 比 S 多如下一个子句。

(8) $\neg(\neg T(A))$ 即 $T(A)$

应用归结原理对 S_2 进行归结。

(9) $\neg T(A) \vee T(C)$　　(1)和(7)归结

(10) $\neg T(A)$　　(2)和(9)归结

(11) NIL　　(8)和(10)归结

所以 A 不是老实人。

同样，可证明 B 也不是老实人。

由上面的例子可以看出，在归结过程中，一个子句可以多次被用来进行归结，也可以不被用来归结。在归结时并不一定要把子句集的全部子句都用到，只要在定理证明时能归结出空子句，

在求取问题答案时能归结出 *ANSWER* 就可以了。

对子句集进行归结时,关键的一步是从子句集中找出可以进行归结的一对子句。由于事先不知道哪两个子句可以进行归结,更不知道通过对哪些子句对的归结可以尽快地得到空子句,因而必须对子句集中的所有子句逐对地进行比较,对任何一对可归结的子句对都进行归结。这样不仅要耗费许多时间,而且还会因为归结出了许多无用的归结式而多占用了许多存储空间,造成了时空的浪费,降低了效率。为解决这些问题,人们研究出了多种归结策略。这些归结策略大致可分为两大类:一类是删除策略,另一类是限制策略。前一类通过删除某些无用的子句来缩小归结的范围,后一类通过对参加归结的子句进行种种限制,尽可能地减少归结的盲目性,使其尽快地归结出空子句。关于归结策略可以参看有关书籍。

3.7 小结

1. 推理的概念

从初始证据出发,按某种策略不断运用知识库中的已知知识,逐步推出结论的过程称为推理。

演绎推理是从一般性知识推出适合于某一具体情况的结论。这是一种从一般到个别的推理过程。归纳推理是从足够多的事例中归纳出一般性结论的推理,是一种从个别到一般的推理过程。默认推理是在知识不完全的情况下假设某些条件已经具备所进行的推理。

所谓确定性推理是指推理时所用的知识与证据都是确定的,推出的结论也是确定的。所谓不确定性推理是指推理时所用的知识与证据不都是确定的,推出的结论也是不确定的。

单调推理是在推理过程中随着推理向前推进及新知识的加入,推出的结论越来越接近最终目标。非单调推理是在推理过程中由于新知识的加入,不仅没有加强已推出的结论,反而要否定它,使推理退回到前面的某一步,然后重新开始。

若按推理中是否运用与推理有关的启发性知识来划分,推理可分为启发式推理与非启发式推理。

正向推理是以已知事实作为出发点的一种推理。逆向推理是以某个假设目标作为出发点的一种推理。既有正向又有逆向的推理称为混合推理。

2. 推理的方法

从一组已知为真的事实出发,直接运用经典逻辑的推理规则推出结论的过程称为自然演绎推理。

原子谓词公式及其否定,称为文字。任何文字的析取式称为子句。

把谓词公式可以化成子句集。谓词公式不可满足的充要条件是其子句集不可满足。

鲁宾逊归结原理是机器定理证明的基础,是一种证明子句集不可满足性,从而实现定理证明的一种理论及方法。它的基本方法是:将要证明的定理表示为谓词公式,并化为子句集,然后进行归结,一旦归结出空子句,则定理得证。

应用归结原理求解问题的方法:把已知前提用谓词公式表示出来,并且化为相应的子句集,把待求解的问题也用谓词公式表示出来,然后把它否定并与谓词 *ANSWER* 构成析取式,化为子句集;对子句集进行归结,若得到归结式 *ANSWER*,则答案就在 *ANSWER* 中。

思考题

3.1　什么是推理、正向推理、逆向推理、混合推理？试列出常用的几种推理方式并列出每种推理方式的特点。

3.2　什么是冲突？在产生式系统中解决冲突的策略有哪些？

3.3　什么是子句？什么是子句集？请写出求谓词公式子句集的步骤。

3.4　谓词公式与它的子句集等价吗？在什么情况下它们才会等价？

3.5　引入鲁宾逊归结原理有何意义？什么是归结原理？什么是归结式？

3.6　请写出利用归结原理求解问题答案的步骤。

习题

3.1　设已知下述事实：A；B；$A \rightarrow C$；$B \wedge C \rightarrow D$；$D \rightarrow Q$。求证：Q 为真。

3.2　将下列谓词公式化为相应的子句集。

$$\neg \exists x \forall y \exists z \forall w P(x,y,z,w)$$

3.3　将下列逻辑表达式化为不含存在量词的前束形。

$$(\exists x)(\forall y)[(\forall z)P(x,z) \rightarrow R(x,y,f(a))]$$

3.4　把下列谓词公式分别化为相应的子句集。

(1) $(\forall z)(\forall y)(P(z,y) \wedge Q(z,y))$

(2) $(\forall x)(\forall y)(P(x,y) \rightarrow Q(x,y))$

(3) $(\forall x)(\exists y)(P(x,y) \vee Q(x,y) \rightarrow R(x,y))$

(4) $(\forall x)(\forall y)(P(x,y) \vee Q(x,y) \rightarrow R(x,y))$

(5) $(\forall x)(\forall y)(\exists z)(P(x,y) \rightarrow Q(x,y) \vee R(x,z))$

(6) $(\exists x)(\exists y)(\forall z)(\exists u)(\forall v)(\exists w)(P(x,y,z,u,v,w) \wedge (Q(x,y,z,u,v,w) \vee \neg R(x,z,w)))$

(7) $(\forall x)\{(\forall y)\ P(x,y) \rightarrow \neg (\forall y)[Q(x,y) \rightarrow R(x,y)]$

3.5　判断下列子句集中哪些是不可满足的。

(1) $S = \{\neg P \vee Q, \neg Q, P, \neg P\}$

(2) $S = \{P \vee Q, \neg P \vee Q, P \vee \neg Q, \neg P \vee \neg Q\}$

(3) $S = \{P(y) \vee Q(y), \neg P(f(x)) \vee R(a)\}$

(4) $S = \{\neg P(x) \vee Q(x), \neg P(y) \vee R(y), P(a), S(a), \neg S(z) \vee \neg R(z)\}$

(5) $S = \{\neg P(x) \vee \neg Q(y) \vee \neg L(x,y), P(a), \neg R(z) \vee L(a,z), R(b), Q(b)\}$

3.6　对下列各题分别证明 G 为 $F_1, F_2, \cdots, F_n$ 的逻辑结论。

(1) F_1：$(\exists x)(\exists y)P(x,y)$

G：$(\forall y)(\exists x)P(x,y)$

(2) F_1：$(\forall x)(P(x) \wedge (Q(a) \vee Q(b)))$

G：$(\exists x)(P(x) \wedge Q(x))$

(3) $F_1:(\exists x)(\exists y)(P(f(x))\wedge Q(f(y)))$

$G:P(f(a))\wedge P(y)\wedge Q(y)$

(4) $F_1:(\forall x)(P(x)\rightarrow(\forall y)(Q(y)\rightarrow\neg L(x,y)))$

$F_2:(\exists x)(P(x)\wedge(\forall y)(R(y)\rightarrow L(x,y)))$

$G:(\forall x)(R(x)\rightarrow\neg Q(x))$

(5) $F_1:(\forall x)(P(x)\rightarrow(Q(x)\wedge R(x)))$

$F_2:(\exists x)(P(x)\wedge S(x))$

$G:(\exists x)(S(x)\wedge R(x))$

(6) $F_1:(\forall z)(A(z)\wedge\neg B(z)\rightarrow(\exists y)(D(z,y)\wedge C(y)))$

$F_2:(\exists z)(E(z)\wedge A(z)\wedge(\forall y)(D(z,y)\rightarrow E(y)))$

$F_3:(\forall z)(E(z)\rightarrow\neg B(z))$

$G:(\exists z)(E(z)\wedge C(z))$

3.7 已知:

① 能够阅读的都是有文化的。

② 海豚是没有文化的。

③ 某些海豚是有智能的。

用归结原理证明:某些有智能的并不能阅读。

3.8 已知前提:每个储蓄钱的人都获得利息。

用归结原理证明:如果没有利息,那么就没有人去储蓄钱。

3.9 已知:每个使用 Internet 网络的人都想从网络获得信息。

用归结原理证明:如果没有信息就不会有人使用 Internet。

3.10 设有如下关系:

① 如果 x 是 y 的父亲,y 又是 z 的父亲,则 x 是 z 的祖父。

② 老李是大李的父亲。

③ 大李是小李的父亲。

用归结原理回答:上述人员中谁和谁是祖孙关系?

3.11 已知:

F_1:小张(*Zhang*)在哪里,则小李(*Li*)就去那里。

F_2:小张在学校里。

用归结原理回答:小李在哪里?

3.12 设 TONY、MIKE 和 JOHN 属于 ALPINE 俱乐部,ALPINE 俱乐部的成员不是滑雪运动员就是登山运动员。登山运动员不喜欢雨,而且任何不喜欢雪的人不是滑雪运动员。MIKE 讨厌 TONY 所喜欢的一切东西,而喜欢 TONY 所讨厌的一切东西。TONY 喜欢雨和雪。试用谓词公式的集合表示这段知识,用归结原理回答问题:"谁是 ALPINE 俱乐部的一个成员?他是一个登山运动员但不是一个滑雪运动员。"

第4章 不确定性推理方法

上一章讨论了建立在经典逻辑基础上的确定性推理。这是一种运用确定性知识，从确定的事实或证据进行精确推理得到确定性结论的推理方法。但现实世界中的事物以及事物之间的关系是极其复杂的。由于客观上存在的随机性、模糊性以及某些事物或现象暴露得不充分性，导致人们对它们的认识往往是不精确、不完全的，具有一定程度的不确定性。这种认识上的不确定性反映到知识以及由观察所得到的证据上来，就分别形成了不确定性的知识及不确定性的证据。人们通常是在信息不完善、不精确的情况下运用不确定性知识进行思维、求解问题的，推出的结论也是不确定的。因而还必须对不确定性知识的表示及推理进行研究。这就是本章将要讨论的不确定性推理。

下面首先讨论不确定性推理中的基本问题，然后着重介绍基于概率论的有关理论发展起来的不确定性推理方法，主要介绍可信度方法、证据理论，最后介绍目前在专家系统、信息处理、自动控制等领域广泛应用的依据模糊理论发展起来的模糊推理方法。

4.1 不确定性推理的概念

不确定性推理是从不确定性的初始证据出发，通过运用不确定性的知识，最终推出具有一定程度的不确定性但却是合理或者近乎合理的结论的思维过程。

在不确定性推理中，知识和证据都具有某种程度的不确定性，这就为推理机的设计与实现增加了复杂性和难度。它除了必须解决推理方向、推理方法、控制策略等基本问题外，一般还需要解决不确定性的表示与度量、不确定性匹配、不确定性的传递算法以及不确定性的合成等重要问题。

1. 不确定性的表示与度量

在不确定性推理中，“不确定性”一般分为两类：一是知识的不确定性，二是证据的不确定性。它们都要求有相应的表示方式和度量标准。

(1) 知识不确定性的表示

知识的表示与推理是密切相关的两个方面，不同的推理方法要求有相应的知识表示模式与之对应。在不确定性推理中，由于要进行不确定性的计算，因而必须用适当的方法把不确定性及不确定的程度表示出来。

在确立不确定性的表示方法时，有两个直接相关的因素需要考虑：一是要能根据领域问题的特征把其不确定性比较准确地描述出来，满足问题求解的需要；二是要便于推理过程中对不确定性的推算。只有把这两个因素结合起来统筹考虑，相应的表示方法才是实用的。

目前，在专家系统中知识的不确定性一般是由领域专家给出的，通常是一个数值，它表示相应知识的不确定性程度，称为知识的静态强度。

静态强度可以是相应知识在应用中成功的概率，也可以是该条知识的可信程度或其他，其值的大小范围因其意义与使用方法的不同而不同。今后在讨论各种不确定性推理模型时，将具体地给出静态强度的表示方法及其含义。

(2) 证据不确定性的表示

在推理中，有两种来源不同的证据：一种是用户在求解问题时提供的初始证据，例如，病人的症状、化验结果等；另一种是在推理中用前面推出的结论作为当前推理的证据。对于前一种情况，即用户提供的初始证据，由于这种证据多来源于观察，因而通常是不精确、不完全的，即具有不确定性。对于后一种情况，由于所使用的知识及证据都具有不确定性，因而推出的结论当然也具有不确定性，当把它用作后面推理的证据时，它亦是不确定性的证据。

一般来说，证据不确定性的表示方法应与知识不确定性的表示方法保持一致，以便于推理过程中对不确定性进行统一的处理。在有些系统中，为便于用户的使用，对初始证据的不确定性与知识的不确定性采取了不同的表示方法，但这只是形式上的，在系统内部亦作了相应的转换处理。

证据的不确定性通常也用一个数值表示。它代表相应证据的不确定性程度，称之为动态强度。对于初始证据，其值由用户给出；对于用前面推理所得结论作为当前推理的证据，其值由推理中不确定性的传递算法通过计算得到。

(3) 不确定性的度量

对于不同的知识及不同的证据，其不确定性的程度一般是不相同的，需要用不同的数据表示其不确定性的程度，同时还需要事先规定它的取值范围，只有这样每个数据才会有确定的意义。例如，在专家系统 MYCIN 中，用可信度表示知识及证据的不确定性，取值范围为[-1,1]。当可信度取大于零的数值时，其值越大表示相应的知识或证据越接近于“真”；当可信度的取值小于零时，其值越小表示相应的知识或证据越接近于“假”。

在确定一种度量方法及其范围时，应注意以下几点：

① 度量要能充分表达相应知识及证据不确定性的程度。

② 度量范围的指定应便于领域专家及用户对不确定性的估计。

③ 度量要便于对不确定性的传递进行计算，而且对结论算出的不确定性度量不能超出度量规定的范围。

④ 度量的确定应当是直观的，同时应有相应的理论依据。

2. 不确定性匹配算法及阈值

推理是一个不断运用知识的过程。在这一过程中，为了找到所需的知识，需要用知识的前提条件与数据库中已知的证据进行匹配，只有匹配成功的知识才有可能被应用。

对于不确定性推理，由于知识和证据都具有不确定性，而且知识所要求的不确定性程度与证据实际具有的不确定性程度不一定相同，因而就出现了“怎么才算匹配成功”的问题。对于这个

问题，目前常用的解决方法是，设计一个算法用来计算匹配双方相似的程度，另外再指定一个相似的“限度”，用来衡量匹配双方相似的程度是否落在指定的限度内。如果落在指定的限度内，就称它们是可匹配的，相应知识可被应用，否则就称它们是不可匹配的，相应知识不可应用。上述用来计算匹配双方相似程度的算法称为不确定性匹配算法，用来指出相似的“限度”称为阈值。

3. 组合证据不确定性的算法

在基于产生式规则的系统中，知识的前提条件既可以是简单条件，也可以是用 AND 或 OR 把多个简单条件连接起来构成的复合条件。进行匹配时，一个简单条件对应于一个单一的证据，一个复合条件对应于一组证据，称这一组证据为组合证据。在不确定性推理中，由于结论的不确定性通常是通过对证据及知识的不确定性进行某种运算得到的，因而需要有合适的算法计算组合证据的不确定性。目前，关于组合证据不确定性的计算已经提出了多种方法，如最大最小方法、Hamacher 方法、概率方法、有界方法、Einstein 方法等。每种方法都有相应的适应范围和使用条件，如概率方法只能在事件之间完全独立时使用。

4. 不确定性的传递算法

不确定性推理的根本目的是根据用户提供的初始证据，通过运用不确定性知识，最终推出不确定性的结论，并推算出结论的不确定性程度。因此，需要解决下面两个问题：

① 在每一步推理中，如何把证据及知识的不确定性传递给结论。

② 在多步推理中，如何把初始证据的不确定性传递给最终结论。

对于第一个问题，在不同的不确定性推理方法中所采用的处理方法各不相同，这将在下面的几节中分别进行讨论。

对于第二个问题，各种方法所采用的处理方法基本相同，即把当前推出的结论及其不确定性度量作为证据放入数据库中，供以后推理使用。由于最初那一步推理的结论是用初始证据推出的，其不确定性包含了初始证据的不确定性对它所产生的影响，因而当它又用作证据推出进一步的结论时，其结论的不确定性仍然会受到初始证据的影响。由此一步步地进行推理，必然就会把初始证据的不确定性传递给最终结论。

5. 结论不确定性的合成

推理中有时会出现这样一种情况：用不同知识进行推理得到了相同的结论，但不确定性的程度却不相同。此时，需要用合适的算法对它们进行合成。在不同的不确定性推理方法中所采用的合成方法各不相同。

以上简要地列出了不确定性推理中一般应该考虑的一些基本问题，但这并不是说任何一个不确定性推理都必须包括上述各项内容。

长期以来，概率论的有关理论和方法都被用作度量不确定性的重要手段，因为它不仅有完善的理论，而且还为不确定性的合成与传递提供了现成的公式，因而它被最早用于不确定性知识的表示与处理，像这样纯粹用概率模型来表示和处理不确定性的方法称为纯概率方法或概率方法。

纯概率方法虽然有严密的理论依据，但它通常要求给出事件的先验概率和条件概率，而这些数据又不易获得，因此其应用受到了限制。为了解决这个问题，人们在概率理论的基础上发展起来了一些新的方法及理论，主要有主观 Bayes 方法、可信度方法、证据理论等。

基于概率的方法虽然可以表示和处理现实世界中存在的某些不确定性，在人工智能的不确

定性推理方面占有重要地位,但它们都没有把事物自身所具有的模糊性反映出来,也不能对其客观存在的模糊性进行有效的处理。扎德等人提出的模糊集理论及其在此基础上发展起来的模糊逻辑弥补了这一缺憾,对由模糊性引起的不确定性的表示及处理开辟了一种新途径,得到了广泛应用。

下面详细讨论几种简单实用的不确定性推理方法。

4.2 可信度方法

可信度方法是肖特里菲(E. H. Shortliffe)等人在确定性理论(theory of confirmation)的基础上,结合概率论等提出的一种不确定性推理方法。它首先在专家系统 MYCIN 中得到了成功的应用。由于该方法比较直观、简单,而且效果也比较好,因而受到人们的重视。目前,许多专家系统都是基于这一方法建造起来的。

人们在长期的实践活动中,对客观世界的认识积累了大量的经验,当面临一个新事物或新情况时,往往可用这些经验对问题的真、假或为真的程度作出判断。这种根据经验对一个事物或现象为真的相信程度称为可信度(certainty)。

显然,可信度带有较大的主观性和经验性,其准确性难以把握。但由于人工智能所面向的多是结构不良的复杂问题,难以给出精确的数学模型,先验概率及条件概率的确定又比较困难,因而用可信度来表示知识及证据的不确定性仍不失为一种可行的方法。另外,由于领域专家有丰富的专业知识及实践经验,也不难对领域内的知识给出其可信度。

C-F 模型是基于可信度表示的不确定性推理的基本方法,其他可信度方法都是在此基础上发展起来的。

1. 知识不确定性的表示

在 C-F 模型中,知识是用产生式规则表示的,其一般形式为

IF E THEN H $(CF(H,E))$

其中,$CF(H,E)$是该条知识的可信度,称为可信度因子(certainty factor)。

$CF(H,E)$反映了前提条件与结论的联系强度。它指出当前提条件 E 所对应的证据为真时,它对结论 H 为真的支持程度,$CF(H,E)$的值越大,就越支持结论 H 为真。例如

IF 头痛 AND 流涕 THEN 感冒 (0.7)

表示当病人确实有“头痛”及“流涕”的症状时,则有七成的把握认为他是患了感冒。

$CF(H,E)$在$[-1,1]$上取值,$CF(H,E)$的值要求领域专家直接给出。其原则是:若由于相应证据的出现增加结论 H 为真的可信度,则取 $CF(H,E)>0$,证据的出现越是支持 H 为真,就使 $CF(H,E)$的值越大;反之,取 $CF(H,E)<0$,证据的出现越是支持 H 为假,就使 $CF(H,E)$的值越小;若证据的出现与否与 H 无关,则取 $CF(H,E)=0$。

2. 证据不确定性的表示

在 C-F 模型中,证据的不确定性也是用可信度因子表示的。例如,$CF(E)=0.6$ 表示 E 的可信度为 0.6。

证据可信度值的来源分两种情况:对于初始证据,其可信度的值由提供证据的用户给出;对于用先前推出的结论作为当前推理的证据,其可信度的值在推出该结论时通过不确定性传递算

法计算得到。

证据 E 的可信度 $CF(E)$ 也是在 $[-1,1]$ 上取值的。对于初始证据，若对它的所有观察 S 能肯定它为真，则取 $CF(E)=1$；若肯定它为假，则取 $CF(E)=-1$；若它以某种程度为真，则取 $CF(E)$ 为 $(0,1)$ 中的某一个值，即 $0<CF(E)<1$；若它以某种程度为假，则取 $CF(E)$ 为 $(-1,0)$ 中的某一个值，即 $-1<CF(E)<0$；若它还未获得任何相关的观察，此时可看作观察 S 与它无关，则取 $CF(E)=0$。

在该模型中，尽管知识的静态强度与证据的动态强度都是用可信度因子 CF 表示的，但它们所表示的意义不相同。静态强度 $CF(H,E)$ 表示的是知识的强度，即当 E 所对应的证据为真时对 H 的影响程度，而动态强度 $CF(E)$ 表示的是证据 E 当前的不确定性程度。

3. 组合证据不确定性的算法

当组合证据是多个单一证据的合取时，即

$$E = E_1 \quad \text{AND} \quad E_2 \quad \text{AND} \quad \cdots \quad \text{AND} \quad E_n$$

若已知 $CF(E_1),CF(E_2),\cdots,CF(E_n)$，则

$$CF(E) = \min\{CF(E_1),CF(E_2),\cdots,CF(E_n)\} \tag{4.1}$$

当组合证据是多个单一证据的析取时，即

$$E = E_1 \quad \text{OR} \quad E_2 \quad \text{OR} \quad \cdots \quad \text{OR} \quad E_n$$

若已知 $CF(E_1),CF(E_2),\cdots,CF(E_n)$，则

$$CF(E) = \max\{CF(E_1),CF(E_2),\cdots,CF(E_n)\} \tag{4.2}$$

4. 不确定性的传递算法

C－F 模型中的不确定性推理从不确定的初始证据出发，通过运用相关的不确定性知识，最终推出结论并求出结论的可信度值。其中，结论 H 的可信度由下式计算：

$$CF(H) = CF(H,E) \times \max\{0,CF(E)\} \tag{4.3}$$

由上式可以看出，当相应证据以某种程度为假，即 $CF(E)<0$ 时，则

$$CF(H) = 0$$

这说明在该模型中没有考虑证据为假时对结论 H 所产生的影响。另外，当证据为真，即 $CF(E)=1$ 时，由上式可推出：

$$CF(H) = CF(H,E)$$

这说明知识中的规则强度 $CF(H,E)$ 实际上就是在前提条件对应的证据为真时结论 H 的可信度。或者说，当知识的前提条件所对应的证据存在且为真时，结论 H 有 $CF(H,E)$ 大小的可信度。

5. 结论不确定性的合成算法

若由多条不同知识推出了相同的结论，但可信度不同，则可用合成算法求出综合可信度。

由于对多条知识的综合可通过两两的合成实现，所以下面只考虑两条知识的情况。

设有如下知识

$$\text{IF} \quad E_1 \quad \text{THEN} \quad H \quad (CF(H,E_1))$$

$$\text{IF} \quad E_2 \quad \text{THEN} \quad H \quad (CF(H,E_2))$$

则结论 H 的综合可信度可分为如下两步算出：

(1) 分别对每一条知识求出 $CF(H)$

$$CF_1(H) = CF(H,E_1) \times \max\{0,CF(E_1)\}$$

$$CF_2(H) = CF(H,E_2) \times \max\{0, CF(E_2)\}$$

(2) 用下述公式求出 E_1 与 E_2 对 H 的综合影响所形成的可信度 $CF_{1,2}(H)$

$$CF_{1,2}(H) = \begin{cases} CF_1(H) + CF_2(H) - CF_1(H)CF_2(H) & 若\ CF_1(H) \geqslant 0,\quad CF_2(H) \geqslant 0 \\ CF_1(H) + CF_2(H) + CF_1(H)CF_2(H) & 若\ CF_1(H) < 0,\quad CF_2(H) < 0 \\ \dfrac{CF_1(H) + CF_2(H)}{1 - \min\{|CF_1(H)|, |CF_2(H)|\}} & 若\ CF_1(H)CF_2(H) < 0 \end{cases} \tag{4.4}$$

例 4.1 设有如下一组知识：

r_1: IF E_1 THEN H (0.8)

r_2: IF E_2 THEN H (0.6)

r_3: IF E_3 THEN H (−0.5)

r_4: IF E_4 AND (E_5 OR E_6) THEN E_1 (0.7)

r_5: IF E_7 AND E_8 THEN E_3 (0.9)

已知：$CF(E_2)=0.8$，$CF(E_4)=0.5$，$CF(E_5)=0.6$，$CF(E_6)=0.7$，$CF(E_7)=0.6$，$CF(E_8)=0.9$。求 $CF(H)$。

解 第一步：对每一条规则求出 $CF(H)$。

由 r_4 得到

$$\begin{aligned} CF(E_1) &= 0.7 \times \max\{0, CF[E_4 \text{ AND } (E_5 \text{ OR } E_6)]\} \\ &= 0.7 \times \max\{0, \min\{CF(E_4), CF(E_5 \text{ OR } E_6)\}\} \\ &= 0.7 \times \max\{0, \min\{CF(E_4), \max\{CF(E_5), CF(E_6)\}\}\} \\ &= 0.7 \times \max\{0, \min\{0.5, \max\{0.6, 0.7\}\}\} \\ &= 0.7 \times \max\{0, 0.5\} \\ &= 0.35 \end{aligned}$$

由 r_5 得到

$$\begin{aligned} CF(E_3) &= 0.9 \times \max\{0, CF(E_7 \text{ AND } E_8)\} \\ &= 0.9 \times \max\{0, \min\{CF(E_7), CF(E_8)\}\} \\ &= 0.9 \times \max\{0, \min\{0.6, 0.9\}\} \\ &= 0.9 \times \max\{0, 0.6\} \\ &= 0.54 \end{aligned}$$

由 r_1 得到

$$\begin{aligned} CF_1(H) &= 0.8 \times \max\{0, CF(E_1)\} \\ &= 0.8 \times \max\{0, 0.35\} \\ &= 0.28 \end{aligned}$$

由 r_2 得到

$$\begin{aligned} CF_2(H) &= 0.6 \times \max\{0, CF(E_2)\} \\ &= 0.6 \times \max\{0, 0.8\} \end{aligned}$$

$$= 0.48$$

由 r_3 得到

$$\begin{aligned} CF_3(H) &= -0.5 \times \max\{0, CF(E_3)\} \\ &= -0.5 \times \max\{0, 0.54\} \\ &= -0.27 \end{aligned}$$

第二步:根据结论不确定性的合成算法得到

$$\begin{aligned} CF_{1,2}(H) &= CF_1(H) + CF_2(H) - CF_1(H) \times CF_2(H) \\ &= 0.28 + 0.48 - 0.28 \times 0.48 = 0.63 \end{aligned}$$

$$\begin{aligned} CF_{1,2,3}(H) &= \frac{CF_{1,2}(H) + CF_3(H)}{1 - \min\{|CF_{1,2}(H)|, |CF_3(H)|\}} \\ &= \frac{0.63 - 0.27}{1 - \min\{0.63, 0.27\}} = \frac{0.36}{0.73} = 0.49 \end{aligned}$$

即综合可信度为

$$CF(H) = 0.49$$

4.3　证据理论

证据理论(theory of evidence)由德普斯特(A. P. Dempster)于 20 世纪 60 年代首先提出,并由沙佛(G. Shafer)在 20 世纪 70 年代中期进一步发展起来的一种处理不确定性的理论,所以,又称为 D - S 理论。1981 年巴纳特(J. A. Barnett)把该理论引入专家系统中,同年卡威(J. Garvey)等人用它实现了不确定性推理。由于该理论能够区分“不确定”与“不知道”的差异,并能处理由“不知道”引起的不确定性,具有较大的灵活性,因而受到了人们的重视。目前,在证据理论的基础上已经发展了多种不确定性推理模型。本节简单介绍几种不确定性推理模型。

4.3.1　概率分配函数

证据理论是用集合表示命题的。

设 D 是变量 x 所有可能取值的集合,且 D 中的元素是互斥的,在任一时刻 x 都取且只能取 D 中的某一个元素为值,则称 D 为 x 的样本空间。

在证据理论中,D 的任何一个子集 A 都对应于一个关于 x 的命题,称该命题为“x 的值是在 A 中”。例如,用 x 代表打靶时所击中的环数,$D = \{1, 2, \cdots, 10\}$,则 $A = \{5\}$ 表示“x 的值是 5”或者“击中的环数为 5”;$A = \{5, 6, 7, 8\}$ 表示“击中的环数是 5,6,7,8 中的某一个”。又如,用 x 代表所看到的颜色,$D = \{红, 黄, 蓝\}$,则 $A = \{红\}$ 表示“x 是红色”;若 $A = \{红, 蓝\}$,则它表示“x 或者是红色,或者是蓝色”。

证据理论中,为了描述和处理不确定性,引入了概率分配函数、信任函数及似然函数等概念。下面首先介绍概率分配函数的概念。

设 D 为样本空间,领域内的命题都用 D 的子集表示,则基本概率分配函数(basic probability assignment function)定义如下:

定义 4.1 设函数 $M:2^D \to [0,1]$，即对任何一个属于 D 的子集 A，命它对应一个数 $M \in [0,1]$，且满足

$$M(\varnothing) = 0$$

$$\sum_{A \subseteq D} M(A) = 1$$

则称 M 是 2^D 上的基本概率分配函数，$M(A)$ 称为 A 的基本概率数。

关于概率分配函数的定义有以下几点说明：

① 设样本空间 D 中有 n 个元素，则 D 中子集的个数为 2^n 个，定义中的 2^D 就是表示这些子集的。例如，设

$$D = \{红,黄,蓝\}$$

则它的子集个数刚好是 $2^3 = 8$ 个，具体为

$$A_1 = \{红\},\quad A_2 = \{黄\},\quad A_3 = \{蓝\},\quad A_4 = \{红,黄\},$$

$$A_5 = \{红,蓝\},\quad A_6 = \{黄,蓝\},\quad A_7 = \{红,黄,蓝\},\quad A_8 = \varnothing$$

其中，$\varnothing$ 表示空集。

② 概率分配函数的作用是把 D 的任意一个子集 A 都映射为 $[0,1]$ 上的一个数 $M(A)$。概率分配函数实际上是对 D 的各个子集进行信任分配，$M(A)$ 表示分配给 A 的那一部分。例如，设

$$A = \{红\},\quad M(A) = 0.3$$

它表示对命题“x 是红色”的正确性的信任度是 0.3。

当 A 由多个元素组成时，$M(A)$ 不包括对 A 的子集的信任度，而且也不知道该对它如何进行分配。例如，在

$$M(\{红,黄\}) = 0.2$$

中不包括对 $A = \{红\}$ 的信任度 0.3，而且也不知道该把这个 0.2 分配给 $\{红\}$ 还是分配给 $\{黄\}$。

当 $A = D$ 时，$M(A)$ 是对 D 的各子集进行信任分配后剩下的部分，它表示不知道该对这部分如何进行分配。例如，当

$$M(D) = M(\{红,黄,蓝\}) = 0.1$$

时，它表示不知道该对这个 0.1 如何分配，但它不是属于 $\{红\}$，就一定是属于 $\{黄\}$ 或 $\{蓝\}$，只是由于存在某些未知信息，不知道应该如何分配。

③ 概率分配函数与概率不同。例如，设

$$D = \{红,黄,蓝\}$$

且设

$$M(\{红\}) = 0.3,\quad M(\{黄\}) = 0,\quad M(\{蓝\}) = 0.1,\quad M(\{红,黄\}) = 0.2,$$

$$M(\{红,蓝\}) = 0.2,\ M(\{黄,蓝\}) = 0.1,\ M(\{红,黄,蓝\}) = 0.1,\ M(\varnothing) = 0$$

显然，M 符合概率分配函数的定义，但是

$$M(\{红\}) + M(\{黄\}) + M(\{蓝\}) = 0.4$$

若按概率的要求，这三者的和应等于 1。

4.3.2　信任函数

定义 4.2　命题的信任函数(belief function) $Bel:2^D \to [0,1]$,且

$$Bel(A) = \sum_{B \subseteq A} M(B) \qquad \forall A \subseteq D \tag{4.5}$$

其中 2^D 表示 D 的所有子集。

Bel 函数又称为下限函数,*Bel*(*A*)表示对命题 *A* 为真的总的信任程度。

由信任函数及概率分配函数的定义容易推出

$$Bel(\varnothing) = M(\varnothing) = 0$$

$$Bel(D) = \sum_{B \subseteq D} M(B) = 1$$

根据上面给出的数据,可以求得

$$\begin{aligned}
Bel(\{红\}) &= M(\{红\}) = 0.3 \\
Bel(\{红,黄\}) &= M(\{红\}) + M(\{黄\}) + M(\{红,黄\}) \\
&= 0.3 + 0.2 = 0.5 \\
Bel(\{红,黄,蓝\}) &= M(\{红\}) + M(\{黄\}) + M(\{蓝\}) + M(\{红,黄\}) \\
&\quad + M(\{红,蓝\}) + M(\{黄,蓝\}) + M(\{红,黄,蓝\}) \\
&= 0.3 + 0 + 0.1 + 0.2 + 0.2 + 0.1 + 0.1 = 1
\end{aligned}$$

4.3.3　似然函数

似然函数(plausibility function)又称为不可驳斥函数或上限函数,下面给出它的定义。

定义 4.3　似然函数 $Pl:2^D \to [0,1]$,且

$$Pl(A) = 1 - Bel(\neg A) \qquad \forall A \subseteq D \tag{4.6}$$

现在介绍似然函数的含义。由于 *Bel*(*A*)表示对 *A* 为真的信任程度,所以 *Bel*(¬ *A*)就表示对 ¬ *A* 为真,即 *A* 为假的信任程度,由此可推出 *Pl*(*A*)表示对 *A* 为非假的信任程度。下面来看两个例子,其中用到的基本概率数仍为上面给出的数据。

$$\begin{aligned}
Pl(\{红\}) &= 1 - Bel(\neg \{红\}) \\
&= 1 - Bel(\{黄,蓝\}) \\
&= 1 - [M(\{黄\}) + M(\{蓝\}) + M(\{黄,蓝\})] \\
&= 1 - [0 + 0.1 + 0.1] \\
&= 0.8 \\
Pl(\{黄,蓝\}) &= 1 - Bel(\neg \{黄,蓝\}) \\
&= 1 - Bel(\{红\}) \\
&= 1 - 0.3 \\
&= 0.7
\end{aligned}$$

4.3.4　概率分配函数的正交和(证据的组合)

有时对同样的证据会得到两个不同的概率分配函数,例如,对样本空间

$$D = \{a, b\}$$

从不同的来源分别得到如下两个概率分配函数

$$M_1(\{a\}) = 0.3,\quad M_1(\{b\}) = 0.6,\quad M_1(\{a,b\}) = 0.1,\quad M_1(\varnothing) = 0$$

$$M_2(\{a\}) = 0.4,\quad M_2(\{b\}) = 0.4,\quad M_2(\{a,b\}) = 0.2,\quad M_2(\varnothing) = 0$$

此时需要对它们进行组合,德普斯特(A. P. Dempster)提出的组合方法就是对这两个概率分配函数进行正交和运算。

定义 4.4 设 M_1 和 M_2 是两个概率分配函数,则其正交和 $M = M_1 \oplus M_2$ 为

$$M(\varnothing) = 0$$

$$M(A) = K^{-1} \sum_{x \cap y = A} M_1(x) M_2(y) \tag{4.7a}$$

其中,K 由下式计算

$$K = 1 - \sum_{x \cap y = \varnothing} M_1(x) M_2(y) = \sum_{x \cap y \neq \varnothing} M_1(x) M_2(y) \tag{4.7b}$$

如果 $K \neq 0$,则正交和 M 也是一个概率分配函数;如果 $K = 0$,则不存在正交和 M,即没有可能存在概率函数,称 M_1 与 M_2 矛盾。

对于多个概率分配函数 $M_1, M_2, \cdots, M_n$,如果它们可以组合,也可通过正交和运算将它们组合为一个概率分配函数。

下面举例说明求正交和的方法。

例 4.2 设 $D = \{黑,白\}$,且设

$$M_1(\{黑\},\{白\},\{黑,白\},\varnothing) = (0.3, 0.5, 0.2, 0)$$

$$M_2(\{黑\},\{白\},\{黑,白\},\varnothing) = (0.6, 0.3, 0.1, 0)$$

求 M_1 与 M_2 的正交和。

解 由定义 4.4 得到

$$\begin{aligned} K &= 1 - \sum_{x \cap y = \varnothing} M_1(x) M_2(y) \\ &= 1 - [M_1(\{黑\}) M_2(\{白\}) + M_1(\{白\}) M_2(\{黑\})] \\ &= 1 - [0.3 \times 0.3 + 0.5 \times 0.6] = 0.61 \end{aligned}$$

$$\begin{aligned} M(\{黑\}) &= K^{-1} \sum_{x \cap y = \{黑\}} M_1(x) M_2(y) \\ &= \frac{1}{0.61}[M_1(\{黑\}) M_2(\{黑\}) + M_1(\{黑\}) M_2(\{黑,白\}) + M_1(\{黑,白\}) M_2(\{黑\})] \\ &= \frac{1}{0.61}[0.3 \times 0.6 + 0.3 \times 0.1 + 0.2 \times 0.6] = 0.54 \end{aligned}$$

同理可得

$$M(\{白\}) = 0.43$$

$$M(\{黑,白\}) = 0.03$$

所以,经对 M_1 与 M_2 进行组合后得到的概率分配函数为

$$M(\{黑\},\{白\},\{黑,白\},\varnothing) = (0.54, 0.43, 0.03, 0)$$

4.3.5 基于证据理论的不确定性推理

基于证据理论的不确定性推理,大体可分为以下步骤:

① 建立问题的样本空间 D。

② 由经验给出，或者由随机性规则和事实的信度度量计算求得幂集 2^D 的基本概率分配函数。

③ 计算所关心的子集 $A \in 2^D$ 的信任函数值 $Bel(A)$ 或者似然函数值 $Pl(A)$。

④ 由 $Bel(A)$ 或者 $Pl(A)$ 得出结论。

下面通过实例进行说明。

例 4.3　设有规则：

(1) 如果 流鼻涕 则 感冒但非过敏性鼻炎(0.9)
　　或 过敏性鼻炎但非感冒(0.1)

(2) 如果 眼发炎 则 感冒但非过敏性鼻炎(0.8)
　　或 过敏性鼻炎但非感冒(0.05)

又有事实：

(1) 小王流鼻涕(0.9)

(2) 小王眼发炎(0.4)

括号中的数字表示规则和事实的可信度。

用证据理论推理小王患的什么病？

解　首先，取样本空间

$$D = \{h_1, h_2, h_3\}$$

其中，h_1 表示“感冒但非过敏性鼻炎”，h_2 表示“过敏性鼻炎但非感冒”，h_3 表示“同时得了两种病”。

由式(4.3)可以计算该问题的样本空间的基本概率分配函数。根据第一条规则和第一个事实的可信度，得到基本概率分配函数为

$$M_1(\{h_1\}) = 0.9 \times 0.9 = 0.81$$

$$M_1(\{h_2\}) = 0.9 \times 0.1 = 0.09$$

$$M_1(\{h_1, h_2, h_3\}) = 1 - M_1(\{h_1\}) - M_1(\{h_2\}) = 1 - 0.81 - 0.09 = 0.1$$

根据第二条规则和第二个事实的可信度，得到基本概率分配函数为

$$M_2(\{h_1\}) = 0.4 \times 0.8 = 0.32$$

$$M_2(\{h_2\}) = 0.4 \times 0.05 = 0.02$$

$$M_2(\{h_1, h_2, h_3\}) = 1 - M_2(\{h_1\}) - M_2(\{h_2\}) = 1 - 0.32 - 0.02 = 0.66$$

用证据理论将上述两个由不同规则得到的概率分配函数组合，得

$$\begin{aligned} K &= 1 - [M_1(\{h_1\})M_2(\{h_2\}) + M_1(\{h_2\})M_2(\{h_1\})] \\ &= 1 - [0.81 \times 0.02 + 0.09 \times 0.32] \\ &= 1 - 0.045 = 0.955 \end{aligned}$$

$$\begin{aligned} M(\{h_1\}) &= K^{-1}[M_1(\{h_1\})M_2(\{h_1\}) + M_1(\{h_1\})M_2(\{h_1, h_2, h_3\}) + \\ &\quad M_1(\{h_1, h_2, h_3\})M_2(\{h_1\})] \\ &= \frac{1}{0.955} \times 0.8258 = 0.87 \end{aligned}$$

$$
\begin{aligned}
M(\{h_2\}) &= K^{-1}[M_1(\{h_2\})M_2(\{h_2\}) + M_1(\{h_2\})M_2(\{h_1,h_2,h_3\} \\
&\quad + M_1(\{h_1,h_2,h_3\})M_2(\{h_2\})] \\
&= \frac{1}{0.955} \times 0.0632 = 0.066
\end{aligned}
$$

$$
M(\{h_1,h_2,h_3\}) = 1 - M(\{h_1\}) - M(\{h_2\}) = 1 - 0.87 - 0.066 = 0.064
$$

由信任函数的定义得

$$
Bel(\{h_1\}) = M(\{h_1\}) = 0.87
$$

$$
Bel(\{h_2\}) = M(\{h_2\}) = 0.066
$$

由似然函数的定义得

$$
\begin{aligned}
Pl(\{h_1\}) &= 1 - Bel(\neg\{h_1\}) = 1 - Bel(\{h_2,h_3\}) \\
&= 1 - [M(\{h_2\} + M(\{h_3\}) \\
&= 1 - [0.066 + 0] = 0.934 \\
Pl(\{h_2\}) &= 1 - Bel(\neg\{h_2\}) = 1 - Bel(\{h_1,h_3\}) \\
&= 1 - [M(\{h_1\} + M(\{h_3\}) \\
&= 1 - [0.87 + 0] = 0.13
\end{aligned}
$$

综合上述结果得，

“感冒但非过敏性鼻炎”为真的信任度为 0.87，非假的信任度为 0.934；

“过敏性鼻炎但非感冒”为真的信任度为 0.066，非假的信任度为 0.13。

因此，患者是感冒了而非过敏性鼻炎。

4.4 模糊推理方法

4.4.1 模糊逻辑的提出与发展

“模糊”是人类感知万物，获取知识，思维推理，决策实施的重要特征。“模糊”比“清晰”所拥有的信息容量更大，内涵更丰富，更符合客观世界。为了用数学方法描述和处理自然界出现的不精确、不完整的信息，如人类语言信息和图像信息，1965 年美国著名学者加利福尼亚大学教授扎德(L. A. Zadeh)发表了关于“fuzzy set”的论文，首先提出了模糊理论。

在刚提出模糊理论的年代，由于科学技术尤其是计算机技术发展的限制，以及科技界对“模糊”含义的误解，使得模糊理论没有得到应有的发展。在美国、欧洲、中国和日本，只有少数科学家研究模糊理论。虽然理论文章发表了约 5000 篇，但实际应用却寥寥无几。

模糊理论成功的应用首先是在自动控制领域。1974 年，英国伦敦大学教授 Mamdani 首次将模糊理论应用于热电厂的蒸汽机控制，揭开了模糊理论在控制领域应用的新篇章，充分展示了模糊控制技术的应用前景。1976 年，Mamdani 又将模糊理论应用于水泥旋转炉的控制。在所有应用中，模糊控制在欧洲主要用于工业自动化，在美国主要用于军事领域。尽管在此之后的 10 多年内，模糊控制技术应用取得了很好的效果，然而，没有取得根本上的突破。

到 20 世纪 80 年代，随着计算机技术的发展，日本科学家将模糊理论成功地运用于工业控制

和消费品控制，在世界范围内掀起了模糊控制应用高潮。1983 年日本 Fuji Electric 公司实现了饮水处理装置的模糊控制。1987 年日本 Hitachi 公司研制出地铁的模糊控制系统。1987—1990 年在日本申报的模糊产品专利就达 319 种，分布在过程控制、汽车电子、图像识别/图像数据处理、测量技术/传感器、机器人、诊断、家用电器控制等领域。

目前，各种模糊产品充满日本、西欧和美国市场，如模糊洗衣机、模糊吸尘器、模糊电冰箱和模糊摄像机等。各国都将模糊技术作为本国重点发展的关键技术。

在日常生活中，人们往往用“较少”、“较多”、“小一些”、“很小”等模糊语言来进行控制。例如，当我们拧开水阀向水桶放水时，有这样的经验：桶里没有水或水较少时，水阀应开大一些；桶里的水比较多时，水阀应开小一些；水桶快满时应把阀门开得很小；水桶里的水满时，应迅速关掉水阀。这里的水阀开“大一些”、“小一些”，水桶水“比较多”、“快满”、“比较少”等都是模糊概念。

模糊控制是以模糊数学为基础，运用语言规则知识表示方法和先进的计算机技术，由模糊推理进行决策的一种高级控制策略。它无疑属于智能控制范畴，而且发展至今已成为人工智能领域中的一个重要分支。

常规控制一般都要求系统有精确的数学模型。但大多数的工业过程的参数呈现极强的时变非线性特性，一般很难建立数学模型，所以对于不确定性系统，采用常规控制很难实现有效控制，而模糊控制可以利用语言信息却不需要精确的数学模型，从而可以实现对不确定性系统较好的控制。模糊控制技术是由模糊数学、计算机科学、人工智能、知识工程等多门学科相互渗透，且理论性很强的科学技术。

在人工智能领域里，特别是在知识表示方面，模糊逻辑有相当广阔的应用前景。目前在自动控制、模式识别、自然语言理解、机器人及专家系统研制等方面，应用模糊逻辑取得了一定的成果，引起了本领域的越来越多专家学者的关注。

4.4.2　模糊集合

1. 模糊集合的定义

模糊集合(fuzzy sets)是经典集合的扩充。下面首先介绍集合论中的几个名词。

论域：所讨论的全体对象称为论域。一般用 U、E 等大写字母表示论域。

元素：论域中的每个对象。一般用 a,b,c,x,y,z 等小写字母表示集合中的元素。

集合：论域中具有某种相同属性的确定的、可以彼此区别的元素的全体，常用 A,B,C,X,Y,Z 等表示集合。如 $A=\{x \mid f(x)>0\}$。

在经典集合中，元素 a 和集合 A 的关系只有两种：a 属于 A 或 a 不属于 A，即只有两个真值“真”和“假”。

例如，若定义 18 岁以上的人为“成年人”集合，则一位超过 18 岁的人属于“成年人”集合，而另外一位不足 18 岁的人，哪怕只差一天也不属于该集合。

经典集合可用特征函数表示。例如，“成年人”集合可以表示为

$$\mu_{成年人}(x)=\begin{cases}1 & x \geqslant 18\\ 0 & x<18\end{cases}$$

如图 4.1 所示。这是一种对事物的二值描述即二值逻辑。

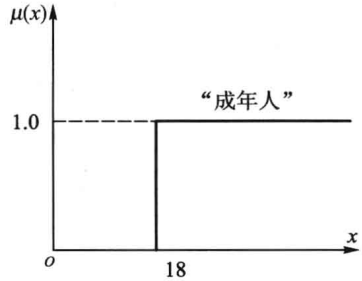


图 4.1　“成年人”特征函数

经典集合只能描述确定性的概念，而不能描述现实世界中模糊的概念。例如，“天气很热”等概念。模糊逻辑模仿人类的智慧，引入隶属度（degree of membership）的概念，描述介于“真”与“假”中间的过程。

模糊集合中每一个元素被赋予一个介于 0 和 1 之间的实数，描述其元素属于这个模糊集合的强度，该实数称为元素属于这个模糊集合的隶属函数。如上述例子中，一个人变成“成年人”的过程可用一连续曲线表示，如图 4.2 所示。

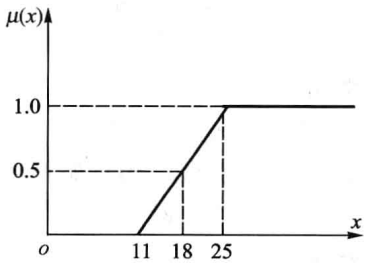


图 4.2 “成年人”隶属函数

2. 模糊集合的表示方法

与经典集合表示不同的是，模糊集合中不仅要列出属于这个集合的元素，而且要注明这个元素属于这个集合的隶属度。

当论域中元素数目有限时，模糊集合 A 的数学描述为

$$A = \{(x, \mu_A(x)), x \in X\} \tag{4.8}$$

其中，$\mu_A(x)$ 为元素 x 属于模糊集 A 的隶属度，X 是元素 x 的论域。

（1）Zadeh 表示法

① 当论域是离散且元素数目有限时，常采用模糊集合的 Zadeh 表示法

$$A = \mu_A(x_1)/x_1 + \mu_A(x_2)/x_2 + \cdots + \mu_A(x_n)/x_n = \sum_{i=1}^{n} \mu_A(x_i)/x_i \tag{4.9}$$

其中，x_i 表示模糊集合所对应的论域中的元素，而 $\mu_A(x_i)$ 表示相应的隶属度，“/”只是一个分隔符号，并不表示分数的意思。符号“+”或者“∑”也不表示求和，而是表示模糊集合在论域上的整体。

上式也可以等价地表示为

$$A = \{\mu_A(x_1)/x_1, \mu_A(x_2)/x_2, \cdots, \mu_A(x_n)/x_n\} \tag{4.10}$$

② 当论域是连续的，或者其中元素数目无限时，Zadeh 将模糊集 A 表示为

$$A = \int_{x \in U} \mu_A(x)/x \tag{4.11}$$

这里的“$\int$”不是数学中的积分符号，只是表示论域中各元素与其隶属度对应关系的总括，是一个记号。

（2）序偶表示法

$$A = \{(\mu_A(x_1), x_1), (\mu_A(x_2), x_2), \cdots, (\mu_A(x_n), x_n)\} \tag{4.12}$$

（3）向量表示法

$$A = [\mu_A(x_1), \mu_A(x_2), \cdots, \mu_A(x_n)] \tag{4.13}$$

在向量表示法中，因为默认模糊集合中的元素依次是 $x_1, x_2, \cdots, x_n$，所以，隶属度为 0 的项不能省略。

3. 隶属函数

模糊集合中所有元素的隶属度全体构成模糊集合的隶属函数。

正确地确定隶属函数（membership function）是运用模糊集合理论解决实际问题的基础。隶属函数是对模糊概念的定量描述。我们遇到的模糊概念不胜枚举，然而准确地反映模糊集合的隶属函数，却无法找到统一的模式。

隶属函数的确定过程，本质上说应该是客观的，但每个人对于同一个模糊概念的认识理解又有差异。因此，隶属函数的确定又带有主观性。实际上，引进隶属度后，将人们对事物认识的模糊性转化为隶属度确定的主观性。隶属函数一般根据经验或统计进行确定，也可由专家给出。对于同一个模糊概念，不同的人会建立不完全相同的隶属函数，尽管形式不完全相同，只要能反映同一模糊概念，仍然能够较好地解决和处理实际模糊信息的问题。

例如，以年龄作论域，取 $U=[0,200]$，扎德给出了“年老”O 与“年青”Y 两个模糊集合的隶属函数为

$$\mu_O(u)=\begin{cases}0 & 0\leqslant u\leqslant 50\\ \left[1+\left(\dfrac{5}{u-50}\right)^2\right]^{-1} & 50<u\leqslant 200\end{cases}$$

$$\mu_Y(u)=\begin{cases}1 & 0\leqslant u\leqslant 25\\ \left[1+\left(\dfrac{u-25}{5}\right)^2\right]^{-1} & 25<u\leqslant 200\end{cases}$$

采用 Zadeh 表示法，“年老”O 与“年青”Y 两个模糊集合可以表示为

$$O=\int_{50<u\leqslant 200}\left[1+\left(\frac{u-50}{5}\right)^{-2}\right]^{-1}\Big/u$$

$$Y=\int_{0\leqslant u\leqslant 25}1/u+\int_{25<u\leqslant 200}\left[1+\left(\frac{u-25}{5}\right)^2\right]^{-1}\Big/u$$

常见模糊隶属函数有：正态分布、三角分布、梯形分布等。

4.4.3　模糊集合的运算

模糊集合是经典集合的推广，所以，经典集合的运算可以推广到模糊集合。但由于模糊集合要由它的隶属函数加以确定，所以，需要重新定义模糊集合的基本运算。

(1) 模糊集合的包含关系

若 $\mu_A(x)\geqslant\mu_B(x)$，则称 A 包含 B，记作 $A\supseteq B$。

(2) 模糊集合的相等关系

若 $\mu_A(x)=\mu_B(x)$，则称 A 与 B 相等，记作 $A=B$。

(3) 模糊集合的交并补运算

设 A、B 是论域 U 中的两个模糊集。

① 交运算(intersection)$A\cap B$：

$$\mu_{A\cap B}(x)=\min\{\mu_A(x),\mu_B(x)\}=\mu_A(x)\wedge\mu_B(x)\tag{4.14}$$

② 并运算(union)$A\cup B$：

$$\mu_{A\cup B}(x)=\max\{\mu_A(x),\mu_B(x)\}=\mu_A(x)\vee\mu_B(x)\tag{4.15}$$

③ 补运算(complement)$\bar{A}$ 或者 A^C：

$$\mu_{\bar{A}}(x)=1-\mu_A(x)\tag{4.16}$$

其中，$\wedge$ 表示取小运算；$\vee$ 表示取大运算。

例 4.4　设论域 $U=\{x_1,x_2,x_3,x_4\}$，A 及 B 是论域 U 上的两个模糊集合，已知

$$A=0.3/x_1+0.5/x_2+0.7/x_3+0.4/x_4$$

$$B = 0.5/x_1 + 1/x_2 + 0.8/x_3$$

求 $\bar{A}$、$\bar{B}$、$A \cap B$、$A \cup B$。

解

$$\bar{A} = 0.7/x_1 + 0.5/x_2 + 0.3/x_3 + 0.6/x_4$$

$$\bar{B} = 0.5/x_1 + 0.2/x_3 + 1/x_4$$

$$A \cap B = \frac{0.3 \wedge 0.5}{x_1} + \frac{0.5 \wedge 1}{x_2} + \frac{0.7 \wedge 0.8}{x_3} + \frac{0.4 \wedge 0}{x_4}$$

$$= 0.3/x_1 + 0.5/x_2 + 0.7/x_3$$

$$A \cup B = \frac{0.3 \vee 0.5}{x_1} + \frac{0.5 \vee 1}{x_2} + \frac{0.7 \vee 0.8}{x_3} + \frac{0.4 \vee 0}{x_4}$$

$$= 0.5/x_1 + 1/x_2 + 0.8/x_3 + 0.4/x_4$$

(4) 模糊集合的代数运算

设 A、B 是论域 U 中的两个模糊集。

① 代数积：

$$\mu_{A \cdot B}(x) = \mu_A(x)\mu_B(x)$$

② 代数和：

$$\mu_{A+B}(x) = \mu_A(x) + \mu_B(x) - \mu_{A \cdot B}(x)$$

③ 有界和：

$$\mu_{A \oplus B}(x) = \min\{1, \mu_A(x) + \mu_B(x)\} = 1 \wedge [\mu_A(x) + \mu_B(x)]$$

④ 有界积：

$$\mu_{A \otimes B}(x) = \max\{0, \mu_A(x) + \mu_B(x) - 1\} = 0 \vee [\mu_A(x) + \mu_B(x) - 1]$$

例 4.5 设论域 $U = \{x_1, x_2, x_3, x_4, x_5\}$，$A$ 及 B 是论域上的两个模糊集合，已知

$$A = 0.2/x_1 + 0.4/x_2 + 0.9/x_3 + 0.5/x_5$$

$$B = 0.1/x_1 + 0.7/x_3 + 1.0/x_4 + 0.3/x_5$$

求 $A \cdot B$、$A + B$、$A \oplus B$、$A \otimes B$。

解

$$A \cdot B = 0.02/x_1 + 0.63/x_3 + 0.15/x_5$$

$$A + B = 0.28/x_1 + 0.4/x_2 + 0.97/x_3 + 1.0/x_4 + 0.65/x_5$$

$$A \oplus B = 0.3/x_1 + 0.4/x_2 + 1.0/x_3 + 1.0/x_4 + 0.8/x_5$$

$$A \otimes B = 0.6/x_3$$

4.4.4 模糊关系与模糊关系的合成

1. 模糊关系

在模糊集合论中，模糊关系占有重要地位。模糊关系是普通关系的推广。普通关系是描述两个集合中的元素之间是否有关联。模糊关系描述两个模糊集合中的元素之间的关联程度。当论域为有限时，可以采用模糊矩阵表示模糊关系。

例 4.6 某地区人的身高论域 $X = \{140, 150, 160, 170, 180\}$（单位：cm），体重论域 $Y = \{40, 50, 60, 70, 80\}$（单位：kg）。身高与体重两个集合的元素之间没有确定的关系，只有一定程度的关联。它们的关联程度如表 4.1 所示。

表 4.1　身高与体重的模糊关系

X \ R \ Y	40	50	60	70	80
140	1	0.8	0.2	0.1	0
150	0.8	1	0.8	0.2	0.1
160	0.2	0.8	1	0.8	0.2
170	0.1	0.2	0.8	1	0.8
180	0	0.1	0.2	0.8	1

它是从 X 到 Y 的一个模糊关系 $\boldsymbol{R}$，可用模糊矩阵表示为

$$\boldsymbol{R} = \begin{bmatrix} 1 & 0.8 & 0.2 & 0.1 & 0 \\ 0.8 & 1 & 0.8 & 0.2 & 0.1 \\ 0.2 & 0.8 & 1 & 0.8 & 0.2 \\ 0.1 & 0.2 & 0.8 & 1 & 0.8 \\ 0 & 0.1 & 0.2 & 0.8 & 1 \end{bmatrix}$$

下面给出模糊关系的定义。

设 A、B 为两个模糊集合，在模糊数学中，模糊关系可用叉积(cartesian product)表示。在模糊逻辑中，这种叉积常用最小算子运算，即

$$\boldsymbol{\mu}_{A\times B}(\boldsymbol{a},\boldsymbol{b}) = \min\{\mu_A(\boldsymbol{a}),\mu_B(\boldsymbol{b})\} \tag{4.17}$$

若 A、B 为离散模糊集，其隶属函数分别为

$$\boldsymbol{\mu}_A = [\mu_A(a_1),\mu_A(a_2),\cdots,\mu_A(a_n)]$$

$$\boldsymbol{\mu}_B = [\mu_B(b_1),\mu_B(b_2),\cdots,\mu_B(b_n)]$$

则其叉积运算为

$$\boldsymbol{\mu}_{A\times B}(\boldsymbol{a},\boldsymbol{b}) = \boldsymbol{\mu}_A^{\mathrm{T}} \circ \boldsymbol{\mu}_B \tag{4.18}$$

其中“$\circ$”为模糊向量乘积。

上述定义的模糊关系，又称为二元模糊关系。通常所谓的模糊关系 $\boldsymbol{R}$，一般是指二元模糊关系。下面举例说明模糊关系的具体求取方法。

例 4.7　已知输入的模糊集合 A 和输出的模糊集合 B 分别为

$$A = 1.0/a_1 + 0.8/a_2 + 0.5/a_3 + 0.2/a_4 + 0.0/a_5$$

$$B = 0.7/b_1 + 1.0/b_2 + 0.6/b_3 + 0.0/b_4$$

求 A 到 B 的模糊关系 $\boldsymbol{R}$。

解

$$\boldsymbol{R} = A \times B = \boldsymbol{\mu}_A^{\mathrm{T}} \circ \boldsymbol{\mu}_B = \begin{bmatrix} 1.0 \\ 0.8 \\ 0.5 \\ 0.2 \\ 0.0 \end{bmatrix} \circ [0.7 \quad 1.0 \quad 0.6 \quad 0.0]$$

$$
= \begin{bmatrix} 1.0 \wedge 0.7 & 1.0 \wedge 1.0 & 1.0 \wedge 0.6 & 1.0 \wedge 0.0 \\ 0.8 \wedge 0.7 & 0.8 \wedge 1.0 & 0.8 \wedge 0.6 & 0.8 \wedge 0.0 \\ 0.5 \wedge 0.7 & 0.5 \wedge 1.0 & 0.5 \wedge 0.6 & 0.5 \wedge 0.0 \\ 0.2 \wedge 0.7 & 0.2 \wedge 1.0 & 0.2 \wedge 0.6 & 0.2 \wedge 0.0 \\ 0.0 \wedge 0.7 & 0.0 \wedge 1.0 & 0.0 \wedge 0.6 & 0.0 \wedge 0.0 \end{bmatrix}
$$

$$
= \begin{bmatrix} 0.7 & 1.0 & 0.6 & 0.0 \\ 0.7 & 0.8 & 0.6 & 0.0 \\ 0.5 & 0.5 & 0.5 & 0.0 \\ 0.2 & 0.2 & 0.2 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix}
$$

可以看出，两个模糊向量的叉积，类似于两个向量的乘积，只是其中的乘积运算用取小运算代替。

2. 模糊关系的合成

设模糊关系 $\boldsymbol{Q} \in X \times Y$，$\boldsymbol{R} \in Y \times Z$，则模糊关系 $\boldsymbol{S} \in X \times Z$ 称为模糊关系 $\boldsymbol{Q}$ 与 $\boldsymbol{R}$ 的合成。模糊关系 $\boldsymbol{Q}$ 与模糊关系 $\boldsymbol{R}$ 的合成 S 是模糊矩阵的叉乘 $\boldsymbol{S} = \boldsymbol{Q} \circ \boldsymbol{R}$。

模糊矩阵的合成可以由多种计算方法得到。例如，下列常用的几种计算方法：

① 最大－最小合成法：写出矩阵乘积 $\boldsymbol{QR}$ 中的每个元素，然后将其中的乘积运算用取小运算代替，求和运算用取大运算代替。

② 最大－代数积合成法：写出矩阵乘积 $\boldsymbol{QR}$ 中的每个元素，然后将其中的求和运算用取大运算代替，而乘积运算不变。

例 4.8 设模糊集合 X、Y、Z 分别为 $X = \{x_1, x_2, x_3, x_4\}$，$Y = \{y_1, y_2, y_3\}$，$Z = \{z_1, z_2\}$

$$
\boldsymbol{Q} = \begin{bmatrix} 0.5 & 0.6 & 0.3 \\ 0.7 & 0.4 & 1 \\ 0 & 0.8 & 0 \\ 1 & 0.2 & 0.9 \end{bmatrix} \qquad \boldsymbol{R} = \begin{bmatrix} 0.2 & 1 \\ 0.8 & 0.4 \\ 0.5 & 0.3 \end{bmatrix}
$$

求模糊关系 $\boldsymbol{Q}$ 与模糊关系 $\boldsymbol{R}$ 的合成 $\boldsymbol{S}$。

解 (1)最大－最小合成法

$$
\boldsymbol{S} = \boldsymbol{Q} \circ \boldsymbol{R} = \begin{bmatrix} 0.5 & 0.6 & 0.3 \\ 0.7 & 0.4 & 1 \\ 0 & 0.8 & 0 \\ 1 & 0.2 & 0.9 \end{bmatrix} \circ \begin{bmatrix} 0.2 & 1 \\ 0.8 & 0.4 \\ 0.5 & 0.3 \end{bmatrix}
$$

$$
= \begin{bmatrix} (0.5 \wedge 0.2) \vee (0.6 \wedge 0.8) \vee (0.3 \wedge 0.5) & (0.5 \wedge 1) \vee (0.6 \wedge 0.4) \vee (0.3 \wedge 0.3) \\ (0.7 \wedge 0.2) \vee (0.4 \wedge 0.8) \vee (1 \wedge 0.5) & (0.7 \wedge 1) \vee (0.4 \wedge 0.4) \vee (1 \wedge 0.3) \\ (0 \wedge 0.2) \vee (0.8 \wedge 0.8) \vee (0 \wedge 0.5) & (0 \wedge 1) \vee (0.8 \wedge 0.4) \vee (0 \wedge 0.3) \\ (1 \wedge 0.2) \vee (0.2 \wedge 0.8) \vee (0.9 \wedge 0.5) & (1 \wedge 1) \vee (0.2 \wedge 0.4) \vee (0.9 \wedge 0.3) \end{bmatrix}
$$

$$=\begin{bmatrix}0.6 & 0.5\\0.5 & 0.7\\0.8 & 0.4\\0.5 & 1\end{bmatrix}$$

（2）最大－代数积合成法

$$\boldsymbol{S}=\boldsymbol{Q}\circ\boldsymbol{R}=\begin{bmatrix}0.5 & 0.6 & 0.3\\0.7 & 0.4 & 1\\0 & 0.8 & 0\\1 & 0.2 & 0.9\end{bmatrix}\circ\begin{bmatrix}0.2 & 1\\0.8 & 0.4\\0.5 & 0.3\end{bmatrix}$$

$$=\begin{bmatrix}(0.5\times0.2)\vee(0.6\times0.8)\vee(0.3\times0.5) & (0.5\times1)\vee(0.6\times0.4)\vee(0.3\times0.3)\\(0.7\times0.2)\vee(0.4\times0.8)\vee(1\times0.5) & (0.7\times1)\vee(0.4\times0.4)\vee(1\times0.3)\\(0\times0.2)\vee(0.8\times0.8)\vee(0\times0.5) & (0\times1)\vee(0.8\times0.4)\vee(0\times0.3)\\(1\times0.2)\vee(0.2\times0.8)\vee(0.9\times0.5) & (1\times1)\vee(0.2\times0.4)\vee(0.9\times0.3)\end{bmatrix}$$

$$=\begin{bmatrix}0.48 & 0.5\\0.5 & 0.7\\0.64 & 0.32\\0.45 & 1\end{bmatrix}$$

4.4.5　模糊推理

1. 模糊知识表示

对于模糊不确定性，一般采用隶属度来刻画。隶属度是一个命题中所描述的事物的属性、状态和关系等的强度。例如，我们用三元组（张三，体型，（胖，0.9））表示命题“张三比较胖”，其中的 0.9 就代替“比较”而刻画了张三“胖”的程度。

这种隶属度表示法，一般是一种针对对象的表示法。模糊知识表示一般形式为

（<对象>，<属性>，（<属性值>，<隶属度>））

可以看出，它实际是通常三元组（<对象>，<属性>，<属性值>）的细化，其中的<隶属度>一项是对前面属性值的精确刻画。

事实上，这种思想和方法还可广泛用于产生式规则、谓词逻辑、框架、语义网络等多种知识表示方法中，从而扩充它们的表示范围和能力。下面举例说明。

人类思维判断的基本形式是

如果　（条件）　→　则　（结论）

其中的条件和结论常常是模糊的。

例如

如果　压力较高且温度在缓慢上升　　则　阀门略开

可以用模糊集合表示这个知识中压力、温度和阀门值的不确定性。

（锅炉，工况，（压力，0.80））∧（锅炉，工况，（温度，0.3））→（阀门，状态，（开，0.2））

例如，模糊规则

如果　患者有些头疼并且发高烧　　则　他患了重感冒

可表示为

(患者,症状,(头疼,0.95))∧(患者,症状,(发烧,1.1))→(患者,疾病,(感冒,1.2))

许多模糊规则实际上是一组多重条件语句,可以表示为从条件论域到结论论域的模糊关系矩阵 $\boldsymbol{R}$。通过条件模糊向量与模糊关系 $\boldsymbol{R}$ 的合成进行模糊推理,得到结论的模糊向量,然后采用"清晰化"方法将模糊结论转换为精确量。

根据模糊集合和模糊关系理论,对于不同类型的模糊规则可用不同的模糊推理方法。

2. 对 IF A THEN B 类型的模糊规则的推理

若已知输入为 A,则输出为 B;若现在已知输入为 $\boldsymbol{A}'$,则输出 $\boldsymbol{B}'$用合成规则求取

$$\boldsymbol{B}' = \boldsymbol{A}' \circ \boldsymbol{R} \tag{4.19}$$

其中 $\boldsymbol{R}$ 为 $\boldsymbol{A}$ 到 $\boldsymbol{B}$ 的模糊关系。

例 4.9 对于例 4.6 所示模糊系统,求当输入为

$$\boldsymbol{A}' = 0.4/a_1 + 0.7/a_2 + 1.0/a_3 + 0.6/a_4 + 0.0/a_5$$

时,系统的输出 $\boldsymbol{B}'$。

解 在例 4.6 中已经得到模糊关系,下面进行模糊合成得到模糊输出。

$$\boldsymbol{B}' = \boldsymbol{A}' \circ \boldsymbol{R} = \begin{bmatrix} 0.4 \\ 0.7 \\ 1.0 \\ 0.6 \\ 0.0 \end{bmatrix}^{\mathrm{T}} \circ \begin{bmatrix} 0.7 & 1.0 & 0.6 & 0.0 \\ 0.7 & 0.8 & 0.6 & 0.0 \\ 0.5 & 0.5 & 0.5 & 0.0 \\ 0.2 & 0.2 & 0.2 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix}$$

$$\begin{aligned}
&= [(0.4 \wedge 0.7) \vee (0.7 \wedge 0.7) \vee (1.0 \wedge 0.5) \vee (0.6 \wedge 0.2) \vee (0.0 \wedge 0.0), \\
&\quad (0.4 \wedge 1.0) \vee (0.7 \wedge 0.8) \vee (1.0 \wedge 0.5) \vee (0.6 \wedge 0.2) \vee (0.0 \wedge 0.0), \\
&\quad (0.4 \wedge 0.6) \vee (0.7 \wedge 0.6) \vee (1.0 \wedge 0.5) \vee (0.6 \wedge 0.2) \vee (0.0 \wedge 0.0), \\
&\quad (0.4 \wedge 0.0) \vee (0.7 \wedge 0.0) \vee (1.0 \wedge 0.0) \vee (0.6 \wedge 0.0) \vee (0.0 \wedge 0.0)] \\
&= [(0.4 \vee 0.7 \vee 0.5 \vee 0.2 \vee 0.0), (0.4 \vee 0.7 \vee 0.5 \vee 0.2 \vee 0.0), \\
&\quad (0.4 \vee 0.6 \vee 0.5 \vee 0.2 \vee 0.0), (0.0 \vee 0.0 \vee 0.0 \vee 0.0 \vee 0.0)] \\
&= (0.7, 0.7, 0.6, 0.0)
\end{aligned}$$

则

$$\boldsymbol{B}' = 0.7/b_1 + 0.7/b_2 + 0.6/b_3 + 0.0/b_4$$

系统的控制规则库一般是由若干条规则组成的,对于每一条推理规则都可以得到一个相应的模糊关系,N 条规则就有 N 个模糊关系:$\boldsymbol{R}_1, \boldsymbol{R}_2, \cdots, \boldsymbol{R}_n$,对于整个系统的全部控制规则所对应的模糊关系 R,可通过对 N 个模糊关系 $\boldsymbol{R}_i (i = 1, 2, \cdots, n)$ 取"并"操作得到,即

$$R = \boldsymbol{R}_1 \cup \boldsymbol{R}_2 \cup \cdots \cup \boldsymbol{R}_n = \bigcup_{i=1}^{n} \boldsymbol{R}_i \tag{4.20}$$

4.4.6 模糊决策

由上述模糊推理得到的结论或者操作是一个模糊向量,不能直接应用,需要先转化为确定值。将模糊推理得到的模糊向量,转化为确定值的过程称为"模糊决策",或者"模糊判决"、"清晰化"、"反模糊化"(defuzzification)等。下面介绍几种简单、实用的模糊决策方法。

1. 最大隶属度法

最大隶属度法是在模糊向量中，取隶属度最大的量作为推理结果。

例如，当得到模糊向量为

$$\boldsymbol{U}' = 0.1/2 + 0.4/3 + 0.7/4 + 1.0/5 + 0.7/6 + 0.3/7$$

由于推理结果隶属于等级 5 的隶属度为最大，所以取结论为

$$U = 5$$

如果有两个以上的元素均为最大（一般依次相邻），则可以取它们的平均值。例如

$$\boldsymbol{U}' = 0.5/-3 + 0.5/-2 + 0.5/-1 + 0.0/0 + 0.0/1 + 0.0/2 + 0.0/3$$

则

$$U = \frac{-3-2-1}{3} = -2$$

这种方法的优点是简单易行，缺点是完全排除了其他隶属度较小的量的影响和作用，没有充分利用推理过程取得的信息。

2. 加权平均判决法

为了克服最大隶属度法的缺点，可以采用加权平均判决法，即

$$U = \frac{\sum_{i=1}^{n} \mu(u_i) u_i}{\sum_{i=1}^{n} \mu(u_i)} \tag{4.21}$$

例如

$$\boldsymbol{U}' = 0.1/2 + 0.6/3 + 0.5/4 + 0.4/5 + 0.2/6$$

则

$$U = \frac{2 \times 0.1 + 3 \times 0.6 + 4 \times 0.5 + 5 \times 0.4 + 6 \times 0.2}{0.1 + 0.6 + 0.5 + 0.4 + 0.2} = 4$$

3. 中位数法

论域上把隶属函数曲线与横坐标围成的面积平分为两部分的元素称为模糊集的中位数。

中位数法就是把模糊集的中位数作为系统控制量。

当论域为有限离散点时，中位数 u^* 可以用下列公式求取

$$\sum_{u_1}^{u^*} \mu(u_i) = \sum_{u^*+1}^{u_n} \mu(u_j) \tag{4.22}$$

例如，$\boldsymbol{U}' = 0.1/-4 + 0.5/-3 + 0.1/-2 + 0.0/-1 + 0.1/0 + 0.2/1 + 0.4/2 + 0.5/3 + 0.1/4$，由于 $u_1 = -4$，$u_9 = 4$，则当 $u^* = u_6$ 时，$\sum_{u_1}^{u_6} \mu(u_i) = \sum_{u_7}^{u_9} \mu(u_i) = 1$，所以中位数为 $u^* = u_6 = 1$，则 $U = 1$。

如果该点在有限元素之间，可用插值的方法来求取。例如，$\boldsymbol{U}' = 0.1/-4 + 0.5/-3 + 0.3/-2 + 0.1/-1 + 0.1/0 + 0.4/1 + 0.5/2 + 0.1/3 + 0.2/4$ 显然，u^* 在元素 0 和 1 之间。这时，可用线性插值处理，即令 $\Delta u = 1.2/(1.1 + 1.2) = 0.522$，所以取 $u^* = u_5 + \Delta u = 0.522$。实际上，模糊推理不需要很精确，因此也可以不用插值方法，直接取 $u^* = 0$ 或者 $u^* = 1$。

与最大隶属度法相比，这种方法利用了更多的信息，但计算比较复杂，特别是在连续隶属度函数时，需要求解积分方程，因此应用场合要比加权平均法少。

加权平均法比中位数法具有更佳的性能，而中位数法的动态性能要优于加权平均法，静态性能则略逊于加权平均法。使用中位数法的模糊控制器类似于多级继电器控制，加权平均法则类似于 PI 控制器。一般情况下，都优于最大隶属度法。

4.4.7 模糊推理的应用

例 4.10 设有模糊控制规则："如果温度低，则将风门开大"。设温度和风门开度的论域为 $\{1,2,3,4,5\}$。"温度低"和"风门大"的模糊量可以表示为

$$\text{“温度低”} = 1/1 + 0.6/2 + 0.3/3 + 0.0/4 + 0/5$$

$$\text{“风门大”} = 0/1 + 0.0/2 + 0.3/3 + 0.6/4 + 1/5$$

已知事实"温度较低"，可以表示为

$$\text{“温度较低”} = 0.8/1 + 1/2 + 0.6/3 + 0.3/4 + 0/5$$

试用模糊推理确定风门开度。

解 （1）确定模糊关系 $\boldsymbol{R}$

$$\boldsymbol{R} = \begin{bmatrix} 1.0 \\ 0.6 \\ 0.3 \\ 0.0 \\ 0.0 \end{bmatrix} \circ [0.0 \quad 0.0 \quad 0.3 \quad 0.6 \quad 1.0]$$

$$= \begin{bmatrix} 1.0 \wedge 0.0 & 1.0 \wedge 0.0 & 1.0 \wedge 0.3 & 1.0 \wedge 0.6 & 1.0 \wedge 1.0 \\ 0.6 \wedge 0.0 & 0.6 \wedge 0.0 & 0.6 \wedge 0.3 & 0.6 \wedge 0.6 & 0.6 \wedge 1.0 \\ 0.3 \wedge 0.0 & 0.3 \wedge 0.0 & 0.3 \wedge 0.3 & 0.3 \wedge 0.6 & 0.3 \wedge 1.0 \\ 0.0 \wedge 0.0 & 0.0 \wedge 0.0 & 0.0 \wedge 0.3 & 0.0 \wedge 0.6 & 0.0 \wedge 1.0 \\ 0.0 \wedge 0.0 & 0.0 \wedge 0.0 & 0.0 \wedge 0.3 & 0.0 \wedge 0.6 & 0.0 \wedge 1.0 \end{bmatrix}$$

$$= \begin{bmatrix} 0.0 & 0.0 & 0.3 & 0.6 & 1.0 \\ 0.0 & 0.0 & 0.3 & 0.6 & 0.6 \\ 0.0 & 0.0 & 0.3 & 0.3 & 0.3 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix}$$

（2）模糊推理

$$\boldsymbol{B}' = \boldsymbol{A}' \circ \boldsymbol{R} = \begin{bmatrix} 0.8 \\ 1.0 \\ 0.6 \\ 0.3 \\ 0.0 \end{bmatrix}^{\mathrm{T}} \circ \begin{bmatrix} 0.0 & 0.0 & 0.3 & 0.6 & 1.0 \\ 0.0 & 0.0 & 0.3 & 0.6 & 0.6 \\ 0.0 & 0.0 & 0.3 & 0.3 & 0.3 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix}$$

$$= (0.0, 0.0, 0.3, 0.6, 0.8)$$

(3) 模糊决策

用最大隶属度法进行决策得风门开度为 5。用加权平均判决法和中位数法进行决策得风门开度为 4。

4.5 小结

1. 不确定性推理中的基本问题

(1) 知识与证据不确定性的表示与度量

(2) 不确定性匹配算法及阈值

(3) 组合证据不确定性的算法

(4) 不确定性的传递算法

(5) 结论不确定性的合成

2. 可信度方法

可信度因子 $CF(H,E)$ 在 $[-1,1]$ 上取值，反映了前提条件与结论的联系强度。若由于相应证据的出现增加结论 H 为真的可信度，则取 $CF(H,E)>0$，证据的出现越是支持 H 为真，就使 $CF(H,E)$ 的值越大；反之，取 $CF(H,E)<0$，证据的出现越是支持 H 为假，就使 $CF(H,E)$ 的值越小；若证据的出现与否与 H 无关，则取 $CF(H,E)=0$。

证据 E 的可信度 $CF(E)$ 也是在 $[-1,1]$ 上取值的。对于初始证据，若对它的所有观察 S 能肯定它为真，则取 $CF(E)=1$；若肯定它为假，则取 $CF(E)=-1$；若它以某种程度为真，则取 $CF(E)$ 为 $(0,1)$ 中的某一个值；若它以某种程度为假，则取 $CF(E)$ 为 $(-1,0)$ 中的某一个值；若它还未获得任何相关的观察，此时可看作观察 S 与它无关，则取 $CF(E)=0$。

3. D－S 证据理论

D－S 证据理论引入概率分配函数、信任函数而非采用概率来度量不确定性，引用似然函数来处理由不知道而引起的不确定性。用概率分配函数的正交和来进行证据组合。

4. 模糊推理

在模糊逻辑中，给集合中每一个元素赋予一个介于 0 和 1 之间的实数，描述其属于一个集合的程度，该实数称为元素属于一个集合的隶属度。集合中所有元素的隶属度全体构成集合的隶属函数。

模糊关系描述两个模糊集合中的元素之间关联程度的多少。当论域为有限时，模糊关系的合成可用模糊矩阵的合成表示。模糊矩阵的合成可以由多种计算方法得到。常用的几种计算方法：最大－最小合成法、最大－代数积合成法。

通过条件模糊向量与模糊关系 $\boldsymbol{R}$ 的合成进行模糊推理，得到结论的模糊向量，然后采用模糊决策将模糊结论转换为精确量。

模糊决策方法有最大隶属度法、加权平均判决法、中位数法等。

思考题

4.1　什么是不确定性推理？有哪几类不确定性推理方法？不确定性推理中需要解决的基

本问题有哪些？

4.2　什么是可信度？由可信度因子 $CF(H,E)$ 的定义说明它的含义。

4.3　简述求取问题结论可信度的步骤。

4.4　说明概率分配函数、信任函数、似然函数的含义。

4.5　概率分配函数与概率相同吗？为什么？

4.6　如何用 D－S 证据理论描述假设、规则和证据的不确定性，并实现不确定性的推理和组合？

4.7　什么是模糊性？它与随机性有什么区别？试举出几个日常生活中的模糊概念。

4.8　模糊推理的一般过程是什么？

习题

4.1　设有如下一组推理规则

r_1:　IF　E_1　THEN　E_2　(0.6)

r_2:　IF　E_2　AND　E_3　THEN　E_4　(0.8)

r_3:　IF　E_4　THEN　H　(0.7)

r_4:　IF　E_5　THEN　H　(0.9)

且已知 $CF(E_1)=0.5$，$CF(E_3)=0.6$，$CF(E_5)=0.4$，结论 H 的初始可信度一无所知。求 $CF(H)$ 为多少？

4.2　已知：规则可信度为

r_1:　IF　E_1　THEN　H_1　(0.7)

r_2:　IF　E_2　THEN　H_1　(0.6)

r_3:　IF　E_3　THEN　H_1　(0.4)

r_4:　IF (H_1 AND E_4) THEN　H_2　(0.2)

证据可信度为：$CF(E_1)=CF(E_2)=CF(E_3)=CF(E_4)=0.5$，$H_1$ 的初始可信度一无所知，H_2 的初始可信度 $CF_0(H_2)=0.3$，计算结论 H_2 的可信度 $CF(H_2)$。

4.3　设有如下规则

r_1:　IF　E_1　THEN　H_1　(0.8)

r_2:　IF　E_2　THEN　H_1　(0.9)

r_3:　IF　E_3　AND　E_4　THEN　E_1　(0.9)

r_4:　IF　E_5　THEN　E_2　(0.7)

r_5:　IF　E_6　OR　E_7　THEN　E_2　(−0.3)

并已知初始证据的可信度为 $CF(E_3)=0.8$，$CF(E_4)=0.9$，$CF(E_5)=0.8$，$CF(E_6)=0.1$，$CF(E_7)=0.5$，试画出推理网络，并用可信度法计算 $CF(H_1)$。

4.4　设样本空间 $D=\{a,b,c,d\}$，M_1、M_2 为定义在 2^D 上的概率分布函数：

M_1: $M_1(\{b,c,d\})=0.7$，$M_1(\{a,b,c,d\})=0.3$，M_1 的其余基本概率数均为 0；

M_2: $M_2(\{a,b\})=0.6$，$M_2(\{a,b,c,d\})=0.4$，M_2 的其余基本概率数均为 0。

求它们的正交和 $M=M_1\oplus M_2$。

4.5　设有论域 $U=\{x_1,x_2,x_3,x_4,x_5\}$，$A$、$B$ 是 U 上的两个模糊集，且有

$$A = 0.85/x_1 + 0.7/x_2 + 0.9/x_3 + 0.9/x_4 + 0.7/x_5$$

$$B = 0.5/x_1 + 0.65/x_2 + 0.8/x_3 + 0.98/x_4 + 0.77/x_5$$

求 $A\cap B$、$A\cup B$ 和 $\bar{A}$。

4.6　设有如下两个模糊关系

$$\boldsymbol{A} = \begin{bmatrix} 0.7 & 0.6 & 0.3 \\ 0.7 & 0.6 & 0.2 \\ 0.5 & 0.5 & 0.2 \end{bmatrix} \qquad \boldsymbol{B} = \begin{bmatrix} 0.8 & 0.4 \\ 0.6 & 0.2 \\ 0.9 & 0.4 \end{bmatrix}$$

求 $\boldsymbol{A}\circ\boldsymbol{B}$。

4.7　设有如下两个模糊关系

$$\boldsymbol{R}_1 = \begin{bmatrix} 0.2 & 0.8 & 0.4 \\ 0.4 & 0 & 1.0 \\ 1.0 & 0.5 & 0 \\ 0.7 & 0.6 & 0.5 \end{bmatrix} \qquad \boldsymbol{R}_2 = \begin{bmatrix} 0.7 & 0.3 \\ 0.4 & 0.8 \\ 0.2 & 0.9 \end{bmatrix}$$

求两个模糊关系的合成 $\boldsymbol{R}_1\circ\boldsymbol{R}_2$。

4.8　设有如下三个模糊关系

$$\boldsymbol{R}_1 = \begin{bmatrix} 1.0 & 0.0 & 0.7 \\ 0.3 & 0.2 & 0.0 \\ 0.0 & 0.5 & 1.0 \end{bmatrix} \quad \boldsymbol{R}_2 = \begin{bmatrix} 0.6 & 0.6 & 0.0 \\ 0.0 & 0.6 & 0.1 \\ 0.0 & 0.1 & 0.0 \end{bmatrix} \quad \boldsymbol{R}_3 = \begin{bmatrix} 1.0 & 0.0 & 0.7 \\ 0.0 & 1.0 & 0.0 \\ 0.7 & 0.0 & 1.0 \end{bmatrix}$$

求模糊关系的合成 $\boldsymbol{R}_1\circ\boldsymbol{R}_2$，$\boldsymbol{R}_1\circ\boldsymbol{R}_3$，$\boldsymbol{R}_1\circ\boldsymbol{R}_2\circ\boldsymbol{R}_3$。

4.9　用 $X=\{x_1,x_2,x_3\}$ 表示病人集合，$Y=\{y_1,y_2,y_3,y_4,y_5\}$ 表示病人症状集合，$Z=\{z_1,z_2,z_3\}$ 表示病名集合。已知病人集合 X 与病人症状集合 Y 之间的模糊关系 $\boldsymbol{Q}$、病人症状集合 Y 与病名集合 Z 之间的模糊关系 $\boldsymbol{R}$ 分别为

$$\boldsymbol{Q} = \begin{bmatrix} 0.1 & 0.8 & 0.2 & 0.6 & 0.1 \\ 0.7 & 0.2 & 0.1 & 0.1 & 0.8 \\ 0.8 & 0.2 & 0.6 & 0.2 & 0.1 \end{bmatrix} \qquad \boldsymbol{R} = \begin{bmatrix} 0.3 & 0.7 & 1.0 \\ 1.0 & 0.3 & 0.2 \\ 0.3 & 1.0 & 1.0 \\ 1.0 & 0.3 & 0.2 \\ 0.3 & 1.0 & 0.7 \end{bmatrix}$$

确定病人集合 X 与病名集合 Z 之间的模糊关系 $\boldsymbol{S}$。

第5章 搜索求解策略

在求解一个问题时，涉及两个方面：一是该问题的表示，如果一个问题找不到一个合适的表示方法，就谈不上对它求解；另一方面则是选择一种相对合适的求解方法。在人工智能中，问题求解的基本方法有搜索法、归约法、归结法、推理法及产生式等。由于绝大多数需要人工智能方法求解的问题缺乏直接求解的方法，因此，搜索不失为一种求解问题的一般方法。搜索求解方法的应用非常广泛，例如在下棋等游戏软件中。

下面首先讨论搜索的基本概念，然后着重介绍状态空间知识表示和搜索策略，主要有回溯策略、宽度优先搜索、深度优先搜索等盲目的图搜索策略，以及 A 及 A^* 搜索算法等启发式图搜索策略。

5.1 搜索的概念

5.1.1 搜索的基本问题与主要过程

1. 在搜索中需要解决的基本问题

① 搜索过程是否一定能找到一个解。

② 当搜索过程找到一个解时，找到的是否最佳解。

③ 搜索过程的时间与空间复杂性如何。

④ 搜索过程是否终止运行或是否会陷入一个死循环。

2. 搜索的主要过程

① 从初始或目的状态出发，并将它作为当前状态。

② 扫描操作算子集，将适用当前状态的一些操作算子作用在其上而得到新的状态，并建立指向其父结点的指针。

③ 检查所生成的新状态是否满足结束状态，如果满足，则得到解，并可沿着有关指针从结束状态反向到达开始状态，给出一解答路径；否则，将新状态作为当前状态，返回第②步再进行搜索。

5.1.2 搜索策略

(1) 从初始状态出发的正向搜索，也称为数据驱动

正向搜索是从问题给出的条件——一个用于状态转换的操作算子集合出发的。搜索的过程

为应用操作算子从给定的条件中产生新条件，再用操作算子从新条件产生更多的新条件，这个过程一直持续到有一条满足目的要求的路径产生为止。数据驱动就是用问题给定数据中的约束知识指导搜索，使其沿着那些已知是正确的线路前进。

(2) 从目的状态出发的逆向搜索，也称为目的驱动

逆向搜索则是先从想达到的目的入手，看哪些操作算子能产生该目的以及应用这些操作算子产生目的时需要哪些条件，这些条件就成为我们要达到的新目的，即子目的。搜索就通过反向的连续的子目的不断进行，直至找到问题给定的条件为止。这样就找到了一条从数据到目的的操作算子所组成的链。

根据搜索过程中是否运用与问题有关的信息，可以分为启发式搜索和盲目搜索。

所谓盲目搜索(blind search)是指在对特定问题不具有任何有关信息的条件下，按固定的步骤(依次或随机调用操作算子)进行的搜索，它能快速地调用一个操作算子。

所谓启发式搜索(heuristic search)则是考虑特定问题领域可应用的知识，动态地确定调用操作算子的步骤，优先选择较适合的操作算子，尽量减少不必要的搜索，以求尽快地到达结束状态，提高搜索效率。

盲目搜索中，由于没有可参考的信息，只要能匹配的操作算子都须运用，这会搜索出更多的状态，生成较大的状态空间显示图；而启发式搜索中，运用一些启发信息，只采用少量的操作算子，生成较小的状态空间显示图，就能搜索到一个解答，但是每使用一个操作算子便需做更多的计算与判断。启发式搜索一般要优于盲目搜索，但不可过于追求更多的甚至完整的启发信息。

5.2　状态空间的搜索策略

5.2.1　状态空间表示法

状态空间(state space)表达法是知识表示的一种基本方法。

所谓状态是用来表示系统状态、事实等叙述型知识的一组变量或数组：

$$\boldsymbol{Q} = [q_1, q_2, \cdots, q_n]^{\mathrm{T}} \tag{5.1}$$

所谓操作是用来表示引起状态变化的过程型知识的一组关系或函数：

$$\boldsymbol{F} = \{f_1, f_2, \cdots, f_m\} \tag{5.2}$$

状态空间是利用状态变量和操作符号，表示系统或问题的有关知识的符号体系。状态空间可以用一个四元组表示：

$$(S, O, S_0, G)$$

其中，S 是状态集合。S 中每一元素表示一个状态，状态是某种结构的符号或数据。O 是操作算子的集合。利用算子可将一个状态转换为另一个状态。S_0 是包含问题的初始状态，是 S 的非空子集，$S_0 \subset S$。G 是包含问题的目的状态，是 S 的非空子集，$G \subset S$。G 可以是若干具体状态，也可以是满足某些性质的路径信息描述。

从 S_0 结点到 G 结点的路径称为求解路径。求解路径上的操作算子序列为状态空间的一个解。例如，操作算子序列 $O_1, \cdots, O_k$ 使初始状态转换为目标状态(如图 5.1 所示)：

$$S_0 \xrightarrow{O_1} S_1 \xrightarrow{O_2} S_2 \xrightarrow{O_3} \cdots \xrightarrow{O_k} G$$

图 5.1 状态空间的解

则 $O_1,\cdots,O_k$ 即为状态空间的一个解。当然,解往往不是唯一的。

任何类型的数据结构都可以用来描述状态,如符号、字符串、向量、多维数组、树和表格等。所选用的数据结构形式要与状态所蕴涵的某些特性具有相似性。如对八数码问题,一个 3×3 的阵列便是一个合适的状态描述方式。

例 5.1 八数码问题的状态空间表示。

八数码问题(重排九宫问题)是在一个 3×3 的方格盘上,放有 1~8 的数码,另一格为空。空格四周上下左右的数码可移到空格。需要找到一个数码移动序列使初始的无序数码转变为一些特殊的排列。例如,图 5.2(a)所示的八数码问题的初始状态为问题的一个布局,需要找到一个数码移动序列使下面的初始布局转变为图 5.2(b)所示的目标状态:

2	3	1
5		8
4	6	7

(a) 初始状态

1	2	3
8		4
7	6	5

(b) 目标状态

图 5.2 八数码问题

该问题可以用状态空间来表示。此时八数码的任何一种摆法就是一个状态,所有的摆法即为状态集 S,它们构成了一个状态空间,其数目为 9!。而 G 是指定的某个或某些状态。对于操作算子,如果着眼在数码上,相应的操作算子就是数码的移动,其操作算子共有 4(方向)×8(数码)=32 个。如着眼在空格上,即空格在方格盘上的每个可能位置的上下左右移动,其操作算子可简化成 4 个:

将空格向上移 Up

将空格向左移 Left

将空格向下移 Down

将空格向右移 Right

移动时要确保空格不会移出方格盘之外,因此并不是在任何状态下都能运用这 4 个操作算子。如空格在方格盘的右上角时,只能运用两个操作算子——向左移 Left 和向下移 Down。

5.2.2 状态空间的图描述

状态空间可用有向图来描述,图的结点表示问题的状态,图的弧表示状态之间的关系,就是求解问题的步骤。初始状态对应于实际问题的已知信息,是图中的根结点。问题的状态空间描述中,寻找从一种状态转换为另一种状态的某个操作算子序列就等价于在一个图中寻找某一路径。

如图 5.3 所示用有向图描述的状态空间。该图表示对状态 S_0 允许使用操作算子 O_1,O_2 及

O_3，并分别使 S_0 转换为 S_1，S_2 及 S_3。这样一步步利用操作算子转换下去，如 $S_{10} \in G$，则 O_2，O_6，O_{10}就是一个解。

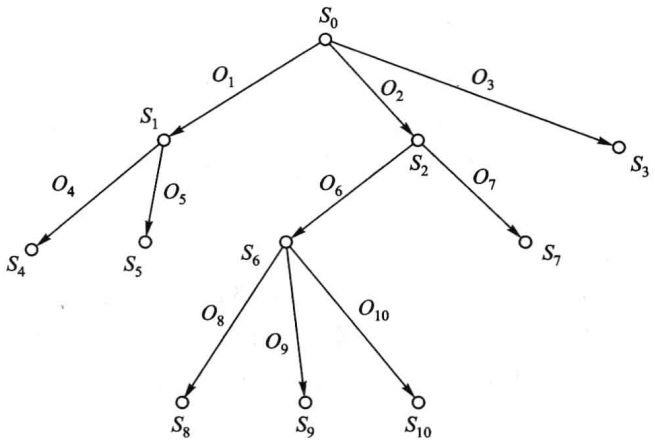


图 5.3　状态空间的有向图描述

上面是较为形式化的说明，下面再以八数码问题为例，讨论具体问题的状态空间的有向图描述。

例 5.2　对于八数码问题，如果给出问题的初始状态，就可以用图来描述其状态空间。其中的弧可用表明空格的 4 种可能移动的 4 个操作算子来标注，即空格向上移 Up、向左移 Left、向下移 Down、向右移 Right。该图的部分描述如图 5.4 所示。

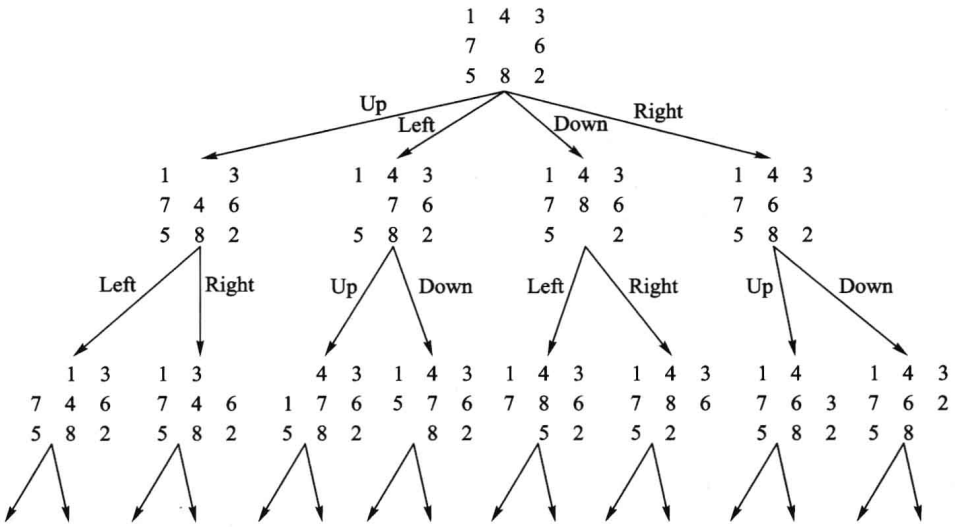


图 5.4　八数码状态空间图（部分）

在某些问题中，各种操作算子的执行是有不同费用的。如在旅行商问题中，两两城市之间的距离通常是不相等的，那么，在图中只需要给各弧线标注距离或费用即可。下面以旅行商问题为例说明这类状态空间的图描述，其终止条件则是用解路径本身的特点来描述，即经过图中所有城市的最短路径找到时搜索便结束。

例 5.3　旅行商问题（traveling salesman problem，TSP）或推销员路径问题是假设一个推销员

从出发地，到若干个城市去推销产品，然后回到出发地。问题是要找到一条最好的路径，使得推销员访问每个城市后回到出发地所经过的路径最短或者费用最少。

图 5.5 是这个问题的一个实例，图中结点代表城市，弧上标注的数值表示经过该路径的费用（或距离）。我们假定推销员从 A 城出发。

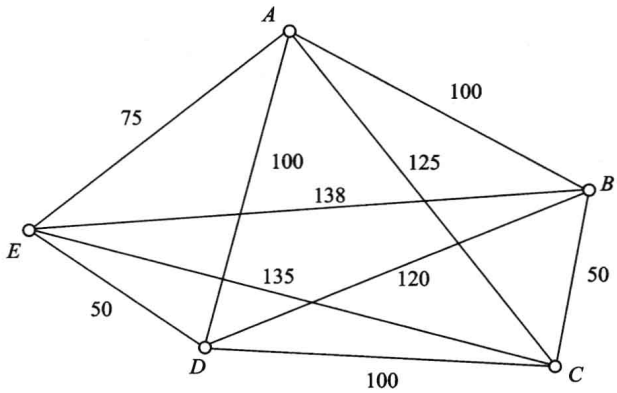


图 5.5 旅行商问题的一个实例

可能的路径有很多，例如，费用为 375 的路径（A,B,C,D,E,A）就是一个可能的旅行路径，但目的是要找具有最小费用的旅行路径。注意，这里对目的的描述是关注整个路径的特性而不是单个状态的特性。

图 5.6 是该问题的部分状态空间表示。

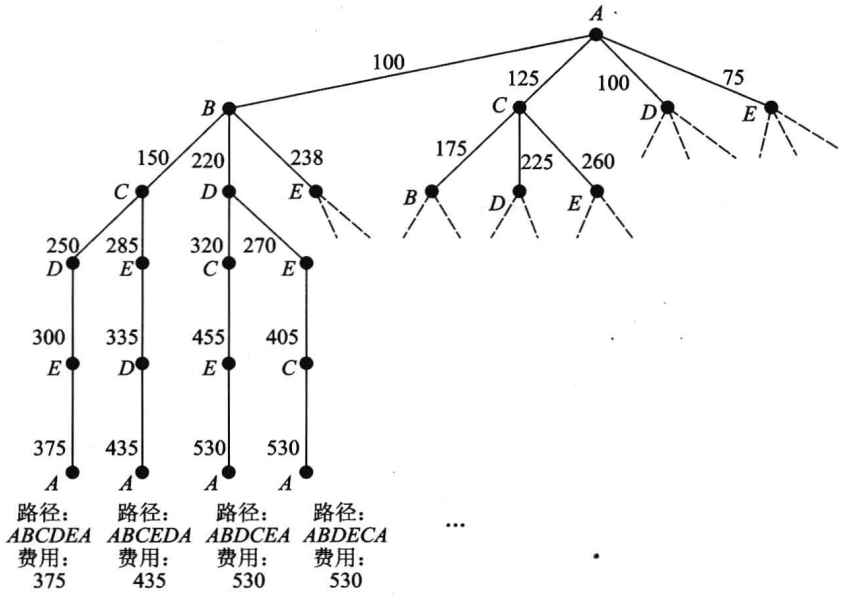


图 5.6 旅行商问题的状态空间图（部分）

上面两个例子中，只绘出了问题的部分状态空间图，当然，完全可以绘出问题的全部状态空间图，但对许多实际问题，要在有限的时间内绘出问题的全部状态图是不可能的。例如旅行商问

题，n 个城市存在$\frac{1}{2}(n-1)!$ 条路径。如果用 10^8 次/秒的计算机进行穷举，则当 $n=7$ 时，搜索时间为 $t=2.5\times10^{-5}$ s；当 $n=15$ 时，$t=1.8$ h；当 $n=20$ 时，$t=350$ 年；当 $n=50$ 时，$t=5\times10^{48}$年；当 $n=100$ 时，$t=5\times10^{142}$年；当 $n=200$ 时，$t=5\times10^{358}$年。因此，这类显式表示对于大型问题的描述是不切实际的，而对于具有无限结点集合的问题则是不可能的。因此要研究能够在有限时间内搜索到较好解的搜索算法。

在状态空间搜索中，搜索某个状态空间以求得操作算子序列的一个解答的过程。这种搜索是状态空间问题求解的基础。

搜索策略的主要任务是确定选取操作算子的方式。它有两种基本方式：盲目搜索和启发式搜索。

5.3　盲目的图搜索策略

5.3.1　回溯策略

不管是正向搜索还是逆向搜索求解问题，都是在状态空间图中找到从初始状态到目的状态的路径。路径上弧的序列对应于解题的步骤。若在选择操作算子求解问题时，能给出绝对可靠的预测或有绝对正确的选择策略，一次性地成功穿过状态空间而达到目的，构造出一条解题路径，那就不需要所谓的搜索了。但事实上，不是总能给出正确的选择，求解实际问题时必须尝试多条路径才能找到目的状态。一个搜索算法的策略就是要决定树或图中状态的搜索次序。回溯策略是一种系统地尝试状态空间中各种不同路径的技术。

回溯策略的搜索是从初始状态出发，不停地、试探性地寻找路径，直到到达目的或“不可解结点”，即“死胡同”为止。回溯策略是当遇到不可解结点时就回溯到路径中最近的父结点上，查看该结点是否还有其他的子结点未被扩展。若有，则沿这些子结点继续搜索；如果找到目标，就成功退出搜索，返回解题路径。

可以看出，回溯策略的搜索过程呈现出递归过程的性质。搜索过程在每个结点上的检查遵循着递归方式。下面给出递归过程。

```
    Step Track (DataList):
    Data: = First(DataList);                          * 当前状态为状态序列表中第一个状态
    if Member(Data, Tail(DataList)) then return FAIL;  * 回老路，退回
    if Goal(Data) then return NIL;                    * 达到目的状态，成功返回
    if DeadEnd(Data) then return FAIL;                * 达到不合理状态，退回
    if Length(DataList) > Bound then return FAIL;     * 已到深度限制，退回
    Rules: = AppRules(Data);                          * 得出可应用的规则集
Loop:if Null(Rules) then return FAIL;                 * 进入死胡同，退回
    R: = First(Rules);                                * 取出第一条可用规则
    Rules: = Tail(Rules);
    Newdata: = Gen(R,Data);                           * 运用规则，生成新状态
    NewDataList: = Cons(Newdata, DataList);
    Path: = Back Track(NewDataList);                  * 递归
```

```
If Path: = FAIL then go Loop
        else return Cons(R, Path);
```

说明:变量符号 Data, DataList, Rules, R, Newdata, NewdataList, Path 分别表示当前状态、状态序列表、规则集序列表、当前运用的规则、生成的新状态、生成的新状态序列表、当前解路径表;常量符号 FAIL, NIL, Bound, loop 分别表示回溯点标记、空表、深度限制值、循环标号;函数 First(y)表示在 y 表中取第一个元素,函数 Tail(x)是取除了第一个元素的 x 表的其余部分,函数 Member(x,y)表示变量 x 是否为 y 表的一个元素,函数 Goal(x)表示变量状态 x 是否为目的状态,函数 DeadEnd(x)表示变量状态 x 是否为不可能解路径上的状态,函数 Length(x)是取 x 表的表长度,函数 AppRules(x)是求出在变量状态 x 上可运用的规则集,函数 Null(x)表示 x 表是否为空表,函数 Gen(x,y)是求 x 规则运用在 y 状态后所生成的新状态,函数 Cons(x,y)是将变量状态 x 加在 y 表的前部。

图 5.7 给出了一个状态空间中应用回溯搜索的示意图,图中虚线箭头的方向表示搜索的轨迹,结点边的数字表示被搜索到的次序。

回溯搜索的算法用 3 张表来保存状态空间中的不同性质结点:

(1) 路径状态表

路径状态(path states, PS)表保存当前搜索路径上的状态。如果找到了目的,PS 就是解路径上的状态有序集。

(2) 新的路径状态表

新的路径状态(new path states, NPS)表包含了等待搜索的状态,其后裔状态还未被搜索到,即未被生成扩展。

(3) 不可解状态表

不可解状态(no solvable states,NSS)表列出了找不到解路径的状态。如果在搜索中扩展出的状态是它的元素,则可立即将之排除,不必沿该状态继续搜索。

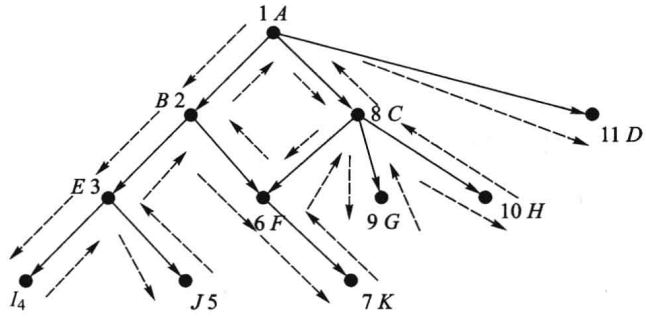


图 5.7 回溯搜索示意图

为了避免造成无穷循环搜索,需要检测并删除多次出现的那些状态。具体检测可通过判断每一个新生成的状态是否在 PS,NPS,NSS 三张表中来实现。如果它属于其中一张表,就说明它已被搜索过而不必再考虑。

当前正在被检测的状态,记作 CS(current state)。CS 总是等于最近加入 PS 中的状态,是当前正在探寻解题路径的"前锋"。各种合适的推理规则或其他问题求解操作都可应用于 PS。应

用后一般便得到一些新状态，即 PS 的子状态的有序集，然后再将该集合中的第一个子状态作为当前状态 CS，并加入到 PS 中，其余的则按序放入 NPS 中，用于以后的搜索。如果应用后 CS 没有子状态，则要从 PS、NPS 中删除它，同时将其加入到 NSS，之后回溯查找 NPS 中表首位置的状态。

```
Function backtrack:
begin
   PS: = [Start]; NPS: = [Start]; NSS: = [ ]; CS: = Start;          * 初始化
   while NPS≠[ ] do
         begin
         if CS = 目的状态 then return(PS);                          * 成功，返回解题路径
         if CS 没有子状态(不包括 PS,NPS 和 NSS 中已有的状态)
            then
              begin
                 while ((PS 非空) and (CS = PS 中的第一个元素))do
                       begin
                          将 CS 加入 NSS                            * 标明此状态不可解
                          从 PS 中删除第一个元素 CS                  * 回溯
                          从 NPS 中删除第一个元素 CS;
                          CS: = NPS 中的第一个元素;
                       end;
              将 CS 加入 PS;
              end
         else
           begin
             将 CS 子状态(不包括 PS、NPS 和 NSS 中已有的)加入 NPS;
             CS: = NPS 中的第一个元素;
                 将 CS 加入到 PS;
           end
         end;
      return FAIL;
end.
```

图 5.7 的回溯轨迹如下：

初值：PS = [*A*]；NPS = [*A*]；NSS = [ ]；CS = *A*

重复	CS	PS	NPS	NSS
0	*A*	[*A*]	[*A*]	[ ]
1	*B*	[*BA*]	[*BCDA*]	[ ]
2	*E*	[*EBA*]	[*EFBDCA*]	[ ]
3	*I*	[*IEBA*]	[*IJEFBCDA*]	[ ]
4	*J*	[*JEBA*]	[*JEFBCDA*]	[*I*]
5	*F*	[*FBA*]	[*FBCDA*]	[*EJI*]
6	*K*	[*KFBA*]	[*KFBCDA*]	[*EJI*]
7	*C*	[*CA*]	[*CDA*]	[*BFKEJI*]
8	*G*	[*GCA*]	[*GHCDA*]	[*BFKEJI*]

上面的搜索过程显示出回溯是状态空间中的一个正向搜索。它将初始条件作为初始状态，对其子状态进行搜索以寻找目的。如将目的作为搜索图的根即初始状态，本算法便可看做逆向搜索。如对算法中“成功，返回解题路径”的判别条件“CS = 目的状态”修改为“搜索路径的性质优劣”，那算法必须通过检查 PS 中的路径来确定是否到达目的。

回溯是状态空间搜索的一个基本算法。各种图搜索算法，包括深度优先、宽度优先、最好优先搜索等都有回溯的思想，其中宽度、深度优先搜索是状态空间的最基本的搜索策略。

5.3.2 宽度优先搜索策略

宽度优先搜索策略是按照图 5.8 所示的次序来搜索状态的。由 S_0 生成状态 1，2，然后扩展状态 1，生成状态 3，4，5，接着扩展状态 2，生成状态 6，7，8，该层扩展完后，再进入下一层，对状态 3 进行扩展，如此一层一层地扩展下去，直到搜索到目的状态（如果目的状态存在）。

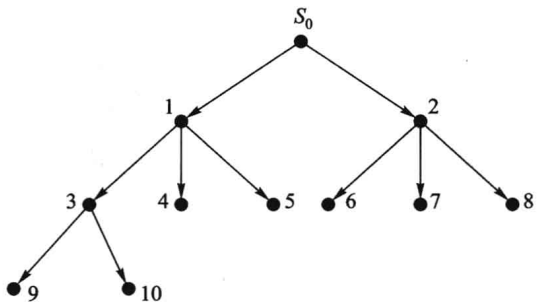


图 5.8 宽度优先搜索策略中状态的搜索次序

在实际宽度优先搜索时，为了保存状态空间搜索的轨迹，用到了两个表：open 表和 closed 表。open 表与回溯算法中的 NPS 表相似，包含了已经生成出来但其子状态未被搜索的状态。open 表中状态的排列次序就是搜索的次序。closed 表记录了已被生成扩展过的状态，它相当于回溯算法中 PS 表和 NSS 表的合并。下面是宽度优先搜索过程。

```
Procedure breadth_first_search
begin
open: = [ start ]; closed: = [ ]                          * 初始化
while open ≠ [ ] do
      begin
        从 open 表中删除第一个状态，称之为 n;
        将 n 放入 closed 表中;
        if n = 目的状态 then return (success);
        生成 n 的所有子状态;
        从 n 的子状态中删除已在 open 或 closed 表中出现的状态;   * 避免循环搜索
        将 n 的其余子状态，按生成的次序加入到 open 表的后段;
        end;
      end;
```

注意，open 表是一个队列结构，即先进先出（FIFO）的数据结构。曾在 open 表或 closed 表中出现过的子状态要删去。

如果过程因 while 循环条件(open≠[ ])不满足而结束,则表明已搜索完整个状态空间但未搜索到目的状态,说明搜索失败了。

如果整个状态空间是无限的并不能满足 while 的循环条件即无解,过程便会一直搜索下去,所以在过程中应增加"搜索超时而结束"的终止部分。

下面举一个宽度优先搜索的例子。

例 5.4　如图 5.9 所示,通过搬动积木块,希望从初始状态达到一个目的状态,即三块积木堆叠在一起。积木 A 在顶部,积木 B 在中间,而积木 C 在底部。

这个问题的唯一操作算子为 $MOVE(X,Y)$,即把积木 X 搬到 Y(积木或桌面)上面。如"搬动积木 A 到桌面上"表示为 $MOVE(A,Table)$。该操作算子可运用的先决条件是:

① 被搬动积木的顶部必须为空。

② 如果 Y 是积木(不是桌面),则积木 Y 的顶部也必须为空。

③ 同一状态下,运用操作算子的次数不得多于一次(可从 open 表和 closed 表加以检查)。

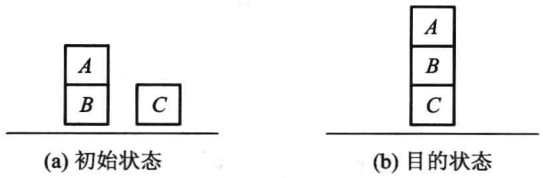


图 5.9　积木问题

图 5.10 表示了由宽度优先搜索所产生的搜索树。各结点是以产生和扩展的先后次序编下标的。当搜索到 S_{10} 目的状态时,过程便结束。此时,open 表包含 S_0 至 S_5,而 closed 表包含 S_6 至 S_{10}。

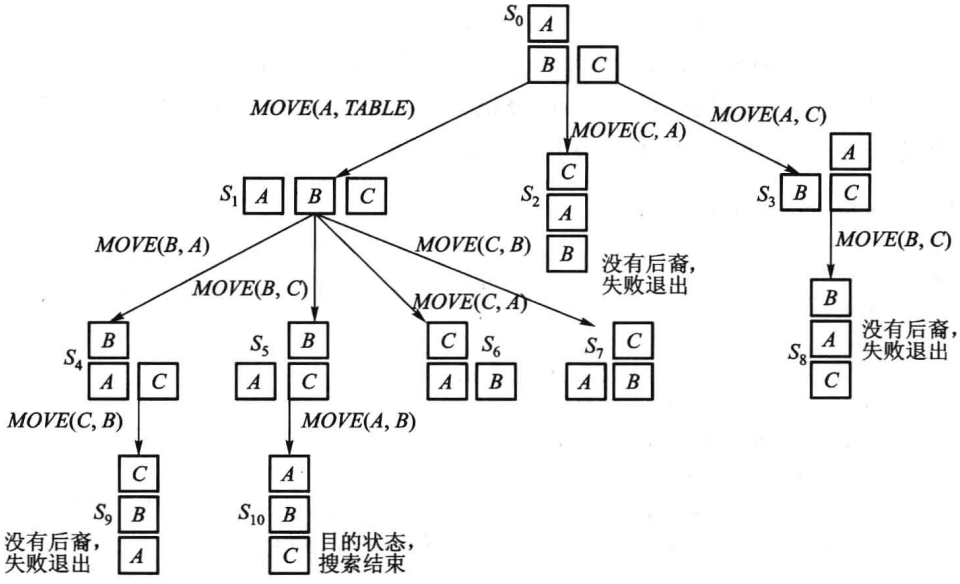


图 5.10　积木问题的宽度优先搜索树

由于宽度优先搜索总是在生成扩展完 N 层的所有结点之后才转向 $N+1$ 层,所以它总能找

到最好的解(如有解),但当图的分支数太多,即状态的后裔数的平均值较大,这种组合爆炸就会使算法耗尽资源,从而在可利用的空间中找不到解。这是由于每层搜索中所有生成的未扩展的结点都要保存到 open 表中,如果解题路径较长,这个数目将会大得使搜索无法进行。

5.3.3 深度优先搜索策略

深度优先搜索策略是按图 5.11 所示的次序来搜索状态的。

搜索从 S_0 出发,沿一个方向一直扩展下去,如状态 1,2,3,直到达到一定的深度(这里假定为 3 层)。如果未找到目的状态或无法再扩展时,便回溯到另一条路径(状态 4)继续搜索;若还未找到目的状态或无法再扩展时,再回溯到另一条路径(状态 5,6)搜索……

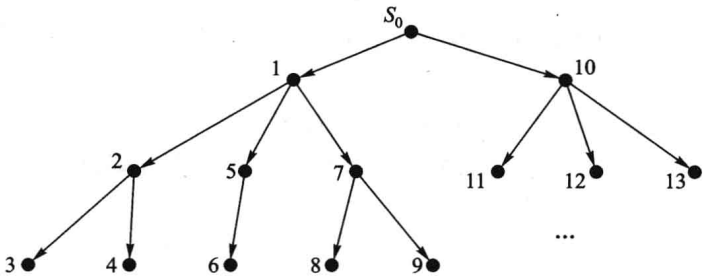


图 5.11 深度优先搜索策略中状态的搜索次序

在深度优先搜索中,当搜索到某一个状态时,它所有的子状态以及子状态的后裔状态都必须先于该状态的兄弟状态被搜索。深度优先搜索在搜索空间时应尽量往深处去,只有再也找不出某状态的后裔状态时,才能考虑它的兄弟状态。

很明显,深度优先搜索法不一定能找到最优解,并且可能由于深度的限制,会找不到解(待求问题存在着解),然而,如果不加深度限制值,则可能会沿着一条路径无限地扩展下去,这当然是不希望的。为了保证找到解,那就应选择合适的深度限制值,或采取不断加大深度限制值的办法,反复搜索,直到找到解。

修改宽度优先搜索过程便可得到深度优先搜索过程:

```
Procedure depth_first_search
  begin
  open: = [start];closed: = [ ];d: = 深度限制值
  while open≠[ ] do
    begin
    从 open 表中删除第一个状态,称之为 n;
    将 n 放入 closed 表中;
    if n = 目的状态 then return (success);
    if n 的深度 < d then continue;
    生成 n 的所有子状态;
    从 n 的子状态中删除已在 open 或 closed 表中出现的状态;
    将 n 的其余子状态,按生成的次序加入到 open 表的前端;
    end;
end;
```

注意，open 表是一个堆栈结构，即先进后出（FILO）的数据结构。open 表用堆栈实现的方法使得搜索偏向于最后生成的状态；曾在 open 表或 closed 表中出现过的子状态要删去。和 breadth_first_search 中一样，此处 open 表列出了所有已生成但未作扩展的状态（搜索的“前锋”），closed 表记录了已扩展过的状态。同 breadth_first_search 一样，两个算法都可以把每个结点同它的父结点一起保存，以便构造一条从起始状态到目的状态的路径。

与宽度优先搜索不同的是，深度优先搜索并不能保证第一次搜索到的某个状态时的路径是到这个状态的最短路径。对任何状态而言，以后的搜索有可能找到另一条通向它的路径。如果路径的长度对解题很关键的话，当算法多次搜索到同一个状态时，它应该保留最短路径。具体可把每个状态用一个三元组来保存（状态，父状态，路径长度）。当生成子状态时，将路径长度加 1，和子状态一起保存起来。当有多条路径可到达某子状态时，这些信息可帮助选择最优的路径。必须指出，深度优先搜索中即使保存这些信息也不能保证算法能得到的解题路径是最优的。

下面举一深度优先搜索的例子。

例 5.5　卒子穿阵问题。要求一卒子从顶部通过图 5.12 所示的阵列到达底部。卒子行进中不可进入到代表敌兵驻守的区域（标注 1），并不准后退。假定深度限制值为 5。

行	1	2	3	4	列
1	1	0	0	0	
2	0	0	1	0	
3	0	1	0	0	
4	1	0	0	0	

图 5.12　阵列图

由深度优先搜索策略产生的搜索树如图 5.13 所示。在结点 S_0，卒子还没有进入阵列，在其他结点，其所处的阵列位置用一对数字（行号，列号）表示，结点的编号代表搜索的次序。

当搜索过程终止时，open 表含有结点 S_{17}（为一目的结点）和 S_{18}，而其他结点（$S_0 \sim S_{16}$）都在 closed 表中。很明显，所求得的解路（$S_0, S_8, S_{14}, S_{15}, S_{16}, S_{17}$）比最优路径（从（1，4）进入）多走一步。

此外，由于本算法把状态空间作为搜索树来考虑，而不是当作一般的搜索图考虑，所以，忽略了两个不同的结点 S_2 和 S_9 实际上是代表了同一个状态这个问题。因而，从结点 S_9 向下的搜索实际是在重复从结点 S_2 向下的搜索。

深度优先搜索能尽快地深入下去，如果已知解题路径很长，深度搜索就不会在开始状态的周围即“浅层”状态上浪费时间；但另一方面，深度优先搜索会在搜索的深处“迷失方向”，找不到通向目的的更短路径或陷入到一个不通往目的的无限长的路径中。深度优先搜索在搜索有大量分支的状态空间时有相当高的效率，它不需要把某一层上的所有结点都进行扩展。

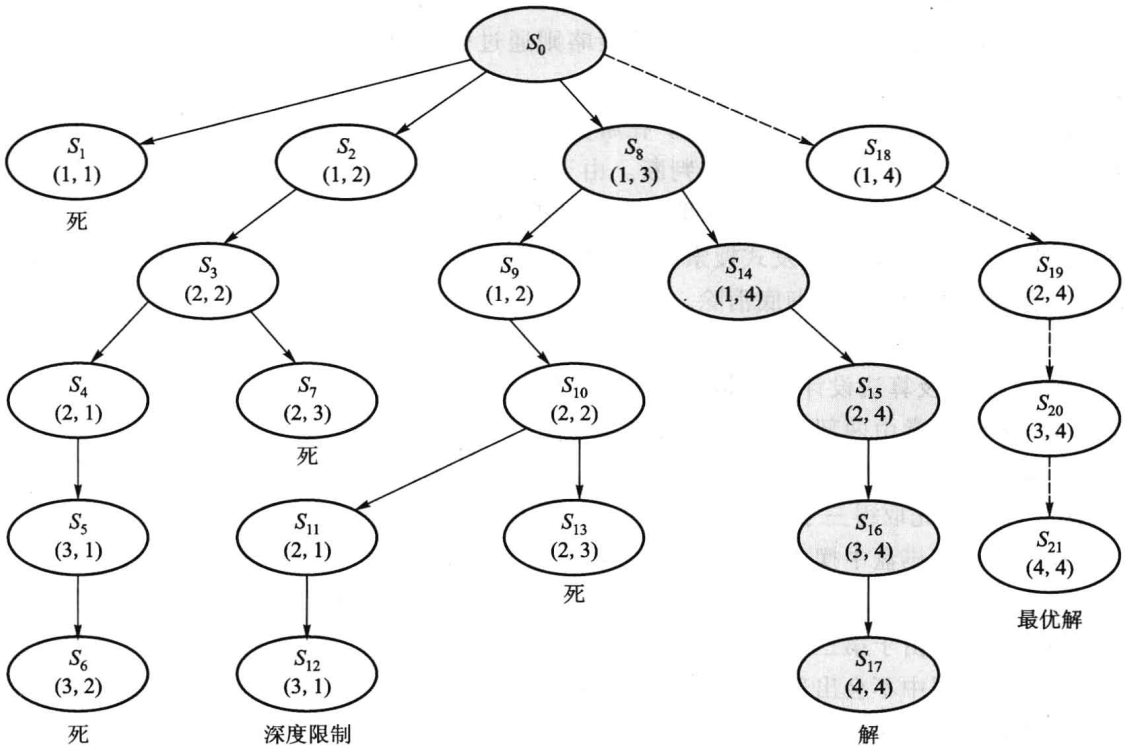


图 5.13 卒子穿阵的深度优先搜索树

5.4 启发式图搜索策略

前节所介绍的大部分搜索方法都是盲目搜索方法，其搜索的复杂性往往是很高的。为了提高算法的效率，必须放弃利用纯数学的方法来决定搜索结点的次序，而需要对具体问题作具体分析，利用与问题有关的信息，从中得到启发来引导搜索，以达到减少搜索量的目的，这就是启发式搜索。

本节先对启发及启发式策略所涉及的问题作一介绍，然后将具体介绍启发式图搜索算法——A 及 A^* 算法，最后对启发式搜索算法的性质进行讨论。

5.4.1 启发式策略

启发式(heuristic)策略就是利用与问题有关的启发信息引导搜索。

在状态空间搜索中，启发式被定义成一系列操作算子，并能从状态空间中选择最有希望到达问题解的路径。

问题求解系统可在两种基本情况下运用启发式策略：

① 由于在问题陈述和数据获取方面固有的模糊性，可能会使一个问题没有一个确定的解，这就要求系统能运用启发式策略作出最有可能的解释。

② 虽然一个问题可能有确定解，但是其状态空间特别大，搜索中生成扩展的状态数会随着

搜索的深度呈指数级增长。穷尽式搜索策略如宽度优先或深度优先搜索，在一个给定的较实际的时空内很可能得不到最终的解，而启发式策略则通过引导搜索向最有希望的方向进行来降低搜索复杂度。

但是，启发式策略也是极易出错的。在解决问题过程中启发仅仅是下一步将要采取措施的一个猜想。它常常根据经验和直觉来判断。由于启发式搜索只利用特定问题的有限的信息，很难准确地预测下一步在状态空间中采取的具体的搜索行为。一个启发式搜索可能得到一个次优解，也可能一无所获。这是启发式搜索固有的局限性，而这种局限性不可能由所谓更好的启发式策略或更有效的搜索算法来彻底消除。

在问题求解中，需要启发式知识来剪枝以减少状态空间，否则只能求解一些模拟的小问题。因此，启发式策略及算法设计一直是人工智能的核心问题。

启发式搜索通常由两部分组成：启发方法和使用该方法搜索状态空间的算法。

例 5.6　一字棋游戏。在九宫棋盘上，从空棋盘开始，双方轮流在棋盘上摆各自的棋子 × 或 ○（每次一枚），谁先取得三子一线（一行、一列或一条对角线）的结果就取胜。

× 和 ○ 能够在棋盘中摆成的各种不同的棋局就是问题空间中的不同状态。在 9 个位置上摆放｛空，×，○｝有 3^9 种棋局。当然，其中大多数不会在实际对局中出现。任一方的摆棋就是状态空间中的一条弧。由于第三层及更底层的某些状态可以通过不同路径到达，所以其状态空间是图而不是树。但图中不会出现回路，因为弧具有方向性（下棋时不允许悔棋）。这样搜索路径时就不必检测是否有回路。

在一字棋游戏中，第一步有 9 个空格便有 9 种可能的走法，第二步 8 种，第三步 7 种……如此递减，所以共有 $9\times8\times7\times\cdots\times1$ 即 9！种不同的棋局状态，其状态空间较大，穷尽搜索的组合数较大。

可以利用启发方法来剪枝以减少状态空间的大小。根据棋盘的对称性可以减少搜索空间的大小。棋盘上很多棋局是等价的，如第一步实际上只有 3 种走法，角、边的中央和棋盘正中，这时状态空间的大小为 $3\times8!$。在状态空间的第二层上，由对称性还可进一步减少到 $3+12\times7!$ 种，当然还可以再进一步地减少状态空间的大小。

此外，使用启发方法进行搜索几乎可以整个地消除复杂的搜索过程。设想将棋子走到棋盘上 × 有最多的赢线的格子上，若两种状态有相等的赢的几率，取其中的一种。最初的三种状态显示在图 5.14 中。这样，可设计一种算法（完全实现启发式搜索），它选择具有最高启发值的棋局状态进行下棋子。在本例的这种情况下，只需搜索 × 占据棋盘正中位置的棋局状态，而其他的各种棋局状态连同它们的延伸棋局状态都不必再考虑了。如图 5.15 所示 2/3 的状态空间就不必搜索了。

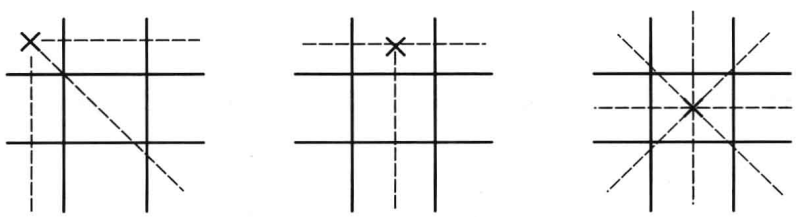


图 5.14　启发式策略的运用

第一步棋下完后，对方只能有两种走法。无论选择哪种走法，我方均可以通过启发式搜索来选择下一步可能的走法。在搜索过程中，每一步只需估价单个结点的子结点便可决定下哪步棋。图 5.15 显示了游戏前三步简化了的搜索过程。每种状态都标记了它的启发值。

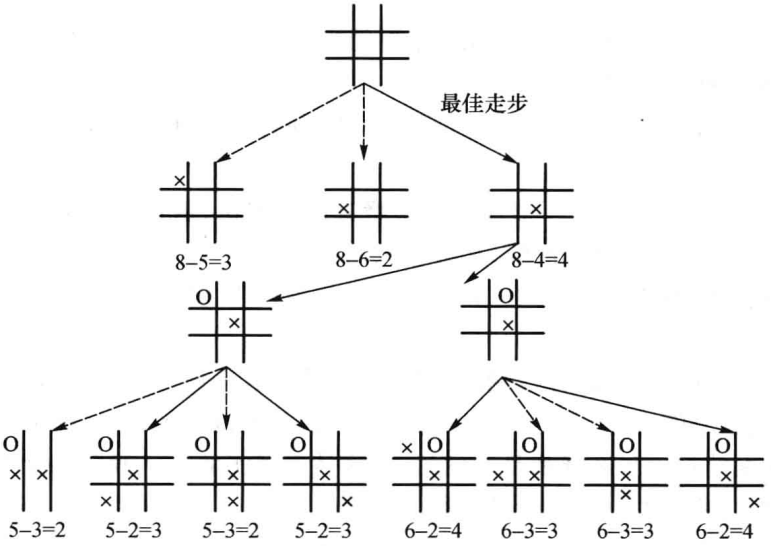


图 5.15 启发式搜索下缩减的状态空间

要精确地计算待搜索的状态数目比较难，但可以大致计算它的上限。一盘棋最多走 9 步，每步的下一步平均有 4,5 种走法，这样大约就是 4.5 ×9，近 40 种状态，这比原来 9！大小的状态空间缩小了很多。

5.4.2 启发信息和估价函数

在实际解决一个具体问题时，人们常常把一个复杂的实际问题抽象化，保留某些主要因素，忽略大量次要因素，从而将这个实际问题转化成具有明确结构的有限或无限的状态空间问题，这个状态空间中的状态和变换规律都是已知的集合，因此可以找到一个求解该问题的算法。

在具体求解中，启发式搜索能够利用与该问题有关的信息来简化搜索过程，称此类信息为启发信息。然而，在求解问题中能利用的大多不是具有完备性的启发信息，而是非完备的启发信息。其原因是：

① 大多数情况下，求解问题系统不可能知道与实际问题有关的全部信息，因而无法知道该问题的全部状态空间，也不可能用一套算法来求解所有的问题。这样就只能依靠部分状态空间、一些特殊的经验和有关信息来求解其中的部分问题。

② 有些问题在理论上虽然存在着求解算法，但是在工程实践中，这些算法不是效率太低，就是根本无法实现。为了提高求解问题的效率，不得不放弃使用这些“完美的”算法，而求助于一些启发信息来进行启发式搜索。

如在博弈问题中，计算机为了保证最后胜利，可以将所有可能的走法都试一遍，然后选择最佳走步。这样的算法是可以找到的，但计算所需的时空代价十分惊人。对于可能有的棋局数来说，一字棋是 9！ $=3.6\times10^5$，西洋跳棋是 10^{78}，国际象棋是 10^{120}，围棋是 10^{761}。假设每步可以搜

索一个棋局，用极限并行速度（10^{-104}年/步）来处理，搜索一遍国际象棋的全部棋局也得 10^{16}年即 1 亿亿年才可以算完，而我们已知的宇宙寿命才 100 亿年！

由此看来，启发式的问题求解，不仅在实践上是需要的，而且在理论上也是必不可少的。

启发信息按运用的方法不同可分为三种：

① 陈述性启发信息。一般被用于更准确、更精炼地描述状态，使问题的状态空间缩小，如待求问题的特定状况等属于此类信息。

② 过程性启发信息。一般被用于构造操作算子，使操作算子少而精，如一些规律性知识等属于此类信息。

③ 控制性启发信息。它是关于表示控制策略方面的知识，包括协调整个问题求解过程中所使用的各种处理方法、搜索策略、控制结构等有关的知识。

为提高搜索效率就需要利用上述三种启发信息作为搜索的辅助性策略。这里主要介绍控制性的启发信息。

利用控制性的启发信息有两种极端的情况：一种是没有任何控制性知识作为搜索的依据，因而搜索的每一步完全是随意的，如随机搜索、宽度搜索、深度搜索等；另一种是有充分的控制知识作为依据，因而搜索的每一步选择都是正确的，但这是不现实的。一般情况介于二者之间。在搜索过程中需要根据这些启发信息估计各个结点的重要性。

估价函数（evaluation function）的任务就是估计待搜索结点的“有希望”程度，并依次给它们排定次序。估价函数 $f(x)$ 可以是任意一种函数，如定义为结点 x 处于最佳路径上的概率，或是 x 结点和目的结点之间的距离或差异，或是 x 格局的得分等。一般来说，估计一个结点的价值，必须综合考虑两方面的因素：已经付出的代价和将要付出的代价。在此，把估价函数 $f(n)$ 定义为从初始结点经过 n 结点到达目的结点的路径的最小代价估计值，其一般形式是

$$f(n) = g(n) + h(n) \tag{5.3}$$

其中，$g(n)$ 是从初始结点到 n 结点的实际代价，而 $h(n)$ 是从 n 结点到目的结点的最佳路径的估计代价。因为实际代价 $g(n)$ 可以根据已生成的搜索树实际计算出来，而估计代价 $h(n)$ 是对未生成的搜索路径作某种经验性的估计。这种估计来源于对问题解的某些特性的认识，希望依靠这些特性来更快地找到问题的解，因此，主要是 $h(n)$ 体现了搜索的启发信息。

一般地，在 $f(n)$ 中，$g(n)$ 的比重越大，越倾向于宽度优先搜索方式，而 $h(n)$ 的比重越大，表示启发性能越强。

$g(n)$ 的作用一般是不可忽略的。因为它代表了从初始结点经过 n 结点到达目的结点的总代价估值中实际已付出的那一部分。保持 $g(n)$ 项就保持了搜索的宽度优先成分。这有利于搜索的完备性，但会影响搜索的效率。在特殊情况下，如果只希望找到达到目的结点的路径而不关心会付出什么代价，则 $g(n)$ 的作用可以忽略。另外，当 $h(n) \gg g(n)$ 时，也可忽略 $g(n)$，这时有 $f(n) = h(n)$，有利于提高搜索的效率，但影响搜索的完备性。

给定一个问题后，根据该问题的特性和解的特性，可以有多种方法定义估价函数，用不同的估价函数指导搜索，其效果可以相差很远。因此，必须尽可能地选择最能体现问题特性的、最佳的估价函数。设计估价函数的目标就是利用有限的信息作出一个较精确的估价函数。

例 5.7　八数码问题的估价函数。

八数码问题的估价函数的设计方法有多种，并且不同的估价函数对求解八数码问题有不同

的影响。

① 最简单的估价函数是取一格局与目的格局位置不符的数码数目。直观感觉这种估价函数很有效,因为在其他条件相同的情况下,某格局位置不符的数码数目越少,则它和最终目的越接近,因而它是下一个搜索格局。但是,这种估价函数并没有充分利用所能获得的信息。它没有考虑数码所需移动的距离。

② 另一个较好的估价函数是各数码移到目的位置所需移动的距离的总和。

这两种估价函数都没有考虑数码逆转(与目的格局中数码排列的先后顺序)的情况。如果两块数码相邻但与目标格局相比位置相反,则至少需移动 3 次才能将它们移到正确的位置上。

③ 第三种估价函数对每一对逆转数码乘以一个倍数。

④ 第四种估价函数将位置不符数码数目的总和与 3 倍数码逆转数目相加。它克服了仅计算数码逆转数目策略的局限。

这个例子说明,设计一个好的估价函数具有相当的难度。好的估价函数的设计是一个经验问题,判断和直觉是很重要的因素,但是衡量其好坏的最终标准是在具体应用时的搜索效果。

5.4.3 *A* 搜索算法

启发式图搜索法的基本特点是:如何寻找并设计一个与问题有关的 $h(n)$ 及构造出 $f(n) = g(n) + h(n)$,然后以 $f(n)$ 的大小来排列待扩展状态的次序,每次选择 $f(n)$ 值最小者进行扩展。

和上节中提到的宽度优先和深度优先搜索算法一样,启发式图搜索法算法使用了两张表来记录状态信息:在 open 表中保留所有已生成而未扩展的状态;在 closed 表中记录已扩展过的状态。算法中有一步是根据某些启发信息来排列 open 表的。它既不同于宽度优先所使用的队列(先进先出),也不同于深度优先所使用的堆栈(先进后出),而是一个按状态的启发估价函数值的大小排列的一个表。进入 open 表的状态不是简单地排在队尾(或队首),而是根据其估值的大小插入到表中合适的位置,每次从表中优先取出启发估价函数值最小的状态加以扩展。

A 算法是基于估价函数的一种加权启发式图搜索算法,具体步骤如下:

步骤 1　把附有 $f(S_0)$ 的初始结点 S_0 放入 open 表;

步骤 2　若 open 表为空,则搜索失败,退出;

步骤 3　移出 open 表中第一个结点 N 放入 closed 表中,并顺序编号 n;

步骤 4　若目标结点把附有 $f(S_0)$ 的初始 $S_g = N$,则搜索成功,结束;

步骤 5　若 N 不可扩展,则转步骤 2;

步骤 6　扩展 N,生成一组附有 $f(x)$ 的子结点,对这组子结点作如下处理。

① 考察是否有已在 open 表或 closed 表中存在的结点。若有则再考察其中有无 N 的先辈结点,若有则删除之,对于其余结点也删除之,但由于它们又被第二次生成,因此需要考虑是否修改已经存在于 open 表或 closed 表中的这些结点及其后裔的返回指针和 $f(x)$ 的值,修改原则是:选 $f(x)$ 值小的路走。

② 为其余子结点配上指向 N 的返回指针后放入 open 表中,并对 open 表按 $f(x)$ 值以升序排序,转步骤 2。

启发式图搜索的 *A* 算法描述如下:

```
procedure heuristic_search
```

```
open: = [start];closed: = [ ];f(s): = g(s) + h(s);                    * 初始化
while open≠[ ]do
begin
  从 open 表中删除第一个状态,称之为 n;
  if n = 目的状态 then return(success);
  生成 n 的所有子状态;
  if n 没有任何子状态 then continue;
  for n 的每个子状态 do
    case 子状态 is not already on open 表 or closed 表;
      begin
        计算该子状态的估价函数值;
        将该子状态加到 open 表中;
      end;
    case 子状态 is already on open 表:
      if 该子状态是沿着一条比在 open 表已有的更短路径而到达
        then 记录更短路径走向及其估价函数值;
    case 子状态 is already on closed 表:
      if 该子状态是沿着一条比在 closed 表已有的更短路径而到达
      then
        begin
        将该子状态从 closed 表移到 open 表中;
        记录更短路径走向及其估价函数值;
        end;
    case end;
    将 n 放入 closed 表中;
    根据估价函数值,从小到大重新排列 open 表;
end;                                                                  * open 表中结点已耗尽
return(failure);
```

每次重复时,A 搜索算法从 open 表中取出第一个状态,如果该状态满足目的条件,则算法返回到该状态的搜索路径,在这里,每个状态都保留了其父状态的信息,以保证能返回完整的搜索路径。

如果 open 表的第一个状态不是目的状态,则算法利用与之相匹配的一系列操作算子进行相应的操作来产生它的子状态。如果某个子状态已在 open 表(或 closed 表)中出现过,即该状态再一次被发现时,则通过刷新它的祖先状态的历史记录,使算法极有可能找到到达目的状态的更短的路径。

接着,用 A 搜索算法 open 表中每个状态的估价函数值,按照值的大小重新排序,将值最小的状态放在表头,使其第一个被扩展。

例 5.8　图 5.16 给出了利用 A 搜索算法求解八数码问题的搜索树,解路为 s,B,E,I,K,L。

图 5.16 中状态旁括号内的数字表示该状态的估价函数值,其估价函数定义为

$$f(n) = d(n) + w(n)$$

其中,$d(n)$ 代表状态的深度,每步为单位代价;$w(n)$ 表示以“不在位”的数码数作为启发信息的度量。

搜索过程中 open 表和 closed 表内状态排列的变化情况如表 5.1 所示。

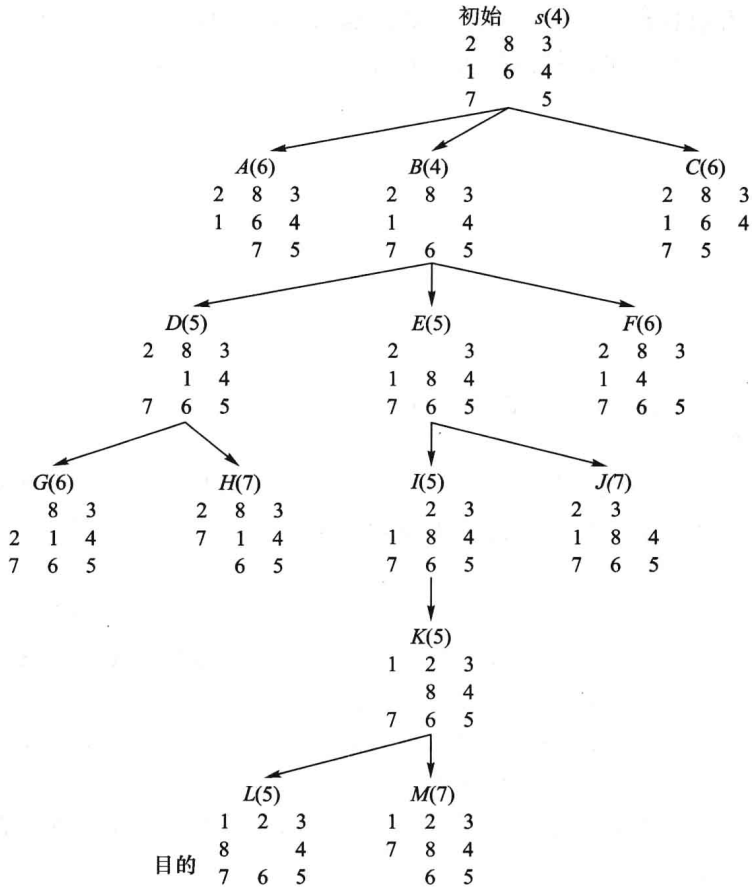


图 5.16 八数码问题的 A 搜索树

表 5.1 搜索过程中 open 表和 closed 表内状态排列的变化情况

open 表	closed 表
初始化:($s(4)$)	( )
一次循环后:	
($B(4)$ $A(6)$ $C(6)$)	($s(4)$)
二次循环后:	
($D(5)$ $E(5)$ $A(6)$ $C(6)$ $F(6)$)	($s(4)$ $B(4)$)
三次循环后:	
($E(5)$ $A(6)$ $C(6)$ $F(6)$ $G(6)$ $H(7)$)	($s(4)$ $B(4)$ $D(5)$)
四次循环后:	
($I(5)$ $A(6)$ $C(6)$ $F(6)$ $G(6)$ $H(7)$ $J(7)$)	($s(4)$ $B(4)$ $D(5)$ $E(5)$)
五次循环后:	
($K(5)$ $A(6)$ $C(6)$ $F(6)$ $G(6)$ $H(7)$ $J(7)$)	($s(4)$ $B(4)$ $D(5)$ $E(5)$ $I(5)$)
六次循环后:	
($L(5)$ $A(6)$ $C(6)$ $F(6)$ $G(6)$ $H(7)$ $J(7)$ $M(7)$)	($s(4)$ $B(4)$ $D(5)$ $E(5)$ $I(5)$ $K(5)$)
七次循环后:	
L 为目的状态,则成功退出,结束搜索	($s(4)$ $B(4)$ $D(5)$ $E(5)$ $I(5)$ $K(5)$ $L(5)$

前面已提到启发信息给得越多即估价函数值越大，则 A 算法需要搜索处理的状态数就越少，其效率就越高。但也不是估价函数值越大越好，因为估价函数值太大会使 A 算法不一定能搜索到最优解。

5.4.4　A^* 搜索算法及其特性分析

定义 $h^*(n)$ 为状态 n 到目的状态的最优路径的代价，则 A 搜索算法的启发函数 $h(n)$ 满足小于等于 $h^*(n)$，即

$$h(n) \leqslant h^*(n), \quad \text{对所有结点 } n \tag{5.4}$$

的条件下，被称为 A^* 搜索算法。

A^* 搜索算法是由著名的人工智能学者 Nilsson 提出的，它是目前最有影响的启发式图搜索算法，也称为最佳图搜索算法。

如果某一问题有解，那么利用 A^* 搜索算法对该问题进行搜索则一定能搜索到解，并且一定能搜索到最优的解。因此，A^* 搜索算法比 A 搜索算法好。它不仅能得到目标解，并且还一定能找到最优解（只要问题有解）。

在例 5.8 所描述的八数码问题中的 $w(n)$ 即为 $h(n)$。它表示了“不在位”的数码数。这个 $w(n)$ 满足了 $h(n) \leqslant h^*(n)$ 的条件，因此，图 5.16 的八数码 A 搜索树也是 A^* 搜索树，所得的解路 (s,B,E,I,K,L) 为最优解路，其步数为状态 $L(5)$ 上所标注的 5。

在一些问题求解中，只要搜索到一个解，就会想到最优解，因此，其搜索效率显得更重要。那么，下面的问题是：是否还有更好的启发策略？在什么意义上称某一启发策略比另一个好？

另外，当通过启发式搜索得到某一状态的路径代价时，是否能保证在以后的搜索中不会出现到达该状态有更小的代价？

就上面这些问题，下面讨论 A^* 搜索算法的有关特性。

1. 可采纳性

当一个搜索算法在最短路径存在时能保证找到它，就称它是可采纳的。

通过估价函数 $f(n)=g(n)+h(n)$，可归纳出一类可采纳性的启发搜索策略的特征。若 n 是状态空间图中的一个状态，$g(n)$ 是衡量某一状态在图中的深度，$h(n)$ 是 n 到目的状态代价的估计值，此时 $f(n)$ 则是从起点出发，通过 n 到达目标的路径的总代价的估计值。

定义最优估价函数

$$f^*(n) = g^*(n) + h^*(n) \tag{5.5}$$

式中，$g^*(n)$ 为起点到 n 状态的最短路径代价值；$h^*(n)$ 是 n 状态到目的状态的最短路径的代价值。这样，$f^*(n)$ 就是起点出发通过 n 状态而到达目的状态的最佳路径的总代价值。

尽管在绝大部分实际问题中并不存在 $f^*(n)$ 这样的先验函数，但可以将 $f(n)$ 作为 $f^*(n)$ 的一个近似估计函数。在 A 及 A^* 搜索算法中，$g(n)$ 作为 $g^*(n)$ 的近似估价。$g(n)$ 与 $g^*(n)$ 可能并不相等，但有 $g(n) \geqslant g^*(n)$。仅当搜索过程已发现了到达 n 状态的最佳路径时，它们才相等。

同样，可以用 $h(n)$ 代替 $h^*(n)$ 作为 n 状态到目的状态的最小代价估计值。虽然在绝大多数情况下无法计算 $h^*(n)$，但是要判别某一 $h(n)$ 是否大于 $h^*(n)$ 还是可能的。如果 A 搜索算法所使用的估计函数 $f(n)$ 能达到 $f(n)$ 中的 $h(n) \leqslant h^*(n)$，则称之为 A^* 搜索算法。

可以证明，所有的 A^* 搜索算法都是可采纳的。

宽度优先算法是 A^* 的一个特例，是一个可采纳的搜索算法。该算法相当于 A^* 算法中取 $h(n)=0$ 和 $f(n)=g(n)+0$。宽度优先搜索时对某一状态只考虑它同起始状态的距离代价。这是由于该算法在考虑 $n+1$ 层状态之前，已考察了 n 层中的任意一种状态，所以每个目的状态都是沿着最短的可能路径而找到的。不幸的是，宽度优先搜索算法的搜索效率太低。

2. 单调性

在 A^* 搜索算法中并不要求 $g(n)=g^*(n)$。这意味着要采纳的启发算法可能会沿着一条非最佳路径搜索到某一中间状态。从 A 搜索算法可看出，在这种情况下，算法需要比较代价、调整路径等，使搜索的效率大大降低。如果对启发函数 $h(n)$ 加上单调性的限制，就可以减少比较代价和调整路径的工作量，从而减少搜索代价。

下面介绍启发函数单调性的概念。

如果某一启发函数满足：

① 对所有状态 n_i 和 n_j，其中 n_j 是 n_i 的后裔，满足 $h(n_i)-h(n_j)\leqslant cost(n_i,n_j)$，其中 $cost(n_i,n_j)$ 是从 n_i 到 n_j 的实际代价。

② 目的状态的启发函数值为 0 或 $h(Goal)=0$。

则称该启发函数 $h(n)$ 是单调的。

搜索算法的单调性可这样描述：在整个搜索空间都是局部可采纳的。一个状态和任一个子状态之间的差由该状态与其子状态之间的实际代价所限定，这就是说，启发策略无论何处都是可采纳的，总是从祖先状态沿着最佳路径到达任一状态。

于是，由于算法总是在第一次发现该点时就已经发现了到达该点状态的最短路径，所以当某一状态被重新搜索时，就无须检验新的路径是否更短，那是不可能的，这就意味着当某一状态被重新搜索时，可以将其立即从 open 表或 closed 表中删除，而无须修改路径的信息。

可以很容易地证明单调性启发策略是可采纳的。这意味着单调性策略中的 $h(n)$，满足 A^* 搜索策略的下界要求，算法是可采纳的。

3. 信息性

何时一个启发策略要比另一个启发策略好？也就是具有更多的启发知识，使其容易找到最短路径，搜索较少的状态，特别是当两种策略都是 A^* 搜索算法时。

在两个 A^* 启发策略的 h_1 和 h_2 中，如果对搜索空间中的任一状态 n 都有 $h_1(n)\leqslant h_2(n)$，就称策略 h_2 比 h_1 具有更多的信息性。如果某一搜索策略的 $h(n)$ 越大，则它所搜索的状态要少得多。

如果启发策略 h_2 的信息性比 h_1 要多，则用 h_2 所搜索的状态集合是 h_1 的一个子集。则算法 A^* 的信息性越多，它所搜索的状态数就越少。必须注意的是，更多的信息性需要更多的计算时间，从而有可能抵消减少搜索空间所带来的益处。

5.5 小结

1. 搜索的概念

在搜索中需要解决是否一定能找到一个解、是否终止运行、找到的解是否最佳解、搜索过程的时间与空间复杂性如何等基本问题。

搜索的方向有正向搜索和逆向搜索。

盲目搜索是在不具有对特定问题的任何有关信息的条件下，按固定的步骤（依次或随机调用操作算子）进行的搜索。

启发式搜索则是考虑特定问题领域可应用的知识，动态地确定调用操作算子的步骤，优先选择较适合的操作算子。

2. 状态空间知识表示方法

状态空间是利用状态变量和操作符号，表示系统或问题的有关知识的符号体系，状态空间是一个四元组(S,O,S_0,G)。

任何类型的数据结构都可以用来描述状态，如符号、字符串、向量、多维数组、树和表格等。

从 S_0 结点到 G 结点的路径被称为求解路径。状态空间的一个解是一个有限的操作算子序列，它使初始状态转换为目标状态。

3. 回溯策略

带回溯策略的搜索是从初始状态出发，不停地、试探性地寻找路径，若它遇到不可解结点就回溯到路径中最近的父结点上，查看该结点是否还有其他的子结点未被扩展。若有，则沿这些子结点继续搜索；如果找到目标，就成功退出搜索，返回解题路径。

回溯是状态空间搜索的一个基本算法。各种图搜索算法，包括深度优先、宽度优先、最好优先搜索等，都有回溯的思想。

4. 宽度优先搜索法

宽度优先搜索法是由 S_0 生成新状态，然后依次扩展这些状态，再生成新状态，该层扩展完后，再进入下一层，如此一层一层地扩展下去，直到搜索到目的状态（如果目的状态存在）。

5. 深度优先搜索法

深度优先搜索法是从 S_0 出发，沿一个方向一直扩展下去，直到达到一定的深度。如果未找到目的状态或无法再扩展时，便回溯到另一条路径继续搜索；若还未找到目的状态或无法再扩展时，再回溯到另一条路径搜索……

6. 启发式图搜索策略

在具体求解中，能够利用与该问题有关的信息来简化搜索过程，称此类信息为启发信息，而称这种利用启发信息的搜索过程为启发式搜索。

A 搜索算法是寻找并设计一个与问题有关的 $h(n)$ 及构造出 $f(n)=g(n)+h(n)$，然后以 $f(n)$ 的大小来排列待扩展状态的次序，每次选择 $f(n)$ 值最小者进行扩展。

定义 $h^*(n)$ 为状态 n 到目的状态的最优路径的代价。对于一具体问题，只要有解，则一定存在 $h^*(n)$。于是，当要求估价函数中的 $h(n)$ 都小于等于 $h^*(n)$ 时，A 搜索算法就成为 A^* 搜索算法。

思考题

5.1　什么是搜索？有哪两大类不同的搜索方法？两者的区别是什么？

5.2　什么是启发式搜索？什么是启发信息？

5.3　用状态空间法表示问题时，什么是问题的解？求解过程的本质是什么？什么是最优

解？最优解唯一吗？

5.4　请写出状态空间图的一般搜索过程。在搜索过程中 open 表和 closed 表的作用分别是什么？有何区别？

5.5　什么是盲目搜索？主要有几种盲目搜索策略？

5.6　在深度优先搜索中，每个结点的子结点是按某种次序生成和扩展的，在决定生成子状态的最优次序时，应该用什么标准来衡量？

5.7　宽度优先搜索与深度优先搜索有何不同？分析深度和宽度优先的优缺点。在何种情况下，宽度优先搜索优于深度优先搜索？在何种情况下，深度优先搜索优于宽度优先搜索？

5.8　什么是 A^* 搜索算法？它的估价函数是如何确定的？A^* 搜索算法与 A 搜索算法的区别是什么？

习题

5.1　修道士和野人的问题。设有 3 个修道士和 3 个野人来到河边，打算用一条船从河的左岸渡到河的右岸。但该船每次只能装载 2 个人，在任何岸边野人的数目都不得超过修道士的人数，否则修道士就会被野人吃掉。假设野人服从任何一种过河安排，如何规划过河计划才能把所有人安全地渡过河去。用状态空间表示法表示修道士和野人的问题，画出状态空间图。

5.2　用状态空间搜索法求解农夫、狐狸、鸡、小米问题。农夫、狐狸、鸡、小米都在一条河的左岸，现在要把它们全部送到右岸去，农夫有一条船，过河时，除农夫外，船上至多能载狐狸、鸡和小米中的一样。狐狸要吃鸡，鸡要吃小米，除非农夫在那里。试规划出一个确保全部安全的过河计划。（提示：a. 用四元组（农夫、狐狸、鸡、小米）表示状态，其中每个元素都可为 0 或 1。0 表示在左岸，1 表示在右岸。b. 每次过河的一种安排作为一个算符，每次过河都必须有农夫，因为只有他可以划船。）

5.3　用有界深度优先搜索方法求解图 5.17 所示八数码难题。初始状态为 S_0，目标状态为 S_g，要求寻找从初始状态到目标状态的路径。

2		8
1	6	3
7	5	4

S_0

1	2	3
8		4
7	6	5

S_g

图 5.17　初始状态和目标状态

第6章 遗传算法及其应用

遗传算法(genetic algorithms, GA)是一类借鉴生物界自然选择和自然遗传机制的随机搜索算法,非常适用于处理传统搜索方法难以解决的复杂和非线性优化问题。遗传算法可广泛应用于组合优化、机器学习、自适应控制、规划设计和人工生命等领域,是21世纪有关智能计算中的重要技术之一。

本章首先详细介绍遗传算法的基本算法,然后介绍双倍体、双种群、自适应等比较典型的改进遗传算法,最后简单介绍了遗传算法在生产调度中的应用。在此基础上,读者可以进一步学习遗传算法以及其他进化算法的内容。

6.1 遗传算法的产生与发展

6.1.1 遗传算法的生物学背景

遗传算法类似于生物进化,需要经过长时间的成长演化,最后收敛到最优化问题的一个或者多个解。因此,了解一些生物进化过程,有助于理解遗传算法的工作过程。

"适者生存"揭示了大自然生物进化过程中的一个规律:最适合自然环境的群体往往产生了更大的后代群体。生物进化的基本过程如图6.1所示。

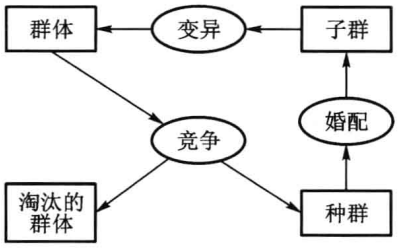


图6.1 生物进化的基本过程

生物的遗传物质的主要载体是染色体(chromosome),DNA是其中最主要的遗传物质,而基因(gene)又是扩展生物性状的遗传物质的功能单元和结构单位。染色体中基因的位置称作基因座(locus),而基因所取的值又叫做等位基因(alleles)。基因和基因座决定了染色体的特征,也决定了生物个体(individual)的性状,如头发的颜色是黑色、棕色或者金黄色等。

以一个初始生物群体(population)为起点,经过竞争后,一部分群体被淘汰而无法再进入这个循环圈,而另一部分则成为种群。竞争过程遵循生物进化中"适者生存,优胜劣汰"的基本规律,所以都有一个竞争标准,或者生物适应环境的评价标准。适应程度高的并不一定进入种群,只是进入种群的可能性比较大。而适应程度低的并不一定被淘汰,只是进入种群的可能性比较

小。这一重要特性保证了种群的多样性。

生物进化中种群经过婚配产生子代群体(简称子群)。在进化的过程中,可能会因为变异而产生新的个体。每个基因编码了生物机体的某种特征,如头发的颜色、耳朵的形状等。综合变异的作用,子群成长为新的群体而取代旧群体。在新的一个循环过程中,新的群体代替旧的群体而成为循环的开始。

6.1.2 遗传算法的基本思想

遗传算法模拟上述生物进化过程求解优化问题。在遗传算法中,染色体对应的是优化问题的解,通常是由一维的串结构数据来表示的。串上各个位置对应上述的基因座,而各位置上所取的值对应上述的等位基因。遗传算法处理的是染色体,或者称为基因型个体。一定数量的个体组成了群体。各个个体对环境的适应程度叫适应度。适应度大的个体被选择进行遗传操作产生新个体,体现了生物遗传中适者生存的原理。选择两个染色体进行交叉产生一组新的染色体的过程,类似生物遗传中的婚配。编码的某一个分量发生变化的过程,类似生物遗传中的变异。

遗传算法包含两个数据转换操作,一个是从表现型到基因型的转换,将搜索空间中的参数或解转换成遗传空间中的染色体或个体,这个过程称为编码(coding)。另一个是从基因型到表现型的转换,即将个体转换成搜索空间中的参数,这个过程称为译码(decode)。

遗传算法在求解问题时从多个解开始,然后通过一定的法则进行逐步迭代以产生新的解。这多个解的集合称为一个种群,记为 $p(t)$。这里 t 表示迭代步,称为演化代。一般情况下,$p(t)$ 中元素的个数在整个演化过程中是不变的,可将群体的规模记为 N。$p(t)$ 中的元素称为个体或染色体,依次记为 $x_1(t),x_2(t),\cdots,x_N(t)$。在进行演化时,要选择当前解进行交叉以产生新解。这些当前解称为新解的父解(parent),产生的新解称为后代解(offspring)。

6.1.3 遗传算法的发展历史

遗传算法的研究兴起于 20 世纪 80 年代末和 90 年代初,但它的历史起源可追溯到 20 世纪 60 年代初期。早期的研究大多以对自然遗传系统的计算机模拟为主,其特点是侧重于对某些复杂操作的研究。虽然其中像自动博弈、生物模拟、模式识别和函数优化等给人以深刻的印象,但总的来说,这是一个无明确目标的发展时期。许多学者并未认识到自然遗传算法可以转化为人工遗传算法,而 Holland 教授和他的学生不久就认识到这一转化的重要性。Holland 认为比起寻找这种或那种具体的求解问题的方法来说,开拓一种能模拟自然选择遗传机制的带有一般性的理论和方法更有意义。在这一时期,Holland 不但发现了基于适应度的人工遗传选择的基本作用,而且还对群体操作等进行了认真的研究。1965 年,他首次提出了人工遗传操作,并把这些应用于自然系统和人工系统中。

1967 年,Bagley 在他的论文中首次提出了遗传算法这一术语,并讨论了遗传算法在博弈中的应用。他所提出的选择、交叉和变异操作与目前遗传算法中的相应操作十分接近,尤其是他对选择做了十分有意义的研究。他认识到,在遗传进化过程的前期和后期,选择概率应合适地变动。为此,他引入了适应度定标概念。他还首次提出了遗传算法自我调整概念,即把交叉和变异的概率融于染色体本身的编码中,从而可使算法自我调整优化。尽管 Bagley 没有进行计算机模拟实验,但这些思想对于后来遗传算法的发展具有重要的意义。

在同一时期，Rosenberg 也对遗传算法进行了研究。他的研究依然是以模拟生物进化为主，但他在遗传操作方面提出了不少独特的设想。1970 年，Cavicchio 把遗传算法应用于模式识别中。实际上他并未直接涉及模式识别，而仅用遗传算法设计一组应用于识别的检测器。Cavicchio 对于遗传操作以及遗传算法的自动调整也做了不少有特色的研究。

1971 年，Hollstien 第一个把遗传算法用于函数优化。他在论文《计算机控制系统中人工遗传自适应方法》中阐述了遗传算法用于数字反馈控制的方法，但实际上，主要讨论了对于二变量函数的优化问题，其中，对于优势基因控制、交叉和变异以及各种编码技术进行了深入的研究。

1975 年是遗传算法研究历史上十分重要的一年。这一年，美国的 Holland 出版了他的专著《自然系统和人工系统的适配》，系统地阐述了遗传算法的基本理论和方法，并提出了对遗传算法的理论研究和发展极为重要的模式理论。同年，DeJong 完成了重要论文《遗传自适应系统的行为分析》，把 Holland 的模式理论与他的计算实验结合起来。这是遗传算法发展中的又一个里程碑。

20 世纪 80 年代以后，遗传算法进入兴盛发展时期，无论是理论研究还是应用研究都成了十分热门的课题。遗传算法广泛应用于自动控制，生产计划，图像处理，机器人等研究领域。1989 年，David Goldbery 出版了著作《遗传算法在搜索、优化及机器学习中的应用》，系统地总结了遗传算法领域的研究工作，给出了大量的应用程序和应用实例，显著地增强了遗传算法的影响力。

在 20 世纪 60 年代，与遗传算法同时出现的还有进化策略、进化规划，三者非常相似，都是基于生物进化机制，因此用“进化计算”作为它们的统称，相应的算法称为“进化算法”。为了区别于传统的符号式人工智能，一般将神经网络、进化计算、模糊系统等称为计算智能。

6.1.4　设计遗传算法的基本内容

（1）编码方案

遗传算法求解问题不是直接作用在问题的解空间上，而是作用于解的某种编码。因此，编码表示方式有时对算法的性能、效率等会产生很大影响。

（2）适应度函数

适应值是对解的质量的一种度量。它通常依赖于解的行为与环境（即种群）的关系，一般以目标函数的形式来表示。解的适应性是进化过程中进行选择的唯一依据。

（3）选择策略

优胜劣汰的选择机制使得适应值大的解有较高的存活概率。这是遗传算法与一般搜索算法的主要区别之一。不同的选择策略对算法的性能也有较大的影响。

（4）控制参数

控制参数主要包括种群的规模、算法执行的最大代数、执行不同遗传操作的概率以及其他一些辅助性的控制参数。

（5）遗传算子

遗传算子是模拟生物基因遗传的操作，从而实现优胜劣汰的进化过程。它主要包括三个基本遗传算子：选择（selection），交叉（crossover），变异（mutation）。

（6）算法的终止准则

由于遗传算法没有利用目标函数的梯度等信息，所以，在演化过程中无法确定个体在解空间的位置，从而无法用传统的方法来判定算法收敛与否以终止算法。常用的办法是预先规定一个

最大的演化代数,或当算法在连续多少代以后解的适应值没有什么明显的改进时,令算法终止。

6.2 遗传算法的基本算法

遗传算法中包含了五个基本要素:参数编码,初始群体的设定,适应度函数的设计,遗传操作设计,控制参数设定。

6.2.1 编码

由于遗传算法不能直接处理问题空间的参数,因此,必须通过编码将要求解的问题表示成遗传空间的染色体或者个体。它们由基因按一定结构组成。由于遗传算法的鲁棒性,对编码的要求并不苛刻。对一个具体的应用问题如何编码是应用遗传算法的首要问题,也是遗传算法应用的难点。事实上,还不存在一种通用的编码方法,特殊的问题往往采用特殊的方法。

1. 位串编码

将问题空间的参数编码为一维排列的染色体的方法,称为一维染色体编码方法。一维染色体编码中最常用的符号集是二值符号集{0,1},即采用二进制编码(binary encoding)。

(1) 二进制编码

二进制编码是用若干三进制数表示一个个体,将原问题的解空间映射到位串空间 $B=\{0,1\}$ 上,然后在位串空间上进行遗传操作。

优点:二进制编码类似于生物染色体的组成,从而使算法易于用生物遗传理论来解释,并使得遗传操作如交叉、变异等很容易实现。另外,采用二进制编码时,算法处理的模式数最多。

缺点:

① 相邻整数的二进制编码可能具有较大的 Hamming 距离。例如,15 和 16 的二进制表示为 01111 和 10000,因此,算法要从 15 改进到 16 则必须改变所有的位。这种缺陷造成了 Hamming 悬崖(Hamming cliffs),将降低遗传算子的搜索效率。

② 二进制编码时,一般要先给出求解的精度。但求解的精度确定后,就很难在算法执行过程中进行调整,从而使算法缺乏微调(fine-tuning)的功能。若在算法一开始就选取较高的精度,那么串长就很大,这样也将降低算法的效率。

③ 在求解高维优化问题时,二进制编码串将非常长,从而使得算法的搜索效率很低。

(2) Gray 编码

Gray 编码是将二进制编码通过一个变换进行转换得到的编码。设二进制串 $\langle\beta_1\beta_2\cdots\beta_n\rangle$ 对应的 Gray 串 $\langle\gamma_1\gamma_2\cdots\gamma_n\rangle$,则从二进制编码到 Gray 编码的变换为

$$\gamma_k=\begin{cases}\beta_1 & k=1\\ \beta_{k-1}\oplus\beta_k & k>1\end{cases}\tag{6.1}$$

式中,⊕表示模 2 的加法。从一个 Gray 串到二进制串的变换为

$$\beta_k=\sum_{i=1}^{k}\gamma_i(\bmod 2)=\begin{cases}\gamma_1 & k=1\\ \beta_{k-1}\oplus\gamma_k & k>1\end{cases}\tag{6.2}$$

Gray 编码的优点是克服了二进制编码的 Hamming 悬崖的缺点。

2. 实数编码

为克服二进制编码的缺点，对问题的变量是实向量的情形，可以直接采用实数编码。

实数编码是用若干实数表示一个个体，然后在实数空间上进行遗传操作。

采用实数表达法不必进行数制转换，可直接在解的表现型上进行遗传操作。从而可引入与问题领域相关的启发式信息来增加算法的搜索能力。近年来，遗传算法在求解高维或复杂优化问题时一般使用实数编码。

3. 多参数级联编码

对于多参数优化问题的遗传算法，常采用多参数级联编码。其基本思想是把每个参数先进行二进制编码得到子串，再把这些子串连成一个完整的染色体。多参数级联编码中的每个子串对应各自的编码参数，所以，可以有不同的串长度和参数的取值范围。

6.2.2　群体设定

由于遗传算法是对群体进行操作的，所以，必须为遗传操作准备一个由若干初始解组成的初始群体。群体设定主要包括两个方面：初始种群的产生和种群规模的确定。

1. 初始种群的产生

遗传算法中初始群体中的个体可以是随机产生的，但最好采用如下策略设定：

① 根据问题固有知识，设法把握最优解所占空间在整个问题空间中的分布范围，然后，在此分布范围内设定初始群体。

② 先随机产生一定数目的个体，然后从中挑选最好的个体加到初始群体中。这种过程不断迭代，直到初始群体中个体数目达到了预先确定的规模。

2. 种群规模的确定

群体中个体的数量称为种群规模。

种群规模影响遗传优化的结果和效率。当种群规模太小时，遗传算法的优化性能一般不会太好，容易陷入局部最优解。而当种群规模太大时，则计算复杂。

种群规模的确定受遗传操作中选择操作的影响很大。模式定理表明：若种群规模为 M，则遗传操作可从这 M 个个体中生成和检测 M^3 个模式，并在此基础上能够不断形成和优化积木块，直到找到最优解。

显然，种群规模越大，遗传操作所处理的模式就越多，产生有意义的积木块并逐步进化为最优解的机会就越高。种群规模太小，会使遗传算法的搜索空间范围有限，因而搜索有可能停止在未成熟阶段，出现未成熟收敛现象使算法陷入局部解。因此，必须保持种群的多样性，即种群规模不能太小。

另一方面，种群规模太大会带来若干弊病：一是群体越大，其适应度评估次数增加，所以计算量也增加，从而影响算法效率；二是群体中个体生存下来的概率大多采用和适应度成比例的方法，当群体中个体非常多时，少量适应度很高的个体会被选择而生存下来，但大多数个体却被淘汰，这会影响配对库的形成，从而影响交叉操作。

6.2.3　适应度函数

遗传算法遵循自然界优胜劣汰的原则，在进化搜索中基本上不用外部信息，而是用适应度值

表示个体的优劣,作为遗传操作的依据。适应度是评价个体优劣的标准。个体的适应度高,则被选择的概率就高,反之就低。适应度函数(fitness function)是用来区分群体中的个体好坏的标准,是算法演化过程的驱动力,是进行自然选择的唯一依据。改变种群内部结构的操作都是通过适应值加以控制的。因此,适应度函数设计非常重要。

在具体应用中,适应度函数的设计要结合求解问题本身的要求而定。一般而言,适应度函数是由目标函数变换得到的。下面讨论将目标函数变换成适应度函数的方法。

1. 将目标函数映射成适应度函数的方法

最直观的方法是直接将待求解优化问题的目标函数作为适应度函数。

若目标函数为最大化问题,则适应度函数为

$$Fit(f(x)) = f(x) \tag{6.3}$$

若目标函数为最小化问题,则适应度函数为

$$Fit(f(x)) = \frac{1}{f(x)} \tag{6.4}$$

2. 适应度函数的尺度变换

在遗传算法中,将所有妨碍适应度值高的个体产生,从而影响遗传算法正常工作的问题统称为欺骗问题(deceptive problem)。

在设计遗传算法时,群体的规模一般在几十至几百,与实际物种的规模相差很远。因此,个体繁殖数量的调节在遗传操作中就显得比较重要。如果群体中出现了超级个体,即该个体的适应值大大超过群体的平均适应值,则按照适应值比例进行选择时,该个体很快就会在群体中占有绝对的比例,从而导致算法较早地收敛到一个局部最优点,这种现象称为过早收敛。这是一种欺骗问题。在这种情况下,应该缩小这些个体的适应度,以降低这些超级个体的竞争力。另一方面,在搜索过程的后期,虽然群体中存在足够的多样性,但群体的平均适应值可能会接近群体的最优适应值。在这种情况下,群体中实际上已不存在竞争,从而搜索目标也难以得到改善,出现了停滞现象。这也是一种欺骗问题。在这种情况下,应该改变原始适应值的比例关系,以提高个体之间的竞争力。

对适应度函数值域的某种映射变换称为适应度函数的尺度变换(fitness scaling)或者定标。

(1) 线性变换

设原适应度函数为f,定标后的适应度函数为f',则线性变换可采用下式表示

$$f' = af + b \tag{6.5}$$

式中,系数a和b可以有多种途径设定,但要满足两个条件:

① 变换后的适应度的平均值f'_{avg}要等于原适应度平均值f_{avg},以保证适应度为平均值的个体在下一代的期望复制数为1,即

$$f'_{avg} = f_{avg} \tag{6.6}$$

② 变换后适应度函数的最大值$f'_{\max}$要等于原适应度函数平均值f_{avg}的指定倍数,以控制适应度最大的个体在下一代中的复制数。

$$f'_{\max} = C_{mult} \cdot f_{avg} \tag{6.7}$$

式中,C_{mult}是为得到所期待的最优群体个体的复制数。实验表明,对于不太大的群体($n = 50 \sim 100$),C_{mult}可在1.2~2.0范围内取值。

根据上述条件，可以确定线性变换的系数。

$$a = \frac{(C_{mult} - 1)f_{avg}}{f_{\max} - f_{avg}} \tag{6.8a}$$

$$b = \frac{(f_{\max} - C_{mult} f_{avg})f_{avg}}{f_{\max} - f_{avg}} \tag{6.8b}$$

线性变换法变换了适应度之间的差距，保持了种群的多样性，计算简便，易于实现。如果种群里某些个体适应度远远低于平均值时，有可能出现变换后适应度值为负的情况。为满足最小适应度值非负的条件，可以进行如下变换：

$$a = \frac{f_{avg}}{f_{avg} - f_{\min}} \tag{6.9a}$$

$$b = \frac{-f_{\min} f_{avg}}{f_{avg} - f_{\min}} \tag{6.9b}$$

（2）非线性变换

幂函数变换法变换公式为

$$f' = f^K \tag{6.10}$$

式中，幂指数 K 与求解问题有关，而且在算法过程中可按需要修正。

指数变换法变换公式为

$$f' = e^{-af} \tag{6.11}$$

这种变换方法的基本思想来源于模拟退火过程，式中的系数 a 决定了复制的强制性，其值越小，复制的强制性就越趋向于那些具有最大适应度的个体。

6.2.4　选择

选择操作也称为复制（reproduction）操作是从当前群体中按照一定概率选出优良的个体，使它们有机会作为父代繁殖下一代子孙。判断个体优良与否的准则是各个个体的适应度值。显然这一操作借用了达尔文适者生存的进化原则，即个体适应度越高，其被选择的机会就越多。

需要注意的是：如果总挑选最好的个体，遗传算法就变成了确定性优化方法，使种群过快地收敛到局部最优解。如果只作随机选择，则遗传算法就变成完全随机方法，需要很长时间才能收敛，甚至不收敛。因此，选择方法的关键是找一个策略，既要使得种群较快地收敛，也要维持种群的多样性。

选择操作的实现方法很多。这里，介绍几种常用的选择方法。

1. 个体选择概率分配方法

在遗传算法中，哪个个体被选择进行交叉是按照概率进行的。适应度大的个体被选择的概率大，但不是说一定能够选上。同样，适应度小的个体被选择的概率小，但也可能被选上。所以，首先要根据个体的适应度确定被选择的概率。个体选择概率的常用分配方法有以下两种：

（1）适应度比例方法

适应度比例方法（fitness proportional model）也称为蒙特卡罗法（Monte Carlo），是目前遗传算法中最基本也是最常用的选择方法。适应度比例法中，各个个体被选择的概率和其适应度值成比例。设群体规模大小为 M，个体 i 的适应度值为 f_i，则这个个体被选择的概率为

$$p_{si} = \frac{f_i}{\sum_{i=1}^{M} f_i} \tag{6.12}$$

(2) 排序方法

排序方法(rank-based model)是计算每个个体的适应度后,根据适应度大小顺序对群体中个体进行排序,然后把事先设计好的概率按排序分配给个体,作为各自的选择概率。选择概率仅仅取决于个体在种群中的序位,而不是实际的适应度值。排在前面的个体有较多的被选择的机会。

它的优点是克服了适应值比例选择策略的过早收敛和停滞现象,而且对于极大值或极小值问题,不需要进行适应值的标准化和调节,可以直接使用原始适应值进行排名选择。排序方法比比例方法具有更好的鲁棒性,是一种比较好的选择方法。

① 线性排序。线性排序选择最初由 J. E. Baker 提出,他首先假设群体成员按适应值大小从好到坏依次排列为 $x_1, x_2, \cdots, x_M$,然后根据一个线性函数给第 i 个个体 x_i 分配选择概率 p_i,即

$$p_i = \frac{a - bi}{M(M+1)} \tag{6.13}$$

式中,a,b 是常数。然后按类似于转盘式选择的方式选择父体以进行遗传操作。

② 非线性排序。Z. Michalewicz 提出将群体成员按适应值从好到坏依次排列,并按下式分配选择概率。

$$p_i = \begin{cases} q(1-q)^{i-1} & i = 1, 2, \cdots, M-1 \\ (1-q)^{M-1} & i = M \end{cases} \tag{6.14}$$

式中,i 为个体排序序号。q 是一个常数,表示最好的个体的选择概率。

也可使用其他非线性函数来分配选择概率 p_i,只要满足以下条件:

a. 若 $P = \{x_1, x_2, \cdots, x_M\}$ 且 $f(x_1) \geqslant f(x_2) \geqslant \cdots \geqslant f(x_M)$,则分配的概率 p_i 满足 $p_1 \geqslant p_2 \geqslant \cdots \geqslant p_M$。

b. $\sum_{i=1}^{M} p_i = 1$。

2. 选择个体方法

选择操作是根据个体的选择概率确定哪些个体被选择进行交叉、变异等操作。基本的选择方法如下。

(1) 轮盘赌选择

轮盘赌选择(roulette wheel selection)策略在遗传算法中使用得最多。

在轮盘赌选择方法中先按个体的选择概率产生一个轮盘,轮盘每个区的角度与个体的选择概率成比例,然后产生一个随机数,它落入转盘的哪个区域就选择相应的个体交叉。

显然,选择概率大的个体被选中的可能性大,获得交叉的机会就大。

在实际计算时,可以按照个体顺序求出每个个体的累积概率,然后产生一个随机数,它落入累积概率的哪个区域就选择相应的个体交叉。例如,表 6.1 所示 11 个个体的适应度、选择概率和累积概率。为了选择交叉个体,需要进行多轮选择。例如,第 1 轮产生一个随机数为 0.81,落在第 5 个和第 6 个个体之间,则第 6 个个体被选中。第 2 轮产生一个随机数为 0.32,落在第 1 个和第 2 个个体之间,则第 2 个个体被选中。依此类推。

表 6.1　个体适应度、选择概率和累积概率

个体	1	2	3	4	5	6	7	8	9	10	11
适应度	2.0	1.8	1.6	1.4	1.2	1.0	0.8	0.6	0.4	0.2	0.1
选择概率	0.18	0.16	0.15	0.13	0.11	0.09	0.07	0.05	0.03	0.02	0.01
累积概率	0.18	0.34	0.49	0.62	0.73	0.82	0.89	0.94	0.97	0.99	1.00

(2) 锦标赛选择方法

锦标赛选择方法(tournament selection model)是从群体中随机选择 k 个个体,将其中适应度最高的个体保存到下一代。这一过程反复执行,直到保存到下一代的个体数达到预先设定的数量为止。参数 k 称为竞赛规模。

锦标赛选择方法的优点是克服了基于适应值比例选择和基于排名的选择在群体规模很大时,其额外计算量(如计算总体适应值或排序)很大的问题。它常常比轮盘赌选择得到更加多样化的群体。

显然,这种方式也使得适应值好的个体具有较大的生存机会。同时,由于它只使用适应值的相对值作为选择的标准,而与适应值的数值大小不成直接比例,从而也能避免超级个体的影响,一定程度上避免了过早收敛和停滞现象的发生。

作为锦标赛选择方法的一种特殊情况,随机竞争方法(stochastic tournament)是每次按轮盘赌选择方法选取一对个体,然后让这两个个体进行竞争,适应度高者获胜。如此反复,直到选满为止。

(3) 最佳个体保存方法

最佳个体保存方法或称为精英选拔方法(elitist model)是把群体中适应度最高的一个或者多个个体不进行交叉而直接复制到下一代中,保证遗传算法终止时得到的最后结果一定是历代出现过的最高适应度的个体。使用这种方法能够明显提高遗传算法的收敛速度,但可能使种群过快收敛,从而只找到局部最优解。实验结果表明:保留种群个体总数的 2% ~5% 的适应度最高的个体,效果最为理想。

在使用其他选择方法时,一般都同时使用最佳个体保存方法,以保证不会丢失最优个体。

6.2.5　交叉

当两个生物机体配对或者复制时,它们的染色体相互混合,产生一个由双方基因组成的全新的染色体组。这一过程称为重组(recombination)或者交叉(crossover)。

交叉得到的后代可能继承了上代的优良基因,后代会比它们父母更加优秀,但也可能继承了上代的不良基因,后代则会比它们父母差,难以生存,甚至不能再复制自己。越能适应环境的后代越能继续复制自己并将其基因传给后代。由此形成一种趋势:每一代总是比其父母一代生存和复制得更好。

这里举个简单的例子。假设雌性动物仅仅青睐大眼睛的雄性。这样眼睛尺寸越大的雄性越受到雌性的青睐,生出更多的后代。可以说动物的适应性正比于它的眼睛的直径。因此,从一个具有不同大小眼睛的雄性群体出发,当动物进化时,在同位基因中,能够产生大眼睛雄性动物的

基因，相对于产生小眼睛雄性动物的基因，就更有可能复制到下一代。当进化几代以后，大眼睛雄性群体将会占据优势。生物逐渐向一种特殊遗传类型收敛。

遗传算法中起核心作用的是交叉算子，也称为基因重组(recombination)。采用的交叉方法应能够使父串的特征遗传给子串。子串应能够部分或者全部地继承父串的结构特征和有效基因。

1. 基本的交叉算子

(1) 一点交叉

一点交叉(single-point crossover)又称为简单交叉。其具体操作是：在个体串中随机设定一个交叉点，实行交叉时，该点前或后的两个个体的部分结构进行互换，并生成两个新的个体。

(2) 二点交叉

二点交叉(two-point crossover)的操作与一点交叉类似，只是设置了两个交叉点(仍然是随机设定)，将两个交叉点之间的码串相互交换。类似于二点交叉，可以采用多点交叉(multiple-point crossover)。

2. 修正的交叉方法

由于交叉，可能出现不满足约束条件的非法染色体。为解决这一问题，可以采取构造惩罚函数的方法，但试验效果不佳，使本已复杂的问题更加复杂。另一种处理方法是对交叉、变异等遗传操作作适当的修正，使其自动满足优化问题的约束条件。例如，在 TSP 问题中采用部分匹配交叉(partially matched crossover，PMX)，顺序交叉(order crossover，OX)和循环交叉(cycle crossover，CX)等。这些方法对于其他一些问题也同样适用。

下面简单介绍部分匹配交叉 PMX。PMX 是由 Goldberg D. E. 和 R. Lingle(1985)提出的。在 PMX 操作中，先依据均匀随机分布产生两个位串交叉点，定义这两点之间的区域为一匹配区域，并使用位置交换操作交换两个父串的匹配区域。例如，在任务排序问题中，两父串及匹配区域为

$$A = 9\ \ 8\ \ 4\ \ |\ \ 5\ \ 6\ \ 7\ \ |\ \ 1\ \ 3\ \ 2$$

$$B = 8\ \ 7\ \ 1\ \ |\ \ 2\ \ 3\ \ 9\ \ |\ \ 5\ \ 4\ \ 6$$

首先交换 A 和 B 的两个匹配区域，得到

$$A' = 9\ \ 8\ \ 4\ \ |\ \ 2\ \ 3\ \ 9\ \ |\ \ 1\ \ 3\ \ 2$$

$$B' = 8\ \ 7\ \ 1\ \ |\ \ 5\ \ 6\ \ 7\ \ |\ \ 5\ \ 4\ \ 6$$

显然，A'和 B'中出现重复的任务，所以是非法的调度。解决的方法是将 A'和 B'中匹配区域外出现的重复任务，按照匹配区域内的位置映射关系进行交换，从而使排列成为可行调度。

$$A'' = 7\ \ 8\ \ 4\ \ |\ \ 2\ \ 3\ \ 9\ \ |\ \ 1\ \ 6\ \ 5$$

$$B'' = 8\ \ 9\ \ 1\ \ |\ \ 5\ \ 6\ \ 7\ \ |\ \ 2\ \ 4\ \ 3$$

交叉概率是用来确定两个染色体进行局部的互换以产生两个新的子代的概率。采用较大的交叉概率 P_c 可以增强遗传算法开辟新的搜索区域的能力，但高性能模式遭到破坏的可能性增加。采用太低的交叉概率会使搜索陷入迟钝状态。P_c 一般取为 0.25 ~ 1.00。实验表明交叉概率通常取为 0.7 左右是理想的。每次从群体中选择两个染色体，同时生成 0 和 1 之间的一个随机数，然后根据这个随机数确定这两个染色体是否需要交叉。如果这个随机数低于交叉概率(0.7)，就进行交叉。然后沿着染色体的长度随机地选择一个位置，并把此位置之后的所有的位进行互换。

6.2.6 变异

如果生物繁殖仅仅是上述交叉过程，那么即使经历成千上万代以后，适应能力最强的成员的

眼睛尺寸也只能像初始群体中的最大眼睛一样。而根据对自然的观察中可以看到，人类的眼睛尺寸实际存在一代比一代大的趋势。这是因为当基因传递给子孙后代的过程中，会有很小的概论发生差错，从而使基因发生微小的改变，这就是基因的变异。发生变异的概率通常都很小，但在经历许多代以后变异就会很明显。一些变异对生物是不利的，另一些对生物的适应性可能没有影响，但也有一些可能会给生物带来一些好处，使它们超过其他同类的生物。例如前面的例子，变异可能会产生眼睛更大的生物。当经历许多代以后，眼睛会越来越大。

进化机制除了能够改进已有的特征，也能够产生新的特征。例如，可以设想某个时期动物没有眼睛，而是靠嗅觉和触觉来躲避捕食它们的动物。然而，两个动物有次交配时一个基因突变发生在它们后代的头部皮肤上，发育出了一个具有光敏效应的细胞，使它们的后代能够识别周围环境是亮还是暗的。它能够感知捕食者的到来，能够得到现在是白天还是夜晚，它在地上还是地下等信息，有利于它的生存。这个光敏细胞会进一步突变，逐渐形成为一个区域，从而成为眼睛。

在遗传算法中，变异是将个体编码中的一些位进行随机变化。变异的主要目的是维持群体的多样性，为选择、交叉过程中可能丢失的某些遗传基因进行修复和补充。变异算子的基本内容是对群体中的个体串的某些基因座上的基因值作变动。变异操作是按位进行的，即把某一位的内容进行变异。变异概率是在一个染色体中按位进行变化的概率。主要变异方法有：

（1）位点变异

位点变异是指对群体中的个体码串，随机挑选一个或多个基因座，并对这些基因座的基因值以变异概率 P_m 作变动。对于二进制编码的个体来说，若某位原为 0，则通过变异操作就变成了 1，反之亦然。对于整数编码，将被选择的基因变为以概率选择的其他基因。为了消除非法性，将其他基因所在的基因座上的基因变为被选择的基因。

（2）逆转变异

在个体码串中随机选择两点（称为逆转点），然后将两个逆转点之间的基因值以逆向排序插入到原位置中。

（3）插入变异

在个体码串中随机选择一个码，然后将此码插入随机选择的插入点中间。

（4）互换变异

随机选取染色体的两个基因进行简单互换。

（5）移动变异

随机选取一个基因，向左或者向右移动一个随机位数。

在遗传算法中，变异属于辅助性的搜索操作。变异概率 P_m 一般不能大，以防止群体中重要的、单一的基因可能被丢失。事实上，变异概率太大将使遗传算法趋于纯粹的随机搜索。通常取变异概率 P_m 为 0.001 左右。

6.2.7　遗传算法的一般步骤

综上所述，遗传算法步骤如下：

Step 1　使用随机方法或者其他方法，产生一个有 N 个染色体的初始群体 $pop(1)$，$t:=1$；

Step 2　对群体 $pop(t)$ 中的每一个染色体 $pop_i(t)$，计算它的适应值

$$f_i = fitness(pop_i(t))$$

Step 3　若满足停止条件，则算法停止；否则，以概率

$$p_i = f_i / \sum_{j=1}^{N} f_j$$

从 $pop(t)$ 中随机选择一些染色体构成一个新种群

$$newpop(t+1) = \{pop_j(t) \mid j = 1,2,\cdots,N\}$$

Step 4　以概率 P_c 进行交叉产生一些新的染色体，得到一个新的群体

$$crosspop(t+1)$$

Step 5　以一个较小的概率 P_m 使染色体的一个基因发生变异，形成 $mutpop(t+1)$；$t := t+1$，成为一个新的群体 $pop(t) = mutpop(t+1)$；返回 Step 2。

遗传算法的基本流程如图 6.2 所示。

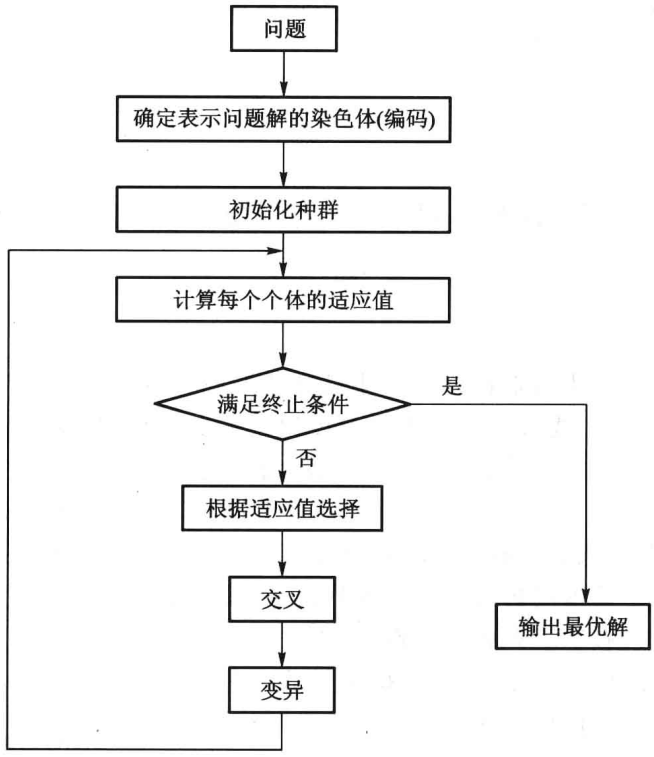


图 6.2　遗传算法的基本流程

6.2.8　遗传算法的特点

遗传算法比起其他普通的优化搜索，采用了许多独特的方法和技术。归纳起来，主要有以下几个方面：

① 遗传算法的编码操作使得它可以直接对结构对象进行操作。所谓结构对象泛指集合、序列、矩阵、树、图、链和表等各种一维、二维、甚至三维结构形式的对象。因此，遗传算法具有非常广泛的应用领域。

② 遗传算法是一个利用随机技术来指导对一个被编码的参数空间进行高效率搜索的方法，而不是无方向的随机搜索。这与其他随机搜索是不同的。

③ 许多传统搜索方法都是单解搜索算法，即通过一些变动规则，将问题的解从搜索空间中的当前解移到另一解。对于多峰分布的搜索空间，这种点对点的搜索方法常常会陷于局部的某个单峰的优解。而遗传算法采用群体搜索策略，即采用同时处理群体中多个个体的方法，同时对搜索空间中的多个解进行评估，从而使遗传算法具有较好的全局搜索性能，减少了陷于局部优解的风险，但还是不能保证每次都得到全局最优解。遗传算法本身也十分易于并行化。

④ 在基本遗传算法中，基本上不用搜索空间的知识或其他辅助信息，而仅用适应度函数值来评估个体，并在此基础上进行遗传操作，使种群中个体之间进行信息交换。特别是遗传算法的适应度函数不仅不受连续可微的约束，而且其定义域也可以任意设定。对适应度函数的唯一要求是能够算出可以比较的正值。遗传算法的这一特点使它的应用范围大大扩展，非常适合于传统优化方法难以解决的复杂优化问题。

6.3　遗传算法的改进算法

为了改进遗传算法的优化性能，人们提出了许多改进算法。下面介绍几种主要的改进算法。

6.3.1　双倍体遗传算法

1. 基本思想

Holland 提出的遗传算法通常被称作基本遗传算法（simple genetic algorithms，SGA）。SGA 是一种“单倍体遗传”，每个基因型由一条染色体组成。自然界中一些简单的植物采用这种遗传，而大多数动物和高级植物都采用双倍体遗传，即每个基因型由一对染色体组成。

双倍体遗传算法采用显性和隐性两个染色体同时进行进化。因此，双倍体遗传算法提供了一种记忆以前有用的基因块的功能：在某些低适应度染色体中，其局部基因块十分有用，是最优解中的基因片段，但由于基因块在当前染色体中的位置不适等因素，导致当前染色体的适应度值不高。保留这些基因块，有利于提高物种的适应能力。

双倍体遗传算法采用显性遗传，如图 6.3 所示。当一对染色体对应的基因块不同时，显性基因遗传给后代，图中大写字母为显性基因。

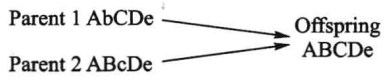


图 6.3　显性遗传

这种双倍体遗传延长了有用基因块的寿命，提高了算法的收敛能力，并且在变异概率低的情况下，也能保持一定水平的多样性。Goldberg 用动态 Knapsack 问题进行了比较研究，实验表明双倍体比单倍体的动态跟踪能力强。

2. 双倍体遗传算法的设计

（1）编码/解码

对于双倍体遗传算法，群体中的每个个体具有两个染色体，一个是显性染色体，一个是隐性染色体。每个染色体的编码/解码方式与基本的遗传算法相同。

（2）复制算子

当进行复制操作时，计算显性染色体的适应度，按照显性染色体的复制概率将个体复制到下

一代群体当中。

(3) 交叉算子

对于交叉操作,从群体中选取两个个体,两个个体的显性染色体进行交叉操作,隐性染色体也同时进行交叉操作。显性交叉率和隐性交叉率可以不同,为了工程实现方便也可将显性交叉率和隐性交叉率设为相同,然后进入下一代群体。

(4) 变异算子

对于变异操作,个体的显性染色体按照正常的变异概率 P_m 执行变异操作,而个体的隐性染色体则按照较大的变异概率执行变异操作。

(5) 双倍体遗传算法显隐性重排算子

当三个遗传算子都执行完成以后,将个体的染色体显隐性进行重新排定,个体中适应度较大的染色体设定为显性染色体,适应度较小的染色体设定为隐性染色体。

双倍体遗传算法程序流程图如图 6.4 所示。

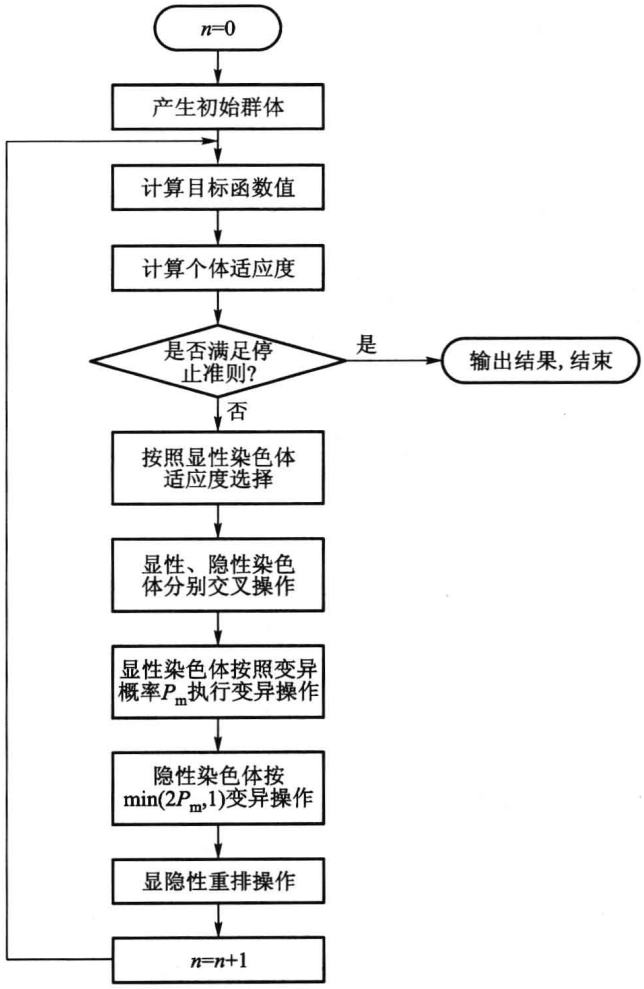


图 6.4 双倍体遗传算法程序流程图

6.3.2　双种群遗传算法

1. 基本思想

基本遗传算法针对一个宏观的种群进行复制、交叉、变异三种操作，类似于人类进化过程，一群人随着时间的推移而不断地进化，并具备越来越多的优良品质。然而由于他们的生长、演化、环境和原始祖先的局限性，经过相当长的时间后，他们将逐渐进化到某些特征相对优势的状态，称之为平衡态。当一个种群进化到这种状态，这个种群的特性就不会再有很大的变化。为了解决这个问题，多种群遗传算法是使用多种群同时进化，并交换种群之间优秀个体所携带的遗传信息，以打破种群内的平衡态达到更高的平衡态，有利于算法跳出局部最优。就本质而言，多种群遗传算法是一种并行算法，可以提高算法的效率。

下面介绍双种群遗传算法。

2. 双种群遗传算法的设计

建立两个遗传算法群体，分别独立地运行复制、交叉、变异操作，同时当每一代运行结束以后，选择两个种群中的随机个体及最优个体分别交换。

(1) 编码/解码设计

编码/解码方法与基本遗传算法相同。

(2) 交叉算子、变异算子

两个种群分别进行选择、交叉、变异等操作，且交叉概率、变异概率不同。

(3) 杂交算子

设种群 A 与种群 B，当 A 与 B 种群都完成了选择、交叉、变异算子后，产生一个随机数 num，随机选择 A 中 num 个个体与 A 中最优个体，随机选择 B 中 num 个个体与 B 中最优个体，交换两者，以打破平衡态。

双种群遗传算法程序流程图如图 6.5 所示。

6.3.3　自适应遗传算法

1. 基本思想

遗传算法的交叉概率 P_c 和变异概率 P_m 的选择是影响遗传算法行为和性能的关键参数，直接影响算法的收敛性。P_c 越大，新个体产生的速度就越快；然而，P_c 过大时遗传模式被破坏的可能性也越大，使得具有高适应度的个体结构很快被破坏；但是如果 P_c 过小，会使搜索过程缓慢，以致停滞不前。对于变异概率 P_m，如果 P_m 过小，就不易产生新的个体结构；如果 P_m 取值过大，则遗传算法变成了纯粹的随机搜索算法。针对不同的优化问题，需要反复实验来确定 P_c 和 P_m。这是一件烦琐的工作，而且很难找到适应于每个问题的最佳值。

Srinvivas M，Patnaik L M 等在 1994 年提出一种自适应遗传算法(adaptive genetic algorithms，AGA)。自适应遗传算法是使交叉概率 P_c 和变异概率 P_m 能够随适应度变化自动改变。当种群各个体适应度趋于一致或者趋于局部最优时，使 P_c 和 P_m 增加，以跳出局部最优；而当群体适应度比较分散时，使 P_c 和 P_m 减少，以利于优良个体的生存。同时，对于适应度高于群体平均适应度的个体，选择较低的 P_c 和 P_m，使得该解得以保护进入下一代；对低于平均适应度的个体，选择较高的 P_c 和 P_m 值，使该解被淘汰。因此，自适应的 P_c 和 P_m 能够提供相对某个解的最佳 P_c 和

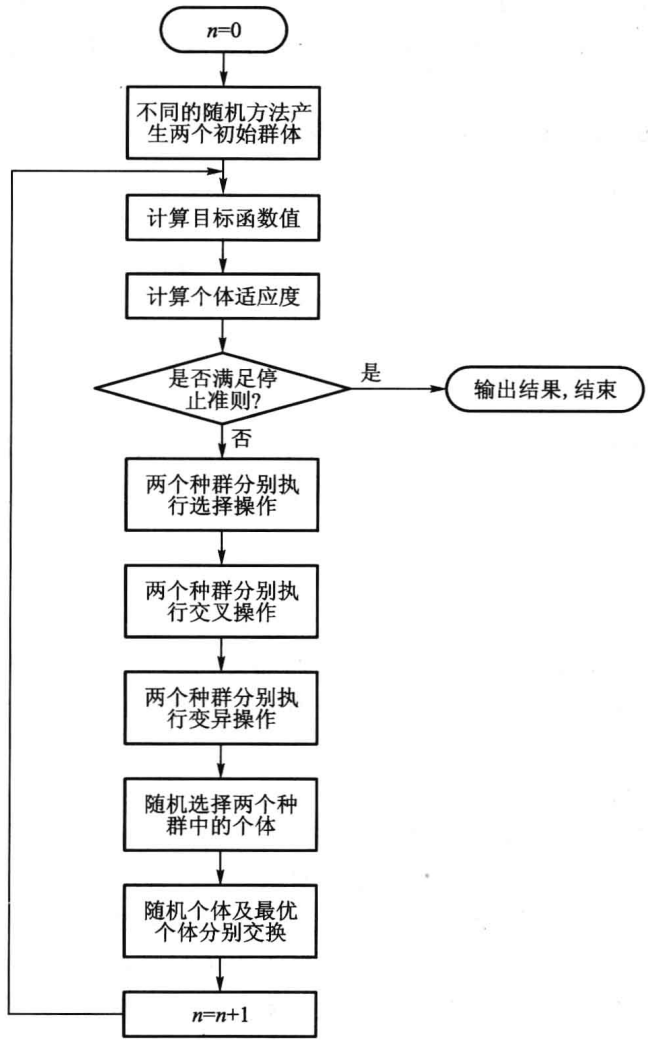


图 6.5 双种群遗传算法程序流程图

P_m。自适应遗传算法在保持群体多样性的同时,保证遗传算法的收敛性。

2. 自适应遗传算法的步骤

Step1 编码/解码设计同基本遗传算法。

Step2 用初始种群产生的一些方法,产生 N(N 是偶数)个候选解,组成初始解集。

Step3 定义适应度函数为 $f=1/ob$,计算适应度 f_i。

Step4 按照轮盘赌规则选择 N 个个体,计算群体的平均适应度 f_{avg} 和最大适应度 f_{max}。

Step5 将群体中的各个个体随机搭配成对,共组成 $N/2$ 对,对每一对个体,按照自适应公式计算自适应交叉概率 P_c,以 P_c 为交叉概率进行交叉操作,即随机产生 $R(0,1)$,如果 $R<P_c$ 则对该对染色体进行交叉操作。

Step6 对于群体中的所有个体,共 N 个,按照自适应变异公式计算自适应变异概率 P_m,以 P_m 为变异概率进行变异操作,即随机产生 $R(0,1)$,如果 $R<P_m$ 则对该染色体进行交叉操作。

Step7　计算由交叉和变异生成新个体的适应度，新个体与父代一起构成新群体。

Step8　判断是否达到预定的迭代次数，是则结束寻优过程，否则转 Step4。

3. 自适应的交叉概率与变异概率

在自适应遗传算法中，P_c 和 P_m 按照以下公式进行调整。

$$P_c = \begin{cases} \dfrac{k_1(f_{\max} - f')}{f_{\max} - f_{avg}}, & f' > f_{avg} \\ k_2, & f' \leqslant f_{avg} \end{cases} \tag{6.15}$$

$$P_m = \begin{cases} \dfrac{k_3(f_{\max} - f)}{f_{\max} - f_{avg}}, & f > f_{avg} \\ k_4, & f \leqslant f_{avg} \end{cases} \tag{6.16}$$

式中，$f_{\max}$ 是群体中最大的适应度值；f_{avg} 是每代群体的平均适应度值；f' 是要交叉的两个个体中较大的适应度值；f 是要变异的个体的适应度值。k_1, k_2, k_3, k_4 是(0,1)之间的常数。

6.4　遗传算法的应用

下面以生产调度这个典型的大规模优化求解问题为例，介绍遗传算法的应用。由于生产调度问题的解容易进行编码，而且，遗传算法可以处理大规模问题，所以，遗传算法成为求解生产调度问题的重要方法之一。

1. 流水车间调度问题

流水车间调度问题(flow-shop scheduling problem，FSP)是与城市不对称情况下的旅行商问题难度相当的同一类型的 NP 完全问题中最困难的问题之一。自从 Johnson 在 1954 年发表第一篇关于 FSP 的文章以来，FSP 引起了许多学者的关注。整数规划和分支定界法是寻求最优解的常用方法，但 FSP 是 NP 完全问题，对于一些大规模甚至中等规模的问题，整数规划和分支定界方法仍是很困难的。在多数情况下很难用数学方法求解生产调度问题，数学计算和智能算法的结合往往是有效的。

FSP 一般可以描述为 n 个工件要在 m 台机器上加工，每个工件需要经过 m 道工序，每道工序要求不同的机器，n 个工件在 m 台机器上的加工顺序相同。工件在机器上的加工时间是给定的，设为 $t_{ij}(i=1,\cdots,n;j=1,\cdots,m)$。问题的目标是确定 n 个工件在每台机器上的最优加工顺序，使最大流程时间达到最小。

对该问题常常作如下假设：

① 每个工件在机器上的加工顺序是给定的。

② 每台机器同时只能加工一个工件。

③ 一个工件不能同时在不同的机器上加工。

④ 工序不能预定。

⑤ 工序的准备时间与顺序无关，且包含在加工时间中。

⑥ 工件在每台机器上的加工顺序相同，且是确定的。

令 $c(j_i,k)$ 表示工件 j_i 在机器 k 上的加工完工时间，$\{j_1,j_2,\cdots,j_n\}$ 表示工件的调度，那么，对于

无限中间存储方式，n 个工件、m 台机器的流水车间调度问题的完工时间可表示为

$$
\begin{aligned}
&c(j_1,1) = t_{j_1 1} \\
&c(j_1,k) = c(j_1,k-1) + t_{j_1 k} k = 2,\cdots,m \\
&c(j_i,1) = c(j_{i-1},1) + t_{j_i 1} i = 2,\cdots,n \\
&c(j_i,k) = \max\{c(j_{i-1},k),c(j_i,k-1)\} + t_{j_i k} \\
&\qquad i = 2,\cdots,n;k = 2,\cdots,m
\end{aligned}
\tag{6.17}
$$

最大流程时间为

$$c_{\max} = c(j_n,m) \tag{6.18}$$

调度目标为确定$\{j_1,j_2,\cdots,j_n\}$，使得 $c_{\max}$最小。

2. 求解 FSP 的遗传算法设计

由于遗传算法所固有的全局搜索与收敛特性，由它得到的次优解往往优于传统方法得到的局部极值解，加之搜索效率比较高，因而被认为是一种切实有效的方法，得到了日益广泛的研究。

下面介绍遗传算法求解 FSP 的编码与适应度函数的设计。

(1) FSP 的编码方法

对于调度问题，通常不采用二进制编码，而使用实数编码。将各个生产任务编码为相应的整数变量。一个调度方案是生产任务的一个排列，其排列中每个位置对应于每个带编号的任务。GA 算法根据一定评价函数，求出最优的排列。

对于 FSP，最自然的编码方式是用染色体表示工件的顺序，例如，对于有四个工件的 FSP，第 k 个染色体 $v_k = [1,2,3,4]$，表示工件的加工顺序为：j_1,j_2,j_3,j_4。

(2) FSP 的适应度函数

令 $c_{\max}^k$表示 k 个染色体 v_k 的最大流程时间，那么，FSP 的适应度函数取为

$$eval(v_k) = \frac{1}{c_{\max}^k} \tag{6.19}$$

3. 求解 FSP 的遗传算法实例

例 6.1 由 Ho 和 Chang(1991)给出的 5 个工件、4 台机器问题的加工时间如表 6.2 所示。

表 6.2 加工时间表

工件 j	t_{j_1}	t_{j_2}	t_{j_3}	t_{j_4}
1	31	41	25	30
2	19	55	3	34
3	23	42	27	6
4	13	22	14	13
5	33	5	57	19

为了便于比较，用穷举法求得最优解为 4－2－5－1－3，加工时间为 213；最劣解为 1－4－2－3－5，加工时间为 294；平均解的加工时间为 265。

下面用遗传算法求解。选择交叉概率 $p_c=0.6$，变异概率 $p_m=0.1$，种群规模为 20，迭代次数 $N=50$。运算结果如表 6.3 和图 6.6 ~ 图 6.8 所示。

表 6.3　遗传算法运行的结果

总运行次数	最好解	最坏解	平均	最好解的频率	最好解的平均代数
20	213	221	213.95	0.85	12

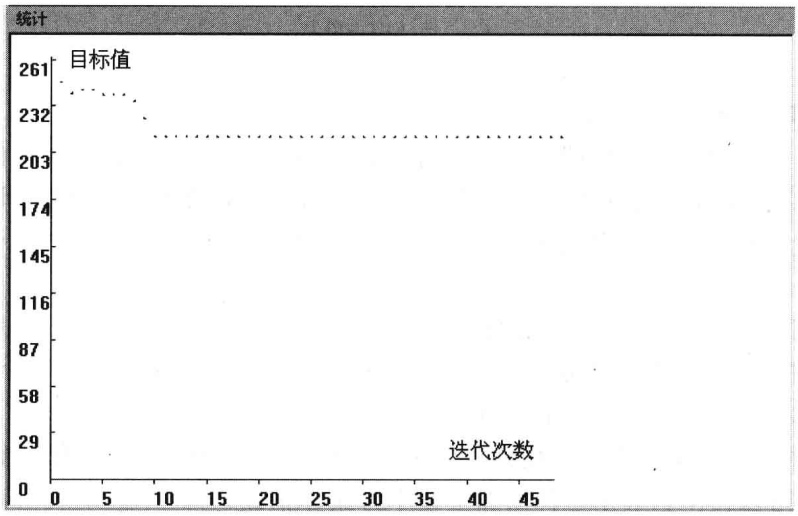


图 6.6　最优解收敛图

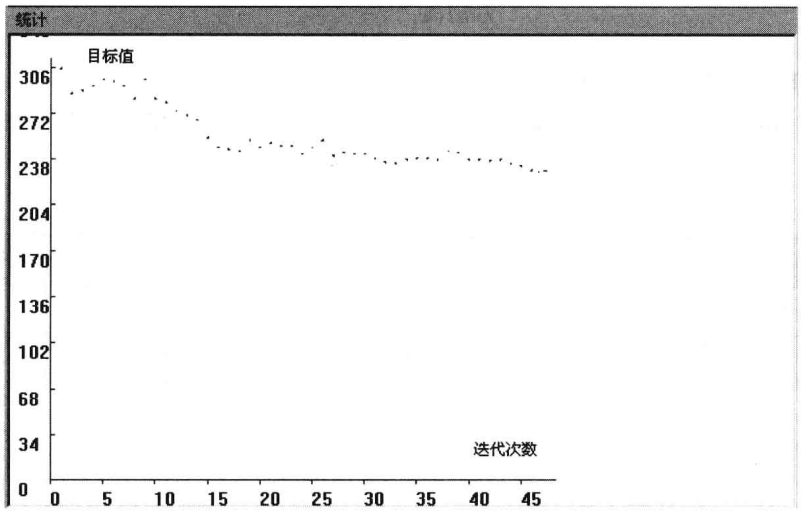


图 6.7　平均值收敛图

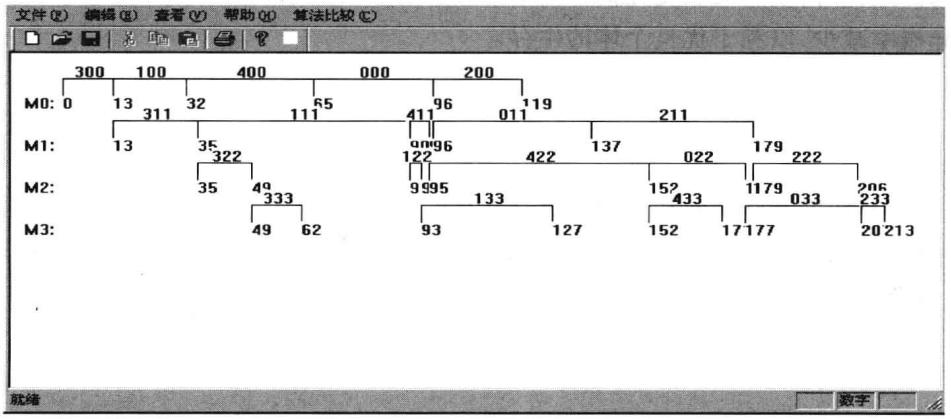


图 6.8 机器甘特图

6.5 小结

1. 基本遗传算法

遗传算法主要借用生物进化中“适者生存”的规律。

遗传算法的设计包括编码、适应度函数、选择、控制参数、交叉与变异等遗传算子。

遗传算法常用的编码方案有位串编码(二进制编码、Gray 编码)、实数编码等。

遗传算法中初始群体中的个体可以是随机产生的。群体规模太小时,遗传算法的优化性能一般不会太好,容易陷入局部最优解。而当群体规模太大时,则计算复杂。

遗传算法的适应度函数是用来区分群体中的个体好坏的标准。适应度函数一般是由目标函数变换得到的,但必须将目标函数转换为求最大值的形式,而且保证函数值非负。为了防止发生欺骗问题,对适应度函数值域的某种映射变换,称为适应度函数的尺度变换或者定标。

个体选择概率的常用分配方法有适应度比例方法、排序方法等。选择个体方法主要有轮盘赌选择、锦标赛选择方法、最佳个体保存方法等。

遗传算法中起核心作用的是交叉算子。主要有一点交叉、二点交叉等基本的交叉算子,有部分匹配交叉等修正的交叉方法。

变异操作主要有位点变异、逆转变异、插入变异、互换变异、移动变异等变异方法。

2. 改进遗传算法

双倍体遗传算法群体中的每个个体具有一个显性染色体,一个隐性染色体。每个染色体的编码/解码方式与基本的遗传算法相同。双倍体遗传算法采用显性遗传,即计算显性染色体的适应度,按照显性染色体的适应度进行选择、交叉、变异操作,隐性染色体也同时进行操作。当三个遗传算子都执行完成以后,将个体的染色体显隐性进行重新排定,个体中适应度较大的染色体设定为显性染色体,适应度较小的染色体设定为隐性染色体。

双种群遗传算法建立两个遗传算法群体,分别独立地运行复制、交叉、变异操作,同时当每一代运行结束以后,选择两个种群中的随机个体及最优个体分别交换。

自适应遗传算法的交叉和变异概率能够随适应度自动改变。当种群各个体适应度趋于一致

或者趋于局部最优时，使交叉和变异概率增加，以跳出局部最优；而当群体适应度比较分散时，使交叉和变异概率减少，以利于优良个体的生存。

思考题

6.1　遗传算法的基本步骤和主要特点是什么？

6.2　适应度函数在遗传算法中的作用是什么？试举例说明如何构造适应度函数。

6.3　选择的基本思想是什么？

习题

6.1　已知 10 个个体的适应度如表 6.4 所示，用幂函数变换法求出调整后的适应度值（$K=2$），然后采用适应度比例法分别求出调整前后各个个体的选择概率。

表 6.4　适 应 度 表

个体编号	原适应度	调整后的适应度	原选择概率	调整后的选择概率
1	2.5			
2	1.0			
3	3.0			
4	1.2			
5	2.1			
6	0.8			
7	2.3			
8	1.5			
9	0.9			
10	1.8			

6.2　用遗传算法求解下列函数的最大值

$$f(x) = 0.4 + sinc(4x) + 1.1sinc(4x+2) + 0.8sinc(x-2) + 0.7sinc(6x-4), x \in [-2,2]$$

其中

$$sinc(x) = \begin{cases} 1, & x = 0 \\ \dfrac{\sin(\pi x)}{\pi x}, & x \neq 0 \end{cases}$$

（提示：函数解用一个 16 位二进制数表示，种群规模 30，$P_c = 0.3$，$P_m = 0.01$，迭代次数 $N = 400$，最大值为 $f(-0.507179) = 1.501564$）。

6.3　编制遗传算法的计算程序，具体求解一个优化问题。记录算法结束时的迭代次数，画出最优个体的适应度变化曲线和群体适应度的平均值变化曲线。

第7章 专家系统与机器学习

自1968年研制成功第一个专家系统DENDRAL以来，专家系统技术发展非常迅速，已经应用到数学、物理、化学、医学、地质、气象、农业、法律、教育、交通运输、机械、艺术以及计算机科学本身，甚至渗透到政治、经济、军事等重大决策部门，产生了巨大的社会效益和经济效益，成为人工智能的重要分支。

下面首先介绍专家系统的产生与发展过程、基本概念，然后着重介绍专家系统的工作原理，介绍机器学习的基本概念和方法，以及知识发现和数据挖掘的基本概念和方法。然后简要介绍建立专家系统的方法以及几个著名的专家系统实例。这几个专家系统已经成为当前开发专家系统的骨架系统，具有很广泛的应用。

7.1 专家系统的产生和发展

专家系统(expert system)的第一个里程碑是斯坦福大学费根鲍姆等人于1968年研制成功分析化合物分子结构的专家系统——DENDRAL系统，它达到了专家的水平。此后，相继建立了各种不同功能、不同类型的专家系统。MYCSYMA系统是由麻省理工学院(MIT)于1971年开发成功并投入应用的专家系统，用LISP语言实现对特定领域的数学问题进行有效的处理，包括微积分运算、微分方程求解等。DENDRAL和MYCSYMA系统是专家系统发展的第一阶段。这个时期专家系统的特点是：高度的专业化，专门问题求解能力强，但结构、功能不完整，移植性差，缺乏解释功能。

20世纪70年代中期，专家系统进入了第二阶段——技术成熟期，出现了一批成功的专家系统。具有代表性的专家系统是MYCIN、PROSPECTOR、AM、CASNET等系统。MYCIN系统是斯坦福大学研制的用于细菌感染性疾病的诊断和治疗的专家系统，能成功地对细菌性疾病作出专家水平的诊断和治疗。它是第一个结构较完整、功能较全面的专家系统。它第一次使用了知识库的概念，引入了可信度的方法进行不精确推理，能够给出推理过程的解释，用英语与用户进行交互。MYCIN系统对形成专家系统的基本概念、基本结构起了重要的作用。PROSPECTOR系统是由斯坦福研究所开发的一个探矿专家系统。由于它首次实地分析华盛顿某山区一带的地质资料，发现了一个钼矿，成为第一个取得显著经济效益的专家系统。CASNET是一个与MYCIN几乎同时开发的专家系统，由拉特格尔(Rutger)大学开发，用于青光眼诊断与治疗。AM系统是由

斯坦福大学于 1981 年研制成功的专家系统，它能模拟人类进行概括、抽象和归纳推理，发现某些数论的概念和定理。

第二阶段的专家系统的特点是：

① 单学科专业型专家系统。

② 系统结构完整，功能较全面，移植性好。

③ 具有推理解释功能，透明性好。

④ 采用启发式推理、不精确推理。

⑤ 用产生式规则、框架、语义网络表达知识。

⑥ 用限定性英语进行人机交互。

20 世纪 80 年代以来，专家系统的研制和开发明显地趋向于商业化，直接服务于生产企业，产生了明显的经济效益。例如 DEC 公司与卡内基 - 梅隆大学合作开发了专家系统 XCON，用于为 VAX 计算机系统制订硬件配置方案，节约资金近 1 亿美元。另一个重要发展是出现专家系统开发工具，从而简化了专家系统的构造。如骨架系统 EMYCIN、KAS、EXPERT，通用知识工程语言 OPS5、RLL，模块式专家系统工具 AGE 等。

7.2　专家系统的概念

7.2.1　专家系统的定义

专家系统是基于知识的系统，它在某种特定的领域中运用领域专家多年积累的经验和专业知识，求解只有专家才能解决的困难问题。专家系统作为一种计算机系统，继承了计算机快速、准确的特点，在某些方面比人类专家更可靠、更灵活，可以不受时间、地域及人为因素的影响。

专家系统的奠基人、斯坦福大学的费根鲍姆（E. A. Feigenbaum）教授，把专家系统定义为："专家系统是一种智能的计算机程序，它运用知识和推理来解决只有专家才能解决的复杂问题。"也就是说，专家系统是一种模拟专家决策能力的计算机系统。

7.2.2　专家系统的特点

1. 具有专家水平的专业知识

具有专家的专业水平是专家系统的最大特点。专家系统中的知识按其在问题求解中的作用可分为三个层次，即数据级、知识库级和控制级。数据级知识是指具体问题所提供的初始事实及在问题求解过程中所产生的中间结论、最终结论。数据级知识通常存放于数据库中。知识库知识是指专家的知识。这一类知识是构成专家系统的基础。控制级知识也称为元知识，是关于如何运用前两种知识的知识，如在问题求解中的搜索策略、推理方法等。专家系统具有的知识越丰富，质量越高，解决问题的能力就越强。

2. 能进行有效的推理

专家系统的核心是知识库和推理机。专家系统要利用专家知识来求解领域内的具体问题，必须有一个推理机构，能根据用户提供的已知事实，通过运用知识库中的知识，进行有效的推理，以实现问题的求解。专家系统不仅能根据确定性知识进行推理，而且能根据不确定的知识进行

推理。领域专家解决问题的方法大多是经验性的,表示出来往往是不精确的,仅以一定的可能性存在。要解决的问题本身所提供的信息往往也是不确定的。专家系统的特点之一就是能综合利用这些不确定的信息和知识进行推理,得出结论。

3. 具有启发性

专家系统除能利用大量专业知识以外,还必须利用经验的判断知识来对求解的问题作出多个假设。依据某些条件选定一个假设,使推理继续进行。

4. 具有灵活性

专家系统的知识库与推理机既相互联系又相互独立。相互联系保证了推理机利用知识库中的知识进行推理以实现对问题的求解;相互独立保证了当知识库作适当修改和更新时,只要推理方式不变,推理机部分就可以不变,使系统易于扩充,具有较大的灵活性。

5. 具有透明性

在使用专家系统求解问题时,不仅希望得到正确的答案,而且还希望知道得到该答案的依据。专家系统一般都有解释机构,向用户解释推理过程,回答用户"为什么(Why)"、"结论是如何得出的(How)"等问题。

6. 具有交互性

专家系统一般都是交互式系统,具有较好的人机界面。一方面它需要与领域专家和知识工程师进行对话以获取知识,另一方面它也需要不断地从用户那里获得所需的已知事实,并回答用户的询问。

专家系统本身是一个程序,但它与传统程序又不同。主要体现在以下几个方面:

① 从编程思想来看,传统程序是依据某个确定的算法和数据结构来求解某个确定的问题,而专家系统求解的许多问题没有可用的数学方法,而是依据知识和推理来求解,即

$$\text{传统程序} = \text{数据结构} + \text{算法}$$

$$\text{专家系统} = \text{知识} + \text{推理}$$

这是专家系统与传统程序的最大区别。

② 传统程序把关于问题求解的知识隐含于程序中,而专家系统则将知识与运用知识的过程即推理机分离。这种分离使专家系统具有更大的灵活性,便于修改。

③ 从处理对象来看,传统程序主要是面向数值计算和数据处理,而专家系统面向符号处理。传统程序处理的数据是精确的,而专家系统处理的数据和知识大多是不精确的、模糊的。

④ 传统程序一般不具有解释功能,而专家系统一般具有解释机构解释自己的行为。因为专家系统依赖于推理,它必须能够解释这个过程。

⑤ 传统程序根据算法求解问题,每次都能产生正确的答案,而专家系统则像人类专家那样工作,一般能产生正确的答案,但有时也会产生错误的答案,这也是专家系统存在的问题之一。但专家系统有能力从错误中吸取教训,改进对某一问题的求解能力。

⑥ 从系统的体系结构来看,传统程序与专家系统具有不同的结构。关于专家系统的结构在后面将作专门的介绍。

7.2.3 专家系统的类型

若按专家系统的特性及功能分类,专家系统可分为10类,如表7.1所示。

表 7.1　专家系统类型

专家系统类型	解决的问题
解释	根据感知数据推理情况描述
诊断	根据观察结果推理系统是否有保障
预测	指导给定情况可能产生的后果
设计	根据给定的要求进行相应的设计
规划	设计动作
控制	控制整个系统的行为
监督	比较观察结果和期望结果
修理	执行计划来实现规定的补救措施
教学	诊断、调整、修改学生行为
调试	建议故障的补救措施

1. 解释型专家系统

解释型专家系统能根据感知数据，经过分析、推理，从而给出相应解释。如化学结构说明、图像分析、语言理解、信号解释、地质解释、医疗解释等专家系统。代表性的解释型专家系统有 DENDRAL，PROSPECTOR 等。

2. 诊断型专家系统

诊断型专家系统能根据取得的现象、数据或事实推断出系统是否有故障，并能找出产生故障的原因，给出排除故障的方案。这是目前开发、应用得最多的一类专家系统。如医疗诊断、机械故障诊断、计算机故障诊断等专家系统。代表性的诊断型专家系统有 MYCIN、CASNET、PUFF（肺功能诊断系统）、PIP（肾脏病诊断系统）、DART（计算机硬件故障诊断系统）等。

3. 预测型专家系统

预测型专家系统能根据过去和现在的信息（数据和经验）推断可能发生和出现的情况。例如用于天气预报、地震预报、市场预测、人口预测、灾难预测等领域的专家系统。

4. 设计型专家系统

设计型专家系统能根据给定要求进行相应的设计。例如用于工程设计、电路设计、建筑及装修设计、服装设计、机械设计及图案设计的专家系统。对这类系统一般要求在给定的限制条件下能给出最佳的或较佳的设计方案。代表性的设计型专家系统有 XCON（计算机系统配置系统）、KBVLSI（VLSI 电路设计专家系统）等。

5. 规划型专家系统

规划型专家系统能按给定目标拟定总体规划、行动计划、运筹优化等，适用于机器人动作控制、工程规划、军事规划、城市规划、生产规划等。这类系统一般要求在一定的约束条件下能以较小的代价达到给定的目标。代表性的规划型专家系统有 NOAH（机器人规划系统）、SECS（制定有机合成规划的专家系统）、TATR（帮助空军制定攻击敌方机场计划的专家系统）等。

6. 控制型专家系统

控制型专家系统能根据具体情况，控制整个系统的行为，适用于对各种大型设备及系统进行

控制。为了实现对控制对象的实时控制,控制型专家系统必须具有能直接接收来自控制对象的信息,并能迅速地进行处理,及时地作出判断和采取相应行动的能力。所以控制型专家系统实际上是专家系统技术与实时控制技术相结合的产物。代表性的控制型专家系统是 YES/MVS(帮助监控和控制 MVS 操作系统的专家系统)。

7. 监督型专家系统

监督型专家系统能完成实时的监控任务,并根据监测到的现象做出相应的分析和处理。这类系统必须能随时收集任何有意义的信息,并能快速地对得到的信号进行鉴别、分析和处理。一旦发现异常,能尽快地作出反应,如发出报警信号等。代表性的监督型专家系统是 REACTOR(帮助操作人员检测和处理核反应堆事故的专家系统)。

8. 修理型专家系统

修理型专家系统是用于制订排除某类故障的规划并实施排除的一类专家系统,要求能根据故障的特点制订纠错方案,并能实施该方案排除故障。当制订的方案失效或部分失效时,能及时采取相应的补救措施。

9. 教学型专家系统

教学型专家系统主要适用于辅助教学,并能根据学生学习过程中所产生的问题进行分析、评价、找出错误原因,有针对性地确定教学内容或采取其他有效的教学手段。代表性的教学型专家系统有 GUIDON(讲授有关细菌传染性疾病方面的医学知识的计算机辅助教学系统)。

10. 调试型专家系统

调试型专家系统用于对系统进行调试,能根据相应的标准检测出被检测对象存在的错误,并能从多种纠错方案中选出适用于当前情况的最佳方案,排除错误。

表 7.1 是根据专家系统的特性及功能对专家系统进行分类的。这种分类往往不是很确切,因为许多专家系统不止一种功能。还可以从其他的角度对专家系统进行分类。例如,可以根据专家系统的应用领域进行分类。

7.2.4 专家系统的应用

当前专家系统主要的应用领域有医学、计算机系统、电子学、工程、地质学、军事科学、过程控制等。表 7.2 按应用领域列出了专家系统的典型应用。

表 7.2 典型领域的专家系统

领域	系统	功能
医学	MYCIN	细菌感染性疾病诊断和治疗
	CASNET	青光眼的诊断和治疗
	PIP	肾脏病诊断
	INTERNIST	内科病诊断
	PUFF	肺功能试验结果解释
	ONCOCIN	癌症化学治疗咨询
	VM	人工肺心机监控

续表

领域	系统	功能
地质学	PROSPECTOR	帮助地质学家评估某一地区的矿物储量
	DIPMETER ADVISOR	油井记录分析
	DRILLING ADVISOR	诊断和处理石油钻井设备的“钻头粘着”问题
	MUD	诊断和处理同钻探泥浆有关的问题
	HYDRO	水源总量咨询
	ELAS	油井记录解释
计算机系统	DART	计算机硬件系统故障诊断
	RI/XCON	配置 VAX 计算机
	YES/MVS	监控和控制 MVS 操作系统
	PTRANS	管理 DEC 计算机系统的建造和配置
	IDT	定位 PDP 计算机中有缺陷的单元
化学	DENDRAL	根据质谱数据来推断化合物的分子结构
	MOLGEN	DNA 分子结构分析和合成
	CRYSALIS	通过电子云密度图推断一个蛋白质的三维结构
	SECS	帮助化学家制定有机合成规划
	SPEX	帮助科学家设计复杂的分子生物学的实验
数学	MACSYMA	数学问题求解
	AM	从基本的数学和集合论中发现概念
工程	SACON	帮助工程师发现结构分析问题的分析策略
	DELTA	帮助识别和排除机车故障
	REACTOR	帮助操作人员检测和处理核反应堆事故
军事	AIRPLAN	用于航空母舰周围的空中交通运输计划的安排
	HASP	海洋声纳信号识别和舰艇跟踪
	TATR	帮助空军制订攻击敌方机场的计划
	RTC	通过解释雷达图像进行舰船分类

7.3　专家系统的工作原理

7.3.1　专家系统的一般结构

由专家系统的定义可知，专家系统的主要组成部分是知识库和推理机。实际专家系统的功能和结构可能彼此有些差异，但完整的专家系统一般应包括人机接口、推理机、知识库、数据库、知识获取机构和解释机构六部分。各部分的关系如图 7.1 所示。

专家系统的核心是知识库和推理机，其工作过程是根据知识库中的知识和用户提供的事实进行推理，不断地由已知的事实推出未知的结论即中间结果，并将中间结果放到数据库中，作为已知的新事实进行推理，从而把求解的问题由未知状态转换为已知状态。在专家系统的运行过程中，会不断地通过人机接口与用户进行交互，向用户提问，并向用户作出解释。

下面分别对专家系统的各个部分进行简单介绍。

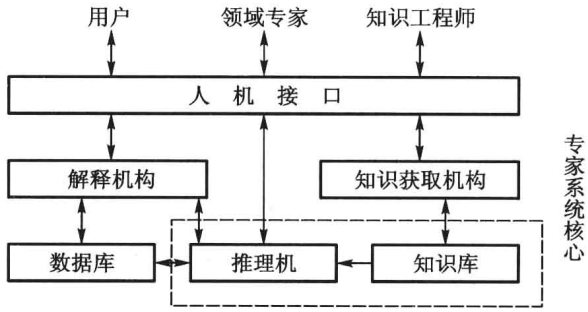


图 7.1 专家系统的一般结构

7.3.2 知识库

知识库(knowledge base)主要用来存放领域专家提供的有关问题求解的专门知识。知识库中的知识来源于知识获取机构,同时它又为推理机提供求解问题所需的知识。

1. 知识表示方法的选择

要建立知识库,首先要选择合适的知识表达方法。对同一知识,一般都可以用多种方法进行表示,但其效果却不同。应根据 2.1.4 节介绍的原则选择知识表达方法,即从能充分表示领域知识、能充分有效地进行推理、便于对知识的组织维护和管理、便于理解与实现等四个方面进行考虑。

2. 知识库的管理

知识库管理系统负责对知识库中的知识进行组织、检索、维护等。专家系统中任何其他部分要与知识库发生联系,都必须通过该管理系统来完成。这样可实现对知识库的统一管理和使用。

在进行知识库维护时,还要保证知识库的安全性。必须建立严格的安全保护措施,以防止由于操作失误等主观原因使知识库遭到破坏,造成严重的后果。一般知识库的安全保护也可以像数据库系统那样,通过设置口令验证操作者的身份,对不同操作者设置不同的操作权限等技术来实现。

7.3.3 推理机

推理机(reasoning machine)的功能是模拟领域专家的思维过程,控制并执行对问题的求解。它能根据当前已知的事实,利用知识库中的知识,按一定的推理方法和控制策略进行推理,直到得出相应的结论为止。

推理机包括推理方法和控制策略两部分。推理方法有确定性推理和不确定性推理。控制策略主要指推理方法的控制及推理规则的选择策略。推理包括正向推理、反向推理和正反向混合推理。推理策略一般还与搜索策略有关。

推理机的性能与构造一般与知识的表示方法有关,但与知识的内容无关,这有利于保证推理机与知识库的独立性,提高专家系统的灵活性。

7.3.4 数据库

数据库又称综合数据库(global database)、动态数据库或黑板,主要用于存放初始事实、问题描述及系统运行过程中得到的中间结果、最终结果等信息。

在开始求解问题时,数据库中存放的是用户提供的初始事实。数据库的内容随着推理过程的进行而变化,推理机会根据数据库的内容从知识库中选择合适的知识进行推理并将得到的中间结果存放于数据库中。数据库中记录了推理过程中的各种有关信息,又为解释机构提供了回答用户咨询的依据。

数据库中还必须具有相应的数据库管理系统,负责对数据库中的知识进行检索、维护等。

7.3.5 知识获取机构

知识获取(knowledge acquisition)是建造和设计专家系统的关键,也是目前建造专家系统的"瓶颈"。知识获取的基本任务是为专家系统获取知识,建立起健全、完善、有效的知识库,以满足求解领域问题的需要。

知识获取通常是由知识工程师与专家系统中的知识获取机构共同完成的。知识工程师负责从领域专家那里抽取知识,并用适合的方法把知识表达出来。知识获取机构把知识转换为计算机可存储的内部形式,然后把它们存入知识库。在存储过程中,要对知识进行一致性、完整性的检测。

不同专家系统的知识获取功能与实现方法差别较大,有的系统采用自动获取知识的方法,而有的系统则采用非自动或半自动知识获取的方法。

7.3.6 人机接口

人机接口(interface)是专家系统与领域专家、知识工程师、一般用户之间进行交互的界面,由一组程序及相应的硬件组成,用于完成输入输出工作。知识获取机构通过人机接口与领域专家及知识工程师进行交互,更新、完善、扩充知识库;推理机通过人机接口与用户交互。在推理过程中,专家系统根据需要不断向用户提问,以得到相应的事实数据,在推理结束时会通过人机接口向用户显示结果;解释机构通过人机接口与用户交互,向用户解释推理过程,回答用户问题。

在输入或输出过程中,人机接口需要内部表示形式与外部表示形式的转换。在输入时,它将把领域专家、知识工程师或一般用户输入的信息转换成系统的内部表示形式,然后分别交给相应的机构去处理;输出时,它将把系统要输出的信息由内部形式转化为人们易于理解的外部形式显示给用户。

在不同的专家系统中,由于硬件、软件环境不同,接口的形式与功能有较大的差别。随着计算机硬件和自然语言理解技术的发展,有的专家系统已经可以用简单的自然语言与用户交互,但有的系统只能通过菜单方式、命令方式或简单的问答方式与用户交互。

7.3.7 解释机构

解释机构(explanator)回答用户提出的问题,解释系统的推理过程。解释机构由一组程序组成。它跟踪并记录推理过程,当用户提出的询问需要给出解释时,它将根据问题的要求分别作相

应的处理，最后把解答用约定的形式通过人机接口输出给用户。

上面讨论的专家系统的一般结构只是专家系统的基本形式。实际上。在具体建造一个专家系统时，随着系统要求的不同，可以在此基础上做适当修改。

7.4 知识获取的主要过程与模式

7.4.1 知识获取的过程

知识获取主要是把用于问题求解的专门知识从某些知识源中提炼出来，并转化为计算机的表示形式存入知识库。知识源包括专家、书本、相关数据库、实例研究和个人经验等。目前专家系统的知识源主要是领域专家，所以知识获取过程需要知识工程师与领域专家反复交流，共同合作完成，如图 7.2 所示。

图 7.2 知识获取的过程

知识获取的基本任务是为专家系统获取知识，建立起健全、完善、有效的知识库，以满足求解领域问题的需要。因此，它需要做以下几项工作。

1. 抽取知识

所谓抽取知识是把蕴涵于知识源中的知识经识别、理解、筛选、归纳等抽取出来，以便用于建立知识库。

知识主要来源是领域专家及相关的专业技术文献，但知识并不都是以某种现成的形式存在于这些知识源中供选择的。领域专家虽然可以处理领域内的各种困难问题，但往往缺少总结，不一定能有条理地说出处理问题的道理和原则；领域专家可以列举大量处理过的实例，但不一定能建立起相互之间的联系，有时甚至是靠直觉或灵感解决问题。而且，领域专家一般都不熟悉专家系统的有关技术。这些都为知识的获取带来了困难。为了从领域专家处得到有用的知识，需要反复多次地与专家交谈，并有目的地引导交谈的内容，然后通过分析、综合、去粗取精，归纳出可供建立知识库的知识。

知识的另一来源是专家系统自身的运行实践。这就需要从实践中学习，总结出新的知识。一般来说，一个专家系统初步建立后，通过运行会发现知识不够健全，需要补充新的知识。此时除了请领域专家提供进一步的知识外，还可由专家系统根据运行经验从已有的知识或实例中演绎、归纳出新的知识，补充到知识库中去。这时专家系统就具有自我学习的能力。

2. 知识的转换

所谓知识的转换是指把知识由一种形式变换为另一种表示形式。

人类专家或科技文献中的知识通常是用自然语言、图形、表格等形式表示的，而知识库中的

知识是用计算机能够识别、运用的形式表示的。两者有较大的差距，所以必须将专家抽取的知识转换成适合知识库存放的知识。知识转换一般分为两步进行：第一步是把专家及文献资料处抽取的知识转换为某种知识表示模式，如产生式规则、框架等；第二步是把该模式表示的知识转换为系统可直接利用的内部形式。前一步通常由知识工程师完成，后一步一般通过输入及编译实现。

3. 知识的输入

把某模式表示的知识经编辑、编译送入知识库的过程称为知识的输入。

目前，知识的输入一般是通过两种途径实现的：一种是利用计算机系统提供的编译软件；另一种是用专门编制的知识编辑系统，称为知识编辑器。

4. 知识的检测

知识库的建立是通过知识抽取、转换、输入等环节实现的。这一过程中的任何环节上的失误都会造成知识的错误，直接影响到专家系统的性能。因此必须对知识进行检测，以便尽早发现并纠正可能存在的不一致、不完整的问题，并采取相应的修正措施。

7.4.2　知识获取的模式

按知识获取的自动化程度划分，知识获取主要有非自动、自动和半自动三种获取模式。

1. 非自动知识获取（人工移植）

非自动知识获取也称为人工移植。在这种方式中，知识获取分两步进行，首先由知识工程师从领域专家或有关的科技文献那里获取知识，然后再由知识工程师用某种知识编辑器输入到知识库中，其工作方式如图 7.3 所示。

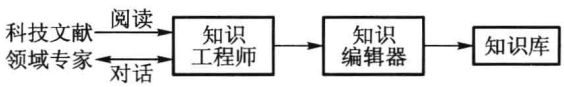


图 7.3　非自动化知识获取

非自动方式是专家系统建造中用得较为普遍的一种知识获取模式。在非自动获取模式中，知识工程师起着关键的作用。其主要任务是：

① 与领域专家进行交谈，阅读有关的文献，获取专家系统所需要的原始知识。

② 对已获得的原始知识进行分析、归纳、整理，形成用自然语言描述的知识条款，然后返回给领域专家检查。

③ 把确定的知识条款用知识表示方法表示出来，用知识编辑器进行编辑输入。

知识编辑器是一种用于知识输入的软件，通常是在建造专家系统时根据实际需要编制的。一般来说，知识编辑器应具有如下主要功能：

① 把用某种模式或语言表示的知识转换成计算机可表示的内部形式并输入到知识库中。

② 检测输入知识库中的知识的语法错误。

③ 检测知识的一致性，报告产生错误的原因，以便进行改正。

2. 自动知识获取

所谓自动知识获取是指系统具有获取知识的能力。它不仅可以直接与领域专家对话，从专

家提供的原始信息中学习到专家系统所需的知识，而且还能从系统自身的运行实践中总结、归纳出新的知识，发现知识中可能存在的错误，不断自我完善，建立起性能优良、知识完善的知识库。为达到这一目的，它至少应具备以下能力：

① 具有识别语言、文字、图像的能力。专家系统中知识主要来源于领域专家以及有关的科技文献资料、图像等。为了实现知识的自动获取，就必须使系统能与领域专家直接对话，能阅读有关的科技资料。这就要求系统应具有识别语言、文字和图像的能力。只有这样，它才能直接获得专家系统所需的原始知识，为知识库的建立奠定基础。

② 具有理解、分析、归纳的能力。领域专家提供的知识通常是处理具体问题的实例，不能直接用于知识库。为了将它变为知识库中的知识，必须在理解的基础上进行分析、归纳、提炼、综合，从中抽取出专家系统的所需的知识并放入知识库中。在非自动知识获取中，这一工作是由知识工程师完成的，而在自动知识获取中，是由系统取代知识工程师完成的。

③ 具有从运行实践中学习的能力。在知识库初步建成投入使用后，随着应用向纵深发展，知识库的不完善性就会逐渐暴露出来。此时知识的自动获取系统应不断地总结经验教训，从运行实践中学习，产生新的知识，纠正可能存在的错误，不断进行知识库的自我完善。

总之，在自动知识获取系统中，原来需要知识工程师做的都由系统取代了，并且还要做更好的工作。自动获取知识的过程如图 7.4 所示。

图 7.4 自动知识获取

3. 半自动知识获取

自动知识获取是一种理想的知识获取方式，涉及人工智能的多个领域，尚处在研究阶段。例如模式识别、自然语言理解、机器学习等，对硬件亦有较高的要求。这几年在自然语言理解、机器学习方面的研究已取得了较大的进展，在人工神经网络的研究中已提出了多种学习算法。这些都为知识的获取提供了有利条件。因此，在建造知识获取系统时，应充分利用这些成果，逐渐向知识的自动获取过渡，提高其智能程度。事实上，在近些年建造的专家系统中，不同程度地做了这方面的尝试与探讨。在非自动知识获取的基础上增加了部分学习功能，使系统能从大量事例中归纳出某些知识。由于这样的系统不同于纯粹的非自动知识获取，但又没有达到完全自动知识获取的程度，因而称之为半自动知识获取。

7.5 机器学习

机器学习是人工智能中一个重要的研究领域，一直受到人工智能及认知心理学家们的普遍关注。关于机器学习的研究，可以追溯到 20 世纪 50 年代中期。当时人们从仿生学的角度开展了研究，希望搞清楚人类大脑及神经系统的学习机理。但由于受到客观条件的限制，未能如愿。之后几经波折，直到 20 世纪 80 年代才获得了蓬勃发展。

近年来，随着专家系统的发展，需要系统具有学习能力，这促进了机器学习的研究，使之获得

了较快的发展，研制出了多种学习系统。但机器学习与计算机科学、心理学等多种学科都有密切的关系，牵涉的面比较广，因此许多理论及技术上的问题尚处于研究之中。

7.5.1　机器学习的基本概念

学习是一个有特定目的的知识获取过程，其内在行为是获取知识、积累经验、发现规律；外部表现是改进性能、适应环境、实现系统的自我完善。

机器学习（machine learning）使计算机能模拟人的学习行为，自动地通过学习来获取知识和技能，不断改善性能，实现自我完善。

作为人工智能的一个研究领域，机器学习主要研究以下三方面问题：

① 学习机理。这是对人类学习机制的研究，即人类获取知识、技能和抽象概念的天赋能力。通过这一研究，将从根本上解决机器学习中存在的种种问题。

② 学习方法。研究人类的学习过程，探索各种可能的学习方法，建立起独立于具体应用领域的学习算法。机器学习方法的构造是在对生物学习机理进行简化的基础上，用计算的方法进行再现。

③ 学习系统。根据特定任务的要求，建立相应的学习系统。

从计算机算法角度研究机器学习问题，与生物学、医学和生理学从生理、生物功能角度，研究生物界特别是人类学习问题有着密切的联系。最近国际上新兴的脑机交互（brain-computer interface，BCI）就是从大脑中直接提取信号，并经过计算机处理加以应用。

一个学习系统一般应该有环境、学习、知识库、执行与评价等四个基本部分组成。各部分之间的关系如图 7.5 所示，其中，箭头表示信息的流向。

图 7.5　学习系统的基本结构

“环境”指外部信息的来源，可以是系统的工作对象，也可以包括工作对象和外界条件。环境是以某种形式表达的外界信息的集合。例如，在医疗系统中，环境就是病人当前的症状、检验的数据和病历；在模式识别中，环境是要识别的图形或图像；在控制系统中，环境是被控对象。环境为系统的学习机构提供有关信息。它通过对环境的搜索取得外部信息，然后经分析、综合、类比、归纳等思维过程获得知识，并将这些知识存入知识库中。

“知识库”用于存储由学习得到的知识，在存储时要进行适当的组织，使它既便于应用又便于维护。

“执行与评价”实际上是由“执行”与“评价”这两个环节组成的。执行环节用于处理系统面临的现实问题，即应用学到的知识求解问题，如定理证明、智能控制、自然语言处理、机器人行动规划等；评价环节用于验证、评价执行环节执行的效果，如结论的正确性等。目前对评价的处理有两种方式：一种是把评价时所需的性能指标直接建立在系统中，由系统对执行环节得到的结果进行评价；另一种是由人来协助完成评价工作。如果采用后一种方式，则图 7.5 中可略去评价环节，但环境、学习、知识库、执行等环节是不可缺少的。

“学习”部分将根据反馈信息决定是否要从环境中索取进一步的信息进行学习，以修改、完善知识库中的知识。这是学习系统的一个重要特征。

上述模型是针对符号学习系统的，不能完全概括所有的机器学习系统。例如，不能概括遗传学习、神经网络学习系统等。

7.5.2 机器学习的分类

机器学习可从不同的角度，根据不同的方式进行分类。例如，若按系统的学习能力分类，机器学习可分为监督学习与非监督学习，两者的主要区别是前者在学习时需要教师的示教或训练，而后者是用评价标准来代替人的监督工作；若按所学知识的表示方式分类，机器学习可分为逻辑表示法学习、产生式表示法学习、框架表示法学习等；若按机器学习的应用领域分类，机器学习可分为专家系统、机器人学、自然语言处理、图像识别、博弈、数学、音乐等；若按学习方法是否为符号表示来分类，机器学习可分为符号学习与非符号学习。我们自下一节开始讨论的学习方法都属于符号学习，关于非符号学习，即连接学习，将在第 8 章神经网络中进行讨论。

下面讨论几种当前常用的分类方法。

1. 按学习方法分类

正如人们有各种各样的学习方法一样，机器学习也有多种学习方法。若按学习时所用的方法进行分类，则机器学习可分为机械式学习、指导式学习、示例学习、类比学习、解释学习等。这是温斯顿在 1977 年提出的一种分类方法。

2. 按学习能力分类

(1) 监督学习(有教师学习)。

监督学习(supervised learning)的基本学习方法如图 7.6 所示。监督学习系统中根据“教师”提供的正确响应调整学习系统的参数和结构。

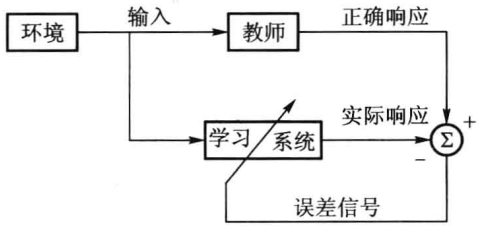


图 7.6 监督学习框图

(2) 强化学习(激励学习)

强化学习(reinforcement learning)的基本学习方法如图 7.7 所示。监督学习是对每个输入模式都有一个正确的目标输出，而强化学习中外部环境对系统输出结果只给出评价信息(奖励或者惩罚)，而不是正确答案，学习系统通过那些受惩的动作改善自身的性能。基于遗传算法的学习方法就是一种强化学习。

(3) 非监督学习(无教师学习)。

非监督学习(unsupervised learning)的基本学习方法如图 7.8 所示。非监督学习系统完全按照环境提供的数据的某些统计规律调节自身的参数或者结构(自组织)，以表示出外部输入的某

种固有特性，如聚类或者某种统计上的分布特征。

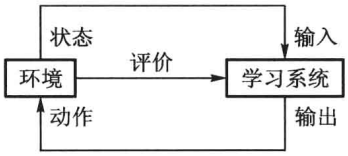


图 7.7　强化学习框图

图 7.8　非监督学习框图

3. 按推理方式分类

若按学习时所采用的推理方式进行分类，机器学习可分为基于演绎的学习及基于归纳的学习。

基于演绎的学习是指以演绎推理为基础的学习。解释学习在其推理过程中主要用的演绎方法，因而可将它划入基于演绎的学习这一类。

基于归纳的学习是指以归纳推理为基础的学习。示例学习、发现学习等在其学习过程中主要使用了归纳推理，因而可划入归纳学习这一类。

早期的机器学习系统一般都使用单一的推理方式，现在则趋于集成多种推理技术来支持学习。例如类比学习就既用到演绎推理又用到归纳推理，解释学习也是这样，只是因它演绎部分所占的比例较大，所以把它归入基于演绎的学习。

4. 按综合属性分类

随着机器学习的发展以及人们对它认识的提高，要求对机器学习进行更科学、更全面的分类。因而近年来有人提出了按学习的综合属性进行分类，它综合考虑了学习的知识表示、推理方法、应用领域等多种因素，能比较全面地反映机器学习的实际情况。用这种方法进行分类，不仅可以把过去已有的学习方法都包括在内，而且反映了机器学习的最近发展。

按照这种分类方法，机器学习可分为归纳学习、分析学习、连接学习以及遗传算法与分类器系统等。

分析学习是基于演绎和分析的学习。学习时从一个或几个实例出发，运用过去求解问题的经验，通过演绎对当前面临的问题进行求解，或者产生能更有效应用领域知识的控制性规则。分析学习的目标不是扩充概念描述的范围，而是提高系统的效率。

7.5.3　机械式学习

机械式学习(rote learning)又称为记忆学习，或者死记式学习，是一种最简单、最原始的学习方法。机械式学习通过直接记忆或者存贮外部环境所提供的信息达到学习的目的，并在以后通过对知识库的检索得到相应的知识直接用来求解问题。

例如，已知输入(前提、条件)是$(x_1, x_2, \cdots, x_n)$时，输出(结论、操作)是$(y_1, y_2, \cdots, y_m)$，则把联想对

$$[(x_1, x_2, \cdots, x_n), (y_1, y_2, \cdots, y_m)]$$

存入知识库中。当以后又出现$(x_1, x_2, \cdots, x_n)$时，只要直接从知识库中检索出$(y_1, y_2, \cdots, y_m)$就可以了，不需要进行计算和推导。

机械式学习实质上是用存储空间来换取处理时间。虽然节省了计算时间，但却多占用了存

储空间。当因学习而积累的知识逐渐增多时，占用的空间就会越来越大，检索的效率也将随着下降。所以，在机械式学习中要权衡时间与空间的关系，这样才能取得较好的效果。

应用机械式学习的一个典型例子是塞缪尔的跳棋程序 CHECKERS。该程序采用极大极小方法搜索博弈树，在给定的搜索深度下用估价函数对格局进行评分，然后通过倒推计算求出上层结点的倒推值，以决定当前的最佳走步。CHECKERS 的学习环节把每个格局的倒推值都记录下来，当下次遇到相同的情况时，就直接利用“记住”的倒推值决定最佳走步，而不必重新计算。例如，设在某一格局 A 时轮到 CHECKERS 走步，它向前搜索三层，得到如图 7.9 所示的搜索树。

在图 7.9 中，根据对端结点的静态估值，可求得 A 的倒推值为 6，最佳走步是走向 C。这时 CHECKERS 记住 A 及其倒推值 6。假若在以后的对弈中又出现了格局 A 且轮到它走步，则它就可以通过检索直接得到 A 的倒推值，而不必再做倒推计算。如果博弈时出现了图 7.10 所示的情况，格局 A 是搜索树的端点，此时使用 A 的倒推值比使用它的静态估值将更准确，同时由于对 A 使用了所记忆的倒推值，因而对格局 Q 来说，相当于搜索深度扩大到六层。

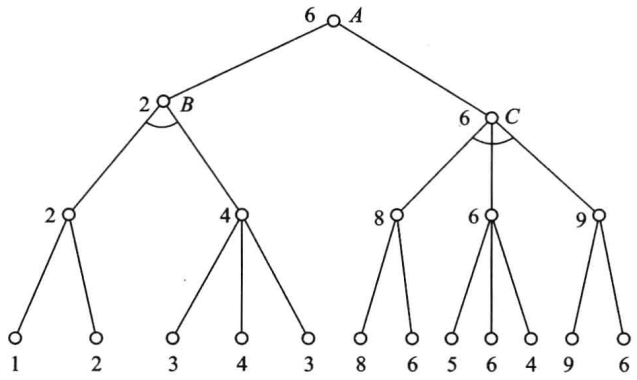


图 7.9 博弈搜索树

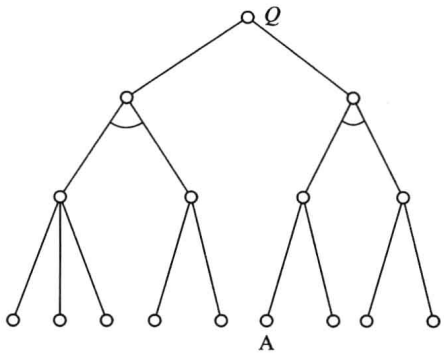


图 7.10 以 A 为结点的博弈树

7.5.4 指导式学习

指导式学习(learning by being told)又称为嘱咐式学习或教授式学习。指导式学习是由外部环境向系统提供一般性的指示或建议，系统把它们具体地转化为细节知识并送入知识库中。在学习过程中要反复对形成的知识进行评价，使其不断完善。

一般地说，指导式学习的学习过程由下列四个步骤组成。

1. 征询指导者的指示或建议

指导式学习的第一步工作是征询指导者的指示或建议。其征询方式可以是简单的，也可以是复杂的；可以是主动的，也可以是被动的。所谓简单征询是指导者给出一般性的意见，系统将其具体化；所谓复杂征询是指系统不仅要求指导者给出一般性的建议，而且还要具体地鉴别知识库中可能存在的问题，并给出修改意见；所谓被动征询是指系统只是被动地等待指导者提供意见；所谓主动征询是指系统不只是被动地接受指示，而且还能主动地提出询问，把指导者的注意力集中在特定的问题上。

理论上讲，为了实现征询，系统应具有识别、理解自然语言的能力。这样才能使系统直接与

指导者进行对话。但由于目前还不能完全实现这一要求，因而目前征询通常使用某种约定的语言进行。

2. 把征询意见转换为可执行的内部形式

征询意见的目的是为了获得知识，以便用这些知识求解问题。为此学习系统应具有把用约定形式表示的征询意见转化为计算机内部可执行形式的能力，并且能在转化过程中进行语法检查及适当的语义分析。

3. 加入知识库

经转化后的知识就可加入知识库。在加入过程中要对知识进行一致性检查，以防止出现矛盾、冗余、环路等问题。

4. 评价

为了检验新知识的正确性，需要对它进行评价。最简单也是最常用的评价方法是对新知识进行经验测试，即执行一些标准例子，然后检查执行情况是否与已知情况一致。如果出现了不一致，表示新知识中存在某些问题，此时可把有关信息反馈给指导者，请他给出另外的指导意见。

指导式学习是一种比较实用的学习方法，可用于专家系统的知识获取。它既可以避免由系统自己进行分析、归纳从而产生新知识所带来的困难，又无需领域专家了解系统内部知识表示和组织的细节，因此目前应用得较多。

7.5.5　示例学习

人们要解决一个新的问题，常常是将过去成功解决的类似的案例，用于求解新的问题。例如，医生在对某个病人作了检查后，会想到以前看过的病人的情况，找出几个在重要病症上相似的病人，将那些病人的诊断和治疗方案用于这个病人。这就是示例学习的基本思想。

示例学习(learning from examples)又称为实例学习或从例子中学习。示例学习是通过从环境中取得若干与某概念有关的例子，经归纳得出一般性概念的一种学习方法。在这种学习方法中，外部环境(教师)提供一组例子(正例和反例)，然后从这些特殊知识中归纳出适用于更大范围的一般性知识，它将覆盖所有的正例并排除所有反例。例如，如果我们用一批动物作为示例，并且告诉学习系统哪一个动物是“马”，哪一个动物不是，当示例足够多时，学习系统就能概括出关于“马”的概念模型，使自己能识别马，并且能把马与其他动物区别开来，这一学习过程就是示例学习。

示例学习的过程是：首先从示例空间中选择合适的训练示例，然后经解释归纳出一般性的知识，最后再从示例空间中选择更多的示例对它进行验证，直到得到可实用的知识为止。示例学习的学习模型如图 7.11 所示。

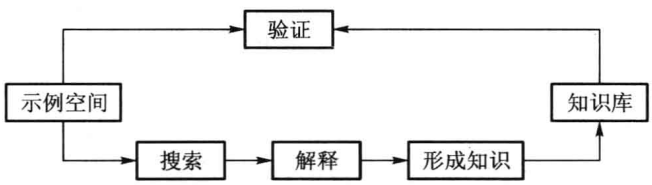


图 7.11　示例学习的学习模型

“示例空间”是所有可对系统进行训练的示例集合。与示例空间有关的主要问题是示例的质量、数量以及它们在示例空间中的组织。示例的质量和数量将直接影响到学习的质量，而示例的组织方式将影响到学习效率。

“搜索”的作用是从示例空间中查找所需的示例。为了提高搜索的效率，需要设计合适的搜索算法，并把它与示例空间的组织进行统筹考虑。

“解释”是从搜索到的示例中抽象出所需的有关信息供形成知识使用。当示例空间中的示例与知识的表示形式有较大差别时，需要将其转换为某种适合于形成知识的过渡形式。

“形成知识”是指把经解释得到的有关信息通过综合、归纳等形成一般性的知识。关于形成知识的方法，将在下面讨论。

“验证”的作用是检验所形成的知识的正确性，为此需从示例空间中选择大量的示例。如果通过验证发现形成的知识不正确，则需进一步获得示例，对刚才形成的知识进行修正。重复这一过程，直到形成正确的知识为止。

7.6 知识发现与数据挖掘

随着计算机和网络技术的迅速发展，出现了以数据库和数据仓库为存储单位的海量数据，而且这种数据仍然在以惊人的速度不断增长。如何对这些海量数据进行有效处理，特别是如何从这些数据中归纳、提取出高一级的更本质、更有用的规律性信息，就成了信息领域的一个重要课题。事实上，这些海量数据不仅承载着大量的信息，同时也蕴藏着丰富的知识。正是在这样的背景下，知识发现与数据挖掘技术应运而生。

知识发现与数据挖掘现已成为人工智能和信息科学领域的一个热门方向，其应用范围非常广泛，如企业数据、商业数据、科学实验数据、管理决策数据等。

7.6.1 知识发现与数据挖掘的概念

知识发现的全称是从数据库中发现知识(knowledge discovering from data base，KDD)。数据挖掘(data mining，DM)是从数据库中挖掘知识。KDD 和 DM 的本质含义是一样的，只是知识发现的概念主要流行于人工智能和机器学习领域，而数据挖掘的概念则主要流行于统计、数据分析、数据库和管理信息系统领域。所以，现在有关文献中一般都把二者同时列出。

知识发现和数据挖掘的目的就是从数据集中抽取和精化一般规律或模式。其涉及的数据形态包括数值、文字、符号、图形、图像、声音，甚至视频和 Web 网页等等。数据组织方式可以是有结构的、半结构的或非结构的。知识发现的结果可以表示成各种形式，包括概念、规则、定律、公式、议程等。本节仅对知识发现与数据挖掘技术作一简单介绍。

7.6.2 知识发现的一般过程

知识发现过程可粗略地划分为数据准备、数据挖掘以及结果的解释和评价等三步。

1. 数据准备

数据准备又可分为三个子步骤：数据选取、数据预处理和数据变换。数据选取就是确定目标数据，即操作对象，它是根据用户的需要从原始数据库中抽取的一组数据。数据预处理一般可能

包括消除噪声、推导计算缺值数据、消除重复记录、完成数据类型转换等。当数据开采的对象是数据仓库时，一般来说，数据预处理已经在生成数据仓库时完成了。数据变换的主要目的是消减数据维数，即从初始特征中找出真正有用的特征以减少数据开采时要考虑的特征或变量个数。

2. 数据挖掘

数据挖掘阶段首先要确定挖掘的任务或目的是什么，如数据总结、分类、聚类、关联规则或序列模式等。确定了挖掘任务后，就要决定使用什么样的挖掘算法。同样的任务可以用不同的算法来实现，选择实现算法有两个考虑因素：一是不同的数据有不同的特点，因此需要用与之相关的算法来挖掘；二是用户或实际运行系统的要求，有的用户可能希望获取描述型的、容易理解的知识，而有的用户系统的目的是获取预测准确度尽可能高的预测型知识。

3. 结果的解释和评价

数据挖掘阶段发现的知识模式中可能存在冗余或无关的模式，所以还要经过用户或机器的评价。若发现所得模式不满足用户要求，则需要退回到发现阶段之前，如重新选取数据，采用新的数据变换方法，设定新的数据挖掘参数值，甚至换一种挖掘算法。另外，由于 KDD 最终是面向人的，因此可能要对发现的模式进行可视化，或者把结果转换为用户易懂的另一种表示，如把分类决策树转换为“if-then”规则。

7.6.3　知识发现的任务

所谓知识发现的任务，就是知识发现所要得到的具体结果。

1. 数据总结

数据总结的目的是对数据进行浓缩，给出它的紧凑描述。传统的也是最简单的数据总结方法是计算出数据库的各个字段上的求和值、平均值、方差值等统计值，或者用直方图、饼状图等图形方式表示。数据挖掘主要关心从数据泛化的角度来讨论数据总结。数据泛化是一种把数据库中的有关数据从底层抽象到高层次上的过程。

2. 概念描述

有两种典型的描述：特征描述和判别描述。特征描述是从学习任务相关的一组数据中提取出关于这些数据的特征式，这些特征式表达了该数据集的总体特征；而判别描述则描述了两个或多个类之间的差异。

3. 分类

分类是数据挖掘中一项非常重要的任务，目前在商业应用最多。分类的目的是提出一个分类函数或分类模型（也常常称作分类器），该模型能把数据库中的数据项映射到给定类别中的一个。

4. 聚类

聚类是根据数据的不同特征，将其划分为不同的类。它的目的是使得属于同一类别的个体之间的差异尽可能的小，而不同类别的个体间的差异尽可能的大。聚类方法包括统计方法、机器学习方法、神经网络方法和面向数据库的方法等。

5. 相关性分析

相关性分析的目的是发现特征之间或数据之间的相互依赖关系。数据相关性关系代表一类重要的可发现的知识。一个依赖关系存在于两个元素之间。如果从一个元素 A 的值可以推出另

一个元素 B 的值,则称 B 依赖于 A。这里所谓元素可以是字段,也可以是字段间的关系。

6. 偏差分析

偏差分析包括分类中的反常实例、例外模式、观测结果对期望值的偏离以及量值随时间的变化等,其基本思想是寻找观察结果与参照量之间的有意义的差别。通过发现异常,可以引起人们对特殊情况加倍注意。

7. 建模

建模就是通过数据挖掘,构造出能描述一种活动、状态或现象的数学模型。

7.6.4 知识发现的方法

知识发现主要有以下几种方法。

1. 统计方法

事物的规律性,一般从其数量上会表现出来。而统计方法就是从事物的外在数量上的表现去推断事物可能的规律性。因此,统计方法就是知识发现的一个重要方法。常见的统计方法有回归分析、判别分析、聚类分析以及探索分析等。

2. 粗糙集

粗糙集(rough set)理论是由 Zdziskew Pawlak 在 1982 年提出的,它是一种新的数学工具,用于处理含糊性和不确定性,粗糙集在数据挖掘中也可发挥重要作用。什么是粗糙集呢?由于篇幅所限,这里不给出其精确的数学定义,简单地说,粗糙集是由集合的下近似、上近似来定义的。下近似中的每一个成员都是该集合的确定成员,若不是上近似中的成员肯定不是该集合的成员。粗糙集的上近似是下近似和边界区的合并。边界区的成员可能是该集合的成员,但不是确定的成员。可以认为粗糙集是具有三值隶属函数的模糊集,即是、不是、也许。与模糊集一样,它是一种处理数据不确定性的数学工具,常与规则归纳、分类和聚类方法结合起来使用。

3. 可视化

可视化(visualization)就是把数据、信息和知识转化为图形的表现形式的过程。可视化可使抽象的数据信息形象化。于是,人们便可以直观地对大量数据进行考察、分析,发现其中蕴藏的特征、关系、模式和趋势等。因此,信息可视化也是知识发现的一个有用的手段。

4. 传统机器学习方法

包括符号学习和连接学习。

7.6.5 知识发现的对象

1. 数据库

数据库是当然的知识发现对象。当前研究比较多的是关系数据库的知识发现。其主要研究课题有超大数据量、动态数据、噪声、数据不完整性、冗余信息和数据稀疏等。

2. 数据仓库

随着计算机技术的迅猛发展,到 20 世纪 80 年代,许多企业的数据库中积累了大量的数据。于是,便产生了进一步使用这些数据的需求,就是想通过对这些数据的分析和推理,为决策提供依据。但对于这种需求,传统的数据库系统却难以实现,主要有两个原因:一是传统数据库一般只存储短期数据,而决策需要大量历史数据;二是决策信息涉及许多部门的数据,而不同系统的

数据难以集成。在这种情况下,数据仓库(data warehouse)技术应运而生。

目前,人们对数据仓库有很多不同的理解。Inmon 将数据仓库明确定义为:数据仓库是面向主题的、集成的、内容相对稳定的、不同时间的数据集合,用以支持经营管理中的决策制定过程。

具体来讲,数据仓库收集不同数据源中的数据,将这些分散的数据集中于一个更大的库中,最终用户从数据仓库中进行查询和数据分析。数据仓库中的数据应是良好定义的、一致的、不变的,数据量也应足够支持数据分析、查询、报表生成,此外还包括长期积累分布的、跨平台的数据,使用过程中可忽略许多技术细节。总之,数据仓库有四个基本特征:

① 数据仓库的数据是面向主题的。

② 数据仓库的数据是集成的。

③ 数据仓库的数据是稳定的。

④ 数据仓库的数据是随时间不断变化的。

数据仓库是面向决策分析的,数据仓库从事务型数据抽取并集成得到分析型数据后,需要各种决策分析工具对这些数据进行分析和挖掘,才能得到有用的决策信息。而数据挖掘技术具备从大量数据中发现有用信息的能力,于是数据挖掘自然成为数据仓库中进行数据深层分析的一种必不可少的手段。

数据挖掘往往依赖于经过良好组织和预处理的数据源,数据的好坏直接影响数据挖掘的效果,因此数据的前期准备是数据挖掘过程中一个非常重要的阶段。而数据仓库具有从各种数据源中抽取数据,并对数据进行清理、聚集和转移等各种处理的能力,恰好为数据挖掘提供了良好的进行前期数据准备工作的环境。

因此,数据仓库和数据挖掘技术的结合成为必然的趋势。数据挖掘为数据仓库提供深层次数据分析的手段,数据仓库为数据挖掘提供经过良好预处理的数据源。目前许多数据挖掘工具都采用了基于数据仓库的技术。例如,中科院计算所智能信息处理开放实验室开发的知识发现平台 DBMiner 就是一个典型的例子。

3. Web 信息

随着 Web 的迅速发展,分布在 Internet 上的 Web 网页已构成了一个巨大的信息空间。在这个信息空间中也蕴藏着丰富的知识。因此,Web 信息也就理所当然地成为一个知识发现对象。

Web 知识发现主要分内容发现和结构发现。内容发现是指从 Web 文档的内容中提取知识;结构发现是指从 Web 文档的结构信息中推导知识。Web 内容发现又可分为对文本文档(包括 text,HTML 等格式)和多媒体文档(包括 image、audio、video 等类型)的知识发现。Web 结构发现包括文档之间的超链接结构、文档内部的结构、文档 URL 中的目录路径结构等。

4. 图像和视频数据

图像和视频数据中也存在有用的信息需要挖掘。比如,地球资源卫星每天都要拍摄大量的图像或录像,对同一个地区而言,这些图像存在着明显的规律性,白天和黑夜的图像不一样,当可能发生洪水时与正常情况下的图像又不一样。通过分析这些图像的变化,我们可以推测天气的变化,可以对自然灾害进行预报。这类问题,在通常的模式识别与图像处理中都需要通过人工来分析这些变化规律,从而不可避免地漏掉了许多有用的信息。

7.7 专家系统的建立

专家系统是人工智能中一个正在发展的研究领域，虽然目前已建立了许多专家系统，但是尚未形成建立专家系统的一般方法。下面简单介绍专家系统的一般建立过程。

专家系统是一个计算机软件系统，但与传统程序又有区别，因为知识工程与软件工程在许多方面有较大的差别，所以专家系统的开发过程在某些方面与软件工程类似，但某些方面又有区别。例如，软件工程的设计目标是建立一个用于事物处理的信息处理系统，处理的对象是数据，主要功能是查询、统计、排序等，其运行机制是确定的；而知识工程的设计目标是建立一个辅助人类专家的知识处理系统，处理的对象是知识和数据，主要的功能是推理、评估、规划、解释、决策等，其运行机制难以确定。另外从系统的实现过程来看，知识工程比软件工程更强调渐进性、扩充性。因此，在设计专家系统时软件工程的设计思想及过程虽可以借鉴，但不能完全照搬。

专家系统的开发步骤一般分为问题识别、概念化、形式化、实现和测试等阶段，如图 7.12 所示。

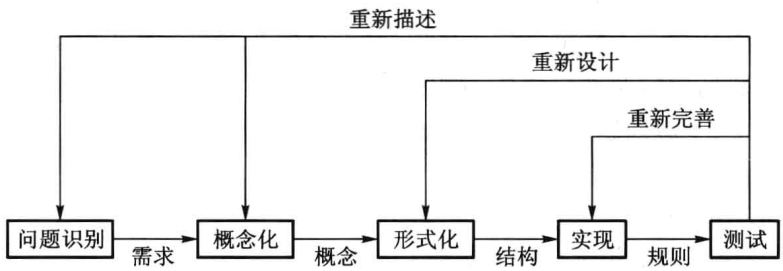


图 7.12 专家系统的开发步骤

(1) 问题识别阶段

在问题识别阶段，知识工程师和专家将确定问题的主要特点：

① 确定人员和任务，选定包括领域专家和知识工程师在内的参加人员，并明确各自的任务。

② 问题识别，描述问题的特征及相应的知识结构，明确问题的类型和范围。

③ 确定资源，确定知识源、时间、计算设备以及经费等资源。

④ 确定目标，确定问题求解的目标。

(2) 概念化阶段

概念化阶段的主要任务是揭示描述问题所需要的关键概念、关系和控制机制，子任务、策略和有关问题求解的约束。这个阶段需要考虑的问题有：

① 什么类型的数据有用，数据之间的关系如何？

② 问题求解时包括哪些过程，这些过程有哪些约束？

③ 如何将问题划分为子问题？

④ 信息流是什么？哪些信息是由用户提供的，哪些信息是需要导出的？

⑤ 问题求解的策略是什么？

(3) 形式化阶段

形式化阶段是把概念化阶段概括出来的关键概念、子问题和信息流特征形式化地表示出来。究竟采用什么形式,要根据问题的性质选择适当的专家系统构造工具或适当的系统框架。在这个阶段,知识工程师起着更积极的作用。

在形式化过程中,三个主要的因素是假设空间、基本的过程模型和数据的特征。为了理解假设空间的结构,必须对概念形式化并确定它们之间的关系,还要确定概念的基元和结构。为此需要考虑以下问题:

① 把概念描述成结构化的对象,还是处理成基本的实体?

② 概念之间的因果关系或时空关系是否重要,是否应当显式地表示出来?

③ 假设空间是否有限?

④ 假设空间是由预先确定的类型组成的,还是由某种过程生成的?

⑤ 是否应考虑假设的层次性?

⑥ 是否有与最终假设相关的不确定性或其他的判定性因素?

⑦ 是否考虑不同的抽象级别?

找到可以用于产生解答的基本过程模型是形式化知识的重要一步。过程模型包括行为和数学的模型。如果专家使用一个简单的行为模型,对它进行分析就能产生很多重要的概念和关系。数学模型可以提供附加的问题求解信息,或用于检查知识库中因果关系的一致性。

在形式化知识中,了解问题领域中数据的性质也是很重要的。为此应当考虑下述问题:

① 数据是不足的、充足的还是冗余的?

② 数据是否有不确定性?

③ 对数据的解释是否依赖于出现的次序?

④ 获取数据的代价是多少?

⑤ 数据是如何得到的?

⑥ 数据的可靠性和精确性如何?

⑦ 数据是一致的和完整的吗?

(4) 实现阶段

在形式化阶段,已经确定了知识表示形式和问题的求解策略,也选定了构造工具或系统框架。在实现阶段,要把前一阶段的形式化知识变成计算机软件,即要实现知识库、推理机、人机接口和解释系统。

在建立专家系统的过程中,原型系统的开发是极其重要的步骤之一。对于选定的表达方式,任何有用的知识工程辅助手段(如编辑、智能编辑或获取程序)都可以用来完成原型系统知识库。另外,推理机应能模拟领域专家求解问题的思维过程和控制策略。

(5) 测试阶段

这一阶段的主要任务是通过运行实例评价原型系统以及用于实现它的表达形式,从而发现知识库和推理机的缺陷。通常导致性能不佳的因素有如下三种:

① 输入输出特性,即数据获取与结论表示方法存在缺陷。例如,提问难于理解、含义模糊,使得存在错误或不充分的数据进入系统;结论过多或者太少,没有适当地组织和排序。

② 推理规则有错误、不一致或不完备。

③ 控制策略有问题，不是按专家采用的“自然顺序”解决问题。

专家系统必须先在实验室环境下进行精化和测试，然后才能够进行实地领域测试。在测试过程中，实例的选择应照顾到各个方面，要有较宽的覆盖面，即要涉及典型的情况，也要涉及边缘的情况。测试的主要内容有：

① 可靠性。通过实例的求解，检查系统得到的结论是否与已知结论一致。

② 知识的一致性。当向知识库输入一些不一致、冗余等有缺陷的知识时，检查它是否可把它们检测出来；当要求系统求解一个不应当给出答案的问题时，检查它是否会给出答案；如果系统具有某些自动获取知识的功能，则检测获取知识的正确性。

③ 运行效率。检测系统在知识查询及推理方面的运行效率，找出薄弱环节及求解方法与策略方面的问题。

④ 解释能力。对解释能力的检测主要从两个方面进行：一是检测它能回答哪些问题，是否达到了要求；二是检测回答问题的质量，即是否有说服力。

⑤ 人机交互的便利性。为了设计出友好的人机接口，在系统设计之前和设计过程中也要让用户参与。这样才能准确地表达用户的要求。

对人机接口的测试主要由最终用户来进行。根据测试的结果，应对原型系统进行修改。测试和修改过程应反复进行，直到系统达到满意的性能为止。

7.8 专家系统实例

目前专家系统的研究几乎已经遍及人类生活的各个方面。为了使读者对专家系统有更加具体的认识，下面介绍几个著名的实例。

7.8.1 医学专家系统——MYCIN

MYCIN 系统是由斯坦福大学 1972 年开始研制的用于对细菌感染性疾病进行诊断和治疗的专家系统。MYCIN 的功能是帮助内科医生诊断细菌感染疾病，并给出建议性的诊断结果和处方。MYCIN 系统是将产生式规则从通用问题求解的研究转移到解决专门问题的一个成功的典范，在专家系统的发展中占有重要的地位，许多专家系统就是在它的基础上建立起来的。

1. MYCIN 系统总体结构

MYCIN 系统是用 INTER LISP 语言编写的，知识库中大约有二百多条关于细菌血症的规则，可以识别约 50 种细菌。整个系统占 245 KB，其中 INTER LISP 系统占 160 KB，编译后的 MYCIN 系统占 50 KB，知识库占 8 KB，其余 27 KB 存放临床参数和作为工作空间，有咨询解释功能。

MYCIN 系统处理一个患者的咨询过程如图 7.13 所示。这个过程中的每一步都包含着规则的调用，人机对话。从询问中取得疾病状态、化验参数等通过直接观察得到的数据。

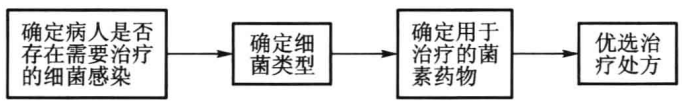


图 7.13 MYCIN 系统的咨询过程

MYCIN 系统的结构图如图 7.14 所示。从图中可以看出，MYCIN 系统主要由咨询、解释和知识获取三个模块以及知识库、动态数据库组成。

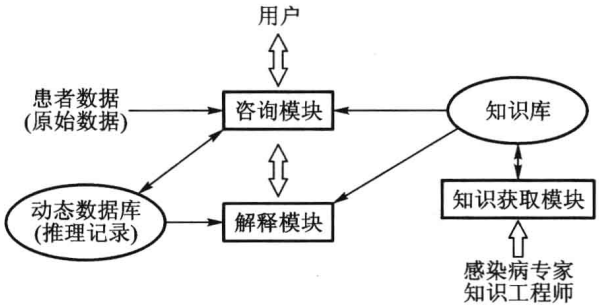


图 7.14　MYCIN 系统结构图

(1) 咨询模块

它相当于推理机和用户接口。当医生使用 MYCIN 系统时，首先启动这一子系统。此时 MYCIN 系统将给出提示，要求医生输入有关的信息，如病人的姓名、年龄、症状等，然后利用知识库中的知识进行推理，得出患者所患的疾病及治疗方案。MYCIN 系统采用反向推理的控制策略。推理过程将形成由若干条规则链构造成的与/或树。MYCIN 系统采用深度优先法进行搜索。在 MYCIN 系统中，还使用了基于可信度的不精确推理。

(2) 解释模块

它用于回答用户(医生)的询问。在咨询子系统的运行过程中，可以随时启动解释子系统，要求系统回答“为什么要求输入这一参数”、“结论是怎样得出的”等问题，MYCIN 系统通过记录系统所形成的与/或树来实现解释功能。

(3) 知识获取模块

用于从医生那里获取新的知识，完善知识库。当发现有医学知识被遗漏，或者发现新知识时，医生和知识工程师可以利用该模块增加或修改知识库。

2. MYCIN 动态数据库中的数据表示

动态数据库用于存放与患者有关的数据、化验结果以及系统推出的结论等动态变化的信息。动态数据库中的数据按照它们之间的关系组成一棵上下文树(context tree)。上下文树是在咨询过程中形成的。树中的结点称为上下文。每个结点对应一个具体的对象，描述该对象的所有数据都存储在该结点上。每一个结点旁注明结点名，括号中为该结点的上下文类型。上下文的类型能够指示出哪些规则可能被调用。因此，一个上下文树就构成了对病人的完整描述。

图 7.15 所示为上下文树的一个实例，表示从病人 PATIENT - 1 身上当前提取了两种培养物 CULTURE - 1 和 CULTURE - 2，先前曾提取过一种培养物 CULTURE - 3，从这些培养物中分别分离出相应的有机体。从 ORGANISM - 1 至 ORGANISM - 4，每种有机体有相应的药物进行治疗。对病人进行手术时使用过药物 DRUG - 4。通过该上下文树把病人的有关培养物及其使用药物的情况清楚地描述了出来，并且指出了哪种有机体来自于哪一种培养物，对哪种有机体使用了哪种药物。

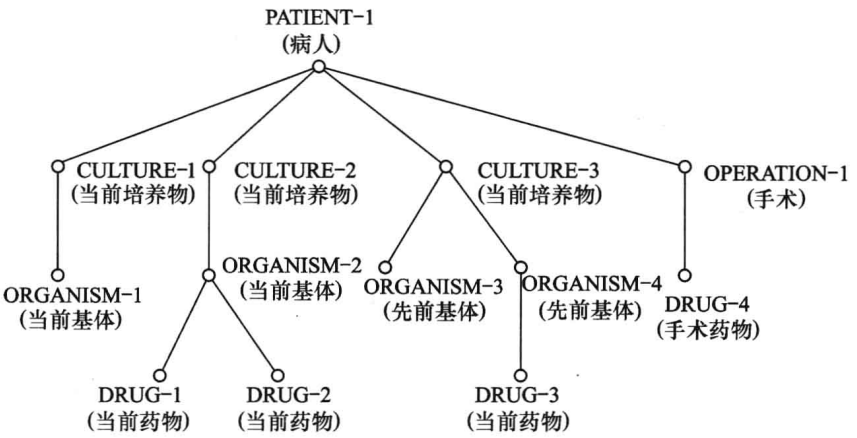


图 7.15 上下文树的一个实例

3. MYCIN 知识库中的知识表示

MYCIN 的知识库主要存放用于诊断和治疗感染性疾病的专家知识,同时还存放了一些进行推理所需要的静态知识,如临床参数的特征表、字典等。该系统用产生式规则表示这些知识。

(1) 领域知识的表示

领域知识用产生式规则表示。例如:

RULE 064 如果:有机体的染色体是革兰氏阳性

且:有机形态是球状的

且:有机体的生长结构呈链状

则:存在证据表明该有机体为链球菌类,可信度为 0.7

规则的每个条件是一个 LISP 函数,它们的返回值为 *T*,*NIL* 或 -1 到 +1 之间的某个数值。规则的行为部分用专门表示动作的行为函数表示。MYCIN 系统中有 3 个专门用于表示动作的行为函数:CONCLUDE,CONCLIST 和 TRANLIST。其中 CONCLUDE 用得最多,其形式为

(CONCLUDE *C P V TALLY CF*)

其中,*C*,*P*,*V* 分别表示上下文、临床参数和值;*TALLY* 是一个变量,用于存放规则前提部分的信任程度;*CF* 是规则强度,由领域专家提供。

(2) 临床参数的表示

每个上下文与一组临床数据相联系。这些数据完全地描述相应的上下文。每个临床参数表示上下文的一个特征,如病人的姓名、培养物的地点、机体的形状、药物的剂量等等。

临床参数可用三元组(上下文、属性、值)表示。例如,三元组(机体 -1,形态,杆状)表示机体 -1 的形态为杆状;三元组(机体 -1,染色体,革兰氏阴性)表示机体 -1 的染色体为革兰氏阴性。

临床数据按其取值方式可分为单值、是非值和多值三种。有的参数如病人的姓名、细菌类别等,可以有许多可能的取值,但各个值互不相容,所以只能取其中一个值,因此属于单值。是非值是单值的一种特殊情形,这时参数限于取"是"或"非"中的一种。例如,药物的剂量是否够,细菌是否重要等。多值参数是那些同时可取一个以上值的参数,如病人的药物过敏、传染的途径等

参数。

MYCIN 系统中有 65 个临床参数，为搜索方便，对参数按照其相对应的上下文分类。

为了避免在推理时过多地询问用户，同时也为了优化存储，MYCIN 系统还把有关的数据列成清单存在知识库中，当推理启用相应的规则时，就直接从清单中找到相应的数据。另外，MYCIN 系统还有一个包含 1400 个单词的词典，主要用于理解用户输入的自然语言。

4. MYCIN 的推理策略

当 MYCIN 系统被启动后，系统首先在数据库中建立一棵上下文树的根结点，并为该结点指定一个名字 PATIENT－1（病人－1），其类型为 PERSON。PERSON 的属性有 NAME，AGE，SEX，REGIMEN，其中 NAME、AGE、SEX 是 LABDATA 参数，即可通过用户询问得到。系统向用户提出询问，要求用户输入病人的姓名、年龄和性别，并以三元组形式存入数据库中。REGIMEN 表示对病人建议的处方。它不是 LABDATA 参数，必须由系统推出。事实上它正是系统进行推理的最终目标。

为了得到 REGIMEN，在推理开始时，首先调用目标规则 092 进行反向推理。规则 092 是系统中唯一在其操作部分涉及 REGIMEN 参数的规则。这个目标规则体现了在 MYCIN 系统中感染性疾病诊断和处方时决策的四个步骤。具体规则如下：

规则 092

IF　存在一种病菌需要处理

　　某些病菌虽然没有出现在目前的培养物中，但已经注意到它们需要处理

THEN　根据病菌对药物的过敏情况，编制一个可能抑制该病菌的处方表

　　从处方表中选择最佳的处方

ELSE　病人不必治疗

规则 092 的前提中涉及两个临床参数 TREATFOR 和 COVERFOR。它们均为非型参数。

TREATFOR 表示需要处理的病菌。它不是 LABDATA 参数，所以系统调用 TREATFOR 的 UPDATED－BY 特征所指出的第一条规则 090，检查它的前提是否为真。为此，如果该前提所涉及的值是可向用户询问的，就直接询问用户，否则再找出可推出该值的规则，判断其前提是否为真。如此反复进行，直到最后推出 PATIENT－1 的主要临床参数 REGIMEN 为止。在此过程中动态生成的关于病人的上下文树如图 7.16 所示。

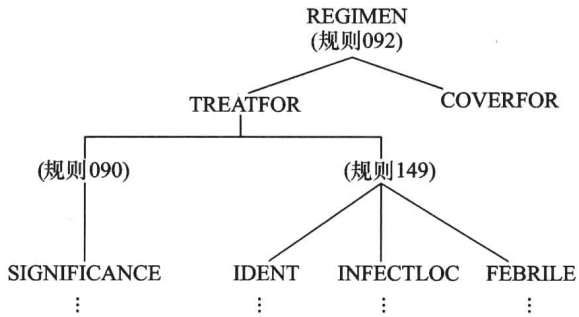


图 7.16　关于病人的上下文树

MYCIN 系统通过两个互相作用的子程序 MONITOR 和 FINDOUT 完成整个咨询和推理过程。

MONITOR 的功能是分析规则的前提条件是否满足，以决定拒绝该规则还是采用该规则，并将每次鉴定一个前提后的结果记录在动态数据库中。如果一个条件中所涉及的临床参数是未知的，则调用 FINDOUT 机制去得到这个消息。

FINDOUT 的功能是检查 MONITOR 所需要的参数，它可能已在动态数据库中，也可以通过用户提问获取。

FINDOUT 根据所需信息种类的不同采取不同的策略。对于化验数据，FINDOUT 首先向用户询问，如果用户不知道，再运用知识库进行推导，即检索知识库中可用推导该参数的规则，并调用 MONITOR 作用于这些规则；对于非化验数据，FINDOUT 首先运用知识库进行推导，如果规则推理不足以得出结论，再向用户询问。

5. 治疗方案选择

当目标规则的前提条件被确认，即诊断“病人患有细菌感染”后，MYCIN 系统开始处理目标规则的结论部分，即选择治疗方案。选择最佳治疗方案分以下两步，首先生成可能的“治疗方案表”，然后从表中选取对该病人的最佳用药配方。

（1）生成可能的“治疗方案表”

MYCIN 系统根据诊断出的细菌特征，选择用药方案。在知识库中存有相应的规则，指示对各种细菌的用药方案。例如有：

IF　细菌的特征是 Pseudomonas

THEN　建议在下列药物中选择治疗：

colistin(0.98)

polynyxin(0.96)

gentamicin(0.96)

carbenicillin(0.96)

sulfisoxazole(0.96)

规则中每个药物后的数值表示该药物对细菌的有效性。

MYCIN 系统应用这些药物选择规则，就能生成针对各种病菌的治疗方案表。这些方案可按其可信度的值进行排序。

（2）选择用药配方

MYCIN 系统根据下列原则从治疗方案中选择相应的用药配方：

该药物对细菌治疗的有效性。

该药物是否已用过。

该药物的副作用。

6. 知识获取

知识库中每条规则是医生的一条独立的经验，知识获取模块用于知识工程师增加和修改规则库中的规则。当输入新规则到规则库时，必须对原有规则进行检查、修改，并修改参数性质表和结点性质表。下面是系统获取一条规则的过程：

① 告诉专家新建立的规则的名字（实质上是规则序号）。

② 逐条获取前提，把前提从英文翻译成相应的 LISP 表达。

③ 逐条获取结论动作，把每一条从英文翻译为 LISP 表达。当有必要时应要求得到相应的规则可信度 *CF*。

④ 用 LISP-english 子程序将规则再翻译成英语，并显示给专家。

⑤ 提问专家是否同意这条翻译的规则；如果规则不正确，专家进行修改并回到步骤④。

⑥ 检查新规则与其他已在规则库中的旧规则之间是否矛盾。如果有必要，可以与专家交互来澄清指出的问题。

⑦ 如果有必要，可调用辅助分类规则对新规则分类。

⑧ 把规则加入到新规则前提中的临床参数性质的 LOOKHEAD 表中。

⑨ 把规则加入到新规则结论中的所有参数 CONTAIED - IN 表和 UPDATED - BY 表中。

⑩ 告诉专家系统新规则已是 MYCIN 系统的规则库中的一部分了。

上述步骤⑨确保 FINDOUT 在新的推导过程中搜索参数的 UPDATED - BY 表时能自动调用新规则。

MYCIN 系统的学习功能是有限的，例如新规则输入时涉及的参数和结点类型要求不超越系统已有的种类。另外。对新旧规则之间的矛盾、不一致等处理也是不全面的。

为了防止不熟练的用户随意输入知识而引起知识的混乱，系统采用二级存储方法。只有新的知识经试运行后证明其可靠，才能并入规则库中。

MYCIN 专家系统之所以重要有几个原因。首先，它证明了人工智能可以应用到实际的现实世界问题；其次，MYCIN 是新概念的试验，如解释机、知识的自动获取和今天可在许多专家系统中找到的智能指导；第三，它证实了专家系统外壳（SHELL）的可行性。

以前的专家系统如 DENDRAL，是一个把知识库中知识与推理机通过软件集成起来的单一系统。MYCIN 明确地把知识库与推理机分开。这对于专家系统技术的发展是极其重要的，因为这意味着专家系统的基本核心可以重用，也就是说，通过清空旧知识装入新领域的知识，新的专家系统可以比 DENDRAL 类型的系统创建快得多。处理推理和解释的 MYCIN 外壳部分，可以用新系统的知识重装。去掉医学知识的 MYCIN 外壳被称为 EMYCIN（基本的或空的 MYCIN）。

专家系统 MYCIN 能识别 51 种病菌，正确地处理 23 种抗菌素，可协助医生诊断、治疗细菌感染性血液病，为患者提供最佳处方。它成功地处理了数百病例，还通过了如下测试：用 MYCIN 与斯坦福大学医学院九名感染病医生分别对十例感染源不清楚的患者进行诊断并给出处方，由八位专家对他们的诊断进行评判，而且被测对象（即 MYCIN 及九位医生）互相隔离，评判专家亦不知道哪一份答卷是谁做的。评判内容包括两个方面，一是所开出的处方是否对症有效；另一是所开出的处方是否对其他可能的病原体也有效且用药又不过量。评判结果是：对第一个评判内容，MYCIN 与另外三名医生处方一致且有效；对第二个评判内容，MYCIN 的得分超过九名医生，显示出了较高的医疗水平。

7.8.2　地质勘探专家系统——PROSPECTOR

著名的地质勘探专家系统 PROSPECTOR 是美国斯坦福人工智能研究中心（SRI）于 1976 年开始研制的。该系统采用 LISP 语言编写。到 1980 年为止，PROSPECTOR 探测到价值 1 亿美元的矿物淀积层，带来了巨大的经济效益，目前它已成为世界上公认的著名专家系统之一。

1. PROSPECTOR 系统概述

(1) PROSPECTOR 系统的结构

系统由推理网络、匹配器、传送器、问答系统、英语分析器、解释系统、网络编译程序和知识获取系统组成,如图 7.17 所示。

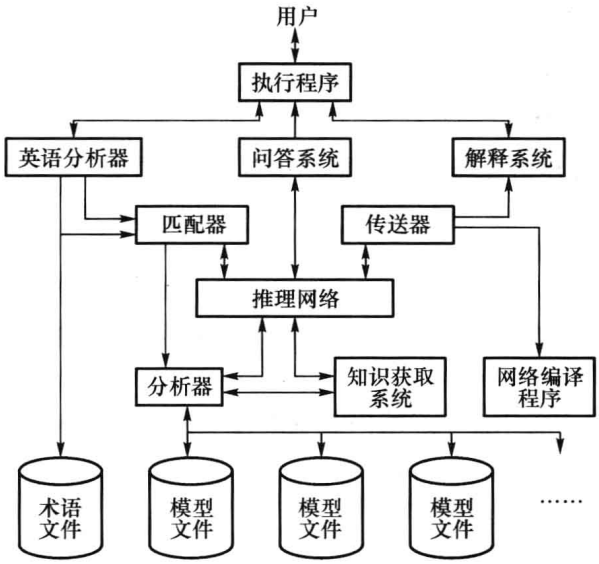


图 7.17 PROSPECTOR 系统的总体结构

PROSPECTOR 系统用语义网络表达知识。知识库由模型文件库和术语文件库组成,推理机具有层次结构,采用"从顶至底"的目标驱动推理控制策略,采用似然推理、逻辑推理、上下文推理相结合的推理方法。

PROSPECTOR 系统的各个组成部分的工作原理如下:

① 模型文件(模型知识库)。PROSPECTOR 系统有 12 个由模型文件组成的模型知识库。在系统内表达成推理规则网络,共有 1100 多条规则。规则的前提是地质勘探数据,结论是地质假设如矿床分类、含量、分布等。每个矿床模型以文件形式存放在磁盘上,可以由分析器调用。

② 术语文件(术语知识库)。有 400 种岩石、地质名字、地质年代和在语义网络中用的其他术语,也以外文件形式存储,作为术语知识库供系统调用。

③ 分析器。用于将矿床模型知识库中的模型文件转换成系统内部的推理网络。

④ 推理网络。PROSPECTOR 系统的推理网络是具有层次结构的与/或树,它将勘探数据和有关地质假设联系起来,进行从顶到底的逐级推理,上一级的结论作为下一级的证据,直到结论是可由勘探数据直接证实的端结点为止。

⑤ 匹配器。用于进行语义网络匹配。把一个模型和另一个模型连接在一起,同时也把用户输入的信息和一些模型连接起来。

⑥ 传送器。用于修正推理网络中模型空间状态变化的概率值。

⑦ 英语分析器。对用户以简单的英语陈述句输入的信息进行分析,并变换到语义网络上。

⑧ 问答系统。检查推理网络的推理过程及模型的运行情况,用户可以随时对系统进行查

询，系统也可以对用户提出问题，要求提供勘探证据。

⑨ 网络编译程序。通过钻井定位模型，根据推理结果，编制钻井井位选择方案，输出图像信息。

⑩ 解释系统。对用户解释有关结论和断言的推理过程、步骤和依据。

⑪ 知识获取系统。获取专家知识，增删、修改推理网络。

（2）PROSPECTOR 系统的功能

① 勘探结果评价。根据岩石标本及地质勘探数据，对矿区勘探结果进行综合评价。

② 矿区勘探评测。根据矿区勘探结果的综合评价，对矿藏资源进行估计和预测；对矿床分布、藏量、品位、开采价值等作出合理的地质假设（推理）和估算。

③ 编制井位计划。根据矿藏资源预测和估计及矿藏的分布、藏量、地质特性等，编制合理的开发计划和钻井井位布局方案。

应用 PROSPECTOR 系统，对美国华盛顿州的钼矿进行勘测，其结果完全为实际钻探所证实。PROSPECTOR 系统的成功应用，对专家系统进入商业市场，起到了重大的推动作用。

2. 推理网络

推理网络实际上是一个矿床模型经编码而成的网络，把探区证据和一些重要地质假设连接成一个有向图。在网络中，证据和假设是相对的，一个假设对于进一步推理来说又是证据，而一个证据对于下一级的推理来说又是假设。

PROSPECTOR 系统提供三种推理方法。

① 似然推理。根据 Bayes 原理的概率关系进行推理，用“似然率”表示规则的强度，描述不同的勘探证据对同一地质假设有不同的支持程度，说明某种结论的概率变化对其他结论的影响。规则强度由专家在矿床模型设计时提供，用语言表达，如完全肯定、有点可能等，然后转换成相应的概率值。在推理过程中采用 Bayes 公式进行概率计算。

② 逻辑推理，基于布尔逻辑关系的推理。在推理网络中，某些规则的证据（前提）和假设之间，具有布尔量的逻辑与、或、非关系，可用布尔代数进行推理。当证据与假设之间具有不确定的关系时，可采用模糊逻辑方法，合取（AND）中取组合中的最小值，析取（OR）则取最大值。

③ 上、下文推理，基于上、下文语义关系的推理。系统在推理过程中，有时需要考虑上、下文先后次序的语义关系。

7.9　专家系统的开发工具

专家系统的研制和开发是一件复杂、困难、费时的工作。为了提高专家系统设计和开发的效率，缩短研制周期，就需要使用专家系统开发工具，以便于提供系统设计和开发的计算机辅助手段和环境。

7.9.1　骨架系统

骨架系统是由已有的成功的专家系统演化而来的。它抽出了原系统中具体的领域知识，而保留了原系统的体系结构和功能，再把领域专用的界面改为通用界面。

在骨架系统中，知识表示模式、推理机制都是确定的。利用骨架系统作为开发工具，只要将

新的领域知识用骨架系统规定的模式表示出来并装入到知识库中就可以了。

在专家系统的建造中发挥了重要作用的骨架系统主要有 EMYCIN,KAS 和 EXPERT 等。

1. EMYCIN 系统

EMYCIN 系统是由 MYCIN 系统抽去原有的医学领域知识,保留骨架而形成的系统。它采用产生式规则表达知识、目标驱动的反向推理控制策略,特别适合开发领域咨询、诊断型专家系统。

EMYCIN 系统具有 MYCIN 系统的全部功能。

① 解释程序。系统可以向用户解释推理过程。

② 知识编辑程序及类英语的简化会话语言。EMYCIN 系统提供了一个开发知识库的环境,使得开发者可以使用比 LISP 更接近自然语言的规则语言来表示知识。

③ 知识库管理和维护手段。EMYCIN 系统提供的开发知识库的环境,还可以在进行知识编辑及输入时进行语法、一致性、是否矛盾和包含等检查。

④ 跟踪和调试功能。EMYCIN 系统还提供了有价值的跟踪和调试功能,试验过程中的状况都被记录并保留下来。

EMYCIN 系统的工作过程分两步。第一步为专家系统建立过程。在该过程中,首先知识工程师输入专家知识,知识获取和知识库构造模块把知识形式化,并对知识进行语法和语义检查,建立知识库。然后知识工程师调试并修改知识库。知识库调试正确后,一个用 EMYCIN 系统构造的专家系统即可交付使用。第二步为咨询过程。在该过程中,咨询用户提出目标假设,推理机制根据知识库中的知识进行推理,最后提出建议,作出决策,并通过解释模块向用户解释推理过程。

EMYCIN 系统已用于建造医学、地质、工程、农业和其他领域的诊断型专家系统。图 7.18 列出了借助于 EMYCIN 系统开发的一些专家系统。

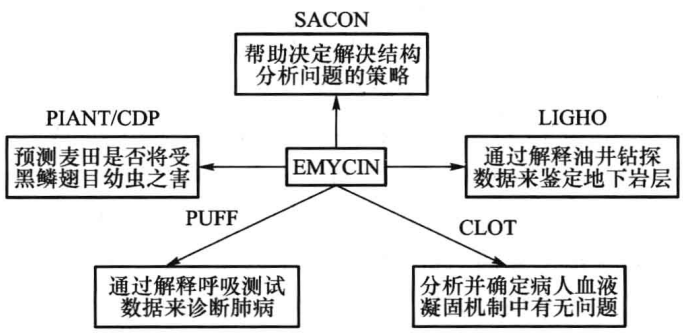


图 7.18 EMYCIN 系统的应用

2. KAS 系统

KAS 系统是由 PROSPECTOR 系统抽去原有的地质勘探知识而形成的。当把某个领域知识用 KAS 所要求的形式表示出来并输入到知识库中后,它就成为一个可用 PROSPECTOR 的推理机构来求解问题的专家系统。

KAS 系统采用产生式规则和语义网络相结合的知识表达方法及启发式正反向混合推理控制策略。KAS 系统在推理过程中的推理方向是不断改变的,其推理过程大致为:在 KAS 系统提示

下，用户以类似自然语言的形式输入信息，KAS 系统对其进行语法检查并将正确的信息转换为语义网络，然后与表示成语义网络形式的规则的前提条件相匹配，从而形成一组候选目标，并根据用户输入的信息使各候选目标得到不同的评分。接着 KAS 系统从这些候选的目标中选出一个评分最高的候选目标进行反向推理，只要一条规则的前提条件不能被直接证实或被否定，则反向推理就一直进行下去。当有证据表明某个规则的前提条件不可能有超过一定阈值的评分时，就放弃沿这条路线进行的推理，而选择其他的路线。

KAS 系统提供了一些辅助工具如知识编辑系统、推理解释系统、用户回答系统、英语分析器等，用来开发和测试规则和语义网络。

KAS 系统具有一个功能很强的网络编辑程序和网络匹配程序。网络编辑程序可以用来把用户输入的信息转化为相应的语义网络，并可用来检测语法错误和一致性等。网络匹配程序用于分析任意两个语义网络之间的关系，看其是否具有等价、包含、相交等关系，从而决定这两个语义网络是否匹配，同时它还可以用来检测知识库中的知识是否存在矛盾、冗余等。

KAS 系统适用于开发解释型专家系统。其典型应用如图 7.19 所示。

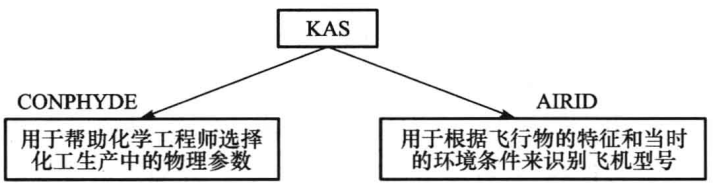


图 7.19　KAS 系统的应用

3. EXPERT 系统

EXPERT 系统是由美国 Rutgers 大学的威斯（Weiss）和库里科斯基（Kulikowski）等人在已成功开发的专家系统及工具如 CASNET 系统（青光眼诊断系统）等的基础上于 1981 年设计完成的一个骨架系统，适用于开发诊断和分类型专家系统。

EXPERT 系统的知识由假设、事实和决策规则三部分组成。与 EMYCIN 系统和 PROSPECTOR 系统不一样，在 EXPERT 系统中事实与假设是严格区分的。事实是有待观察、测量和确定的证据，如人的身高、血压等。事实是以真、假、数值或不知道的形式来回答系统提出的问题的。假设可以是由系统推出来的结论，例如，一个诊断就是一个假设。通常每个假设都有一个不确定性的度量值。规则用来描述事实和假设之间的逻辑关系，在 EXPERT 系统中具有三种形式的规则：FF 规则、FH 规则、HH 规则。

所谓 FF 规则，就是从事实到事实的规则，即从已知的事实推出另一些事实，从而可省去一些不必要的提问。被 FF 规则推出来的事实只能取真、假或不知道。例如

$$F(M,T) \rightarrow F(PREGP,\ F)\text{；如果 } M \text{ 为真，则 } PREGP \text{ 为假}$$

表示如果病人为男性，就不必做妊娠检查。

所谓 FH 规则，就是从事实到假设的规则，用来由事实的逻辑组合推出假设并确立其可信度。例如

$$F(A,T)\&F(B,F)\&[1:F(C,T),F(D,F)] \rightarrow H(E,0.5)$$

以上规则表示如果第一个事实（A 为真）成立、第二个事实（B 为假）成立、第三个事实（C 为真）和

第四个事实(D 为假)中有一个成立,则假设 E 成立的可能性为 0.5。

可信度的取值范围为 -1 到 1。1 表示绝对肯定,-1 表示绝对否定。在推理中可能有几条规则推出同一假设,这时可信度的绝对值最大的规则生效。

所谓 HH 规则,就是从假设到假设的规则,用来从已知的假设推出其他假设。在 EXPERT 系统中的 HH 规则中,出现在规则左部的假设的确定性程度需要一个数值区间表示。例如

$$H(A,0.2:1)\&H(B,0.1:1)\rightarrow H(C,1)$$

它表示如果对假设 A 有 0.2 到 1 的把握,并且对假设 B 有 0.1 到 1 的把握,则得出结论 C,其把握程度为 100%。另外,在 EXPERT 系统中为提高推理效率,还把若干条 HH 规则组成一个模块。在模块前另加条件,称为该规则的上下文。只有上下文为真时,该规则组内的规则才被启用。

EXPERT 系统推理的主要目的是达到正确的结论或提出合理的问题。其推理过程大致描述如下:

① 由事实对所有的 FF 规则进行推理,以取得尽可能多的事实。

② 从已有的事实出发,检查所有的 FH 规则,如果其左部为真,就将其右部的假设存入集合 PH 中。

③ 置集合 DH 为空。

④ 从已有事实出发,检查所有的 HH 规则的上下文,且对上下文条件成立的规则作如下处理:若规则的左部假设出现在 DH 或 PH 中,则令 H 的当前可信度为 PH 和 DH 中同一 H 的各可信度绝对值最大者,按 H 的这个可信度对此规则进行推理,并把结论存入 DH 中。若 DH 中已有这个假设 H,则仅保留其可信度绝对值最大的那一个。

⑤ 按假设所形成的推理网络进行推理,以最终得到假设的可信度值。

⑥ 对假设的选择除可按上述方法选择可信度最大的外,EXPERT 还设置了评分函数。

EXPERT 已被用于建造医疗、地质和其他一些领域的诊断专家系统。其典型应用如图 7.20 所示。

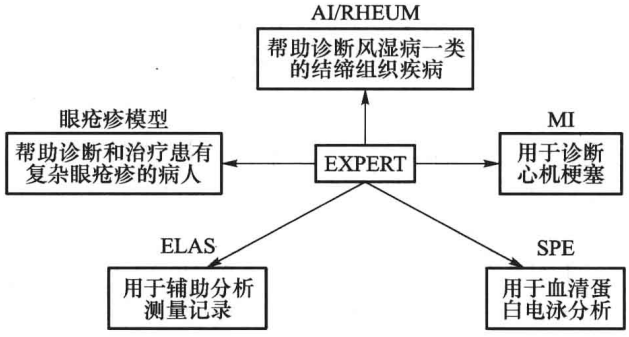


图 7.20 EXPERT 系统的应用

以上讨论了三种骨架系统。用骨架系统开发领域专家系统可以大大减少开发的工作量。但也存在一定的问题,主要问题是骨架系统只适用于建造与之类似的专家系统,因其推理机制和控制策略是固定的,所以局限性较大,灵活性差。

7.9.2　通用型知识表达语言

通用型知识表达语言并不严格地倾向特定的领域和范例系统，所以能够处理许多不同领域和类型的问题。目前这类通用语言已有很多，如 OPS5，ROSIE，HEARSAY Ⅲ，RLL，ART 等。这里只简单地介绍 OPS5。

OPS5 是美国卡内基－梅隆大学开发的一种通用知识表达语言，其特点是将通用的表达和控制结合起来。它提供了专家系统所需的基本机制，并不偏向于某些特定的问题求解策略和知识表达结构。OPS5 允许程序设计者使用符号表示并表达符号之间的关系，但并不事先定义符号和关系的含义。这些含义完全由程序设计者所写的产生式规则确定。

OPS5 由产生式规则库、推理机及数据库三部分组成。规则的一般形式为

P〈规则号〉〈前提〉→〈结论〉

其中前提是条件元的序列，而结论部分是基本动作构成的集合。OPS5 中定义了 12 个基本动作如 MAKE，MODIFY，REMOVE，WRITE 等。数据库用于存储当前求解问题的已知事实及中间结果等。数据库包含了一个不变的符号集合。符号结构有两种类型：符号向量、与属性－值元组相联系的对象。

推理机用规则库中的规则及数据库中的事实进行推理，具体步骤如下：

① 确定哪些规则的前提已满足（匹配）。

② 选择一个前提得到满足的规则。如果得不到满足的规则，则中止运行（解除冲突）。

③ 执行所选择规则的动作。

④ 转向①。

上述动作是行为序列的大框架，用户可以根据其意愿方式加入控制结构，即产生式系统本身确定使用什么样的控制及求解策略。

OPS5 已被用来开发许多专家系统。其典型应用如图 7.21 所示。

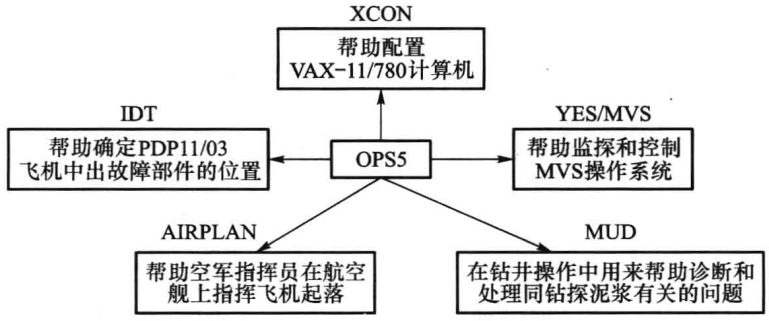


图 7.21　OPS5 系统的应用

7.9.3　专家系统开发环境

专家系统开发环境又称为专家系统开发工具包，它可为专家系统的开发提供多种方便的构件，如知识获取的辅助工具、适用各种不同知识结构的知识表示模式、各种不同的不确定推理机制、知识库管理系统以及各种不同的辅助工具、调试工具等。目前，国内外已有的专家系统开发

环境有 AGE,KEE 等。这里只简单地介绍 AGE。

AGE 是斯坦福大学研制的一个专家系统开发环境。它是一种典型的模块组合式开发工具。AGE 为用户提供了一个通用的专家系统结构框架,并将该框架分解为许多在功能和结构上较为独立的组件部件。这些组件已预先编制成标准模块存在系统中。用户可以通过以下两条途径构造自己的专家系统:

① 用户使用 AGE 现有的各种组件作为构造材料,很方便地来组合设计自己所需的系统。

② 用户通过 AGE 的工具界面,定义和设计各种所需的组成部件,以构成自己的专家系统。

AGE 采用了黑板模型来构造专家系统结构框架。应用 AGE 已经开发了一些专家系统,主要用于医疗诊断、密码翻译、军事科学等方面。

7.9.4 专家系统程序设计语言

PROLOG 和 LISP 是两种最主要的人工智能程序设计语言。

PROLOG 语言是一种以逻辑推理为基础的程序设计语言。无论是它描述求解问题的方式,还是其语言本身都与一般的程序设计语言有很大的差别。它最早在 20 世纪 70 年代初由英国爱丁堡大学的 R. Kowalski 首先提出,1986 年美国推出了 TURBO PROLOG 软件,能够适应个人计算机。现在,PROLOG 语言已经广泛应用于许多人工智能领域,包括定理证明、专家系统、自然语言理解等。

自从 LISP 创立以来,LISP 在美国一直居于人工智能语言的主导地位。由于它易于表达,许多早期专家系统外壳是用 LISP 建立的。但是,传统的计算机不能高效地执行 LISP,对用 LISP 编写的外壳,情况更糟。为了解决这个问题,几个公司开始提供专门设计的机器来执行 LISP 代码。这些 LISP 机完全使用 LISP,甚至把它作为汇编语言。但 LISP 机比传统的机器费用高,且是单用户的。用 LISP 编写的专家系统一般难以嵌入用其他语言编写的程序。

选择人工智能语言的一个重要原因是它提供了一些工具。由于可移植性、效率和速度等原因,许多专家系统工具,现在都用 C 语言编写或转换为 C 语言。由于面向对象程序设计语言以其类、对象、继承等机制,与人工智能的知识表示、知识库等产生了自然的联系。因而,现在面向对象型语言也成为一种人工智能程序设计语言。面向对象的程序设计也被广泛地用于人工智能程序设计,特别是专家系统程序设计。

7.10 小结

专家系统是基于知识的系统,用于在某种特定的领域中运用领域专家多年积累的经验和专业知识,求解只有专家才能解决的困难问题。

专家系统具有专家水平的专业知识,能进行有效的推理,具有启发性、灵活性、透明性、交互性等特点。

专家系统本身是一个程序,但它与传统程序不同。

专家系统一般包括人机接口、推理机、知识库、数据库、知识获取机构和解释机构等六个部分。各部分的关系如图 7.1 所示。

学习是一个有特定目的的知识获取过程,其内在行为是获取知识,积累经验,发现规律;外部

表现是改进性能，适应环境，实现系统的自我完善。

机器学习使计算机能模拟人的学习行为，自动地通过学习获取知识和技能，不断改善性能，实现自我完善。机器学习主要研究学习机理、学习方法、学习系统三方面问题。

能够在一定程度上实现机器学习的系统称为学习系统。一个学习系统一般应该由环境、学习、知识库、执行与评价等四个基本部分组成。

机械式学习通过直接记忆或者存储外部环境所提供的信息达到学习的目的，并在以后通过对知识库的检索得到相应的知识直接用来求解问题。

指导式学习由外部环境向系统提供一般性的指示或建议，系统把它们具体地转化为细节知识并送入知识库中。在学习过程中要反复对形成的知识进行评价，使其不断完善。指导式学习是一种比较实用的学习方法，目前应用得较多。

归纳学习是应用归纳推理进行学习的一类学习方法。按其有无教师指导可分为示例学习及观察与发现学习两种形式。

示例学习是通过从环境中取得若干与某概念有关的例子，经归纳得出一般性概念的一种学习方法。在这种学习方法中，外部环境（教师）提供一组例子（正例和反例），然后从这些特殊知识中归纳出适用于更大范围的一般性知识，它将覆盖所有的正例并排除所有反例。

观察学习用于对事例进行概念聚类，形成概念描述。概念聚类是一种观察学习。概念聚类的基本思想是把事例按一定的方式和准则进行分组，使不同的组代表不同的概念，并且对每一个组进行特征概括，得到一个概念的语义符号描述。

发现学习用于发现规律，产生定律或规则。发现学习是从系统的初始知识、观察事例或经验数据中归纳出规律或规则。

只有在专家系统开发是可能的、合理的并且合适的时候，才应该开发求解这个问题的专家系统。

骨架系统抽出了已成功应用的专家系统中具体的领域知识，而保留了原系统的体系结构和功能，再把领域专用的界面改为通用界面。在骨架系统中，知识表示模式、推理机制都是确定的。利用骨架系统作为开发工具，只要将新的领域知识用骨架系统规定的模式表示出来并装入到知识库中就可以了。

思考题

7.1　什么是专家系统？它有哪些基本特征？

7.2　专家系统有哪几部分组成？各部分的功能和结构如何？

7.3　专家系统与传统程序有何不同和相似之处？

7.4　什么是学习和机器学习？

7.5　试述机器学习系统的基本结构，并说明各部分的作用。

7.6　试述机器学习的模式。机器学习有哪些重要问题需要研究？

7.7　机械式学习的基本思想是什么？机械式学习有哪些重要问题需要研究？在设计机械式学习系统时，要考虑哪些问题？

7.8　什么是指导式学习？指导式学习的学习过程包括哪些步骤？

7.9　什么是归纳学习？它与演绎学习主要有哪些区别？

7.10　假设把桌子这个概念定义为一切具有大而平的顶部和至少有三条分开的腿的物体。试说明归纳算法如何学习这个概念，并给出一张桌子和其他近似物的描述序列。

7.11　示例学习的基本思想是什么？在示例学习中提供正、反例的信息源有哪些？

7.12　简述类比学习的基本思想。利用类比学习可以学到哪些东西？利用类比学习策略学习新概念的步骤是什么？

7.13　专家系统设计中要注意哪些问题？

7.14　简述专家系统的开发过程。

7.15　专家系统的主要类型和主要的应用领域有哪些？

7.16　描述专家系统中常用的正向推理和反向推理的算法流程。

第8章 人工神经网络及其应用

人工神经网络(artificial neural network,ANN)是一个用大量简单处理单元经广泛连接而组成的人工网络,是对人脑或生物神经网络若干基本特性的抽象和模拟。神经网络理论为机器学习等许多问题的研究提供了一条新的思路,目前已经在模式识别、机器视觉、联想记忆、自动控制、信号处理、软测量、决策分析、智能计算、组合优化问题求解、数据挖掘等方面获得成功应用。

神经网络的研究已经获得许多成果,提出了大量的神经网络模型和算法。本章着重介绍最基本、最典型、应用最广泛的 BP 神经网络和 Hopfield 神经网络及其在模式识别、联想记忆、软测量、智能计算、组合优化问题求解等方面的应用。在此基础上,读者可以进一步学习神经网络的其他内容。

8.1 神经元与神经网络

8.1.1 生物神经元结构

现代人的大脑内约有 10^{11} 个神经元,每个神经元与其他神经元之间约有 1 000 个连接,这样,大脑内约有 10^{14} 个连接。如果将一个人大脑中所有神经细胞的轴突和树突依次连接起来,拉成一条直线,则可以从地球连到月亮,再从月亮连到地球。人的智能行为就是由如此高度复杂的组织产生的。浩瀚的宇宙中,也许只有包含数千亿颗星球的银河系的复杂性能够与大脑相比。

从生物控制与信息处理的角度看,生物神经元结构如图 8.1 所示。

神经元的主体部分为细胞体。细胞体(soma)由细胞核、细胞质、细胞膜等组成。每个细胞体都有一个细胞核(cell nucleus),埋藏在细胞体之中,进行呼吸和新陈代谢等许多生化过程。神经元还包括树突和一条长的轴突,由细胞体向外伸出的最长的一条分支称为轴突即神经纤维。轴突末端部分有许多分支,叫轴突末梢。典型的轴突长 1 cm,是细胞体直径的 100 倍。一个神经元通过轴突末梢与 10 到 10 万个其他神经元相连接。轴突是用来传递和输出信息的,其端部的许多轴突末梢为信号输出端子,将神经冲动传给其他神经元。由细胞体向外伸出的其他许多较短的分支称为树突。树突相当于细胞的输入端,树突的全长各点都能接收其他神经元的冲动。神经冲动只能由前一级神经元的轴突末梢传向下一级神经元的树突或细胞体,不能作反方向的传递。

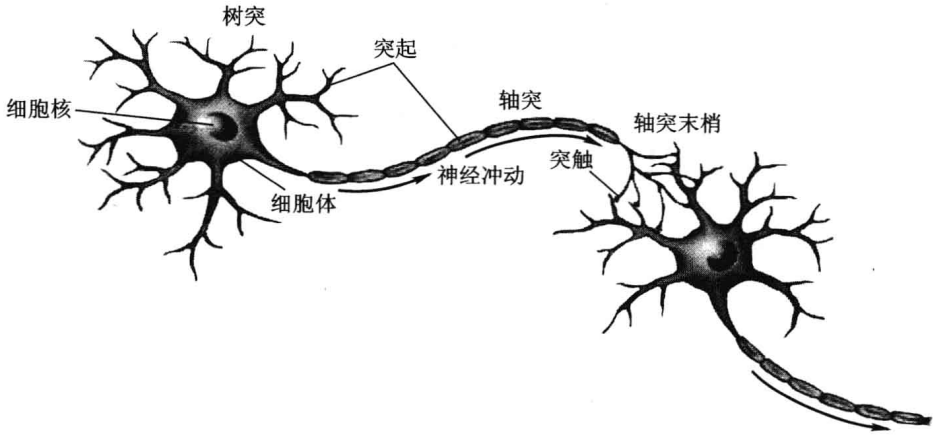


图 8.1 神经元构造

神经元具有两种常规工作状态:兴奋与抑制,即满足“0 - 1”律。当传入的神经冲动使细胞膜电位升高超过阈值时,细胞进入兴奋状态,产生神经冲动并由轴突输出;当传入的冲动使膜电位下降低于阈值时,细胞进入抑制状态,没有神经冲动输出。

8.1.2 神经元数学模型

早在 1943 年,美国神经和解剖学家麦克洛奇和数学家匹兹就提出了神经元的数学模型(M - P模型),从此开创了神经科学理论研究的时代。从 20 世纪 40 年代开始,根据神经元的结构和功能不同,先后提出的神经元模型有几百种之多。下面介绍神经元的一种所谓标准、统一的数学模型,它由三部分组成,即加权求和、线性动态系统和非线性函数映射,如图 8.2 所示。

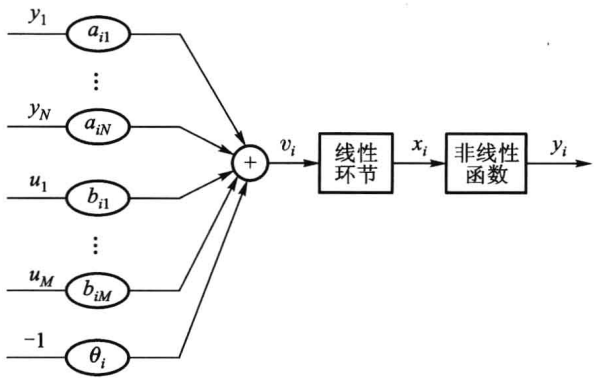


图 8.2 神经元数学模型

图 8.2 中,$y_i(t)$ 为第 i 个神经元的输出,θ_i 为第 i 个神经元的阈值($i=1,2,\cdots,N$); $u_k(t)$ ($k=1,2,\cdots,M$)为外部输入;a_{ij},b_{ik}为权值。

加权求和

$$v_i(t) = \sum_{j=1}^{N} a_{ij} y_j(t) + \sum_{k=1}^{M} b_{ik} u_k(t) - \theta_i \tag{8.1}$$

将式(8.1)记为矩阵形式

$$\boldsymbol{V}(t) = \boldsymbol{A}\boldsymbol{Y}(t) + \boldsymbol{B}\boldsymbol{U}(t) - \boldsymbol{\theta} \tag{8.2}$$

式中，$\boldsymbol{A} = [a_{ij}]_{N\times N}$，$\boldsymbol{B} = [b_{ik}]_{N\times M}$，$\boldsymbol{V} = [v_1, \cdots, v_N]^{\mathrm{T}}$，$\boldsymbol{Y} = [y_1, \cdots, y_N]^{\mathrm{T}}$，$\boldsymbol{U} = [u_1, \cdots, u_M]^{\mathrm{T}}$，$\boldsymbol{\theta} = [\theta_1, \cdots, \theta_N]^{\mathrm{T}}$。

线性环节的传递函数描述为

$$X_i(s) = H(s) V_i(s) \tag{8.3}$$

式中，$H(s)$通常取为：1；$\frac{1}{s}$；$\frac{1}{Ts+1}$；e^{-Ts}及其组合等。

神经元输出 $y_i(t)$与 $y_i(t)$之间的非线性函数关系最常用的有以下两种：

(1) 阶跃函数

$$f(x_i) = \begin{cases} 1 & x_i \geqslant 0 \\ 0 & x_i < 0 \end{cases} \tag{8.4}$$

或

$$f(x_i) = \begin{cases} 1 & x_i \geqslant 0 \\ -1 & x_i < 0 \end{cases} \tag{8.5}$$

(2) S 型函数

它具有平滑和渐近性，并保持单调性，是最常用的非线性函数。最常用的 S 型函数为 Sigmoid 函数

$$f(x_i) = \frac{1}{1 + \mathrm{e}^{-\alpha x_i}} \tag{8.6}$$

式中，α 可以控制其斜率。

对于需要神经元输出在[-1,1]区间时，S 型函数可以选为双曲线正切函数(hyperbolic tangent function)

$$f(x_i) = \frac{1 - \mathrm{e}^{-\alpha x_i}}{1 + \mathrm{e}^{-\alpha x_i}} \tag{8.7}$$

8.1.3　神经网络的结构与工作方式

1. 神经网络的结构

神经网络是由众多简单的神经元连接而成的一个网络。尽管每个神经元结构、功能都不复杂，但神经网络的行为并不是各单元行为的简单相加，网络的整体动态行为则是极为复杂的，可以组成高度非线性动力学系统，从而可以表达很多复杂的物理系统，表现出一般复杂非线性系统的特性(如不可预测性、不可逆性、多吸引子、可能出现混沌现象等)和作为神经网络系统的各种性质。神经网络具有大规模并行处理能力和自适应、自组织、自学习能力以及分布式存储等特点，在许多领域得到了成功的应用，展现了非常广阔的应用前景。

众多的神经元的轴突和其他神经元或者自身的树突相连接，构成复杂的神经网络。根据神经网络中神经元的连接方式可以划分为不同类型的结构。目前人工神经网络主要有前馈型和反

馈型两大类。

(1) 前馈型

前馈型神经网络中,各神经元接受前一层的输入,并输出给下一层,没有反馈。前馈网络可分为不同的层,第 i 层只与第 $i-1$ 层输出相连,输入与输出的神经元与外界相连。后面着重介绍的 BP 神经网络就是一种前馈型神经网络。

(2) 反馈型

在反馈型神经网络中,存在一些神经元的输出经过若干个神经元后,再反馈到这些神经元的输入端。最典型的反馈型神经网络是 Hopfield 神经网络。它是全互联神经网络,即每个神经元和其他神经元都相连。

2. 神经网络的工作方式

当满足兴奋条件时,神经网络中的神经元就会改变为兴奋状态,当不满足兴奋条件时,它就会改变为抑制状态。如果神经网络中各个神经元同时改变状态,则称为同步(synchronous)工作方式;如果神经网络中神经元是一个一个地改变状态,即当某个神经元改变状态时,其他神经元保持状态不变,称为异步(asynchronous)工作方式。神经网络在不同的工作方式下的性能有些差异。

8.1.4 神经网络的学习

神经网络方法是一种知识表示方法和推理方法。神经网络知识表示方法与谓词、产生式、框架、语义网络等完全不同。谓词、产生式、框架、语义网络等方法是知识的显式表示。例如,在产生式系统中知识独立地表示为一条规则。神经网络知识表示是一种隐式的表示方法。在这里,将某一问题的若干知识通过学习表示在同一网络中。

神经网络的学习是指调整神经网络的连接权值或者结构,使输入输出具有需要的特性。

1944 年赫布(Hebb)提出了改变神经元连接强度的 Hebb 学习规则。由于 Hebb 学习规则的基本思想很容易被接受,得到了较为广泛的应用,至今仍在各种神经网络模型的研究中起着重要的作用。但近年来神经科学的许多发现都表明,Hebb 学习规则并没有准确反映神经元在学习过程中突触变化的基本规律。

Hebb 学习规则:当某一突触两端的神经元同时处于兴奋状态,那么该连接的权值应该增强。用数学方式描述调整权值 w_{ij} 的方法为

$$w_{ij}(k+1) = w_{ij}(k) + \alpha y_i(k) y_j(k) \quad (\alpha > 0) \tag{8.8}$$

式中,$w_{ij}(k+1)$ 为权值的下一步值;$w_{ij}(k)$ 为权值的当前值。

8.2 BP 神经网络及其学习算法

8.2.1 BP 神经网络的结构

BP 神经网络(back-propagation neural network)就是多层前向网络,其结构如图 8.3 所示。

设 BP 神经网络具有 m 层。第一层称为输入层,最后一层称为输出层,中间各层称为隐层。输入层起缓冲存储器的作用,把数据源加到网络上,因此,输入层的神经元的输入与输出关系一

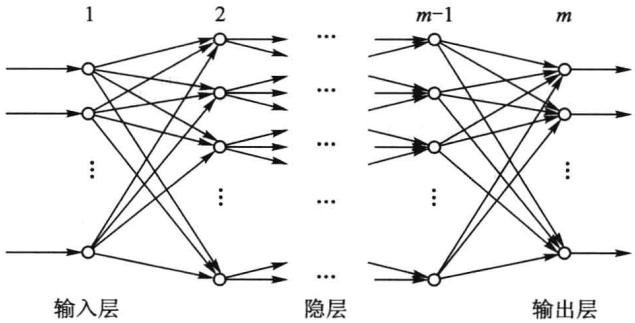


图 8.3　BP 神经网络结构

般是线性函数。隐层中各个神经元的输入与输出关系一般为非线性函数。隐层 k 与输出层中各个神经元的非线性输入与输出关系记为 $f_k(k=2,\cdots,m)$。由第 $k-1$ 层的第 j 个神经元到第 k 层的第 i 个神经元的连接权值为 w_{ij}^k。并设第 k 层中第 i 个神经元输入的总和为 u_i^k，输出为 y_i^k，则各变量之间的关系为

$$y_i^k = f_k(u_i^k)$$
$$u_i^k = \sum_j w_{ij}^{k-1} y_j^{k-1} k = 2,\cdots,m \tag{8.9}$$

当 BP 神经网络输入数据 $X=[x_1,x_2,\cdots,x_{p_1}]^{\mathrm{T}}$（设输入层有 p_1 个神经元），从输入层依次经过各隐层结点，可得到输出数据 $Y=[y_1^m,y_2^m,\cdots,y_{p_m}^m]^{\mathrm{T}}$（设输出层有 p_m 个神经元）。因此，可以把 BP 神经网络看成是一个从输入到输出的非线性映射。

给定 N 组输入输出样本为 $\{X_{si},Y_{si}\}$，$i=1,2,\cdots,N$。如何调整 BP 神经网络的权值，使 BP 神经网络输入为样本 X_{si} 时，神经网络的输出为样本 Y_{si}。这就是 BP 神经网络的学习问题。可见，BP 学习算法是一种有教师学习算法。

要解决 BP 神经网络的学习问题，关键是解决两个问题：

第一，是否存在一个 BP 神经网络能够逼近给定的样本或者函数。

下述定理可以回答这个问题。

Kolmogorov 定理：给定任意 $\varepsilon>0$，对于任意的连续函数 f，存在一个三层 BP 神经网络，其输入层有 p_1 个神经元，中间层有 $2p_1+1$ 个神经元，输出层有 p_m 个神经元，它可以在任意 ε 平方误差精度内逼近 f。

上述定理不仅证明了映射网络的存在，而且说明了映射网络的结构。就是说，总存在一个结构为 $p_1\times(2p_1+1)\times p_m$ 的三层前向神经网络能够精确地逼近任意的连续函数 f。但对多层 BP 神经网络，如何合理地选取 BP 网络的隐层数及隐层的结点数，目前尚无有效的理论和方法。

第二，如何调整 BP 神经网络的权值，使 BP 神经网络的输入与输出之间的关系与给定的样本相同。BP 学习算法给出了具体的调整算法。

8.2.2　BP 学习算法

BP 学习算法最早是由 Werbos 在 1974 年提出的。Rumelhart 等于 1985 年发展了 BP 学习算法，实现了 Minsky 多层感知器的设想。

BP 学习算法是通过反向学习过程使误差最小，因此选择目标函数为

$$\min J = \frac{1}{2}\sum_{j=1}^{p_m}(y_j^m - y_{sj})^2 \tag{8.10}$$

即选择神经网络权值使期望输出 y_{sj} 与神经网络实际输出 y_j^m 之差的平方和最小。

这种学习算法实际上是求目标函数 J 的极小值，约束条件是式(8.9)，可以利用非线性规划中的“最快下降法”，使权值沿目标函数的负梯度方向改变，因此，神经网络权值的修正量为

$$\Delta w_{ij}^{k-1} = -\varepsilon\frac{\partial J}{\partial w_{ij}^{k-1}} \quad (\varepsilon > 0) \tag{8.11}$$

式中，ε 为学习步长，一般小于 0.5。

下面推导 BP 学习算法。先求 $\frac{\partial J}{\partial w_{ij}^{k-1}}$

$$\frac{\partial J}{\partial w_{ij}^{k-1}} = \frac{\partial J}{\partial u_i^k}\frac{\partial u_i^k}{\partial w_{ij}^{k-1}} = \frac{\partial J}{\partial u_i^k}\frac{\partial}{\partial w_{ij}^{k-1}}\left(\sum_j w_{ij}^{k-1}y_j^{k-1}\right) = \frac{\partial J}{\partial u_i^k}y_j^{k-1}$$

记

$$d_i^k = \frac{\partial J}{\partial u_i^k} \quad (k = 2,\cdots,m)$$

则

$$\Delta w_{ij}^{k-1} = -\varepsilon d_i^k y_j^{k-1} \quad (k = 2,\cdots,m) \tag{8.12}$$

下面推导计算 d_i^k 的公式

$$d_i^k = \frac{\partial J}{\partial u_i^k} = \frac{\partial J}{\partial y_i^k}\frac{\partial y_i^k}{\partial u_i^k} = \frac{\partial J}{\partial y_i^k}f'_k(u_i^k)$$

下面分两种情况求 $\frac{\partial J}{\partial y_i^k}$。

① 对输出层(第 m 层)的神经元，即 $k=m$，$y_i^k = y_i^m$，由误差定义式得

$$\frac{\partial J}{\partial y_i^k} = \frac{\partial J}{\partial y_i^m} = y_i^m - y_{si} \tag{8.13}$$

则

$$d_i^m = (y_i^m - y_{si})f'_m(u_i^m) \tag{8.14}$$

② 若 i 为隐单元层 k，则有

$$\frac{\partial J}{\partial y_i^k} = \sum_l \frac{\partial J}{\partial u_l^{k+1}}\frac{\partial u_l^{k+1}}{\partial y_i^k} = \sum_l d_l^{k+1}w_{li}^k$$

则

$$d_i^k = f'_k(u_i^k)\sum_l d_l^{k+1}w_{li}^k \tag{8.15}$$

综上所述，BP 学习算法可以归纳为

$$\Delta w_{ij}^{k-1} = -\varepsilon d_i^k y_j^{k-1} \tag{8.16a}$$

$$d_i^m = (y_i^m - y_{si})f'_m(u_i^m) \tag{8.17a}$$

$$d_i^k = f'_k(u_i^k)\sum_l d_l^{k+1} w_{li}^k \quad (k = m-1,\cdots,2) \tag{8.18a}$$

若取 $f_k(\cdot)$ 为 S 型函数，即

$$y_i^k = f_k(u_i^k) = \frac{1}{1+\mathrm{e}^{-u_i^k}}$$

则

$$\frac{\partial y_i^k}{\partial u_i^k} = f'_k(u_i^k) = \frac{\mathrm{e}^{-u_i^k}}{[1+\mathrm{e}^{-u_i^k}]^2} = y_i^k(1-y_i^k)$$

则 BP 学习算法可以归纳为

$$\Delta w_{ij}^{k-1} = -\varepsilon d_i^k y_j^{k-1} \tag{8.16b}$$

$$d_i^m = y_i^m(1-y_i^m)(y_i^m - y_{si}) \tag{8.17b}$$

$$d_i^k = y_i^k(1-y_i^k)\sum_l d_l^{k+1} w_{li}^k \quad (k = m-1,\cdots,2) \tag{8.18b}$$

从以上公式可以看出，求第 k 层的误差信号 d_i^k，需要上一层的 d_l^{k+1}。因此，误差函数的求取是一个始于输出层的反向传播的递归过程，所以称为反向传播（back propagation，BP）学习算法。通过多个样本的学习，修改权值，不断减少偏差，最后达到满意的结果。

8.2.3　BP 学习算法的实现

BP 学习算法的程序框图如图 8.4 所示。

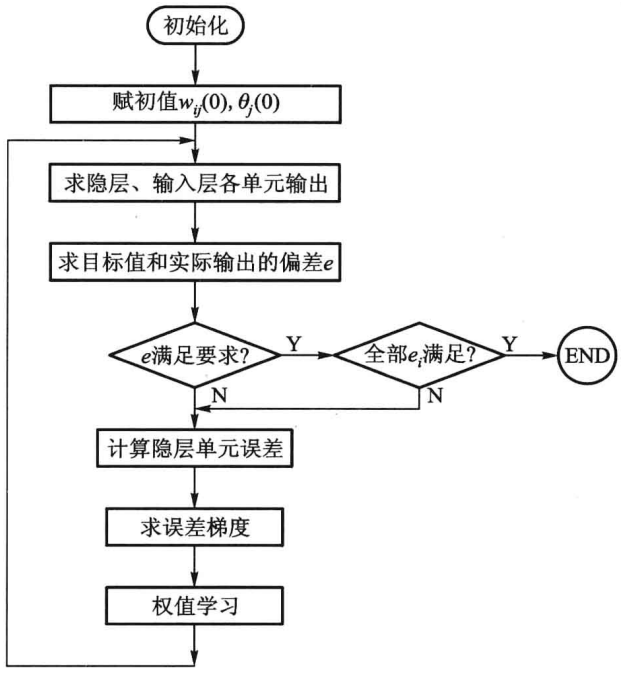


图 8.4　BP 算法程序框图

在 BP 算法实现时，还要注意下列问题：

① 训练数据预处理。预处理过程包含一系列的线性的特征比例变换，将所有的特征变换到[0,1]或者[-1,1]区间内，使得在每个训练集上，每个特征的均值为0，并且具有相同的方差。预处理过程也称为尺度变换(scaling)，或者规格化(normalization)。

② 后处理过程。当应用神经网络进行分类操作时，通常将输出值编码成所谓的名义变量，具体的值对应类别标号。在一个两类分类问题中，可以仅使用一个输出，将它编码成一个二值变量(例如+1，-1)。当具有更多的类别时，应当为每个类别分配一个代表类别决策的名义输出值。例如，对于一个三类分类问题，可以设置三个名义输出，每个名义输出取值为{+1，-1}，对应的各个类别决策为{+1，-1，-1}，{-1，+1，-1}，{-1，-1，+1}。利用阈值可以将神经网络的输出值变换成为合适的名义输出值。

③ 初始权值的影响及设置。和所有梯度下降算法一样，初始权值对 BP 神经网络的最终解有很大的影响。虽然全部设置为0显得比较自然，但从式(8.16)可以看出这将导致很不理想的结果。如果输出层的权值全部为0，则反向传播误差也将为0，输出层前面的权值将不会改变。因此，一般以一个均值为0的随机分布设置 BP 神经网络的初始权值。

8.3 BP 神经网络的应用

8.3.1 BP 神经网络在模式识别中的应用

模式识别主要研究用计算机模拟生物、人的感知，对模式信息(如图像、文字、语音等)进行识别和分类。传统人工智能的研究部分地显示了人脑的归纳、推理等智能。但是，对于人类底层的智能，如视觉、听觉、触觉等方面，现代计算机系统的信息处理能力还不如一个幼儿园的孩子。

神经网络模型模拟了人脑神经系统的特点：处理单元的广泛连接，并行分布式信息储存、处理，自适应学习能力等。神经元网络的研究为模式识别开辟了新的研究途径。与模式识别的传统方法相比，神经网络方法具有较强的容错能力、自适应学习能力、并行信息处理能力。

例 8.1 设计一个三层 BP 网络对数字0至9进行分类。训练数据如图8.5所示，测试数据如图8.6所示。

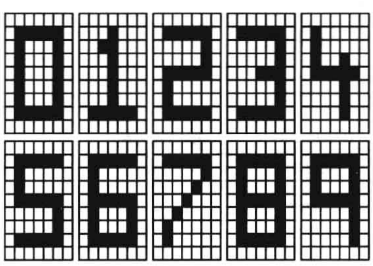


图 8.5 数字分类训练数据

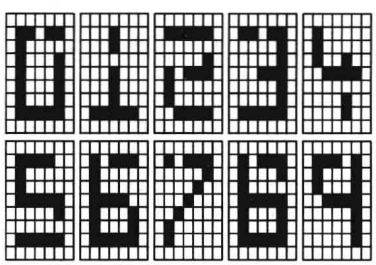


图 8.6 数字分类测试数据

解 该分类问题有10类，且每个目标向量应该是这10个向量中的一个。目标值由数字1至9的9个向量中的一个表示，0是由所有结点的输出全为0来表示。每个数字用9×7的网格表示，灰色像素代表0，黑色像素代表1。将网格表示为0或者1的长位串。位映射由左上角开

始向下直到网格的整个一列，然后重复其他列。

例如，数字“1”的网格的数字串为{0,0,0,0,0,0,0,0,0;0,0,0,0,0,0,0,1,0;0,0,1,0,0,0,0,1,0;0,1,1,1,1,1,1,1,0;0,0,0,0,0,0,0,1,0;0,0,0,0,0,0,0,1,0;0,0,0,0,0,0,0,0,0}。

选择网络结构为 63－6－9。9×7 个输入结点，对应上述网格的映射。9 个输出结点对应 10 种分类。使用的学习步长为 0.3。训练 1000 个周期，如果输出结点的值大于 0.9，则取为 1，如果输出结点的值小于 0.1，则取为 0。

当训练成功后，对如图 8.6 所示测试数据进行测试。图 8.6 所示测试数据都有一个或者多个位丢失。测试结果表明：除了 8 以外，所有被测的数字都能够被正确地识别。对于数字 8，第 8 个结点的输出值为 0.49，而第 6 个结点的输出值为 1，表明第 8 个样本网络是模糊的，可能是数字 6，也可能是数字 8，但也不完全确信是两者之一。实际上，人识别这个数字时也会发生这种错误。

8.3.2　BP 神经网络在软测量中的应用

1. 软测量技术

在工业过程中，为了保证产品的质量和生产的连续平稳，需要对与品质密切相关的过程变量进行实时监控，然而常常存在一些重要的变量无法或难以用传感器直接检测。例如，化学反应器的产品质量、发酵罐的菌体浓度等都是很难在线测量的。有时，这类变量可以采用在线分析仪直接测量，但需要很大的设备投资，并且会有很大的滞后。对于这类变量可以采用软测量方法进行间接测量。

软测量是利用一些可测变量去估计推测那些难以测量的变量。

软测量技术是基于 20 世纪 70 年代 Brosillow 提出的推断控制。近年来，软测量技术在理论研究和工业应用方面都取得了较大进展。理论研究已经历了从线性到非线性、从静态到动态、从无校正到有校正功能的过程。它能够连续计算那些不可测或难以检测的参数，在一定程度上可以代替在线分析仪表。

软测量的基本过程是：依据生产过程中有关的变量间的关联，选择与被估计变量（难以或无法在线检测的）相关的一组可测变量，构造以可测变量为输入、被估计变量为输出的数学模型，用计算机软件实现被估计变量的最佳估计。在软测量系统中，被估计的变量称为主导变量，与被估计变量相关的一组可测变量称为辅助变量。软测量的基本过程如图 8.7 所示。

图 8.7　软测量的基本过程

这类数学模型及相应的计算机软件被称为软测量器或软仪表。软测量的输出可作为控制系统的被控变量或反映过程特征的工艺参数，为优化控制与决策提供基础。

软测量系统的设计主要有辅助变量的选择，数据采集与处理，软测量模型的建立等。

辅助变量的选择。辅助变量的选择对主导变量的估计有重要的影响。辅助变量的选择包括

变量类型、变量数量和检测点位置的选择。这三个方面是互相关联、互相影响的，不但由过程特性决定，还受设备价格和可靠性、安装和维护的难易程度等外部因素制约。辅助变量数目的下限是被估计的变量数，而最佳数目则与过程的自由度、测量噪声以及模型的不确定性有关。辅助变量的选择确定了软测量的输入信息矩阵，因而直接决定了软测量模型的结构和输出。

数据采集与处理。测量数据通过安装在现场的传感器、变送器等仪表获得，受到仪表精度、测量原理和测量方法、生产环境的影响，测量数据都不可避免地含有误差，甚至有严重的误差。如果将这些数据直接用于软测量，则很难得到正确的主导变量估计值。因此，必须对原始数据进行预处理（数据校正和数据变换）以得到精确可靠的数据。这是软测量成败的关键，具有十分重要的意义。

软测量模型的建立。软测量模型是软测量技术的核心。它不同于一般意义下的数学模型，它主要是通过辅助变量来获得对主导变量的最佳估计。目前已经提出了许多建模方法。例如，基于过程机理分析的机理建模方法、基于实验数据的系统辨识方法等。事实上，实际过程的输入和输出的关系可能是复杂的，很难用一个简单的函数表示，特别是很难弄清楚数据中输入和输出的关系。BP 神经网络能够逼近任意复杂函数，具有学习功能，非常适合于作为软测量模型，可以从实验数据中“学习”到这个关系。

下面以污水处理质量指标检测为例，介绍神经网络的软测量方法。

2. 污水处理过程神经网络软测量模型

随着现代工业的迅猛发展，工业污水对人类赖以生存的水资源的破坏越来越大，污水处理成为国内外迫切需要解决的问题。近几年来水处理过程控制成为国内外的研究热点。但是，由于目前检测污水质量指标的传感器实时性差、误差大、价格昂贵，许多方法采用开环控制，或者选择其他间接指标进行控制，影响了控制效果。因此，在线检测污水质量指标成为提高控制质量的关键。

随着能够检测污水生物处理过程参数的检测仪器的发展，很多研究人员对污水生物处理中各种底物（C、N、P）和各种检测参数之间的变化规律进行了研究，试图采用间接指标实现反馈控制。但由于这些关系的复杂性，很难基于机理分析方法给出确切的关系。下面运用 BP 神经网络实现污水处理质量指标软测量。

污水所含的污染物质千差万别，主要指标有物理、化学、生物三大类。从水体有机污染物看，其主要危害是消耗水中溶解氧。一般采用生物化学需氧量（BOD）、化学需氧量（COD）等指标来反映水中需氧有机物的含量。污水中的氮（N）、磷（P）虽为植物营养元素，但过多的 N、P 进入天然水体却易导致富营养化。因此，BOD、COD、N 和 P 是反映水质好坏的重要指标。污水中 COD 特别是 BOD 的测定方法较为烦琐且耗时很长，使测出结果的时间严重滞后实际运行的时间，不能及时反应实际情况。目前，虽然出现了 COD 浓度在线检测仪，但仍存在滞后、较大误差及价格昂贵等问题。

对于污水的监测和处理已经提出了许多方法。目前广泛使用的是序批式活性污泥法（SBR）。SBR 法是由曝气池、沉淀池、污泥回流和剩余污泥排除几个系统组成，如图 8.8 所示。

污水和回流的活性污泥一起进入曝气池形成混合液，通过曝气设备充入空气，空气中的氧溶入污水使混合液产生好氧代谢反应，使混合液得到足够的搅拌而呈悬浮状态，然后流入沉淀池，混合液中的悬浮固体在沉淀池中沉淀下来和水分离，流出沉淀池的就是净化水。沉淀池中的污泥大部分回流，成为回流污泥。回流污泥的目的是使曝气池内保持一定的悬浮固体浓度，也就是

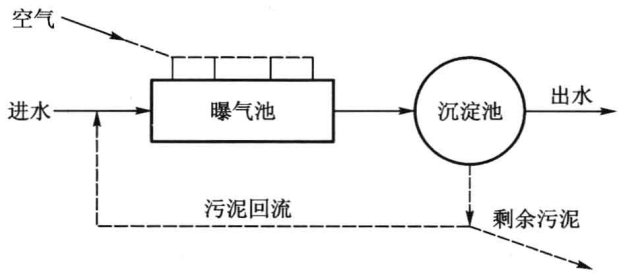


图 8.8　活性污泥法的基本流程

保持一定的微生物浓度。

在 SBR 法中，根据出水 BOD（或 COD）浓度的变化来调节控制反应（曝气）时间是最具有保证的方法。国内外的学者曾采用 ORP、DO、PH 或污泥浓度 MLSS 来间接控制污水处理过程，但这些单个的控制参数在动态的污水处理过程中仍未取得好的效果。单独使用 ORP，有时由于污水性质或 DO 的差异、特征点不明显，而且 ORP 过于敏感，小的误差会带来控制的不稳定；而单独使用 DO、PH 或 MLSS 则不能反映处理的整个过程（有机物、氮和磷的去除）。因此，必须联合这三者作为控制参数才能做到高效节能，取得最满意的效果。

综合上述分析，可以选择 BOD、COD、N 和 P 作为软测量模型的主导变量，选择 ORP、DO、PH 和 MLSS 作为辅助变量。

利用人工神经网络的软测量技术，建立一个三层 BP 网络，如图 8.9 所示。它采用能够在线检测的 ORP、DO、PH 和 MLSS 作为系统输入信号，实现污水的 COD、BOD、N 和 P 等参数的软测量，从而估计进水水质，决定曝气量大小、反应时间，控制出水水质，实现对污水处理过程的实时控制。

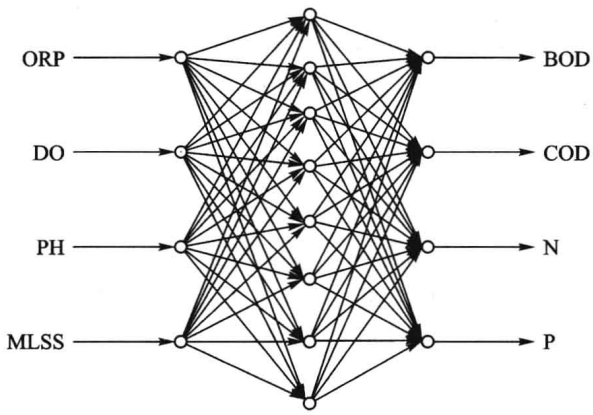


图 8.9　污水质量指标 BP 神经网络图

根据污水处理质量指标神经网络软测量的要求，对实际污水处理运行过程的一些分析仪表得到的实验数据进行处理，可以得到反映污水处理质量指标复杂关系的实验数据，将其作为 BP 神经网络的训练数据，对神经网络进行训练。训练好的 BP 神经网络作为软测量仪表，能够实时估计污水的 COD、BOD、N、P 等参数，有效地解决污水处理质量的实时估计问题。

基于 BP 神经网络的污水处理软测量系统，能够实现污水的 COD、BOD、N、P 等参数的实时测量，估计处理的污水的水质，从而决定曝气量大小、反应时间，控制出水水质，实现各种活性污泥法过程的实时反馈控制，提高污水处理系统的可靠性并节约能耗，具有重要的理论意义和应用价值。

8.4 Hopfield 神经网络及其改进

从 20 世纪 60 年代初到 80 年代初，神经网络的研究处于“冰河期”，到了 20 世纪 80 年代中期，美国加州理工学院生物物理学家霍普菲尔德（J. J. Hopfield）在神经网络建模及应用方面的开创性成果，在世界范围内重新掀起了神经网络的研究热潮。

1982 年和 1984 年，霍普菲尔德先后提出离散型 Hopfield 神经网络和连续型 Hopfield 神经网络，引入“计算能量函数”的概念，给出了网络稳定性判据，尤其是给出了 Hopfield 神经网络的电子电路实现，为神经计算机的研究奠定了基础，同时开拓了神经网络用于联想记忆和优化计算的新途径，从而有力地推动了神经网络的研究。这两种模型是目前最重要的神经优化计算模型。

8.4.1 离散型 Hopfield 神经网络

Hopfield 神经网络（HNN）是全互联反馈神经网络，它的每一个神经元都和其他神经元相连接。

具有 N 个神经元的离散型 Hopfield 神经网络 NN，可由一个 $N \times N$ 阶矩阵 $\boldsymbol{w} = [w_{ij}]_{N \times N}$ 和一个 N 维行向量 $\boldsymbol{\theta} = [\theta_1, \theta_2, \cdots, \theta_N]^{\mathrm{T}}$ 所唯一确定，记为 $NN = (\boldsymbol{w}, \boldsymbol{\theta})$，其中，$w_{ij}$ 为从第 j 个神经元的输出到第 i 个神经元的输入之间的连接权值，表示神经元 i 与 j 的连接强度，且 $w_{ji} = w_{ij}, w_{ii} = 0$；$\theta_i$ 表示神经元 i 的阈值。若用 $v_i(k)$ 表示 k 时刻神经元所处的状态，那么神经元 i 的状态随时间变化的规律（又称演化律）为

二值硬限器（binary hardlimiters）：

$$v_i(k+1) = \begin{cases} 1 & u_i(k) \geqslant 0 \\ 0 & u_i(k) < 0 \end{cases} \tag{8.19}$$

或者为双极硬限器：

$$v_i(k+1) = \begin{cases} 1 & u_i(k) \geqslant 0 \\ -1 & u_i(k) < 0 \end{cases} \tag{8.20}$$

式中

$$u_i(k) = \sum_{\substack{j=1 \\ j \neq i}}^{n} w_{ij} v_j(k) - \theta_i \quad (1 \leqslant i \leqslant N) \tag{8.21}$$

Hopfield 神经网络可以是同步工作方式，也可以是异步工作方式。即神经元更新既可以同步（并行）进行，也可以是异步（串行）进行。在同步进行时，神经网络中所有神经元的更新同时进行。在异步进行时，在同一时刻只有一个神经元更新，而且这个神经元在网络中每个神经元都更新之前不会再次更新。在异步更新时，神经元的更新顺序可以是随机的。

Hopfield 神经网络中的神经元相互作用，不断演化。如果神经网络在演化过程中，从某一时刻开始，神经网络中的所有神经元的状态不再改变，则称该神经网络是稳定的。

Hopfield 神经网络是高维非线性动力学系统，可能有若干个稳定状态。从任一初始状态开始运动，总可以达到某个稳定状态。这些稳定状态可以通过改变各个神经元之间的连接权值得到。

稳定性是 Hopfield 神经网络的最重要的特性。下面分析 Hopfield 神经网络的稳定性。

Hopfield 神经网络是一个多输入多输出带阈值的二态非线性动力学系统，所以，类似于李雅普诺夫稳定性分析方法，在 Hopfield 神经网络中，也可以构造一种 Lyapunov 函数，在满足一定的参数条件下，该函数值在网络运行过程中不断降低，最后趋于稳定的平衡状态。Hopfield 引入这种能量函数作为网络计算求解的工具，因此，常常称它为计算能量函数。

离散型 Hopfield 神经网络的计算能量函数定义为

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N} w_{ij}v_iv_j + \sum_{i=1}^{N}\theta_i v_i \tag{8.22}$$

式中，v_i，v_j 是各个神经元的输出。

下面考察第 m 个神经元的输出变化前后，计算能量函数 E 值的变化。设 $v_m=0$ 时的计算能量函数值为 E_1。

$$E_1 = -\frac{1}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N} w_{ij}v_iv_j + \sum_{i=1}^{N}\theta_i v_i \tag{8.23}$$

将 $i=m$ 项分出来，并注意到 $v_m=0$。

$$E_1 = -\frac{1}{2}\sum_{\substack{i=1\\ i\neq m}}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N} w_{ij}v_iv_j + \sum_{\substack{i=1\\ i\neq m}}^{N}\theta_i v_i \tag{8.24}$$

类似地，当 $v_m=1$ 时的计算能量函数值为 E_2，则有

$$E_2 = -\frac{1}{2}\sum_{\substack{i=1\\ i\neq m}}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N} w_{ij}v_iv_j + \sum_{\substack{i=1\\ i\neq m}}^{N}\theta_i v_i - \sum_{\substack{j=1\\ j\neq m}}^{N} w_{mj}v_j + \theta_m \tag{8.25}$$

当神经元状态由 0 变为 1 时，计算能量函数 E 值的变化量 ΔE 为

$$\Delta E = E_2 - E_1 = -\left(\sum_{\substack{j=1\\ j\neq m}}^{N} w_{mj}v_j - \theta_m\right) \tag{8.26}$$

由于此时神经元的输出由 0 变为 1，因此满足神经元兴奋条件，即

$$\sum_{\substack{j=1\\ j\neq m}}^{N} w_{mj}v_j - \theta_m > 0 \tag{8.27}$$

由式(8.26)，得 $\Delta E<0$。

当神经元状态由 1 变为 0 时，计算能量函数 E 值的变化量 ΔE 为

$$\Delta E = E_1 - E_2 = \sum_{\substack{j=1\\ j\neq m}}^{N} w_{mj}v_j - \theta_m \tag{8.28}$$

由于此时神经元的输出由 1 变为 0，因此

$$\sum_{\substack{j=1\\ j\neq m}}^{N} w_{mj}v_j - \theta_m < 0 \tag{8.29}$$

也得 $\Delta E<0$。

综上所述，神经元状态变化时总有 $\Delta E<0$，这表明神经网络在运行过程中能量将不断降低，

最后趋于稳定的平衡状态。

关于离散型 Hopfield 神经网络的稳定性，早在 1983 年就由科恩（Cohen）与葛劳斯伯格（S. Grossberg）给出了稳定性的证明。Hopfield 等又进一步证明，只要连接权值构成的矩阵是非负对角元的对称矩阵，该网络就具有串行稳定性；若该矩阵为非负定矩阵，则该网络就具有并行稳定性。

8.4.2 连续型 Hopfield 神经网络及其 VLSI 实现

离散型 Hopfield 神经网络中的神经元与生物神经元的差别较大，因为生物神经元的输入与输出是连续的，而且生物神经元存在时滞。1984 年，Hopfield 又提出一种连续时间神经网络模型，在这种网络中，神经元的状态可以取 0 至 1 之间的任一实数值。

连续型 Hopfield 神经网络的电子线路实现如图 8.10 所示。

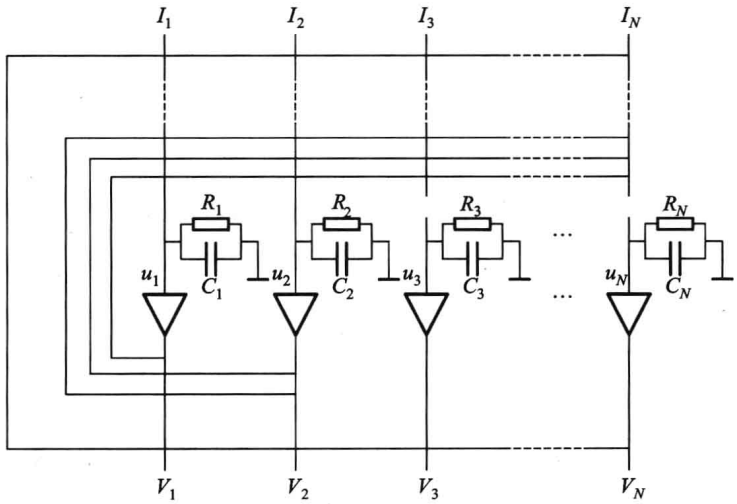


图 8.10 连续型 Hopfield 神经网络电子线路

其中，每一个神经元由电阻 R_i 和电容 C_i 以及具有饱和非线性特性的运算放大器模拟，输出 v_i 同时还反馈至其他神经元，但不反馈自身。u_i 表示神经元 i 的膜电位状态，v_i 表示它的输出，C_i 表示细胞膜输入电容，R_i 表示细胞膜的传递电阻，电阻 R_i 和电容 C_i 并联模拟了生物神经元输出的时间常数，而输出 v_i 对 $u_j(j=1,2,\cdots,N)$ 的影响则模拟了神经元之间互连的突触特性，运算放大器模拟神经元的非线性特性。

由基尔霍夫电流定律得，连续型 Hopfield 神经网络动力学系统方程为

$$\frac{1}{R_i}u_i + C_i\frac{\mathrm{d}u_i}{\mathrm{d}t} = I_i + \sum_{j=1}^{N} w_{ij}v_j \tag{8.30}$$

$$v_i = f(u_i) = \frac{1}{1+\mathrm{e}^{-\frac{2u_i}{u_0}}} \quad (i=1,2,\cdots,N)$$

式中，I_i 为施加在第 i 神经元上的偏置电流，表示系统外部的输入；$w_{ij}=\dfrac{1}{R_{ij}}$ 模拟神经元 j 与 i 之间

互连的突触特性；$f(u_i)$是放大器的非线性饱和特性，近似于 S 形函数。

连续型 Hopfield 神经网络模型，在简化了生物神经元性质的同时，重点突出了以下重要特性：

① 神经元作为一个输入输出变换，其传输特性具有 S 特性。

② 细胞膜具有时空整体作用。

③ 神经元之间存在大量的兴奋和抑制性连接，这种连接主要是通过反馈来实现的。

④ 具有既代表产生动作电位的神经元，又代表按渐近方式工作的神经元的能力。

因此，连续型 Hopfield 神经网络准确地保留了生物神经网络的动态和非线性特征，有助于理解大量神经元之间的协同作用是怎样产生巨大的计算能力的。

连续型 Hopfield 神经网络的计算能量函数 $E(t)$定义为

$$E(t) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}v_i(t)v_j(t) - \sum_{i=1}^{N} v_i(t)I_i + \sum_{i=1}^{N}\frac{1}{R_i}\int_0^{v_i(t)} f^{-1}(v)\,\mathrm{d}v \tag{8.31}$$

定理 8.1　对于连续型 Hopfield 神经网络，若$f^{-1}(\cdot)$为单调递增的连续函数，$C_i>0$，$w_{ij}=w_{ji}$，则$\frac{\mathrm{d}E(t)}{\mathrm{d}t}\leqslant 0$；当且仅当$\frac{\mathrm{d}v_i(t)}{\mathrm{d}t}=0$，$(1\leqslant i\leqslant N)$时，$\frac{\mathrm{d}E(t)}{\mathrm{d}t}=0$。

证明

$$\frac{\mathrm{d}E(t)}{\mathrm{d}t} = \sum_{i=1}^{N}\frac{\partial E(t)}{\partial v_i(t)}\frac{\mathrm{d}v_i(t)}{\mathrm{d}t} \tag{8.32}$$

而

$$\begin{aligned}
\frac{\partial E(t)}{\partial v_i(t)} &= -\frac{1}{2}\sum_{j=1}^{N} w_{ij}v_j(t) - \frac{1}{2}\sum_{j=1}^{N} w_{ji}v_j(t) - I_i + \frac{1}{R_i}f^{-1}(v_i)\\
&= -\frac{1}{2}\sum_{j=1}^{N}(w_{ji}-w_{ij})v_j(t) - \left[\sum_{j=1}^{N} w_{ij}v_j(t) + I_i - \frac{u_i(t)}{R_i}\right]\\
&= -\frac{1}{2}\sum_{j=1}^{N}(w_{ji}-w_{ij})v_j(t) - C_i\frac{\mathrm{d}u_i(t)}{\mathrm{d}t}\\
&= -\frac{1}{2}\sum_{j=1}^{N}(w_{ji}-w_{ij})v_j(t) - C_i\frac{\mathrm{d}f^{-1}(v_i)}{\mathrm{d}t}\\
&= -\frac{1}{2}\sum_{j=1}^{N}(w_{ji}-w_{ij})v_j(t) - C_i\frac{\mathrm{d}f^{-1}(v_i)}{\mathrm{d}v_i}\frac{\mathrm{d}v_i}{\mathrm{d}t}
\end{aligned} \tag{8.33}$$

将式(8.33)代入式(8.32)，得

$$\begin{aligned}
\frac{\mathrm{d}E(t)}{\mathrm{d}t} &= \sum_{i=1}^{N}\frac{\partial E(t)}{\partial v_i(t)}\frac{\mathrm{d}v_i(t)}{\mathrm{d}t}\\
&= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(w_{ji}-w_{ij})v_j(t)\frac{\mathrm{d}v_i(t)}{\mathrm{d}t} - \sum_{i=1}^{N} C_i\frac{\mathrm{d}f^{-1}(v_i)}{\mathrm{d}v_i}\left(\frac{\mathrm{d}v_i}{\mathrm{d}t}\right)^2
\end{aligned} \tag{8.34}$$

由于 $w_{ij}=w_{ji}$，$C_i>0$ 和 $f^{-1}(\cdot)$为单调递增，即有$\frac{\mathrm{d}f^{-1}(v_i)}{\mathrm{d}v_i}>0$，所以从式(8.34)得

$$\frac{\mathrm{d}E(t)}{\mathrm{d}t} = -\sum_{i=1}^{N} C_i\frac{\mathrm{d}f^{-1}(v_i)}{\mathrm{d}v_i}\left(\frac{\mathrm{d}v_i}{\mathrm{d}t}\right)^2 \leqslant 0 \tag{8.35}$$

从式(8.35)可见,若$\frac{dv_i}{dt}=0$,则$\frac{dE(t)}{dt}=0$;反之,若$\frac{dE(t)}{dt}=0$,则必须满足$\frac{dv_i}{dt}=0$

证毕。

上面的定理表明,Hopfield 神经网络的能量函数的值随着时间的推移总是在不断地减小,神经网络趋于某一平衡状态,平衡点就是 $E(t)$ 的极小值点,反之亦然。这说明 Hopfield 神经网络的演变过程就是在$[0,1]^N$空间内寻找极小值稳定点(吸引子)的过程,并在达到这些点后稳定下来。因此,这种神经网络同样具有自动求极小值的计算功能。

无论是离散型 Hopfield 神经网络,还是连续型 Hopfield 神经网络,虽然理论上都能够收敛到极值点,但一般都不能收敛到最优点。为了解决这个问题,下面介绍随机神经网络。

8.4.3 随机神经网络

在 Hopfield 神经网络中,神经元状态为 1 是根据其输入是否大于阈值确定的,是确定性的。而在随机神经网络中,神经元状态为 1 是随机的,服从一定的概率分布。例如,服从玻耳兹曼(Boltzmann)、高斯(Gaussian)、柯西(Cauchy)分布等,从而构成玻耳兹曼机、高斯机、柯西机等随机机。

下面主要以玻耳兹曼机为例,介绍随机神经网络。

1985 年,加拿大多伦多大学教授欣顿(Hinton)等人借助统计物理学的概念和方法,提出了 Boltzmann 机神经网络模型。Boltzmann 机是离散型 Hopfield 神经网络的一种变型,通过对离散型 Hopfield 神经网络加以扰动,使其以概率的形式表达,而网络的模型方程不变,只是输出值类似于 Boltzmann 分布,以概率分布取值。Boltzmann 机是按 Boltzmann 概率分布动作的神经网络。

离散型 Hopfield 神经网络的输出为

$$v_i(k+1)=\operatorname{sgn}\left(\sum_{\substack{j=1\\ j\neq i}}^{N} w_{ij}v_j(k)-\theta_i\right) \tag{8.36}$$

对于 Boltzman 机,设内部状态为

$$I_i=\sum_{\substack{j=1\\ j\neq i}}^{N} w_{ij}v_j(k)-\theta_i \tag{8.37}$$

神经元 i 输出值为 0 和 1 时的概率分别为 $p_i(0)$ 和 $p_i(1)$,即

$$\begin{aligned} p_i(1)&=\frac{1}{1+e^{-I_i/T}}\\ p_i(0)&=1-p_i(1) \end{aligned} \tag{8.38}$$

式中,T 为类似温度的扰动。

从式(8.38)可见,当 T 较大时,$p_i(0)$ 和 $p_i(1)$ 几乎相等;当 T 较小时,两概率值将由系统内部状态决定。因此,在系统运行时,先将 T 置于较大值以使系统能够跃过能量较大的状态来避免陷入局部极小,然后将 T 值逐渐减小,从而使系统最终收敛到全局能量最小的状态。这与模拟退火算法的机制几乎一样。应用模拟退火算法的 Hopfield 神经网络,实际上就是一种最基本的 Boltzmann 机。由式(8.38)可知,I_i 越大,$p_i(1)$ 越大,即神经元变为 1 的可能性越大。

设神经网络由 N 个神经元构成,每个神经元服从二态规律,即只取 0,1 两种状态,且假定神经元之间的连接权矩阵是对称的。在网络中,当神经元的输入加权和发生变化时,将引起神经元

状态的更新，这种更新在各个神经元之间是非同步的，可用概率分析方法描述。设 $\Delta E_i = E(v_i = 0) - E(v_i = 1)$，那么，神经元状态 v_i 为 1 的概率 P_i 服从玻耳兹曼分布。

与 HNN 相类似，Boltzmann 的能量函数定义为

$$E = -\frac{1}{2}\sum_i \sum_{j \neq i} w_{ij} v_i v_j + \sum_i v_i \theta_i \tag{8.39}$$

式中，v_i 和 θ_i 分别是神经元 i 的状态和阈值；w_{ij}为连接权。首先按下式计算神经元 i 状态转换时网络能量的变化

$$\Delta E_i = \sum_j w_{ij} v_j - \theta_i \tag{8.40}$$

然后，依据 ΔE_i 的大小，将神经元 i 以概率

$$p_i = \frac{1}{1 + \exp\left(-\frac{\Delta E_i}{T}\right)} \tag{8.41}$$

改变为状态“1”。

上式中，T 为温度。此式表明：在很高的温度下（$T \gg 1$），p_i 将接近于 0.5，神经元行为具有随机性；在同一温度下，ΔE_i 较大时，状态为“1”的概率增大；在温度很低时，随机性将趋于消失，此时玻耳兹曼机性能与确定性网络相同。

8.5　Hopfield 神经网络的应用

Hopfield 神经网络的许多应用都是利用 Hopfield 神经网络能够收敛到它的一个稳定状态这个特性。例如，利用 Hopfield 神经网络的稳定状态能够实现联想记忆、优化等应用。下面简单介绍基于 Hopfield 神经网络的联想记忆与优化方法。

8.5.1　Hopfield 神经网络在联想记忆中的应用

如果将网络的一个稳态作为一个记忆样本，那么以后当给这个网络一个适当的激励时，网络能够收敛到和输入模式最为相似的样本模式。初态朝稳态的收敛过程便是寻找记忆样本的过程。初态可以认为是给定样本的部分信息，网络改变的过程可以认为是从部分信息找到全部信息，从而实现了联想记忆的功能。

Hopfield 神经网络联想记忆过程，就是非线性动力学系统朝某个稳定状态运行的过程。这就需要调整连接权值使得所要记忆的样本作为 Hopfield 神经网络的能量局部极小点。Hopfield 神经网络联想记忆过程分为学习和联想两个阶段。在给定样本的条件下，调整连接权值，使得存储的样本成为 Hopfield 神经网络的稳定状态，这就是学习阶段。联想是指在已经调整好权值不变的情况下，给出部分不全或者受了干扰的信息，按照动力学规则改变神经元的状态，使神经网络最终变到某个稳定状态。

实现 Hopfield 神经网络联想记忆的关键是网络到达记忆样本能量函数极小点时，决定网络的神经元间连接权值 w_{ij}和阈值 θ_i 等参数。下面介绍按照 Hebb 学习规则设计 Hopfield 神经网络的连接权值。

设给定 m 个样本 $x^{(k)}(k = 1, 2, \cdots, m)$，$x_i^{(k)}$ 表示第 k 个样本中的第 i 个元素。记 w_{ij}是神经元

i 到神经元 j 的权值。

当神经元输出 $x_i \in \{-1, +1\}$ 时，

$$w_{ij} = \begin{cases} \sum_{k=1}^{m} x_i^{(k)} x_j^{(k)} & i \neq j \\ 0 & i = j \end{cases} \tag{8.42a}$$

或者

$$w_{ij}(k) = w_{ij}(k-1) + x_i^{(k)} x_j^{(k)} \quad (k = 1, 2, \cdots, m) \tag{8.42b}$$

$$w_{ij}(0) = 0 \quad w_{ii} = 0$$

当神经元输出 $x_i \in \{0, 1\}$ 时，

$$w_{ij} = \begin{cases} \sum_{k=1}^{m} (2x_i^{(k)} - 1)(2x_j^{(k)} - 1) & i \neq j \\ 0 & i = j \end{cases} \tag{8.43a}$$

或者

$$w_{ij}(k) = w_{ij}(k-1) + (2x_i^{(k)} - 1)(2x_j^{(k)} - 1) \quad (k = 1, 2, \cdots, m) \tag{8.43b}$$

$$w_{ij}(0) = 0 \quad w_{ii} = 0$$

显然，按照上面公式设计的网络连接权值满足对称条件。可以证明，按照上面公式设计网络连接权值时，Hopfield 神经网络的稳定状态是给定样本。

依据上述算法的联想记忆功能，可用于模式识别。但当样本多且彼此相近时，容易引起混淆。在网络结构与参数一定的条件下，要保证联想功能的正确实现，网络所能存储的最大的样本数，称为网络的记忆容量。网络的记忆容量不仅与神经元个数有关，还与连接权值的设计、要求的联想范围大小、样本的性质等有关。当网络要求存储的样本模式是两两正交的，可以有最大的记忆容量。

例 8.2 设计基于 Hopfield 神经网络的分类器。

当人看见苹果和橘子的时候，虽然和以前见过的不完全一样，但通过自联想能力仍然能够识别。利用 Hopfield 神经网络的联想特性，能够设计苹果和橘子的分类器，如图 8.11 所示。输送带将苹果和橘子传送给外形、质地、质量三个传感器检测，Hopfield 神经网络根据传感器检测结果识别，如果识别结果是苹果则执行器就将这个苹果放进苹果筐，否则放进橘子筐。

传感器检测结果为 1 或者 0，其意义如表 8.1 所示。

表 8.1 分类特征编码

	1	0
外形	圆	椭圆
质地	光滑	粗糙
重量	<300 g	>300 g

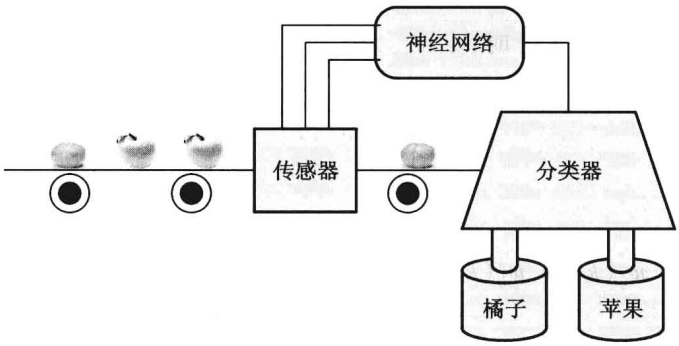


图 8.11　基于 Hopfield 神经网络的分类器

三个传感器输出表示为[外形　质地　质量],则标准橘子表示为:$\boldsymbol{x}^{(1)}=[1,0,1]^{\mathrm{T}}$。标准苹果表示为:$\boldsymbol{x}^{(2)}=[0,1,0]^{\mathrm{T}}$。

(1) 设计 DHNN 结构

设计有 3 个神经元的 Hopfield 神经网络如图 8.12 所示。3 个神经元的阈值都为 0。

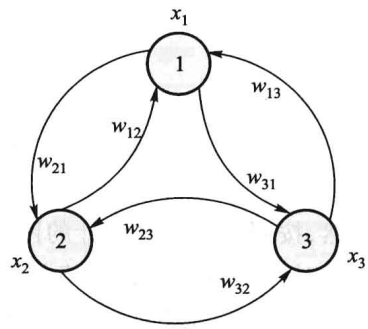


图 8.12　3 个神经元的 Hopfield 神经网络

(2) 设计连接权矩阵

$$w_{ij}=\begin{cases}\sum_{k=1}^{2}(2x_i^{(k)}-1)(2x_j^{(k)}-1) & i\neq j\\ 0 & i=j\end{cases}$$

$$w_{ji}=w_{ij}\quad i=1,2,\cdots,n\quad j=1,2,\cdots,n$$

$$w_{12}=(2\times1-1)\times(2\times0-1)+(2\times0-1)\times(2\times1-1)=-1-1=-2$$

$$w_{21}=w_{12}=-2$$

$$w_{23}=(2\times0-1)\times(2\times1-1)+(2\times1-1)\times(2\times0-1)=-1-1=-2$$

$$w_{32}=w_{23}=-2$$

$$\boldsymbol{W}=\begin{bmatrix}0 & -2 & 2\\ -2 & 0 & -2\\ 2 & -2 & 0\end{bmatrix}$$

(3) 测试

测试用例:$[1,1,1]^{\mathrm{T}}$,取初始状态:$v_1(0)=1,v_2(0)=1,v_3(0)=1$。神经元状态调整次序取为:2→1→3。当 $k=1$ 时,

$$v_2(1)=f\Big(\sum_{j=1}^{3}w_{2j}v_j(0)\Big)=f((-2)\times1+0\times1+(-2)\times1)=f(-4)=0$$

$$v_1(2)=f\Big(\sum_{j=1}^{3}w_{1j}v_j(1)\Big)=f(0\times1+(-2)\times0+2\times1)=f(2)=1$$

神经元状态调整为:$v_1(1)=1,v_2(1)=0,v_3(1)=1$。当 $k=2$ 时,

$$v_1(1)=v_1(0)=1,\quad v_3(1)=v_3(0)=1$$

$$v_2(2) = v_2(1) = 0, \quad v_3(2) = v_3(1) = 1$$

神经元状态调整为：$v_1(2)=1, v_2(2)=0, v_3(2)=1$。

类似地，有：$v_1(3)=1, v_2(3)=0, v_3(3)=1$。

可见输入$[1,1,1]^T$，输出$[1,0,1]^T$。

8.5.2 Hopfield 神经网络优化方法

如果将网络的稳态作为一个优化问题的目标函数的极小点，那么初态朝稳态的收敛过程便是优化计算的过程。该优化计算是在网络演化过程中自动完成的。

1985 年霍普菲尔德和塔克(D. W. Tank)应用神经网络方法求解旅行商问题这一著名的组合优化难题获得成功，引起了世界各国学者的广泛关注。他们的工作不仅对研究神经网络理论具有重要意义，也为组合优化问题求解开辟了新的途径。

1. 基本思想

无论是离散型 Hopfield 神经网络模型还是连续型 Hopfield 神经网络模型，能量函数都表征了网络的动力学演化过程，并揭示了该演化过程与网络稳定状态之间的内在联系。因此，Hopfield 能量函数极小化过程表示了神经网络从初始状态到稳定状态的演化过程。通常，约束优化问题的求解过程实际上就是目标函数的极小化过程，选择合适的能量函数使其最小值对应于问题的最优解。因此，目标函数达到局部极小或者全局最小，相应的解为约束优化问题的局部最优解或者全局最优解。如果将目标函数与能量函数相联系，并通过能量函数将约束优化问题的解映射到神经网络的一个稳定状态上去，那么，就可以利用神经网络的演化过程来实现优化计算。这就是应用神经网络解决优化问题的基本思想。

用神经网络方法求解优化问题的关键是如何把待求解的优化问题映射为一个神经网络。一般可以将求解的组合优化问题的每一个可行解用换位矩阵(permutation matrix)表示。

另一个关键问题是构造能量函数，使其最小值对应于问题的最优解。它决定了一个特定问题是否能够用神经网络方法求解。目前还没有直接将约束优化问题映射为神经网络的方法，通常采用优化理论中的拉格朗日(Lagrange)函数和乘子法，即利用优化问题的目标函数和约束条件构造相应的能量函数。一般使用在计算能量函数 E 中添加一些违反约束条件的惩罚项的简单方法，用罚函数法写出求解优化问题的能量函数，即

$$E = \sum_{i=1}^{m} C_i E_i + C_0 E_0 \tag{8.44}$$

式中，E_i 是违背约束条件的惩罚函数；E_0 是优化的目标函数；C_i 和 C_0 为平衡 E_i 和 E_0 在总能量函数中的作用的比例常数，且如果最小化 E_0，则 $C_0>0$；如果最大化 E_0，则 $C_0<0$；

显然，神经网络和能量函数的形式不是唯一的，如何设计出更好的映射方法及其能量函数构造方法是一个重要的课题。

2. 一般步骤

基于上述基本思想，用神经网络方法求解优化问题的一般步骤为：

① 将求解的优化问题的每一个可行解用换位矩阵表示。

② 将换位矩阵与由 n 个神经元构成的神经网络相对应：每一个可行解所对应的换位矩阵的各元素与相应的神经元稳态输出相对应。

③ 构造能量函数，使其最小值对应于优化问题的最优解，并满足约束条件。

④ 用罚函数法构造目标函数，与 Hopfield 神经网络的计算能量函数表达式相等，确定各连接权 w_{ij} 和偏置参数 I_i 等。

⑤ 给定网络初始状态和网络参数 A,B,C,D 等，使网络（可以是计算机模拟）按动态方程运行，直到达到稳定状态，并将它解释为优化问题的解。

Hopfield 神经网络模型与初始状态有关，例如在 TSP 问题中，原则上可选 $v_{xi}^0 = 1/n$ 为每个神经元的初始值，因为它满足 $\sum_x \sum_i v_{xi}^0 = n$。但由于相同长度的等价路径有 $2n$ 条，系统无法从中抉择，所以要加一定的噪声值（$\pm 0.1 v_{xi}^0$）来打破这种平衡。

当在计算机上实现上述演化时，需要离散化。此时，Δt 的选择很重要，Δt 太大可能导致离散后的算法与原连续算法有很大差异，甚至不收敛；Δt 太小则迭代次数太多，计算时间长。可以证明，只要 Δt 选取得合理，上述算法一定收敛。如果不加其他条件，收敛到的解不一定对应有效的访问路径，可能是不可行解。有些人在理论上对此进行了分析（Aiyer, S. V. B. 等，1990；Gee, A. H. 等，1993），提出了一些改进方法。这些方法的缺点是使权系数的表达式复杂化，降低了收敛速度。

鉴于神经网络动力学性质的复杂性，神经网络优化计算方面仍有许多问题需要进一步深入研究。目前存在的主要问题有：

① 计算结果的不稳定性。

② 系数 A、B、C、D 的确定。

③ 能量函数存在大量局部极小值，求解结果不能保证为最优解。

3. Hopfield 神经网络优化方法求解 TSP

旅行商问题（traveling salesman problem, TSP）或者邮递员路径问题：有 n 个城市，其相互间的距离，或者旅行成本为已知，求合理的路线使每个城市都访问一次，且总路径（或者总成本）为最短。

TSP 是由一个名叫 Karl Menger 的数学家、经济学家在 19 世纪 20 年代首先提出来的，但直到 20 世纪 40 年代一个名叫 Merill Flood 的人开始和他在 RAND 公司的同事们讨论后，TSP 才开始普及起来。那时人们对组合问题特别感兴趣。数学家们非常喜欢研究 TSP，这是因为 TSP 描述起来很简单，但要解决却是非常困难的。当城市数目不断增加时，求解问题所需要的计算量呈指数级增长。TSP 现在仍然被广泛用作新的组合优化方法的测试问题。

对于 TSP，一条访问路径可以用一个换位矩阵表示。以 5 个城市为例，如表 8.2 所示用换位矩阵表示访问 5 个城市的路径顺序为 $C_3 \to C_1 \to C_5 \to C_2 \to C_4 \to C_3$，其路径总长度为

$$l = d_{C_3C_1} + d_{C_1C_5} + d_{C_5C_2} + d_{C_2C_4} + d_{C_4C_3}$$

表 8.2　用换位矩阵表示访问次序

	1	2	3	4	5
C_1	0	1	0	0	0
C_2	0	0	0	1	0

续表

	1	2	3	4	5
C_3	1	0	0	0	0
C_4	0	0	0	0	1
C_5	0	0	1	0	0

如果下标 x,y 表示城市，i 表示第 i 次访问，则路径长度可以表示为下列一般形式：

$$\begin{aligned} l &= \frac{1}{2}\sum_{x}\sum_{y\neq x}\sum_{i} d_{xy}v_{xi}v_{y,i+1} + \frac{1}{2}\sum_{x}\sum_{y\neq x}\sum_{i} d_{xy}v_{xi}v_{y,i-1} \\ &= \frac{1}{2}\sum_{x}\sum_{y\neq x}\sum_{i} d_{xy}v_{xi}(v_{y,i+1} + v_{y,i-1}) \end{aligned} \tag{8.45}$$

式中，d_{xy}表示城市 x,y 之间的距离；v_{xi}表示换位矩阵中的第 x 行第 i 列的元素，其值为 1 时表示第 i 步访问城市 x，其值为 0 时表示第 i 步不访问城市 x。

在表 8.2 中，各行各列只能有一个元素为 1，其余都是 0，否则它表示一条无效的路径。每列中只有一个元素为 1，表示每次只访问一个城市，可以表示为

$$\sum_{x} v_{xi} = 1 \quad \forall i \tag{8.46}$$

每行中只有一个元素为 1，表示每个城市必须且只能访问一次，可以表示为

$$\sum_{i} v_{xi} = 1 \quad \forall x \tag{8.47}$$

综合上述讨论，TSP 可以表示为如下优化问题：

$$\begin{aligned} \min l &= \frac{1}{2}\sum_{x}\sum_{y\neq x}\sum_{i} d_{xy}v_{xi}(v_{y,i+1} + v_{y,i-1}) \\ st. \quad & \sum_{x} v_{xi} = 1 \quad \forall i \quad (\text{每个城市必须访问一次}) \\ & \sum_{i} v_{xi} = 1 \quad \forall x \quad (\text{每个城市只能访问一次}) \end{aligned} \tag{8.48}$$

用罚函数法，将上述约束优化问题表示为下列无约束优化问题：

$$\begin{aligned} J = &\frac{A}{2}\sum_{x}\sum_{i}\sum_{j\neq i} v_{xi}v_{xj} + \frac{B}{2}\sum_{i}\sum_{x}\sum_{y\neq x} v_{xi}v_{yi} + \frac{C}{2}\left(\sum_{x}\sum_{i} v_{xi} - n\right)^2 \\ &+ \frac{D}{2}\sum_{x}\sum_{y\neq x}\sum_{i} d_{xy}v_{xi}(v_{y,i+1} + v_{y,i-1}) \end{aligned} \tag{8.49}$$

令式(8.49)与 Hopfield 神经网络的计算能量函数相等，比较同一变量两端的系数，可得第 x 行第 i 列位置上的神经元与第 y 行第 j 列位置上的神经元之间的连接权值为

$$\begin{aligned} W_{xi,yj} &= -A\delta_{xy}(1-\delta_{ij}) - B\delta_{ij}(1-\delta_{xy}) - C - Dd_{xy}(\delta_{j,i+1} + \delta_{j,i-1}) \\ I_{xi} &= Cn \end{aligned} \tag{8.50}$$

式中

$$\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & \text{其他} \end{cases}$$

Hopfield 神经网络的动态方程为

$$\frac{\mathrm{d}u_{xi}}{\mathrm{d}t} = -\frac{u_{xi}}{\tau} - \frac{\partial E}{\partial v_{xi}}$$

$$= -\frac{u_{xi}}{\tau} - A\sum_{j\neq i} v_{xj} - B\sum_{y\neq x} v_{yi} - C\left(\sum_{x}\sum_{i} v_{xi} - n\right) - D\sum_{y\neq x} d_{xy}(v_{y,i+1} + v_{y,i-1}) \quad (8.51)$$

$$v_{xi} = f(u_{xi}) = \frac{1}{2}\left[1 + \tanh\left(\frac{u_{xi}}{u_0}\right)\right]$$

求解上式，直到收敛，可以得到神经网络的稳态解。在演化过程中，有些神经元的输出 v_{xi} 逐渐增大到 1，而有些神经元的输出 v_{xi} 逐渐减少到 0，最后收敛到稳定状态，所以，神经元输出 0 或者 1。

8.6 小结

1. 人工神经网络的概念

神经元的数学模型由加权求和、线性动态系统和非线性函数映射三部分组成。

前馈型神经网络中，各神经元接受前一层的输入，并输出给下一层，没有反馈。

在反馈型神经网络中，存在一些神经元的输出经过若干个神经元后，再反馈到这些神经元的输入端。最典型的反馈型神经网络是 Hopfield 神经网络。它是全互连神经网络。

2. BP 神经网络的学习

神经网络的学习是指调整神经网络的连接权值或者结构，使输入与输出具有需要的特性。

Hebb 学习规则：当某一突触两端的神经元同时处于兴奋状态，那么该连接的权值应该增强。

存在一个三层 BP 神经网络，可以逼近任意的连续函数。BP 学习算法可以归纳为式(8.16)至式(8.18)。

3. Hopfield 神经网络及其改进

Hopfield 神经网络可能有若干个稳定状态。从任一初始状态开始运动，总可以达到某个稳定状态。这些稳定状态可以通过改变各个神经元之间的连接权值得到。

连续型 Hopfield 神经网络的电子线路实现如图 8.10 所示。

对于连续型 Hopfield 神经网络，若 $f^{-1}(\cdot)$ 为单调递增的连续函数，$C_i > 0$，$w_{ij} = w_{ji}$，则 $\frac{\mathrm{d}E(t)}{\mathrm{d}t} \leqslant 0$。

在随机神经网络中，神经元状态为 1 是随机的，服从一定的概率分布。例如，服从玻耳兹曼(Boltzmann)分布，从而构成玻耳兹曼机。

4. Hopfield 神经网络的应用

如果将 Hopfield 网络的稳态作为一个联想记忆问题的样本，那么初态朝稳态的收敛过程便是联想记忆的过程。

如果将 Hopfield 网络的稳态作为一个优化问题的目标函数的极小点，那么初态朝稳态的收敛过程便是优化计算的过程。

思考题

8.1　为什么说人工神经网络是一个非线性系统？如果 BP 神经网络中所有结点都为线性函数，那么，BP 神经网络还是一个非线性系统吗？

8.2　简述人工神经网络的知识表示形式和推理机制，试举例说明。

8.3　BP 学习算法是什么类型的学习算法？它主要有哪些不足？

8.4　Hopfield 神经网络分为哪两类？两者的区别是什么？

8.5　Hopfield 神经网络优化方法的基本步骤和主要特点是什么？

8.6　Hopfield 神经网络与 BP 神经网络结构有什么区别？

习题

8.1　验证下列神经网络描述了与逻辑关系，如图 8.13 所示。其中 $y=\begin{cases}1, & x>\theta \\ 0, & x\leqslant\theta\end{cases}, \theta=0.5$。

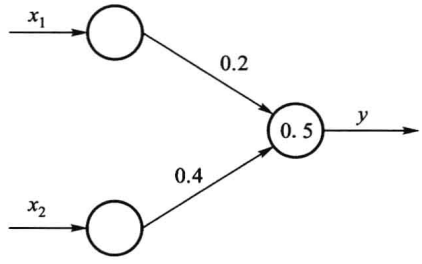


图 8.13　用神经网络表示与逻辑

8.2　异或(Exclusive - OR)(XOR)问题可以用 4 条产生式规则表示：

IF $x_1=0$ AND $x_2=0$ THEN $y=0$

IF $x_1=0$ AND $x_2=1$ THEN $y=1$

IF $x_1=1$ AND $x_2=0$ THEN $y=1$

IF $x_1=1$ AND $x_2=1$ THEN $y=0$

验证异或问题可以表示为图 8.14 中的神经网络：其中 $f(x)=\begin{cases}1, & x\geqslant\theta \\ 0, & x<\theta\end{cases}, \theta=0.5$。

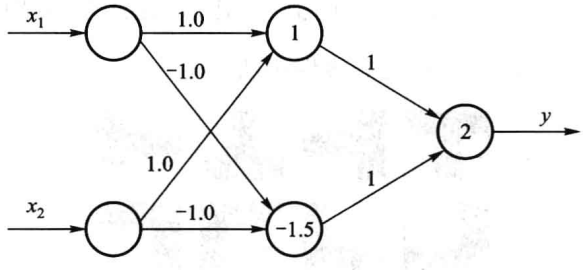


图 8.14　用神经网络表示异或问题

8.3　给定样本为 $X=[1,-1,1]$，$Y=[1,1]$，选 $\varepsilon=0.1$，BP 神经网络的初始权值矩阵选为

$$\boldsymbol{W}_1=\begin{bmatrix}1 & 1 & 1\\ 2 & 0 & 2\\ 3 & 3 & -3\end{bmatrix}\quad \boldsymbol{W}_2=\begin{bmatrix}1 & -1 & 1\\ 0 & 1 & 1\end{bmatrix}$$

如图 8.15 所示。

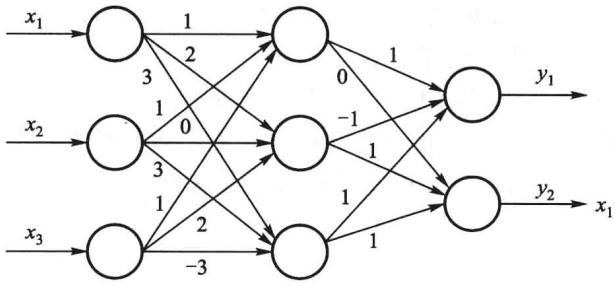


图 8.15　BP 神经网络

隐层和输出层的神经元的输入与输出非线性函数取为 $f(x)=\dfrac{2}{1+\mathrm{e}^{-x}}-1$。试用 BP 学习算法计算权值的调整过程。

8.4　已知下列函数：

① $f(x)=\dfrac{1}{x}$，$1\leqslant x\leqslant 100$。

② $f(x)=\lg x$，$1\leqslant x\leqslant 10$。

③ $f(x)=\mathrm{e}^{-x}$，$1\leqslant x\leqslant 10$。

④ $f(x)=\sin x$，$1\leqslant x\leqslant \dfrac{\pi}{2}$。

利用 BP 算法及 sigmoid 函数，对上面的函数完成如下工作。

① 获取两组数据，一组作为训练集，一组作为测试集。

② 利用训练集，训练一个单隐层的网络。

③ 利用测试集检验训练结果，改变隐层神经元个数，研究它对逼近效果的影响。

8.5　用三层 BP 神经网络解决字母 T 和 L 的识别问题。如图 8.16 所示，每个字母用 3×3 的二维二值图表示，令黑方格为 1，白方格为 0，每个字母有 4 个样本，包括字母正常位置及旋转 90°，180°和 270°的图像，希望输入不同的位置下的 T 时，网络输出为 1，而输入不同的位置下的 L 时，网络输出为 0。用 BP 算法求出权系数和阈值。（建议：选择网络结构为 9－3－1。隐单元非线性函数为 $f(x)=\dfrac{2}{1+\mathrm{e}^{-\alpha x}}-1$，输出单元非线性函数为 $f(x)=\dfrac{1}{1+\mathrm{e}^{-\alpha x}}$。）

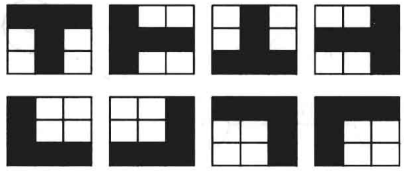


图 8.16　字母样本

8.6　澳大利亚野兔眼睛晶状体的质量与其年龄的关系如表 8.3 所示。不能用一个单值函数表示之间的关系，可以用如下非线性最小平方模型描述：

$$Y = 233.846(1 - \exp(-0.006042x)) + \varepsilon$$

其中，ε 是误差项。试利用反向传播算法，设计一个 BP 神经网络，能够为这个数据集提供一个非线性最小平方逼近，并与上述最小平方模型比较结果。

表 8.3　澳大利亚野兔眼睛晶状体质量与其年龄的关系

年龄/天	质量/mg	年龄/天	质量/mg	年龄/天	质量/mg	年龄/天	质量/mg
15	21.66	75	94.6	218	174.18	338	203.23
15	22.75	82	92.5	218	173.03	347	188.38
15	22.3	85	105	219	173.54	354	189.7
18	31.25	91	101.7	224	178.86	357	195.31
28	44.79	91	102.9	225	177.68	375	202.63
29	40.55	97	110	227	173.73	394	224.82
37	50.25	98	104.3	232	159.98	513	203.3
37	46.88	125	134.9	232	161.29	535	209.7
44	52.03	142	130.68	237	187.07	554	233.9
50	63.47	142	140.58	246	176.13	591	234.7
50	61.13	147	155.3	258	183.4	648	244.3
60	81	147	152.2	276	186.26	660	231
61	73.09	150	144.5	285	189.66	705	242.4
64	79.09	159	142.15	300	186.09	723	230.77
65	79.51	165	139.81	301	186.7	756	242.57
65	65.31	183	153.22	305	186.8	768	232.12
72	71.9	192	145.72	312	195.1	860	246.7
75	86.1	195	161.1	317	216.41		

8.7　已知有 3 个神经元的离散 Hopifeld 神经网络的权值矩阵为：

$$\boldsymbol{W} = \begin{bmatrix} 0 & 1 & -2 \\ 1 & 0 & 3 \\ -2 & 3 & 0 \end{bmatrix}, \quad \boldsymbol{\theta} = \begin{bmatrix} -1 \\ 2 \\ 1 \end{bmatrix}$$

计算该神经网络当状态为 $V(0) = \{1,0,1\}$ 时的能量。

8.8　已知有 4 个神经元的离散 Hopifeld 神经网络的权值矩阵为：

$$\boldsymbol{W}=\begin{bmatrix}0 & 2.8 & 2.5 & -2.6\\ 2.8 & 0 & 4.5 & -1.1\\ 2.5 & 4.5 & 0 & -5.5\\ -2.6 & -1.1 & -5.5 & 0\end{bmatrix},\quad \boldsymbol{\theta}=\begin{bmatrix}5.2\\ -3.2\\ -1.6\\ -7.6\end{bmatrix}$$

计算该神经网络当状态为 $V(0)=\{1,0,1,0\}$ 时的能量。

8.9　已知只有 3 个神经元的离散型 Hopfield 神经网络的权值矩阵为

$$\boldsymbol{W}=\begin{bmatrix}0 & -\frac{2}{3} & \frac{2}{3}\\ -\frac{2}{3} & 0 & -\frac{2}{3}\\ \frac{2}{3} & -\frac{2}{3} & 0\end{bmatrix}$$

3 个神经元的阈值取为 0。因为有 3 个神经元，所以有 $2^3=8$ 个状态。试计算验证：(1, -1, 1) 和 (-1, 1, -1) 是稳定状态，其他状态都会收敛到与之邻近的稳定状态上。

8.10　已知有 4 个神经元的离散型 Hopfield 神经网络的权值矩阵为

$$\boldsymbol{W}=\begin{bmatrix}0 & 3.4 & 2.8 & -3.1\\ 3.4 & 0 & 4.7 & -1.2\\ 2.8 & 4.7 & 0 & -5.9\\ -3.1 & -1.2 & -5.9 & 0\end{bmatrix}$$

4 个神经元的阈值取为(6.3, -4.3, -2.5, -9.6)。任意给定一个初始状态 $V(0)=\{1,0,1,0\}$，计算这个状态的能量，确定一个稳定状态。

第9章 智能体与多智能体系统

随着计算机网络和信息技术的发展，智能体技术得到广泛应用。多智能体不仅具备自身的问题求解能力和行为目标，而且能够相互协作，来达到共同的整体目标。因此，能够解决现实中广泛存在的复杂的大规模问题。

本章在介绍智能体与多智能体系统概念的基础上，简要介绍多智能体系统中的通信、协作、协调等基本技术。

9.1 智能体的概念与结构

9.1.1 智能体的概念

智能体(Agent)技术是当前人工智能研究的热点之一。Agent 在英语中是个多义词，国内学术界将 Agent 翻译为“主体”、“智能体”、“智能代理”等，尚无统一的译法。本章将 Agent 翻译为“智能体”，或者直接引用英文原文。

在人工智能领域中，Agent 可以看做是一个程序或者一个实体，它嵌入在环境中，通过传感器(sensor)感知环境，通过效应器(effector)自治地作用于环境并满足设计要求。Agent 与环境的交互作用如图 9.1 所示。

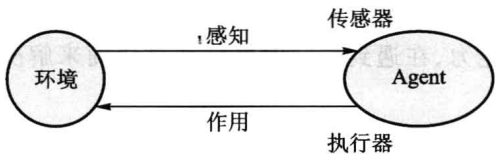


图 9.1 Agent 与环境的交互作用

实际上，一个人可以看成是一个 Agent，眼睛、耳朵等器官如同传感器，而手、脚和嘴如同执行器。一个机器人也可以看成是一个 Agent，通过摄像头、红外传感器等传感设备感知外界环境，各种各样的马达作为执行器作用于外界环境。

一个软件 Agent，使用经过编码的二进制符号序列作为感知与动作的表示。传统的整体设计和集中控制的软件开发方法越来越显示出局限性。智能和分布式的软件系统已经成为软件系统

设计的一个重要方向。综合集成分布于 Internet 的异构软件以支持社团组织完成多部门甚至多社团组织之间合作的具有空间、时间和功能分布性的复杂任务，正在成为建立新型计算环境的主要动力。

1977 年，Hewitt 提出的并发 Actor 模型。这个模型中包含具有自控行为、相互作用、并发执行的对象，并被称为 Actors。他们不仅封装了内部状态，还通过消息传递机制进行通信和并发工作。Actors 被认为是最早的 Agent。

进入 20 世纪 80 年代，主要研究 Agent 之间的交互通信、协调合作，强调 Agent 之间的紧密群体合作而非个体能力的自治和发挥。80 年代末，分布式计算环境的普及推动了多 Agent 技术的发展。

目前，Agent 的能力不断加强，能越来越多地模拟人的思维和行为。近年来，在软硬件领域，并行计算和分布式处理技术的研究都取得了很大进展，使得早期研究者探索的一些 Agent 问题已经在许多新领域广泛开展，如分布式人工智能、机器人学、人工生命、分布式对象计算、人机交互、智能和适应性界面、智能搜索和筛选、信息检索、知识获取、终端用户程序设计等。

9.1.2　智能体的特性

Agent 作为独立的智能实体应该具备以下特性：

1. 自主性

一个 Agent 应该具有独立的局部于自身的知识和知识处理方法，在自身的有限计算资源和行为控制机制下，能够在没有人类和其他 Agent 的直接干预和指导的情况下持续运行，以特定的方式响应环境的要求和变化，并能够根据其内部状态和感知到的环境信息自主决定和控制自身的状态和行为。

自主性是 Agent 区别于过程、对象等其他抽象概念的一个重要特征。

2. 反应性

Agent 能够感知、影响环境。不只是简单被动地对环境的变化做出反应，而是可以表现出受目标驱动的自发行为。Agent 的行为是为了实现自身内在的目标，在某些情况下，Agent 能够采取主动的行为，改变周围的环境，以实现自身的目标。

3. 社会性

如同现实世界中的生物群体一样，Agent 往往不是独立存在的，经常有很多 Agent 同时存在，形成多智能体系统，模拟社会性的群体。因此，Agent 不仅能够自主运行，同时应该具有和外部环境中其他 Agent 相互协作的能力，在遇到冲突时能够通过协商来解决问题。

4. 进化性

Agent 应该能够在交互过程中逐步适应环境，自主学习，自主进化。能够随着环境的变化不断扩充自身的知识和能力，提高整个系统的智能性和可靠性。

9.1.3　智能体的结构

Agent 结构接受传感器的输入，然后运行 Agent 程序，并把执行的结果传送到效应器进行动作。Agent 系统的结构直接影响到系统的性能。人工智能的任务就是设计 Agent 程序，实现 Agent 从感知到动作的映射函数。这种 Agent 程序需要在某种称为结构的计算机设备上运行。简单的 Agent 结构可能只是一台计算机，复杂的 Agent 结构可能包括用在特定任务上的特殊硬件设备，如图像采

集设备或者声音滤波设备等。Agent 结构可能还包括隔离纯硬件和 Agent 程序的软件平台。

Agent、体系结构和程序之间具有如下关系：

Agent = 体系结构 + 程序

在计算机系统中，Agent 含有独立的外部设备、输入/输出驱动装备、各种功能操作处理程序、数据结构和相应的输出。程序的核心部分是决策生成器或问题求解器，它接收全局状态、任务和时序等信息，指挥相应的功能操作程序模块工作，并把内部工作状态和所要执行的重要结果送至全局数据库。Agent 的全局数据库设有存放 Agent 状态、参数和重要结果的数据库，供总体协调使用。Agent 的运行是一个或多个进程，并接受总体调度。各个 Agent 在多个计算机上并行运行，其运行环境由体系结构支持。体系结构还提供共享资源、Agent 间的通信工具和 Agent 间的总体协调，以使各个 Agent 在同一目标下并行、协调工作。

Agent 的结构表示了 Agent 内部各模块集合的组成和相互作用的关系。模块集合及其相互作用规定了 Agent 如何根据所获得的数据和它的运作策略来决定和修改 Agent 的输出。

Agent 结构需要解决以下问题：

① Agent 由哪些模块组成。

② 这些模块之间如何交互信息。

③ Agent 感知到的信息如何影响它的行为和内部状态。

④ 如何将这些模块用软件或硬件的方式组合起来形成一个有机的整体。

单个 Agent 的结构按属性可以分为：反应式体系结构（reactive architecture）、慎思式体系结构（deliberative architecture）和复合式体系结构（complex architecture）。

9.1.4 反应式 Agent

反应式 Agent 是一种具备对当时处境的实时反应能力的 Agent。图 9.2 为反应式 Agent 的结构示意图。

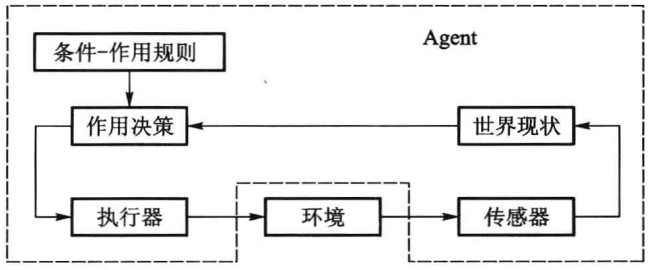


图 9.2 反应式 Agent 的结构示意图

图 9.2 中，Agent 的 If-then 规则使 Agent 将感知和动作连接起来。反应式 Agent 内部预置相关的知识，包含了行为集和约束关系。在外界刺激符合一定的条件后，直接调用预置的知识，产生相应的输出。每个 Agent 既是客户又是服务器，根据程序提出请求或做出回答。其行为以感知的外界信息为激发条件，中间不需要逻辑表示和推理。该 Agent 不制订将来行为计划，由设计者根据工作要求给出 Agent 执行的动作。

反应式 Agent 能及时快速地响应外来信息和环境的变化，但其智能程度较低，也缺乏足够的

灵活性，因此它适用于任务简单的实时环境。

9.1.5　慎思式 Agent

慎思式 Agent 是一种基于知识的系统，包括环境描述和丰富的智能行为的逻辑推理能力。慎思式 Agent 结构示意图如图 9.3 所示，Agent 接收的外部环境信息，依据内部状态进行信息融合，以产生修改当前状态的描述。然后，在知识库的支持下制订规划，再在目标指引下形成动作序列，对环境发生作用。

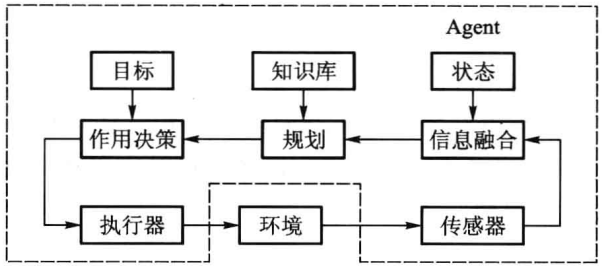


图 9.3　慎思式 Agent 的结构示意图

慎思式结构中的环境模型一般是预先知道的，因而对动态环境存在一定的局限性，不适用于未知环境。由于缺乏必要的知识资源，在 Agent 执行时需要向模型提供有关环境的新信息，而这往往难以实现。慎思式 Agent 具有较高的智能，但无法对环境的变化做出快速响应，而且执行效率相对较低。

9.1.6　复合式 Agent

复合式 Agent 综合了反应式 Agent 和慎思式 Agent 的优点，具有较强的灵活性和快速的响应性。复合式 Agent 是在一个 Agent 内组合多种相对独立和并行执行的智能形态，其结构包括感知、动作、反应、建模、规划、通信和决策等模块，如图 9.4 所示。

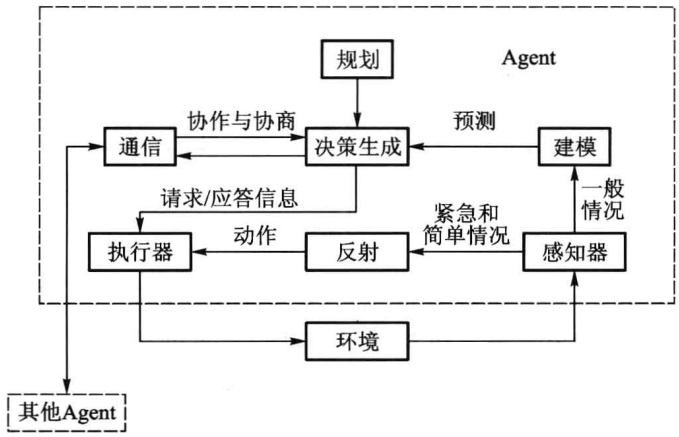


图 9.4　复合式 Agent 的结构示意图

Agent 通过感知模块来感知外部环境信息，再送到不同的处理模块。若感知紧急和简单情况，信息就被送入反映模块，作出决定，并把动作命令送到行动模块，产生相应的动作。

9.1.7 Agent 的应用

随着 Agent 技术的逐步成熟，Agent 已经被应用于很多领域，主要包括：

① 电信。该领域主要是利用 Agent 的自主性、协作性、可移动性和自适应性去解决复杂系统和网络管理方面的任务，包括负载均衡、故障预测、问题分析和信息综合等。

② 兴趣匹配。Agent 更多应用于商业网站向用户提供建议。例如，MIT 多媒体实验室的研究人员在这个领域做了很多的工作，相应的研究成果也曾被用于亚马逊书店和一些销售唱片和影碟的网站中。

③ 用户助理。用 Agent 协助用户更好地完成特定的任务。所使用的 Agent 都体现在用户界面层次，为用户完成某些特定的任务提供相应的信息和建议。

④ 组织结构。由多个 Agent 构造一个类似于人类组织的系统，不同的 Agent 代表着系统内的不同角色，通过这些 Agent 之间的通信和协作来完成具体的任务。目前主要应用于电子商务。例如，一个典型的多 Agent 供应链系统中就包含购买者 Agent、供应商 Agent 和中介 Agent 等多种 Agent。

⑤ 智能信息检索。Agent 可以通过利用相关知识检索一些特定信息。用于信息服务的智能 Agent 能够告诉用户所需要的资源在哪里，根据网上资源回答用户特定主题的问题，按照用户指定的条件过滤筛选信息，以及帮助用户整理下载的信息，还能够从大量的公共原始数据中筛选和提炼有价值的信息，向有关用户发布。

⑥ 决策支持系统。一般决策支持系统通常都需要大量综合性的信息，以及对信息经过深度加工的结果和知识。在这类系统中 Agent 能够监控系统的一些关键信息，在系统可能出现问题的时候，提醒相应的操作员，并在数据挖掘技术和决策支持模型的协助下，为复杂的决策提供有效的支持。

⑦ 移动计算。Agent 的自适应性将使网络服务更有效地适应于各种类型的数据通信模式和移动终端，而且 Agent 的离线计算能力还能为移动应用提供自然有效且稳定的离线计算模式，即使在移动用户断开与网络的连接之后，Agent 仍然能够继续完成尚未完成的任务，并在移动用户再次连上网络之后把结果反馈给用户。除此之外，Agent 还能够为移动用户提供友好个性化的界面，从而为用户提供个性化的服务。

⑧ 远程教育。在远程教育系统中引入 Agent，作为虚拟教师、虚拟学习伙伴、虚拟实验设备、虚拟图书管理员等，实现虚拟的教学、练习和实验环节等。在单机系统中，可以采用 Agent 技术设计人性化的角色，实施对学习者进行导航的模式，增加教学内容的趣味性，改善计算机辅助教学效果。

⑨ 数字娱乐。在网络数字娱乐系统中引入 Agent，可以增强娱乐效果。例如，在个性化的节目中插入点播服务，在游戏、动画中进行更加人性化的角色设计。

9.2　多智能体系统的概念与结构

对于现实中复杂的大规模问题，只靠单个的智能体往往无法描述和解决。因此，一个应用系统中往往包含多个智能体，这些智能体不仅具备自身的问题求解能力和行为目标，而且能够相互协作，达到共同的整体目标，这样的系统称为多智能体系统（multi-agent system，MAS）。

9.2.1　多智能体系统的特点

① MAS 中每个智能体具有独立性和自主性，能够解决给定的子问题，自主地推理和规划并选择恰当的策略，并以特定的方式影响周围的环境。

② MAS 支持分布式应用，具有良好的模块性，易于扩展，设计简单灵活，克服了建造大规模知识库时的知识管理和扩展的困难。

③ MAS 按面向对象的方法构造多层次、多元化的智能体，降低了系统的复杂性，也降低了各个智能体问题求解的复杂性。

④ MAS 是一个协调式的系统，各个智能体之间相互协调合作可以解决大规模的复杂问题；MAS 也是一个集成系统，它采用信息集成技术，将各子系统信息集成。

⑤ 在 MAS 中，智能体之间相互通信，彼此协调，并行地求解问题，提高了问题求解效率。

⑥ 同一个 MAS 中各个智能体可以是异构的。因此，多智能体技术为各种复杂系统提供了一种统一的模型。

⑦ 在 MAS 中，不同领域的专家系统、同一领域不同的专家系统可以协作求解单一专家系统难以解决的问题。多智能体技术打破了当前知识工程领域中仅使用一个专家系统的限制。

9.2.2　多智能体系统的基本类型

针对不同的应用环境，从不同角度提出了多种类型的多智能体模型，包括 BDI 模型、协商模型、协作规划模型和自协调模型等。

1. BDI 模型

BDI 模型是一个概念和逻辑上的理论模型，是研究 Agent 理性和推理机制的基础。在把 BDI 模型扩展到 MAS 的研究时，提出了联合意图、社会承诺、合理行为等 Agent 行为的形式化定义。联合意图为 Agent 建立复杂动态环境下的协作框架，对共同目标和共同承诺进行描述。当所有 Agent 都同意这个目标时，就一起承诺去实现该目标。联合承诺可以用来描述合作推理和协商，社会承诺给出社会承诺机制。

2. 协商模型

Agent 的协作行为一般通过协商而产生。虽然各个 Agent 的行动目标是使自身效用最大化，然而在完成全局目标时，就需要各 Agent 在全局上建立一致的目标。对资源缺乏的 Agent 动态环境，任务分解、任务分配、任务监督和任务评价就是一种必要的协商策略。合同网协议就是协商模型的典型代表，主要解决任务分配、资源冲突和知识冲突等问题。

3. 协作规划模型

多智能体系统的规划模型主要用于制订其协调一致的问题规划。每个 Agent 都具有自己求

解目标,考虑其他 Agent 的行动约束,并进行独立规划。网络结点上的部分规则可以用通信方式来协调所有结点,达到所有 Agent 都接受的全局规划。部分全局规划允许各 Agent 动态合作。Agent 的相互作用以通信规划和目标的形式抽象表达,以通信原语描述规划目标,相互告知自己的期望行为,利用规划信息调节自身的局部规划,达到共同目标。

4. 自协调模型

自协调模型是建立在开放和动态环境下的 MAS 模型,能够随环境变化自适应地调整行为。该模型的动态性表现在系统组织结构的分解重组和 MAS 内部的自主协调等方面。

9.2.3 多智能体系统的体系结构

MAS 的体系结构影响系统的异步性、一致性、自主性和自适应性的程度,决定信息的存储方式、共享方式和通信方式。体系结构中必须有共同的通信协议或传递机制。对于不同复杂程度的应用,应选择不同能力的体系结构。

下面介绍几种常见的 MAS 的体系结构。

1. 网络结构

网络结构中的 Agent 之间都是直接通信的,通信和状态知识都是固定的。网络结构的 Agent 系统中的每个 Agent 必须知道消息应该在什么时候发送到什么地方,系统中有哪些 Agent 是可以合作的,都具备什么样的能力等。但是,将通信和控制功能都嵌入到每个 Agent 内部,就是要求系统中的每个 Agent 都拥有有关其他 Agent 的大量信息和知识,而在开放的分布式系统中这往往是做不到的。另外,当系统中 Agent 的数目越来越多时,这种一对一的直接交互将导致低效率。

2. 联盟结构

联盟结构的工作方式是:若干相距较近的 Agent 通过一个叫做协助者的 Agent 来进行交互,而远程 Agent 之间的交互和消息发送是由局部 Agent 群体的协助者 Agent 协作完成的。这些协助者 Agent 可以实现各种各样的消息发送协议。当一个 Agent 需要某种服务时,它就向它所在的局部群体的协助者 Agent 发送一个请求,该协助者 Agent 将以广播方式发送该请求,或者将该请求与其他 Agent 所声明的能力进行匹配,将此信息发送给对它感兴趣的 Agent。这种结构中的 Agent 不需要知道其他 Agent 的详细信息,因此有较大的灵活性。协助 Agent 能够实现一些高层系统服务,如黄页、直接通信、问题分解和监控等。

3. 黑板结构

黑板结构和联盟系统有相似之处,不同的地方在于黑板结构中的局部 Agent 把信息存放在可存取的黑板上,实现局部数据共享。在一个局部 Agent 群体中,控制外壳 Agent(类似于联盟结构中的协助者)负责信息交互,而网络控制者 Agent 负责局部的 Agent 群体之间的远程信息交互。黑板结构的不足之处在于:局部数据共享要求一定范围的 Agent 群体中的 Agent 拥有统一的数据结构或知识表示,这就限制了系统中 Agent 设计和建造的灵活性。因此,开放的分布式系统不宜采用黑板结构。

9.3 多智能体系统的通信

同一环境中的 Agent 之间、不同环境的 Agent 之间都可以进行通信。一个 Agent 被授权后就

可以通过调用另一个 Agent 的方法向它发送消息。

Agent 进行通信就是改变信息载体，将载体发送到接收 Agent 的可观察环境中。采用这种通信观点可以拓展通信的形式，而不仅限于语言通信。

两个 Agent 之间的通信过程如下：

① 发送方将自己的思想翻译成通信所用语言的格式。

② 发送方将语言格式加载到通信传播媒体，如声音、文字和图像。

③ 传播载体到达接收方。

④ 接收方读取载体中的语言代码。

⑤ 接收方在思维空间中将语言代码按其格式翻译为思想，从而熟悉发送方的意识状态。

9.3.1　智能体通信的类型

通常 Agent 采用两种方式进行通信：一是分享一个共同内部表示语言的 Agent，它们无需任何外部语言就能通信；二是无需做出内部语言假设的 Agent，它们共享英语子集作为通信语言。

1. 使用 Tell 和 Ask 通信

这种通信形式的 Agent 分享共同内部表示语言，并通过界面 Tell 和 Ask 直接访问相互的知识库。

Agent A 可以使用 Tell(KB_B，“P”)通信把提议 P 传到 Agent B，就如 A 会使用 Tell(KB_B，“P”)把 P 加到自己的知识库。类似地，Agent A 可以使用 Ask(KB_B，“Q”)查出 B 是否知道 Q。我们称之为灵感通信(Telepathic Communication)。如图 9.5 所示，两个共享内部语言的 Agent 使用 Tell 和 Ask 界面并借助于知识库相互直接通信，其中每个 Agent 除了具有感知和行为端口之外，还有一个到知识库的输入/输出端口。

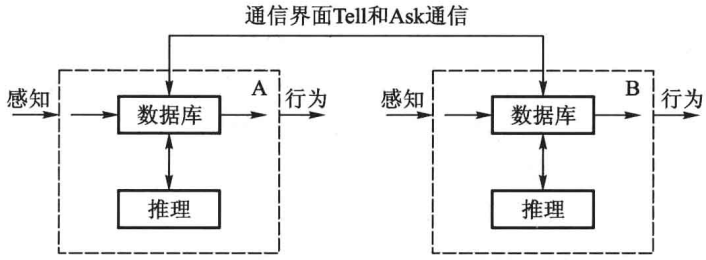


图 9.5　两个 Agent 通过 Tell 和 Ask 通信

2. 使用形式语言通信

大多数 Agent 的通信是通过语言而不是通过直接访问知识库而实现的。两个 Agent 使用语言通信如图 9.6 所示。有的 Agent 可以执行表示语言的行为，而其他 Agent 可以感知这些语言。外部通信语言可以与内部表示语言不同，并且这些 Agent 中的每一个都可以有不同的内部语言。只要每个 Agent 能够可靠地从外部语言映射到自己的内部语言，他们就无需同意任何内部符号。

对于外部通信语言生成和分析的问题，主要由自然语言处理方法来设计这两个过程的算法。但是语言通信最难的部分是前面提到的问题：协调不同的 Agent 的知识库之间的差别。

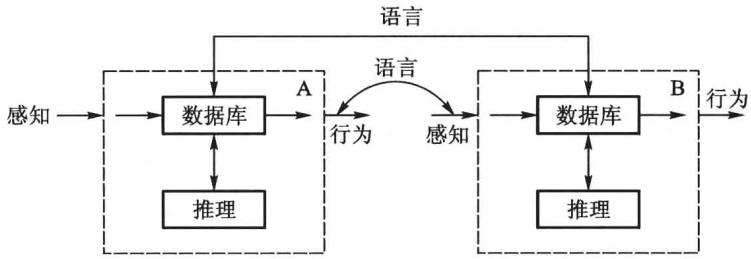


图 9.6 两个 Agent 使用语言通信

9.3.2 Agent 通信的方式

Agent 之间的通信是多智能体系统的基本功能。通信可分为黑板系统和消息/对话系统两种方式。

1. 黑板系统

黑板系统主要支持分布式问题求解。在多智能体系统中,黑板提供公共工作区,参与求解的 Agent 可以"看"到黑板上的问题、数据、求解记录,并将自己对问题的求解结果"写"到黑板上,以供其他 Agent 进行参考、求解。在这一模型中,Agent 之间不能进行直接的通信,信息的交换都是通过黑板这一共享媒介完成的,每个 Agent 独立完成各自求解的问题。

黑板系统可用于任务共享系统和结果共享系统。如果黑板中的 Agent 很多,那么黑板中的数据就会剧增。各个 Agent 在访问黑板时,需要从大量信息中搜索并提取感兴趣的信息。为进行优化处理,黑板应为各 Agent 提供不同的区域。

黑板系统的模型有三个主要组成部分:

① 知识源,即 Agent,是作为求解问题的独立单元,具有不同的专门知识,独立完成特定的任务。

② 黑板,即公共工作区,为知识源提供信息和数据,同时,供知识源进行修改。黑板的结构定义如下:

```
Typedef struct{
int stub_count;
int queue_id;
char semaphoreaPhore;
}BB_ITM bb[BBNUMBER];
```

黑板的消息格式如下:

报文标识	源 Agent 标识	目的 Agent 标识	消息标识	许可权	生命周期	报文或消息

③ 监控机制。根据黑板当前的问题求解状态,以及各知识源的不同求解能力,对其进行监控,使之能适时发现黑板变化,及时进行问题求解。

2. 消息/对话系统

消息/对话系统方式是实现灵活和复杂的协调策略的基础。各 Agent 使用规定的协议相互

交换信息，用于建立通信和协调机制。为了支持协作策略，通信协议必须明确规定通信过程、消息格式和选择通信语言。另外，Agent 之间的通信是知识级的通信，参与通信的 Agent 必须知道通信语言的含义。

有如下两种方式来实现 Agent 间的消息传递：

① 在直接通信方式中，每个 Agent 必须知道消息应该在什么时候发送到什么地方，系统中有哪些 Agent 是可以合作的，这些 Agent 各具备什么样的能力等。这要求系统中的每个 Agent 都拥有关于其他 Agent 的大量信息。

② 通过中介的通信在基于中介的消息传送中，若干相距较近的 Agent 通过通信服务器来进行交互和消息发送，而远程 Agent 之间的交互是由局部群体中的通信服务器协作完成的。

9.3.3　智能体通信语言

智能体之间的交互和协调是通过智能体之间的通信来完成的。当前智能体在完成了对其他智能体的建模之后，如果要对其他智能体进行控制和协调，比如改变其他智能体的目标，知识或者是对环境的置信，就应该通过通信动作来完成。

多智能体系统的一个最显著的特征是各个智能体在满足自身利益的同时，通过相互之间的协商和合作，共同完成单一智能体无法完成的任务。智能体要进行协商和合作，相互之间的通信是必不可少的环节。智能体是具有一定智能性和自主性的软件实体，因此，智能体之间通信所传递的不仅仅是字符流或二进制数据流，而且在知识层上进行表达、理解和交流。

虽然在计算机界已经出现了一些互通的标准和方法，如 CORBA、OLE、ActiveX 等，但这些标准都是从动态、分布环境下运行应用程序困难的角度出发，主要解决数据结构交换和跨平台的远程调用问题。智能体之间的交互并不仅仅局限于数据和方法，因而这些标准都不能彻底解决智能体之间的通信问题。

由美国 KSE（knowledge-sharing effort）机构研究开发的智能体通信语言（agent communication language，ACL）是目前智能体通信研究中最引人注目的。KSE 主要研究开发支持不同系统之间知识共享的基础技术。该机构包括了三个不同的工作组，分别负责中间语言（interlingua）、可共享重用知识（shared reusable knowledge）和外部接口（external interface）的研究工作。

目前国际上最流行的智能体通信语言 ACL 有两种：一种称为知识交换格式语言 KIF；另一种称为知识查询操纵处理语言 KQML。

1. 知识交换格式语言 KIF

知识交换格式语言（knowledge interchange format，KIF）主要是基于谓词逻辑，可以作为描述专家系统、数据库、多智能体的知识表示工具。KIF 负责将一种语言翻译成另一种语言，或者为两种异构智能体的知识表达提供语义共享。可共享重用知识则是一个词汇表，它可以使可共享知识库的内容更容易被理解，同时也为特定的领域提供开发工具和方法。外部接口主要设计软件智能体运行时能够共享知识和信息的通信高层语言（knowledge query and manipulation language，KQML），即知识查询操纵语言。KQML 是自主的异步智能体之间共享知识和实现协作问题求解的通信语言。它既是一种消息格式，也是支持实时智能体之间知识共享的消息处理协议，其目的在于实现基于知识的异构系统之间实现互操作和集成。上述的 KIF、词汇表和 KQML 三个部分正好构成了智能体通信语言（Agent Communication Language，ACL）的主体。

在实际应用中，软件实体之间的互操作和知识共享是应用程序作为软件智能体的重要基础，甚至是必不可少的先决条件。图 9.7 显示了互操作软件智能体的抽象模型。

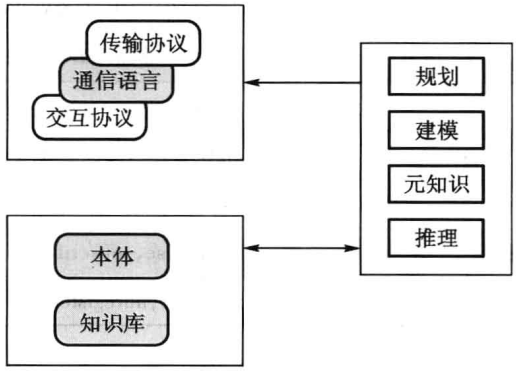


图 9.7 互操作软件智能体的抽象模型

2. 知识查询操纵语言

知识查询操纵语言(knowledge query and manipulation language，KQML)为多智能体通信定义了一套消息表达机制和消息传递格式，从而为多智能体通信提供了一套建立连接识别和交换消息的协议，构建了一种标准通用框架。

KQML 是一种层次结构语言，可以分为 3 层：内容层、通信层和消息层。内容层使用应用程序本身的表达语言来传送消息的实际内容。KQML 可以携带用任何表示语言编写的表达式，包括那些 ASCII 字符串和基于二进制符号表达的内容。通信层主要负责对消息的某些特性进行编码，这些特性描述了底层通信参数，如发送者和接收者的标识符。

消息层是整个 KQML 语言的核心。将一条消息从一个应用程序传送到另一个应用程序时，消息层完成对所传送信息的封装。消息层的一个最基本功能是识别传输消息发送时所使用的协议，并且给消息发送者提供一个附加在内容上的述行语或原语(performative)。除此之外，由于通信内容对参与交互的智能体来说是不透明的，因此消息层还必须包含描述内容的语言、假设的本体和一些内容的类型描述等特征。这些特征使得 KQML 语言在内容不可知的情况下实现对消息的分析、路由和正确的传送。

KQML 语言的语法是基于平衡的插入语列表。列表的初始元素是述行语，其他元素都是述行语的参数，使用关键字/值对进行表达。

尽管 KQML 语言中已经预先定义了一些常用的述行语，但它们既不是一个最小集，也不是包含全部。智能体在完成一次通信中可能只用到了其中一两个述行语。述行语是可以扩充的，如果一组智能体对揭示和协议达成一致，就可以使用附加的述行语。然而智能体要使用预先定义好的述行语，则必须采用标准的方法。表 9.1 中列出了一些 KQML 保留的述行语，其中除了一些标准的通信述行语，如 ask，tell 和面向协议的述行语，如 subscribe 之外，KQML 还包括了一些语言符号使用方法中非协议部分有关的述行语，如 advertise：用以说明智能体处理的消息的种类，recruit：为特定的消息类型找到合适的智能体。

表 9.1　KQML 保留的述行语

述行语类型	述行语名称
Basic query(基本查询)	evaluate, ask-if, ask-about, ask-one, ask-all
Multi-response(多个响应)	stream-about, stream-all, eos
Response(响应)	reply, sorry
Generic information(一般信息)	tell, achieve, cancel, untell, unachieve
Generator(发生器)	standby, ready, next, rest, discard, generator
Capability-definition(定义能力)	advertise, subscribe, monitor, import, export
Networking(网络)	register, unregister, forward, broadcast, route

KQML 语言中有多种在进程之间进行信息交换的协议。最简单的情况,一个智能体充当客户端并向另一个充当服务器的智能体发出一个请求,然后等待服务器方的答复,如图 9.8 中智能体 A 和 B。服务器的应答可以是单一的,也可以包含一组回答。一般服务器方的应答并不是完整的回答,而是提供给客户端一个句柄,客户端可以一次请求一个答复元素,如图 9.8 中的智能体 A 和 C。一个简单的例子就是客户端向关系型数据库请求一系列的实例。图 9.8 中智能体 A 和 D 之间的情况稍微有点不同,当客户端在服务器上注册了一个服务输出后,服务器就会不定期地向客户端发送异步应答,而客户端也不知道每一个应答的到达时间。这样做的好处在于客户端可以处理其他任务而不必专门等待服务器的应答信息。

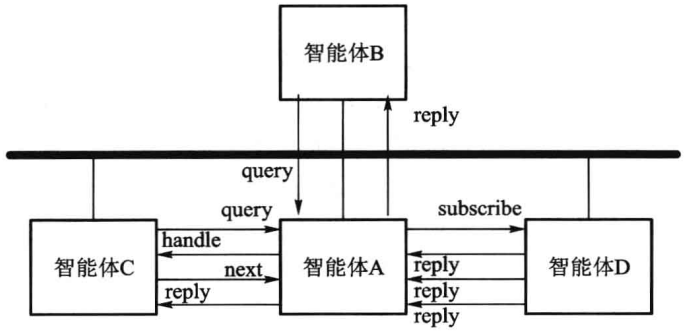


图 9.8　KQML 支持的几种通信协议

9.4　多智能体系统的协调

协调(coordination)和协作(cooperation)是 MAS 研究的核心问题。协调是指一组 Agent 完成一些集体活动时相互作用的性质。协作是非对抗的 Agent 之间保持行为协调的一个特例。Agent 群体协作行为的协调程度、系统的全局连贯性以及多 Agent 组织是否具有自适应性,是关系到整个系统性能和开放性的核心,同时也是一个系统智能水平的重要体现。

MAS 中的协调是指多个 Agent 为了一致和谐的方式工作而进行交互的过程。进行协调是希

望避免 Agent 之间的死锁或活锁。死锁指多个 Agent 无法进行各自的下一步动作,活锁指多个 Agent 不断工作却无任何进展。

当前主要有以下四种协调方法。

(1) 基于集中规划的协调

如果 MAS 中至少有一个 Agent 具备其他 Agent 的知识、能力和环境资源知识,那么该 Agent 可作为主控 Agent 对该 MAS 的目标进行分解,对任务进行规划,并指示或建议其他 Agent 执行相关任务。这种基于集中规划的协调方法特别适用于环境和任务相对固定、动态行为集可预计和需要集中监控的情况。

(2) 基于协商的协调

系统中没有作为规划的主控 Agent,而通过协商来实现任务的分配。协商是 Agent 间交换信息、讨论和达成共识的方式。

(3) 基于对策论的协调

该协调包括有通信的协调和无通信的协调两类。无通信协调是在没有通信的情况下,Agent 根据对方及自身效益模型,按照对策论选择适当行为。在这种协调方式中,Agent 至多也只能达到协调的平衡解。在有通信协调中则可得到协作解。

(4) 基于社会规划的协调

这一类是以每个 Agent 都必须遵循的社会规则、过滤策略、标准和惯例为基础的协调方法。这些规则对于 Agent 的行为加以限制,过滤某些有冲突的意图和行为,保证其他 Agent 必须的行为方式,从而确保本 Agent 行为的可能性,以实现整个 Agent 系统的社会行为的协调。

9.5 多智能体系统的协作

建立多智能体系统的目的在于解决由单一智能体难以处理的复杂问题,即任务在时间或空间上的复杂性超过了单个智能体的能力,仅仅依靠单个智能体行为来实现是不可能、不经济、不完整的,因此协作行为就成为多智能体系统必不可少的行为。在多智能体系统中,智能体之间的协调和合作是研究的核心问题之一,也是衡量系统智能水平的重要指标。

9.5.1 多智能体的协作类型

根据智能体之间目标的关系以及协作的程度,可将协作分为如下五种类型:

① 完全协作型。系统中的智能体都围绕一个共同的全局目标,各个智能体没有自己的局部目标,所有智能体全力以赴地协作。

② 协作型。系统中的智能体具有一个共同的全局目标,同时各个智能体还有与全局目标一致的局部目标。

③ 自私型。系统中不存在共同的全局目标,各智能体都为自己的局部目标工作,而且目标之间可能存在冲突。

④ 完全自私型。系统中不存在共同的目标,各智能体都为自己的局部目标工作,并且不考虑任何协作行为。

⑤ 协作与自私共存型。系统中既存在一些共同的全局目标,某些智能体也可能还具有与全

局目标无直接联系的局部目标。

多智能体系统的协作过程一般分为六个阶段：

① 产生协作需求，即确定协作目标。

② 协作规划，求解合理的协作结构。

③ 寻求协作伙伴。

④ 选择协作方案，即根据协作竞争者反推最佳的协作方案。

⑤ 按协作或交互协议进行协作以实现所确定的目标。

⑥ 结果评估，即判断协作的效果并为以后的协作提供可供参考的经验和教训。

下面介绍几种协作方法。

9.5.2　合同网协作方法

在所有的协作方法中，合同网（contract net）是最著名且应用最广泛的一种协作方法。它是由 Smith 于 1980 年提出的。基本思想来自人们在商务过程中用于管理商品和服务的合同机制。在合同网方法中，所有智能体分为两种角色：管理者（manager）和工作者（worker）。其中，管理者智能体的职责包括：

① 对每一待求解任务建立任务通知书（task announcement），将任务通知书发送给有关的工作者智能体，向工作者发出竞标信息。

② 接受并评估来自工作者的投标值（bid）。

③ 根据事先确定的任务竞标要求，从投标中选择最合适的工作者，并与之签订工作合同（contract）。

④ 在工作者履行合同的同时，对工作者进行监督和管理。当出现紧急情况的时候，采取适当的应急措施，保证任务的顺利完成，并对结果进行综合和分析。

工作者智能体的职责包括：

① 接收相关的任务通知书。

② 评价任务书，根据自身的情况设定任务投标的标值。

③ 向管理者发送投标信息，其中投标信息包含所投的任务、投标的工作者、投标的标值。

④ 若投标获得成功，则和管理者签订合同，并根据合同要求执行合同所分配的任务。

⑤ 当所分配的任务完成时，通知管理者，并且报告求解的结果。

在合同网协作方法中，不需要预先规定智能体的角色，智能体在协作过程中的角色可以变化：任何智能体通过发布任务通知书而成为管理者；相反，任何智能体通过应答任务通知书而成为工作者。这一灵活的特性使得任务能够被层次地分解分配：系统中的每一待求解的任务，由承担该项任务的智能体负责完成。当该智能体无法独立完成该任务时，它就将任务进行分解，为每一子任务发送任务通知书，从而该智能体由工作者变为管理者，履行管理者的职责；然后，从返回的投标中选定“最合适”的工作者智能体，和该工作智能体签订工作合同，将子任务分配给这一智能体。按合同执行子任务的工作者智能体若不能独立完成任务，就需要再次扮演管理者的角色，将子任务再分解，并按照合同网的方式进行分配。如此进行下去，直到所有任务都能独立完成。从以上步骤可以明显看出，合同网系统的任务分解分配过程形成一个动态确定的树结构。

合同网方法还具有性能平滑衰减的优点：如果一个工作者智能体不能提供一个满意的解或

者出现意外不能继续求解时,管理者智能体可以为任务寻找另外一个工作者智能体。这一点对实际生产显得尤为重要,特别是当加工资源(机器)发生故障时,管理者智能体会为分配给该工作者智能体的任务重新寻找合适的新的加工资源,保证任务能够顺利完成,从而提高系统的可靠性和鲁棒性。

9.5.3 黑板模型协作方法

黑板的概念最早是由 Newell 提出的。黑板模型的基本思想为:多个人类专家或智能体专家协同求解一个问题,黑板是一个共享的问题求解工作空间,多个专家都能"看到"黑板。当问题和初始数据记录到黑板上,求解开始。所有专家通过"看"黑板寻找利用其专家经验知识求解问题的机会。当一个专家发现黑板上的信息足以支持他进一步求解问题时,他就将求解结果记录在黑板上。新增加的信息有可能使其他专家继续求解。重复这一过程直到整个问题彻底求解,获得最终结果。

黑板模型由三个基本模块构成:

① 知识源。应用领域根据求解问题专门知识的不同划分成若干相互独立的专家,这些专家称为知识源(即智能体)。每一知识源独立完成一个特定领域的任务。

② 黑板。共享的问题求解工作空间。一般是以层次结构的方式组织,主要存放知识源所需要的信息和求解过程中的解状态数据,如初始数据、部分解、替换解、最终解等,有时也存放控制数据。在问题求解过程中,知识源不断地修改黑板。知识源之间的通信和交互智能通过黑板进行。

③ 监控机制。根据黑板上的问题求解状态和各知识源的求解技能,依据某种控制策略,动态地选择和激活合适的知识源,使知识源能实时地响应黑板的变化。

9.5.4 市场机制协作方法

上述两种智能体协作机制要求智能体之间直接通信,一般适合数量较小的智能体之间的协作求解,对于异构环境下的大量智能体之间的协作,市场机制则是较好的解决方案。

20 世纪 90 年代末期,电子商务和虚拟企业等复杂系统的发展,迫切需要研究异构智能体如何自主地实现协作的问题。这里的异构智能体可能是由不同的人,在不同的时间,利用不同的软件工具、技术,实现从属于不同组织的智能体。这些应用需求导致新的协作技术的出现。

市场机制的基本思想是针对分布式资源分配的待定问题,建立相应的计算经济,使智能体之间通过最少的直接通信来协调多个智能体之间的活动。在该方法中,对智能体关心的所有事务(如技能、资源等)都给予标价。系统中只存在两种类型的智能体:生产者和消费者。生产者能够提供服务,即将某一商品转换为另一商品;消费者能够进行商品交换。智能体以各种价格对商品进行投标,但所有的商品交换都以当前市场价格进行,每一智能体通过投标以便获得最大的利益和效用。

在开放的市场环境中,智能体应该可以自由地选择自己的贸易策略,其行为不一定合乎常规。市场机制假定智能体所给予的偏好是与智能体获得行动结果的知识一致,因此智能体推理行为就是最大化它自身的偏好。

9.6　多智能体系统的协商

协商(negotiation)是 MAS 实现协调、协作、冲突消解和矛盾处理的关键环节。有关 MAS 协商的关键技术可以概括为协商协议、协商策略和协商处理这三方面的内容。

1. 协商协议

协商协议用于处理协商过程中协商方之间的交互和作用,是交易双方交互的规则,决定何时何方采用何种行为,是规范交易协商行为的基础。

它主要研究的内容是 Agent 通信语言的定义、表示、处理和语义解释。协商协议的最简单形式如下:

一条协商通信消息:(<协商原语>, <消息内容>)

其中,协商原语即消息类型,它的定义通常基于言语行为理论;消息内容除包含消息的发送者、接收者、消息号、发送时间等固定信息外,还包括与协商应用的具体领域相关的信息描述。

协商协议的形式化表示通常有三种方法:巴科斯范式表示、有限自动机表示和语义表示。巴科斯范式表示具有简洁明了的特点,是最常用的表示方法。采用纯语义表示的协商工作不多,研究者更多的是给出非形式化的语义解释。常用的协商协议有:根据协商对象的数量分为一对一、一对多、多对多的协议;根据协商的顺序分为轮流出价、同时出价协商协议;根据协商议题的数量分为单属性和多属性协商等。

2. 协商策略

协商策略是 Agent 选择协商协议和通信消息的策略。一般来说,协商策略分为提议评估策略和提议生成策略两部分。提议评估策略用来对收到的提议进行评估,判断是否接受对方给出的提议;提议生成策略用来生成反提议。策略对于协商的效率起着至关重要的作用,根据不同的应用领域可以选择不同的协商策略。

协商策略基本上可以分为五类:单方让步策略、竞争型策略、协作型策略、破坏协商策略和拖延协商策略。单方让步策略只是在协商陷入僵局或协商不再有意义时才起作用,后两类策略显然不利于推进协商进程,所以,只有竞争型和协作型策略才是有意义的。

竞争型策略一般是指协商参与者坚持自己的立场,在协商过程中表现出竞争行为,使协商结果向有利于自身利益方向发展的协商对策。合同网协调模型、劳资协商、基于对策论的协商过程等都属于此类。协作型策略则是指协商各方都从整体利益出发,在协商过程中互相合作,他们采取的协商对策有利于寻找互相能接受的协商结果。

协作型策略的协商过程包括部分全局规划(PGP)、FA/C 等。Agent 应动态地、智能地选择适宜的协商策略,从而在系统运行的不同时刻表现出不同的竞争或协作行为。

策略选择的通用方法是:依据影响协商的多方面因素,给出适宜的策略选择函数。策略选择函数可能包括效用函数、比较或匹配函数、兴趣或爱好函数等几种。策略选择函数的设计除了要综合考虑影响协商的各种因素之外,还要考虑冲突综合消解以及与应用领域有关的属性等。

3. 协商处理

协商处理是对单个协商方及协商系统、协商行为的描述及分析,包括协商算法和系统分析两部分内容。协商算法用于描述 Agent 在协商过程中的行为,包括通信、决策、规划和知识库操作

等。系统分析的任务是分析和评价 Agent 协商的行为和性能，回答协商过程中的问题求解质量、算法效率，以及系统的公平性和死锁等问题。

协商协议主要处理协商过程中 Agent 之间的交互，协商策略主要设计 Agent 内的决策和控制过程，而协商处理则侧重于对单个 Agent 和多个 Agent 协商社会的整体协商行为的描述和分析。前两者描述了多个 Agent 协商的微观方面，而后者则刻画了多个 Agent 协商的宏观层。

9.7 小结

Agent 可以看做是一个程序或者一个实体，它嵌入在环境中，通过传感器感知环境，通过效应器自治地作用于环境并满足设计要求。

Agent 具备特性：自主性、反应性、社会性、进化性。

Agent、体系结构和程序之间的关系：Agent = 体系结构 + 程序。

单个 Agent 的结构可以分为：反应式体系结构、慎思式体系结构和复合式体系结构。

针对不同的应用环境，从不同角度提出了多种类型的 Multi-Agent 模型：BDI 模型、协商模型、协作规划模型和自协调模型等。

多智能体系统的体系结构：网络结构、联盟结构、黑板结构。

智能体通信的类型：使用 Tell 和 Ask 通信、使用形式语言通信。

Agent 通信的方式：黑板系统、消息/对话系统。

智能体通信语言：一是分享一个共同内部表示语言的 Agent，它们无需任何外部语言就能通信；二是无需做出内部语言假设的 Agent，它们共享英语子集作为通信语言。

MAS 中的协调是指多个 Agent 为了一致和谐的方式工作而进行交互的过程。主要有 4 种协调方法：基于集中规划协调、基于协商的协调、基于对策论的协调、基于社会规划的协调。

多智能体的协作是指智能体之间互相配合一起工作。主要 5 种类型：完全协作型、协作型、自私型、完全自私型、协作与自私共存型。

主要协作方法：合同网协作方法、黑板模型协作方法、市场机制协作方法。

合同网是应用最广泛的一种协作方法，类似于人们在商务过程中用于管理商品和服务的合同机制。合同网协作不需要预先规定智能体的角色，智能体在协作过程中的角色可以变化：任何智能体通过发布任务通知书而成为管理者；相反，任何智能体通过应答任务通知书而成为工作者。

黑板模型由知识源、黑板、监控机制三个基本模块构成。

对于异构环境下的大量智能体之间的协作，采用市场机制协作则是较好的解决方案。市场机制的基本思想是针对分布式资源分配的待定问题，建立相应的计算经济，以使智能体间通过最少的直接通信来协调多个智能体之间的活动。系统中只存在两种类型的智能体：生产者和消费者。生产者能够提供服务，即将某一商品转换为另一商品；消费者能够进行商品交换。智能体以各种价格对商品进行投标，但所有的商品交换都以当前市场价格进行，每一智能体通过投标以便获得最大的利益和效用。

协商是 MAS 实现协调、协作、冲突消解和矛盾处理的关键环节，可以概括为协商协议、协商策略和协商处理三个方面。

协商协议用于处理协商过程中协商方之间的交互和作用，是交易双方交互的规则。

一条协商通信消息：(<协商原语>，<消息内容>)

协商协议的形式化表示方法通常有三种：巴科斯范式表示、有限自动机表示和语义表示。

协商策略是 Agent 选择协商协议和通信消息的策略。可以分为 5 类：单方让步策略、竞争型策略、协作型策略、破坏协商策略和拖延协商策略。竞争型和协作型策略才是有意义的。竞争型策略一般是指协商参与者坚持自己的立场，在协商过程中表现出竞争行为，使协商结果向有利于自身利益方向发展的协商对策。协作型策略则是指协商各方都从整体利益出发，在协商过程中互相合作，他们采取的协商对策有利于寻找互相能接受的协商结果。

协商处理是对单个协商方及协商系统、协商行为的描述及分析，包括协商算法和系统分析两部分内容。

思考题

9.1　Agent 有哪些类型的结构？各有什么特点？

9.2　多 Agent 系统有哪些类型的结构？各有什么特点？

9.3　多 Agent 系统中为什么需要通信？目前有哪几种通信方式？

9.4　什么是多 Agent 系统的协调？列举几种协调方法。

9.5　什么是多 Agent 系统的协作？列举几种协作方法。

9.6　什么是多 Agent 系统的协商？列举几种协商方法。

9.7　选择一个你熟悉的领域，编写程序描述 Agent 与环境的作用，并对环境初始状态、Agent 的结构、类型、工作目标加以说明。

第10章 自然语言处理及其应用

如果计算机能够理解、处理自然语言，这将是计算机技术的一项重大突破。自然语言理解的研究在应用和理论两个方面都具有重大的意义。

本章首先介绍自然语言理解的概念以及发展历史，然后从应用角度介绍机器翻译和语音识别技术。

10.1 自然语言理解的概念与发展历史

由于自然语言具有多义性、上下文相关性、模糊性、非系统性、环境相关性等，因此，自然语言理解(natural language understanding)至今尚无一致的定义。

从微观角度，自然语言理解是指从自然语言到机器内部的一个映射。

从宏观角度，自然语言理解是指机器能够执行人类所期望的某种语言功能。这些功能主要包括如下几方面：

① 回答问题。计算机能正确地回答用自然语言输入的有关问题。

② 文摘生成。机器能产生输入文本的摘要。

③ 释义。机器能用不同的词语和句型来复述输入的自然语言信息。

④ 翻译。机器能把一种语言翻译成另外一种语言。

自然语言理解的研究历程，可以分为下列几个时期。

1. 萌芽时期

自然语言理解的研究可以追溯到20世纪40年代末和50年代初期。随着第一台计算机问世，英国的A. Donald Booth和美国的W. Weaver就开始了机器翻译方面的研究。美国、前苏联等国展开的俄、英互译研究工作开启了自然语言理解研究的早期阶段。在这一时期，M. Chomsky提出了形式语言和形式文法的概念，把自然语言和程序设计语言置于相同的层面，用统一的数学方法来解释和定义。M. Chomsky建立了转换生成文法，使语言学的研究进入了定量研究的阶段。Chomsky所建立的文法体系，仍然是目前自然语言理解中文法分析所必须依赖的文法体系，但还不能处理复杂的自然语言问题。

由于20世纪50年代单纯地使用规范的文法规则，再加上当时计算机处理能力的低下，使得机器翻译工作没有取得实质性进展。

2. 以关键词匹配技术为主的时期

从 20 世纪 60 年代开始，已经产生一些自然语言理解系统，用来处理受限的自然语言子集。这些人机对话系统可以作为专家系统、办公自动化及信息检索等系统的自然语言人机接口，具有很大的实用价值。但这些系统大都没有真正意义上的文法分析，而主要依靠关键词匹配技术来识别输入句子的意思。1968 年，美国 MIT 的 B. Raphael 完成的语义信息检索系统 SIR，能记住用户通过英语告诉它的事实，然后对这些事实进行演绎，回答用户提出的问题。美国 MIT 的 J. Weizenbaum 设计的 ELIZA 系统，能模拟一位心理医生（机器）同一位患者（用户）的谈话。在这些系统中，事先存放了大量包含某些关键词的模式，每个模式都与一个或多个解释（又叫响应式）相对应。系统将当前输入的句子同这些模式逐个匹配，一旦匹配成功便立即得到了这个句子的解释，而不再考虑句子中那些非关键词成分对句子意思的影响。匹配成功与否只取决于语句模式中包含的关键词及其排列次序，非关键词不能影响系统的理解。所以，基于关键词匹配的理解系统并非真正的自然语言理解系统，它既不懂文法，又不懂语义，只是一种近似匹配技术。这种方法的最大优点是允许输入的句子不一定要遵循规范的文法，甚至可以是文理不通的。这种方法的主要缺点是技术的不精确性，往往会导致错误的分析。

3. 以句法－语义分析技术为主的时期

20 世纪 70 年代后，自然语言理解的研究在句法－语义分析技术方面取得了重要进展，出现了若干有影响的自然语言理解系统。例如，1972 年美国 BBN 公司 W. Woods 负责设计的 LUNAR，是第一个允许用户用普通英语同计算机对话的人机接口系统，用于协助地质学家查找、比较和评价阿波罗 11 飞船带回来的月球标本的化学分析数据；同年，T. Winograd 设计的 SHEDLU 系统，是一个在“积木世界”中进行英语对话的自然语言理解系统，把句法、推理、上下文和背景知识灵活地结合于一体，模拟一个能够操纵桌子上一些积木玩具的机器人手臂，用户通过人机对话方式命令机器人放置那些积木块，系统通过屏幕给出回答并显示现场的相应情景。

4. 基于知识的自然语言理解发展时期

20 世纪 80 年代后，自然语言理解研究借鉴了许多人工智能和专家系统中的思想，引入了知识的表示和推理机制，使自然语言处理系统不再局限于单纯的语言句法和词法的研究，提高了系统处理的正确性，从而出现了一批商品化的自然语言人机接口和机器翻译系统。例如，美国人工智能公司（AIC）生产的英语人机接口 Intellect，美国弗雷公司生产的 Themis 人机接口。在自然语言理解研究的基础上，机器翻译走出了低谷，出现了一些具有较高水平的机器翻译系统。例如，美国的 META 系统，美国乔治伦敦大学的机译系统 SYSTRAN，欧共体在其基础上实现了英、法、德、西、意及葡等多语对译。

5. 基于大规模语料库的自然语言理解发展时期

由于自然语言理解中知识的数量巨大，特别是由于它们高度的不确定性和模糊性，要想把处理自然语言所需的知识都用现有的知识表示方法明确表达出来是不可能的。为了处理大规模的真实文本，研究人员提出了语料库语言学（corpus linguistics）。语料库语言学认为语言学知识的真正源泉是来自于生活中大规模的资料，我们的任务是使计算机能够自动或半自动地从大规模语料库中获取处理自然语言所需的各种知识。

20 世纪 80 年代，英国 Leicester 大学 Leech 领导的 UCREL 研究小组，利用已带有词类标记的语料库，经过统计分析得出了一个反映任意两个相邻标记出现频率的“概率转移矩阵”。他们设

计的 CLAWS 系统依据这种统计信息(而不是系统内存储的知识),对 LOB 语料库的一百万词的语料进行词类的自动标注,准确率达 96%。

目前市场上已经出现了一些可以进行一定自然语言处理的商品软件,但要让机器像人类那样自如地运用自然语言,仍是一项长远而艰巨的任务。

10.2 语言处理过程的层次

语言虽然表示成一连串文字符号或一串声音流,但其内部是一个层次化的结构,从语言的构成中就可以清楚地看出这种层次性。文字表达的句子的层次是"词素→词或词形→词组或句子",而声音表达的句子的层次是"音素→音节→音词→音句",其中每个层次都受到文法规则的制约。因此,语言的处理过程也应当是一个层次化的过程。

许多现代语言学家把语言处理过程分为三个层次:词法分析、句法分析、语义分析。① 如果接收到的是语音流,那么在上述三个层次之前还应当加入一个语音分析层。② 对于更高层次的语言处理,在进行语义分析后,还应该进行语用分析。虽然这样划分的层次之间并非是完全隔离的,但这种层次化的划分更好地体现了语言本身的构成,并在一定程度上使得自然语言处理系统的模块化成为可能。

1. 词法分析

词法分析是从句子中切分出单词,找出词汇的各个词素,从中获得单词的语言学信息并确定单词的词义。

不同的语言对词法分析有不同的要求。例如,英语和汉语就有较大的差距。在英语等语言中,因为单词之间是以空格自然分开的,切分一个单词很容易,所以找出句子的各个词汇就很方便。但是,由于英语单词有词性、数、时态、派生及变形等变化,要找出各个词素就复杂得多,需要对词尾或词头进行分析。例如,importable 可以是 im-port-able 或 import-able,这是因为 im,port,able 这三个都是词素。词法分析可以从词素中获得许多有用的语言学信息,这些信息对于句法分析是非常有用的。例如,英语中构成词尾的词素 s 通常表示名词复数或动词第三人称单数,ly 通常是副词的后缀,而 ed 通常是动词的过去分词等。另一方面,一个词可以有许多的派生、变形。例如 work 可变化出 works,worked,working,worker,workable 等。如果将这些派生的、变形的词全放入词典,将会产生非常庞大的数据量。实际上它们的词根只有一个。自然语言理解系统中的电子词典一般只放词根,并支持词素分析,这样可以大大压缩电子词典的规模。

在汉语中,每个字就是一个词素,所以要找出各个词素是相当容易的,但要切分出各个词就非常困难,不仅需要构词的知识,还需要解决可能遇到的切分歧义。如"优秀人才学人才学",可以是"优秀人才 - 学人才学",也可以是"优秀人 - 才学人才学"。

2. 句法分析

句法分析是对句子或短语结构进行分析,以确定构成句子的各个词、短语之间的关系以及各自在句子中的作用等,将这些关系用层次结构加以表达,并对句法结构进行规范化。

在计算机科学中,形式语言是某个字母表上一些有限字串的集合,而形式文法是描述这个集合的一种方法。形式文法与自然语言中的文法相似。最常见的文法分类是乔姆斯基(N. Chomsky)在 1950 年根据形式文法中所使用的规则集提出的。他定义了下列四种形式的文法:短语结构文法,又

称 0 型文法;上下文有关文法,又称 1 型文法;上下文无关文法,又称 2 型文法;正则文法,又称 3 型文法。型号愈高所受约束愈多,能表达的语言集就越小,也就是说型号越高描述能力就越弱。但由于上下文无关文法和正则文法能够高效率地实现,成为四类文法中最重要的两种文法类型。

3. 语义分析

句法分析后一般还不能理解所分析的句子,至少还需要进行语义分析。语义分析是把分析得到的句法成分与应用领域中的目标表示相关联。简单的做法就是依次使用独立的句法分析程序和语义解释程序。但这样做使得句法分析和语义分析相分离,在很多情况下无法决定句子的结构。为有效地实现语义分析,并能与句法分析紧密结合,已经提出了多种语义分析方法,如语义文法和格文法。

扩充转移网络 ATN 是将语义信息加进句法分析,并充分支持语义解释。

语义文法是将文法知识和语义知识组合起来,以统一的方式定义为文法规则集。语义文法不仅可以排除无意义的句子,而且具有较高的效率,对语义没有影响的句法问题可以忽略。但是实际应用该文法时需要很多的文法规则,因此一般适用于受到严格限制的领域。

格文法是为了找出动词和跟动词处在结构关系中的名词的语义关系,同时也涉及动词或动词短语与其他的各种名词短语之间的关系。也就是说,格文法的特点是允许以动词为中心构造分析结果。格文法是一种有效的语义分析方法,有助于删除句法分析的歧义性,并且易于使用。

4. 语音分析

构成单词发音的最小独立单元是音素。对于一种语言,如英语,必须将声音的不同单元识别出来并分组。在分组时,应该确保语言中的所有单词都能被区分,两个不同的单词最好由不同的音素组成。

语音分析是根据音位规则,从语言流中区分出各个独立的音素,再根据音位形态规则找出各个音节及其对应的词素或词。

词语以声波传送。语音分析系统传送声波这种模拟信号,并从中抽取诸如能量、频率等特征。然后,将这些特征映射为称做音素的单个语音单元。最后将音素序列转换成单词序列。

语音的产生是将单词映射为音素序列,然后传送给语音合成器,单词的声音通过说话者从语音合成器发出。

5. 语用分析

语用分析就是研究语言所存在的外界环境对语言使用所产生的影响,是自然语言理解中更高层次的内容。

10.3　机器翻译

10.3.1　机器翻译方法概述

人类对机器翻译(machine translation,MT)系统的研究开发已经持续了 50 多年。起初,机器翻译系统主要是基于双语字典进行直接翻译,几乎没有句法结构分析。直到 20 世纪 80 年代,一些机器翻译系统采用了间接方法。在这些方法中,源语言文本被分析转换成抽象表达形式,随后利用一些程序,通过识别词结构(词法分析)和句子结构(句法分析)解决歧义问题。其中有一种

方法将抽象表达设计为一种与具体语种无关的“中间语言”，可以作为许多自然语言的中介。这样，翻译就分成两个阶段：从源语言到中间语言，从中间语言到目标语言。另一种更常用的间接方法是将源语言表达转化成为目标语言的等价表达形式。这样，翻译便分成三个阶段：分析输入文本并将它表达为抽象的源语言；将源语言转换成抽象的目标语言；最后，生成目标语言。

机器翻译系统可以分成下列几种类型：

1. 直译式机器翻译系统

直译式机器翻译系统（direct translation MT systems）通过快速的分析和双语词典，将原文译出，并且重新排列译文的词汇，以符合译文的句法。直译式机器翻译系统如图 10.1 所示。

图 10.1 直译式翻译

大多数著名的大型机器翻译系统本质上都是直译式系统，如 Systran、Logos 和 Fujitsu Atlas。这些系统是高度模块化的系统，很容易被修改和扩展。例如，著名的 Systran 系统在开始设计时只能完成从俄文到英文的翻译，但现在已经可以完成很多语种之间的互译。Logos 开始只针对德语到英语的翻译，而现在可以将英语翻译成法语、德语、意大利语，以及将德语翻译成法语和意大利语。只有 Fujitsu Atlas 系统至今仍把自己局限于英日、日英的翻译。

2. 规则式机器翻译系统

规则式机器翻译系统（rule-based MT systems）是先分析原文内容，产生原文的句法结构，再转换成译文的句法结构，最后再生成译文。基于规则翻译系统通过识别、标注兼类多义词的词类，对多义词意义进行排歧；对某些同类词性的多义词再按其词法规则不同消除歧义。规则式机器翻译系统如图 10.2 所示。

当前主流的机器翻译还是基于规则的机器翻译系统。

3. 中介语式机器翻译系统

中介语式机器翻译系统（inter-lingual MT systems）先生成一种中介的表达方式，而非特定语言的结构，再由中介的表达式转换成译文。程序语言的编译常采取此策略。中介语式机器翻译系统如图 10.3 所示。

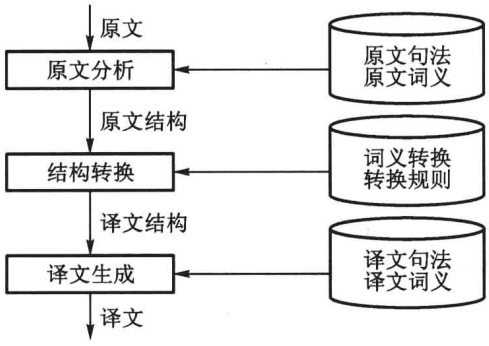


图 10.2 规则式翻译

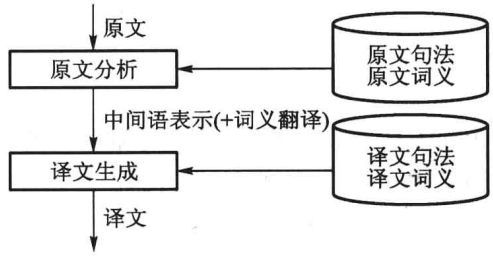


图 10.3 中介语式翻译

最重要的大型机中介语式机器翻译系统是 METAL。20 世纪 80 年代初期，德国西门子公司提供了大部分资金支持开发 METAL，直到 20 世纪 80 年代末才面市。目前最有名的两个中介语式机器翻译系统是 Grenoble 的 Ariane 和欧共体资助的 Eurotra。Ariane 有望成为法国国家机器翻译系统，而 Eurotra 是非常复杂的机器翻译系统之一。20 世纪 80 年代末，日本政府出资支持开发用于亚洲语言之间互译的中间语言系统，中国、泰国、马来西亚和印度尼西亚等国的研究人员均参加了这一研究。

4. 知识库式机器翻译系统

知识库式机器翻译系统（knowledge-based MT systems）是建立一个翻译需要的知识库，构成翻译专家系统。由于知识库的建立十分困难，因此目前此类研究多半有限定范围，并且使用知识获取工具，自动或半自动地大量收集相关知识充实知识库。

5. 统计式机器翻译系统

1994 年，IBM 公司的 A. Berger，P. Brown 等用统计方法和各种不同的对齐技术，给出了统计式机器翻译系统（statistics-based MT systems）Candide。

源语言中任何一个句子都可能与目标语言中的某些句子相似，这些句子的相似程度可能都不相同，统计式机器翻译系统能找到最相似的句子。

6. 范例式机器翻译系统

范例式机器翻译系统（example-based MT systems）是将过去的翻译结果当成范例，产生一个范例库。在翻译一段文字时，参考范例库中近似的例子，并处理差异处。

实际的机器翻译系统往往是混合式机器翻译系统（hybrid MT systems），即同时采用多种翻译策略，以达到正确翻译的目标。

10.3.2　翻译记忆

由于目前还没有一种机器翻译产品的效果能让人满意，因此，目前广泛采用翻译记忆（translation memory，TM）技术辅助专业翻译。以欧盟为例，每天都有大量的文件需要翻译成各成员国的文字，翻译工作量极大，自 1997 年采用德国塔多思（TRADOS）公司的翻译记忆软件以来，欧盟的翻译工作效率大大提高。如今，欧盟、国际货币基金组织等国际组织，微软、SAP、Oracle 和德国大众等跨国企业以及许多世界级翻译公司都以翻译记忆软件作为信息处理的基本工具。

翻译记忆是一种通过计算机软件来实现的专业翻译解决方案。与期望完全替代人工翻译的机器翻译技术不同，翻译记忆实际只是起辅助翻译的作用，也就是计算机辅助翻译（computer aided translation，CAT）。因此，翻译记忆与机器翻译有着本质的区别。

翻译记忆的基本原理是：用户利用已有的原文和译文，建立起一个或多个翻译记忆库，在翻译过程中，系统将自动搜索翻译记忆库中相同或相似的翻译资源（如句子、段落等），给出参考译文，使用户避免重复劳动，只需专注于新内容的翻译。翻译记忆库同时在后台不断学习和自动存储新的译文，变得越来越“聪明”。

由于翻译记忆实现的是原文和译文的比较和匹配，因此能够支持多语种之间的双向互译。以德国塔多思公司为例，该公司的产品基于 UNICODE（统一字符编码），支持 55 种语言，覆盖了几乎所有语言版本的 Windows95/98/NT。

10.4 语音识别

10.4.1 语音识别的概念

用语音实现人与计算机之间的交互，主要包括语音识别（speech recognition）、自然语言理解和语音合成（speech synthesis）。语音识别是完成语音到文字的转换。自然语言理解是完成文字到语义的转换。语音合成是用语音方式输出用户想要的信息。

现在已经有许多场合允许使用者用语音对计算机发命令，但是，目前还只能使用有限词汇的简单句子，因为计算机还无法接受复杂句子的语音命令。因此，需要研究基于自然语言理解的语音识别技术。

相对于机器翻译，语音识别是更加困难的问题。机器翻译系统的输入通常是印刷文本，计算机能清楚地区分单词和单词串。而语音识别系统的输入是语音，其复杂度要大得多，特别是口语有很多的不确定性。人与人交流时，往往是根据上下文提供的信息猜测对方所说的是哪一个单词，还可以根据对方使用的音调、面部表情和手势等来得到很多信息。特别是说话者会经常更正所说过的话，而且会使用不同的词来重复某些信息。显然，要使计算机像人一样识别语音是很困难的。

按照服务对象划分，针对某个用户的语音识别系统，称为特定人工作方式。针对任何人的语音识别系统，则称为非特定人工作方式。

通俗地说，特定人的语音识别是要识别说话人是谁，而非特定人语音识别是要识别说的什么话。

本节将简单介绍语音识别的基本过程，包括语音信号的采集与处理、特征参数的提取与识别等主要过程，然后着重介绍基于隐马尔可夫模型的语音识别方法。

10.4.2 语音识别的主要过程

语音识别过程包括从一段连续声波中采样，将每个采样值量化，得到声波的压缩数字化表示。采样值位于重叠的帧中，对于每一帧，抽取出一个描述频谱内容的特征向量。然后，根据语音信号的特征识别语音所代表的单词。

1. 语音信号采集

语音信号采集是语音信号处理的前提。语音通常通过话筒输入计算机。话筒将声波转换为电压信号，然后通过 A/D 装置（如声卡）进行采样，从而将连续的电压信号转换为计算机能够处理的数字信号。

目前多媒体计算机已经非常普及，声卡、音箱、话筒等已是个人计算机的基本设备。其中声卡是计算机对语音信号进行加工的重要部件，它具有对信号滤波、放大、A/D 和 D/A 转换等功能。而且，现代操作系统都附带录音软件，通过它可以驱动声卡采集语音信号并保存为语音文件。

对于现场环境不好，或者空间受到限制，特别是对于许多专用设备，目前广泛采用基于单片机、DSP 芯片的语音信号采集与处理系统。

2. 语音信号预处理

语音信号在采集后首先要进行滤波、A/D 变换，预加重（Preemphasis）和端点检测等预处理，然后才能进入识别、合成、增强等实际应用。

滤波的目的有两个：一是抑制输入信号中频率超出 $f_s/2$ 的所有分量（f_s 为采样频率），以防止混叠干扰；二是抑制 50 Hz 的电源工频干扰。因此，滤波器应该是一个带通滤波器。

A/D 变换是将语音模拟信号转换为数字信号。A/D 变换中要对信号进行量化，量化后的信号值与原信号值之间的差值为量化误差，又称为量化噪声。

预加重处理的目的是提升高频部分，使信号的频谱变得平坦，保持在低频到高频的整个频带中，能用同样的信噪比求频谱，便于频谱分析。

端点检测是从包含语音的一段信号中确定出语音的起点和终点。有效的端点检测不仅能减少处理时间，而且能排除无声段的噪声干扰。目前主要有两类方法：时域特征方法和频域特征方法。时域特征方法是利用语音音量和过零率进行端点检测，计算量小，但对气音会造成误判，不同的音量计算也会造成检测结果不同。频域特征方法是用声音的频谱的变异和熵的检测进行语音检测，计算量较大。

3. 语音信号的特征参数提取

人说话的频率在 10 kHz 以下。根据香农采样定理，为了使语音信号的采样数据中包含所需单词的信息，计算机的采样频率应是需要记录的语音信号中包含的最高语音频率的两倍以上。一般将信号分割成若干块，信号的每个块称为帧。为了保证可能落在帧边缘的重要信息不会丢失，应该使帧有重叠。例如，当使用 20 kHz 的采样频率时，标准的一帧为 10 ms，包含 200 个采样值。

话筒等语音输入设备可以采集到声波波形，如图 10.4 所示。虽然这些声音的波形包含了所需单词的信息，但用肉眼观察这些波形却得不到多少信息。因此，需要从采样数据中抽取那些能够帮助辨别单词的特征信息。在语音识别中，常用线性预测编码（linear predictive coding，LPC）技术抽取语音特征。

线性预测编码的基本思想是：语音信号采样点之间存在相关性，可用过去的若干采样点的线性组合预测当前和将来的采样点值。线性预测系数可以通过使预测信号和实际信号之间的均方误差最小来唯一确定。

语音线性预测系数作为语音信号的一种特征参数，已经广泛应用于语音处理各个领域。

声波的采样数据可以绘制成一个 $x-y$ 平面图，x 轴表示时间，y 轴表示振幅，如图 10.4 所示。声波有两个主要特征：振幅和频率。图 10.4 所示的声波波形实际上由 3 个正弦波叠加组成，但用肉眼很难分辨。为了能够看清楚声波中包含的主要频率波形，通常将采样信号经过傅里叶变换得到相应的频谱，再从频谱中看出波形中不同音素相匹配的主控频率组成成分。图 10.5 是图 10.4 中声波波形的频谱，清楚地显示了这段声音波形包含的正弦波的振幅和频率。可以看出，声波中主要有 3 个正弦波，它们的幅值分别是三个峰值的幅值，它们的频率分别是三个峰值对应的频率。

4. 向量量化

向量量化（vector quantization，VQ）技术是 20 世纪 70 年代后期发展起来的一种数据压缩和编码技术。经过向量量化的特征向量也可以作为后面隐马尔可夫模型中的输入观察符号。

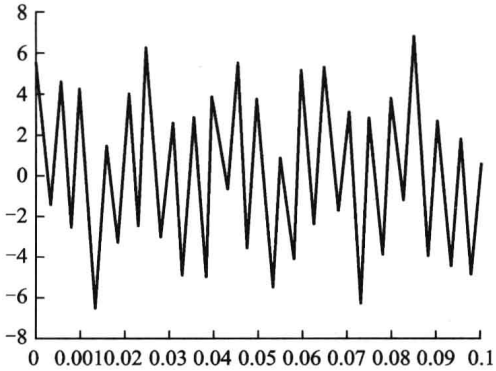


图 10.4 声波波形

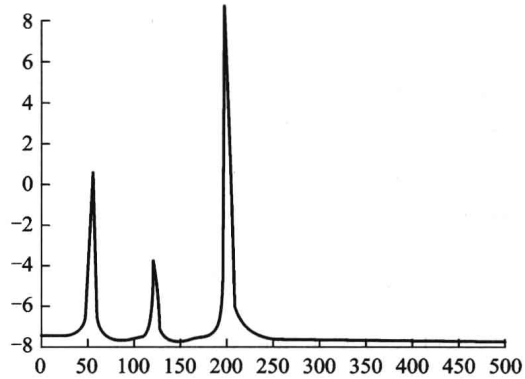


图 10.5 声波频谱

在标量量化中整个动态范围被分成若干个小区间，每个小区间有一个代表值，对于一个输入的标量信号，量化时落入小区间的值就用这个代表值代替。因为这时的信号量是一维的标量，所以称为标量量化。

向量量化的概念是用线性空间的观点，把标量改为一维的向量，对向量进行量化。和标量量化一样，向量量化是把向量空间分成若干个小区域，每个小区域寻找一个代表向量，量化时落入小区域的向量就用这个代表向量代替。

向量量化的基本原理是将若干个标量数据组成一个向量（或者是从一帧语音数据中提取的特征向量）在多维空间给予整体量化，从而可以在信息量损失较小的情况下压缩数据量。

5. 识别

当提取声音特征集合以后，就可以识别这些特征所代表的单词。本节重点关注单个单词的识别。识别系统的输入是从语音信号中提取出的特征参数，如 LPC 预测编码参数，当然，单词对应于字母序列。语音识别所采用的方法一般有模板匹配法、随机模型法和概率语法分析法三种。这三种方法都是建立在最大似然决策贝叶斯（Bayes）判决的基础上的。

（1）模板（template）匹配法

在训练阶段，用户将词汇表中的每一个词依次说一遍，并且将其特征向量作为模板存入模板库。在识别阶段，将输入语音的特征向量序列，依次与模板库中的每个模板进行相似度比较，将相似度最高者作为识别结果输出。

（2）随机模型法

随机模型法是目前语音识别研究的主流。其突出的代表是隐马尔可夫模型。语音信号在足够短的时间段上的信号特征近似于稳定，而总的过程可看成是依次相对稳定的某一特性过渡到另一特性。隐马尔可夫模型则用概率统计的方法来描述这样一种时变的过程。

（3）概率语法分析法

这种方法是用于大长度范围的连续语音识别。语音学家通过研究不同的语音语谱图及其变化发现，虽然不同的人说同一些语音时，相应的语谱及其变化有种种差异，但是总有一些共同的特点足以使他们区别于其他语音，也即语音学家提出的“区别性特征”。另一方面，人类的语言要受词法、语法、语义等约束，人在识别语音的过程中充分应用了这些约束以及对话环境的有关

信息。于是,将语音识别专家提出的“区别性特征”与来自构词、句法、语义等语用约束相互结合,就可以构成一个“自底向上”或“自顶向下”的交互作用的知识系统,不同层次的知识可以用若干规则来描述。

除了上面的三种语音识别方法外,还有许多其他的语音识别方法。例如,基于人工神经网络的语音识别方法,是目前的一个研究热点。目前用于语音识别研究的神经网络有 BP 神经网络、Kohonen 特征映射神经网络等。

10.4.3　隐马尔可夫模型

马尔可夫模型是俄国数学家 Markov 在 1907 年研究俄国文学家普希金作品《奥涅金》中不同音的出现规律时所提出的一个数学模型。它是研究概率随时间传递的一种方法。

如果一个过程的“将来”仅仅依赖“现在”,而不依赖“过去”,则此过程称为马尔可夫过程。时间和状态都是离散的马尔可夫过程称为马尔可夫链。

隐马尔可夫模型(hidden Markov model,HMM)是马尔可夫模型的一个拓展形式,是表示序列可能出现的一种方法。所谓的“隐”是指马尔可夫模型的状态集合观测不到。

隐马尔可夫模型数学描述:设 X_n 是状态集合,X_n 中的元素可组成马尔可夫模型,并设初始状态为 μ_0,其中转移概率为 $a_{ji}=P(X_n=i \mid X_{n-1}=j)$。每个不可观测的状态 X_n 都有生成字符与可观察的序列 Y_n 中的字符相联系,生成概率记为 $b_{ij}=P(Y_n=j \mid X_n=i)$。由不可观测的状态 X_n、初始状态 μ_0、可观测模型 Y_n 一起构成了隐马尔可夫模型。

在 HMM 中,圆圈表示状态,边表示状态之间的合法转换。每条边上有一个权值,表示从一个状态转移到另一个状态的概率。对于任何状态,只能顺着箭头的方向进行状态转移,而从一个状态发出的所有箭头上的概率之和为 1。如果状态不会再转向其他状态,被认为是终止状态。状态可以代表组成单词的字母,但这里只讨论通常的状态。

图 10.6 给出了一个具体的隐马尔可夫模型。模型中有 4 个状态,分别标记为 1 ~ 4。图 10.6 中下面部分的值是观察权值,每个状态可以发出它下面列出的符号之一,权值是概率,显示发出每个符号的相对频率。注意:一个符号可以被多个状态发出。

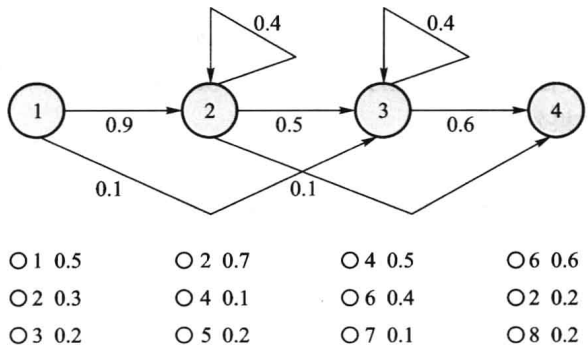


图 10.6　隐马尔可夫模型

图 10.6 中的模型可以看做一个序列生成器。例如,若从状态 1 开始,在状态 4 结束,下面是可能生成的一些序列:1 2 3 4;1 2 2 3 3 3 4;1 2 3 3 4;1 2 2 2 2 3 4。

生成某个序列的概率就是生成该序列路径上的所有概率之积。例如,对于序列 1 2 3 3 4,路径是下列边的集合{1 -2,2 -3,3 -3,3 -4},概率为 0.9 * 0.5 * 0.4 * 0.6 = 0.108。

10.4.4 基于隐马尔可夫模型的语音识别方法

20 世纪 80 年代至今,美国在语音识别方面进行的一些重大研究项目,都采取以 HMM 为基本框架的统计途径。其中包括 AT&T 公司、Bell 实验室 L. R. Rabiner 等在连接数字识别和语音响应(voice response)等方面的工作、IBM 公司 F. Jelinek 等在语音打字机方面所做的工作。HMM 成为目前语音识别主流的研究方法。

语音信号在足够短的时间段上的特征近似于稳定,而总的过程可看成是依次相对稳定的某一特性过渡到另一特性。而这个相对稳定的过程,就是语音内容的一个音素单位。在分析一个大的单词库时,要计算这种特性与特性之间过渡的概率大小。例如,字母 y 跟在 ph 后面出现的概率要大于跟在 t 后面出现的概率。

在语音识别中,输入数据是从声波中抽取出的特征。马尔可夫模型中的状态相当于声音的单元(如音素)。使用者不知道输入的特征相当于什么状态。即使特征并不准确地对应于隐马尔可夫模型中的状态,使用者也可以对可能的状态做出较好的猜测。尽管音素有一些共同的声音特征,但是不同的音素发音不同,音素间的差异可以使人们猜出某个音素到底是什么。于是,给定一个特征,可以知道哪些状态更有可能与此特征相对应。尽管不能确定到底是哪一个状态,但从隐马尔可夫模型的输出概率的大小可以猜出哪个状态的可能性更大。假设有一个特征序列,识别器获取了第一个特征,识别器并不清楚这个特征相当于哪一个状态,但识别器可以通过猜测来减少可能状态的数目(去掉那些输出特征相应观察符号的概率很小的状态)。然后,识别器获取了第二个特征,继续减少可能的状态数。在获取第三个特征后仍然以这种方式继续。当识别器获取更多的特征时,将能进一步减少可能出现的状态数量。隐马尔可夫模型建立了单词特征及一个特征出现在另一个特征之后的概率模型。

图 10.6 显示了每个状态的观察符号列表。现将模型看做一个生成器,模型发出的是一个观察符号序列,而不是状态序列。如果识别器运行 100 次,从状态 1 开始,使用者期望大约 50% 的序列以符号○1 开始,30% 的序列以符号○2 开始,20% 的序列以符号○3 开始。这些百分比就是这些符号从状态 1 产生的概率。从状态 1 出发,最可能转向的状态是 2,但有 10% 的可能会转向状态 3。因此,○2,○4,○5,○6,○7 可能跟在○1 之后。符号○2 最有可能出现在序列中的下一个位置,因为状态 2 跟在状态 1 后出现的可能性较大,并且在状态 2 产生的符号中○2 的概率远远大于其他几个符号。注意,同一个观察符号值可以被不止一个状态产生。例如,○2 可以由状态 1,2,4 产生。

给定一个观察序列,通常对两类计算感兴趣。第一,找出马尔可夫模型中最可能的路径。最可能的路径确定哪个状态序列连同状态激活的次序最有可能产生观察序列。第二,计算模型生成这个观察序列的概率。

第一类计算解释:在模型中,可能会有几条路径都能产生输入的观察序列,序列的可能性应为这几条路径上出现的概率之和。考虑下面的序列:

○1　○2　○4　○4　○6　○6

每个符号对应于一个不同的时间步骤。在时间 1 接收○1,在时间 2 接收○2,在时间 3 和时间 4

接收O4,在时间 5 和时间 6 接收O6。在这里,不关心时间间隔的大小。第 1 个观察符号是O1,O1 只能由状态 1 生成。因此,在该例中,识别器从状态 1 开始。状态 1 只能转向状态 2 或状态 3。下一个观察符号是O2,不能由状态 3 生成,所以,序列中的下一个状态应是状态 2。而从状态 2 可以转向状态 3、状态 4 或维持状态 2 不变。因为状态 4 不能生成O4,因此,必须转向状态 3 或保持状态 2 不变。现在,实际的状态是隐藏的,识别器并不知道O4 的第一次出现是在状态 2 还是在状态 3。但是,识别器可以确定产生O4 的最可能状态。由此,可以计算出模型输出相应的观察符号最有可能的状态转移路径。

第二类计算解释:在识别问题中,假设当前已经有训练成功的几个隐马尔可夫模型代表几个不同的语言单词。输入的是观察序列,而观察序列是由信号处理抽取得到的特征。不同的单词有不同的转移状态和概率,识别器的任务是对于当前的观察符号序列,确定哪一个单词模型是最可能的。也就是计算每个模型输出相应观察序列的概率,并把概率最大的那个模型作为识别结果。

基于 HMM 的语音识别过程如图 10.7 所示。首先是采集语音信号,进行解噪、分帧等预处理,然后是语音分析和特征提取,得到一组随机特征向量,再通过量化技术,将特征向量转化为一组观察序列,最后计算这组符号序列在每个 HMM 上的输出概率,其中输出概率最大的 HMM 对应的孤立字(词),就是识别结果。

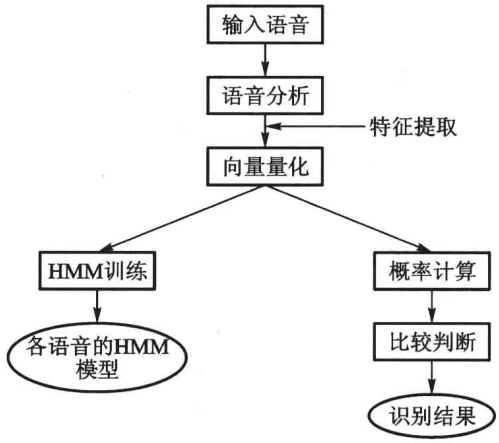


图 10.7　基于 HMM 的孤立字(词)识别

10.5　小结

自然语言理解是指机器能够执行人类所期望的某种语言功能,包括回答问题、文摘生成、释义、翻译。

自然语言理解的五个层次:语音分析、词法分析、句法分析、语义分析和语用分析。

词法分析是从句子中切分出单词,找出词汇的各个词素,从中获得单词的语言学信息并确定单词的词义。

机器翻译系统可以分成下列几种类型:直译式、规则式、中介语式、知识库式、统计式、范

例式。

翻译记忆是用户利用已有的原文和译文，建立起一个或多个翻译记忆库，在翻译过程中，系统将自动搜索翻译记忆库中相同或相似的翻译资源（如句子、段落等），给出参考译文，使用户避免重复劳动，只需专注于新内容的翻译。翻译记忆库同时在后台不断学习和自动储存新的译文，变得越来越“聪明”。

语音识别包括语音信号的采集与处理、特征参数的提取与识别等。语音识别所采用的方法一般有模板匹配法、随机模型法和概率语法分析法三种。

思考题

10.1　什么是自然语言理解？

10.2　自然语言理解过程有哪些层次？各层次的功能如何？

10.3　什么是机器翻译？有几种主要类型？

10.4　什么是语音识别？有几种主要方法？

第 11 章 人工智能在游戏设计中的应用

人工智能游戏的快速发展，为计算机游戏产业提供了新的机遇，目前人工智能技术已经成为优秀计算机游戏开发中不可缺少的部分。

本章首先介绍人工智能游戏的概念以及基本的游戏人工智能技术，然后介绍游戏角色的指导与运动、追逐与躲避、群聚、路径搜索等智能游戏角色设计的基本技术，还介绍智能游戏开发方法与开发工具，最后简单介绍扫雷机智能游戏开发。

11.1 人工智能游戏

计算机游戏(computer game)始于 1958 年的“两人网球”(Tennis for Two)游戏。但直到 20 世纪 70 年代，Atari 公司成功开发 Pong 打砖块游戏，才使更多人重视计算机游戏开发，迅速发展成为新兴的游戏产业。人工智能技术的快速发展，为计算机游戏业提供了新机遇。无论玩家是在任天堂的 Wii 游戏机上与马里奥赛车的车手比赛，还是用微软的 Xbox 360 游戏手柄在 Halo 3 上与外来入侵者对抗，人工智能技术是目前优秀计算机游戏开发中不可缺少的部分。

应用人工智能技术设计的游戏称为人工智能游戏(AI Game)，或简称为智能游戏。

人工智能游戏软件给人以某种程度智能的感觉，让玩家感觉更“好玩”，更能令人沉迷于其中，成为计算机游戏产品能否畅销的一个决定性因素。

对游戏的智能性，具有代表性的几种观点是：

① 如经典游戏“小精灵”(Pac-Man)里的魔鬼、“Unreal”第一人称射击游戏(FPS)里的虚拟战友以及其他游戏角色，它们看起来都是具有智能的生命。这种游戏可以认为是有智能的。

② 有人将游戏中的路径搜索、碰撞检测等，也列入游戏智能的范畴。

③ 游戏角色从简单的追逐、闪躲、移动，到复杂的神经网络和遗传算法应用等，能够体现角色行动的“自主性”，则说明游戏具有智能性。

目前最有影响的人工智能游戏比赛是国际机器人图灵测试比赛。比赛内容是让程序员开发出一个软件“机器人”来控制游戏人物，使其被一群专业玩家误认为是真人。机器人图灵测试是由艾伦·图灵(Alan Turing)设计的图灵测试的一个变种，使机器仅仅在文本谈话下，使评委感觉它是真人。

澳大利亚佩斯的埃迪斯·科文大学计算机与信息学院菲利普·亨斯顿(Philip Hingston)说：

“机器人图灵测试对于游戏当中的人工智能非常重要，通过向游戏玩家提供更有趣的对手，人工智能可以使游戏变得更加好玩。这也对广义的人工智能意义重大，因为它强调了人工智能的核心问题，即人类的智能如何与计算机的智能联系起来？”

史诗游戏公司首席程序员史蒂夫·波巨（Steve Polge）认为创建一个好的机器人是极具挑战性的。该公司开发了虚幻竞技场游戏。他说：“你不能总要求人工智能可以像人一样，因为人可以成为非常惹人厌的、可憎的对手。”相反，波巨认为，开发人员是在开发“能够制定出人意料的计划，并给玩家突然袭击或者设定意外挑战的人工智能，这无疑会使游戏变得更好玩。”

得克萨斯大学的计算机科学与神经科学教授里斯托·米库莱宁（Risto Miikkulainen）是机器人图灵测试参赛者之一，他试图探索合理的人机融合，设计机器人可以快速学习。他说：“当人们玩游戏时，他们适应得很快，所以在制造机器人时，你不能让他们100%精确，因为适应性是不精确的。”

包括“模拟人生”（The Sims）和“孢子”（Spore）在内的最畅销仿真游戏开发者威尔·怀特（Will Wright）希望，机器人图灵测试能够鼓励人工智能研究人员来实现人最难以捉摸的特性——情感。“机器响应成为我们生活中无处不在的部分，但它们总不那么令人满意。”怀特说，“所以，在人工智能中，认知我们感情的维度是一项有趣的任务。”

从商业角度，游戏产业主要关心的是采用新技术所带来的效益。现代游戏的制作都是十分庞大的工程，目前还不能利用人工智能来创作整个游戏。人工智能程序还不是游戏开发的重中之重。游戏开发人员还没有足够地重视人工智能技术在游戏开发中的应用，他们主要的注意力仍然放在图画的质量上，包括三维、纹理映射和实时程序渲染语言等计算机图形学技术。实际上，用花在提高图画质量上的人力和财力，可以开发出更好的智能游戏。特别是计算机图形学总有一天将不再成为游戏技术发展的驱动力。因为效用递减规律将使普通玩家无法区分当前的图形技术与以前的图形技术的差别。而人工智能技术将成为游戏革新的新驱动力，带来新风格的游戏，为主流玩家提供令人兴奋的效果，提供有助于市场竞争的新的计算机游戏产品。

11.2 游戏人工智能

11.2.1 游戏人工智能的概念与分类

适合于游戏开发的人工智能技术称为游戏人工智能（game AI）。

与一般的人工智能技术不同，游戏人工智能算法不一定要满足通用性，只要能够使角色的行为在某些场合内合理就行了。实际上，即使出现了一些人工智能错误，只要不是很明显，而且不降低游戏的娱乐性，玩家不会太在乎，甚至给玩家带来意想不到的兴奋。而对于医疗诊断等专家系统，这些错误就不能存在了。

游戏人工智能分为定性（deterministic）和非定性（nondeterministic）两类。

定性技术是游戏人工智能的基础。用定性技术设计的角色行为是特定的、可预测的。游戏中的角色分为玩家和非玩家两种角色。例如，可以用定性技术设计一个非玩家角色（NPC），沿着坐标轴前进，向某一个目标点移动，直到该角色坐标值与目标点的坐标相同，就停止前进。定性技术容易实现，理解方便，也便于游戏软件的调试和测试。在用定性技术设计游戏时，游戏开发

者必须考虑所有的可预测行为，而且无法帮助非玩家角色的学习或演化。玩家只要重复玩几次用定性技术设计的游戏，就可知道 NPC 的行为，这样，玩家就会失去玩这些游戏的兴趣，从而影响游戏软件的“生命”。

非定性技术是定性技术的一种提高。用非定性技术设计的角色行为具有某种程度的不确定性和不可预测性。在用非定性技术设计的游戏中，NPC 能够学习到玩家的作战行为，并推出新行为，甚至引发突现行为。例如，用非定性技术设计海里的一群鱼，鱼群总体上都会跟着最前面带路的那条鱼游动，而且不会出现鱼和鱼之间的穿透现象，这是属于 NPC 群聚行为，使鱼群表现得更加逼真。由于非定性的不可预测性，会给游戏软件的调试和测试带来一定的难度，但也增加玩家对游戏的兴趣，延长游戏软件的“生命”。

未来的游戏人工智能越来越注重非定性技术的研究与应用。如 Creatures，Black&White，Battlecruiser3000AD，Dirt Track Racing，Fields of Battle，Heavy Gear 等一些主流游戏，都采用了非定性技术，并具有一定的学习功能。所有非玩家角色的行为不再是事先安排好的，而是随着游戏的进展，NPC 从玩家那里不断学习，使玩家难以预测 NPC 的行为，从而使游戏更具挑战性，也扩展了游戏的生命周期。

成功的游戏软件应该采用定性技术和非定性技术相结合的方法。用定性技术解决软件的一部分调试和测试问题，用非定性技术增强软件的智能性，赋予软件更强的生命力和挑战性。任何一种非定性技术，即使功能很强，也不可能解决游戏中的所有问题，体现整个软件的智能性。所以，在使用非定性技术时，恰当的运用定性技术，对软件开发的周期、软件质量等环节都是非常有益的。

目前，在人工智能技术中，采用神经网络、遗传算法、贝叶斯技术、有限状态机等用来实现游戏中的非定性行为。

11.2.2　基本的游戏人工智能技术

目前，基本的游戏人工智能技术有：

1. 搜索技术

搜索技术在现代游戏中应用非常广泛，特别是用于路径规划。在追捕游戏中，只要追逐角色与任一非追逐角色相邻，所对应的状态就是目标状态。对于简单的情况，广泛采用宽度优先和深度优先等非启发式搜索算法。但这些算法需要指数级内存来存储游戏状态的实例。在游戏设计中广泛应用 A^* 算法，例如，在追捕游戏中以两点间欧氏距离为启发函数。A^* 算法能够保证以最少的搜索找到最优的路径。现代游戏设计中特别需要研究对抗搜索等方法。

2. 遗传算法

遗传算法是一种随机优化搜索技术，已经广泛地应用于智能游戏设计。例如，游戏设计中经常需要为某个角色寻找最优路径，往往只考虑距离是远远不够的。游戏设计中利用了一个 3D 地形引擎，需要考虑路径上的地形坡度。当角色走上坡路时应该慢些，而且更费油料。当在泥泞里跋涉时应该比行使在公路上慢。采用遗传算法进行游戏设计时，可以定义一个考虑所有这些要素的适应度函数，从而在移动距离、地形坡度、地表属性之间达到较好的平衡。可以为游戏中不同的地表面创建不同的障碍值或者惩罚值并加入适应度函数。如果道路泥泞则惩罚值大，该道路总的适应度就小，选择这条路径的可能性就小。当然，如果这条路径比较短，使得适应度增加，

选择这条路径的可能性变大。对地形坡度的处理也是类似的。最终路径的选择是所有因素的折中考虑。NPC 要平衡好这些因素是比较困难的,但最后一般都会像真人选择路径那样能够为各种地形找出最优路径,而不是仅仅找到距离最短的路径。

3. 模糊逻辑

在游戏设计中广泛应用模糊逻辑方法。例如,用模糊逻辑控制队友,或者其他非玩家角色,能够实现平滑运动,看上去更自然。在战争游戏中,计算机军队经常得配置防卫兵力,以抵抗潜在威胁的敌军。计算机军队可以根据敌军的距离以及规模等用模糊逻辑评估玩家对计算机军队即非玩家的威胁。其中,距离可以用“很近”、“较近”、“较远”和“很远”等表示;规模可以用“零星”、“少量”、“中等”、“大量”等表示;而威胁程度可以用“无”、“小”、“中”、“大”等表示。可以根据玩家或者非玩家角色的体力、武器熟练度、被击中的次数、盔甲等级等因素,将玩家或者非玩家的战斗能力分为“弱”、“较弱”、“一般”、“较强”、“很强”五个等级。

4. 神经网络

神经网络是基于生物大脑和神经系统中的神经连接结构的一系列机器学习算法的总和。在具体使用中,通过反复调节神经网络中互连结点之间的参数值(权重)来获得针对不同学习任务的最优和近似最优反馈值。本章最后将介绍一个用神经网络控制扫雷机的实例。

5. 一阶谓词逻辑

在游戏设计中,用一阶谓词逻辑描述变化世界的方法,称为情景演算(situation calculus)。用一阶谓词逻辑来计算给定情况下智能主体的应有动作,用自动推理来决定达到最优状态所需采取的动作序列。正是由于需要智能角色去思考变化的世界,而情景演算正好适合描述变化的事物,因此得到较为广泛的应用。但它不适用于对性能要求很高的实时游戏中,而且用逻辑语言来描述复杂游戏是比较困难的。

6. 专家系统

专家系统用于模拟专业玩家的行为,游戏开发人员编写知识库控制角色的行为。尽管智能游戏的知识库在表达上不需要像其他专家系统的知识库那样复杂,但随着游戏的日益复杂化,专家系统越来越难以建立。现在少数游戏专家系统已引入机器学习。机器学习将在未来智能游戏开发中得到越来越广泛的应用。

7. 多智能体

通过多个彼此竞争和协作的智能主体描述角色之间的交互。游戏中存在的众多角色彼此既有竞争也有合作,因此可以用多智能体来自然地产生智能行为。

8. 人工生命

由简单的个体行为组合成复杂模式的研究,称为人工生命(artificial life)。这是多智能体系统中的一种,着重研究如何为虚拟环境中的智能主体赋予某些生物体的共性。游戏 The Sims 和 SimCity 的成功证明了人工生命技术的有效性和娱乐价值。例如,用人工生命设计群聚,控制对象的智能化运动,协调多个智能主体的动作,使它们在整体上看起来像逼真的动物群。

9. 基于范例的推理

这种技术分析数据库中存放的(历史)输入数据和相应的最优输出结果,然后通过对比现有输入数据和历史数据来推知输出结果。这种技术模拟了人们在处理新的情况时参考以前相似经历的做法。

10. 有限状态机(finite state machine,FSM)

这是一种简单的基于规则的系统,它包含了有限个“状态”和状态之间的“转移”,彼此连成一个有向图。有限状态机在每一时刻都只能处于某一状态。有限状态机在智能游戏设计中应用最为广泛,其主要原因是游戏中的非玩家角色常可以设计为含有单个状态变量,而变量值可以表达为有限状态机当前结点的形式。结点的输入和输出则驱动角色在那个状态下的行为。有限状态机的缺点是当状态数目和状态转移数目增加时复杂度大大增加。

11. 决策树(decision tree)

决策树表达一系列产生式规则 if-then 形式的条件判断,容易转化为一组产生式规则。在决策过程中,从决策树的根结点输入一组数据,在每个分叉处根据某个输入值选择其中一个子结点,依此类推。国际象棋和西洋双陆棋等许多棋类游戏中成功采用了游戏树及其搜索方法。决策树在计算机游戏中经常被用来表达控制器。决策树的编程容易实现。非专业程序设计人员可以通过图像用户界面方便地建立与维护决策树。如果每个叶结点所对应的不是单一的行动选择,而是一个所有可能行动的概率分布函数,那么,就成为随机决策树,可用于表达随机化控制器。

12. 置信网络

置信网络采用了概率理论来解决现实世界中的不确定性和不完全知识等问题,是描述不同现象之间内在因果关系的工具。还可以用于推导现实世界的状态和预测各种动作的可能结果。这种技术特别适用于解决智能游戏中涉及的许多子问题。

人工智能游戏设计中或多或少地采用了上述技术,所获得的效果也不尽相同。当开发期限紧张和资源有限的情况下,很多游戏开发人员都倾向于采用基于规则的系统,因为编写、理解和调试基于规则的智能游戏程序比较容易。

11.3　游戏中的角色与分类

游戏中的活动对象分为两类。一类是背景中的活动对象,如天上飘着的云、飞过的鸟等。这类对象的造型和行为要显得逼真也不容易,需要掌握 2D 或者 3D 图形和动画技术,还需要有艺术修养。但它们在游戏中无需人工干预,变化也不多,控制的逻辑比较简单。

另一类活动对象是游戏中的各种角色,或者称为游戏代理,如虚拟的人、兽、怪物、机器人等。角色的活动方式必须变化多端才行,否则游戏就不好玩,所以需要设计比较复杂的角色控制逻辑。尤其是玩家对手的代理的控制逻辑最复杂。例如,要开发一个猫捉老鼠的游戏。假设你是玩家,你的代理是猫,即猫的行为由你操纵,而老鼠的行为则完全需要由计算机程序来控制。当猫不出现时,老鼠必须到处觅食或者打洞,以解决生存必须的吃住问题。一旦发现有猫,则立即躲进洞里。如果附近没有洞,则要立即逃窜。逃窜的方向应该和猫是反方向的。如果途中遇到障碍物挡住了去路,则要改变方向,可以向左或者向右,但不应该回头跑,除了前面是死胡同。老鼠能遵循这样的逻辑来行动,就是游戏编程中为老鼠设计的智能。虽然游戏里的猫也需要人工智能,但很简单,那就是听话,能够听从你用键盘或者鼠标进行的指挥。

所有角色扮演类游戏都需要智能。越是好玩的游戏越是需要复杂的智能。但并不是所有的游戏都需要人工智能。例如,Windows 提供的接龙和挖地雷等游戏就没有人工智能问题。网上

提供的两人对弈的象棋、围棋、军棋类游戏也不需要人工智能技术。但一旦要求机器与人对弈，那就需要很高的智能了。

游戏角色的智能水平是一款游戏可玩性的决定性因素之一，因而也是游戏开发中需要考虑的重要问题。人工智能技术能够实现智能角色，增强游戏的体验并改善游戏的可玩性。

游戏中的角色除了根据角色的属性分为玩家和非玩家两种类型，还可以有下面几种分类方法：

① 根据运动方向与朝向的相关性，游戏角色分为靠转向力改变的角色和自由运动的角色两种类型。靠转向力改变运动方向的角色，如汽车、飞机之类的物体，它们的速度向量的方向必须与其朝向相同或相反，速度方向的改变只能靠施加在头部或尾部的转向力(steering force)。特别是有的角色要求当速度大小不为0时，施加转向力才有效。例如，静止的汽车，对其方向盘的操作是毫无效果的。

② 根据角色的智能性，游戏角色分为智能性角色和无智能性角色两种。角色运动的结果主要是位置和朝向的变化，而运动的方式可有多种。对玩家来说，通过操控键盘、鼠标或游标来实现玩家化身的运动，如行走、跑步、转向、坐蹲等。对智能非玩家角色来说，通过预置各种运动算法，当运动条件和环境符合时，自动产生运动方式。对无智能性非玩家角色，则不产生任何运动，只能靠场景切换或摄影机移动来实现隐藏或现身。

11.4 智能游戏角色设计基本技术

角色的智能性具体表现在指导与运动、追逐与躲避、群聚、路径搜索、智能搜索引擎等方面，下面做个简单介绍。

11.4.1 游戏角色的指导与运动

有两种方法指导角色行为：预定义行为(predefined behavior)和目标导向行为(goal-directed behavior)。区分预定义行为和目标导向行为的依据是：角色是否唯一地选择行动。

预定义行为指导角色行为方法中，角色的所有行为都是动画师和程序设计人员预先设计的。这里讨论的主要是如何处理预先并不完全确定的角色行为的技术。两种比较流行的方法是反应行为规则(reactive behavior rules)和分层有限状态机(hierarchical finite state machine，HFSM)。下面简单介绍反应行为规则。

当角色的行为仅仅取决于当前外部环境的感知时，称为反应行为。在反应行为中，角色不记忆以前经历的情景，没有自己内部情景的表达，因此，无论角色收到的信号顺序如何，只要是同样的刺激信号，其反应行为就是一样的。

反应行为规则是目前最普及的生成角色行为的方法，用一些简单的产生式规则就能够产生复杂的行为。例如，下面简单的左转规则：

若　有障碍在前面　则　向左转　否则　直走

IF blocked ahead THEN turn-left ELSE go-straight

角色没有必要知道关于迷宫的知识，只要依据上面的规则执行，就能够找到走出迷宫的道路。

由于行为规则是预先确定的，所以一旦选定了行动，就不能再改变。许多情况下，角色依据领域知识，很快可以预测所选的行动是不合适的。但是，不管是否有其他更合适的行动方案，自主角色都必须按照当前情景预定义行为规则行动。因此，象分层有限状态机为角色保留了某些简单的内部状态信息，设计控制器就比较容易。

在游戏设计中，许多角色选择行动是非确定性的，有多个方案可以选择。当角色选择了某个行动方案后，如果后来看出这个方案不能达到期望的目标，可以返回选择其他方案，直到选出合适的方案。非确定性选择行动的角色容易接受指导，通过给角色指定目标，使角色能够利用自己的领域知识，了解它选择的行动方案是否能够达到目标。但非确定性选择行动的响应速度比较慢。游戏设计时应该折中考虑预定义行为和目标导向行为指导角色的方法。因此，指导角色的第三种方案是将预定义行为和目标导向行为结合起来。

11.4.2　游戏角色的追逐与躲避

在游戏中，追逐与躲避是角色最普遍的行为。它与随机移动一起构成了早期游戏中主要使用的运动模式。

1. 随机移动

随机移动能够产生角色漫无目的徘徊的感觉。对于自由运动的角色，可以不断地产生随机的速度向量；对于靠转向力来改变速度方向的角色，可以不断产生随机的转向力。若使用完全的随机数来控制角色的移动，行动看起来不稳定，所以应该保持随机过程的一致性，可采用噪声函数法、投射目标法，也可以记录从前走过的路，避免走回头路等来体现随机性。

2. 追逐与躲避

追逐与躲避是功能相反的两种运动模式，在游戏中到处可见。例如，在太空战机射击游戏或角色扮演游戏，游戏中的非玩家角色都有机会试着追逐或躲避玩家角色；在角色扮演冒险游戏中，也想要个巨人或某种可爱的生物，追着你的玩家角色；在第一人称射击游戏和飞行模拟游戏中，需要算出导弹的轨迹，攻击玩家或玩家的飞行器等。对于任何一种情况，都需要一种逻辑手段，使角色进行追逐或躲避。

3. 躲避障碍物

玩家与非玩家角色在游戏中都应该符合现实场景的客观规律，不能出现互相穿透现象。如玩家穿过非玩家角色、玩家走出虚拟场景、玩家穿越地面等。对于这些现象，在设计过程中，可以理解为是追逐与躲避的一种特例。躲避障碍物的方法往往采用传感器、球包围盒等方法，自动检测玩家与物体的距离，当达到测定距离时，可以自动实现玩家不能靠近障碍物。

使用传感器来判断障碍物的基本方法是利用传感器从当前位置沿当前速度向前方一段距离作投射。传感器的末端表示按照当前的速度将要到达的位置。若在这个投射上发生了碰撞，则表示按当前速度行进将会发生碰撞，以此要求角色改变速度。传感器的长度可以与角色的速率成正比。转向力的方向为障碍物表面的碰撞点的法向量的方向，转向力的大小可以与传感器的末端穿进障碍物的深浅成正比。

球包围盒或长方体包围盒是将物体抽象地定义为一个位置坐标和一个碰撞半径来检测角色是否将与物体相碰撞。

11.4.3 游戏角色的群聚

角色的群聚(flocking)行为是基于 Craig Reynolds 提出的 Boids 算法,该算法由三个简单的规则组成:

① 聚合(cohesion):每一个成员移向它的邻居成员的平均位置。

② 对齐(alignment):每一个成员朝向它的邻居成员的平均朝向。

③ 分离(separation):每一个成员与邻居保持一定的距离,以免相撞。

这些规则都可以利用追逐或躲避算法来实现,可以产生非常逼真的群聚效果。

团队中的各个成员所考虑的邻居成员的数量可以通过该成员的视角和视野来控制,只有在视角和视野范围内的成员才予以计算。设置大的视角会使队形向横向铺开来,而小的视角会使队形纵向排开,形成细长的队列。可以为不同的规则制定不同的视角和视野,这些需要在游戏的实现过程中进行调整。

一个成员的行为结果是上述三个规则计算出的转向力的组合。实际上这三个规则计算出的转向力还可以和由其他行为,如追逐、躲避、跟随首领等计算出的转向力组合。有如下几种组合方法:

① 为每一个转向力设置一个权值,将所有的转向力加权,再截取不超过转向力的最大值。这种方法的缺点是权值难以设定和调整,因为有些转向力的施加是必须的,例如,障碍物躲避是必须的,即使它与群聚的转向力发生冲突,也应该保证角色不与障碍物相撞。若过多增加该转向力的权值,会产生不理想的行为,在两个成员接近时行为不稳定,且容易使团队分离。

② 在①的基础上为每个转向力增加一个优先级。除了按照权值组合外,每一次更新按照转向力的优先级的高低顺序组合,若组合的结果大于转向力的最大值,则进行截取并忽略低优先级的转向力。

③ 不设定权值,而是为每个转向力增加一个概率和一个优先级。每一次更新只有一个转向力起作用。首先考虑优先级高的转向力,只有在概率命中,并且结果不为 0 才进行组合;否则,考虑下一个优先级的转向力。在这种方法中,由于每一次更新只计算一个转向力,减少了计算量,但却是以降低精度为代价的。

11.4.4 游戏角色的路径搜索

路径搜索是智能游戏软件中最基本的问题之一。有效的路径搜索方法可以让角色看起来很真实,使游戏变得更有趣味性。

在简单的情况下,有以下几种路径搜索算法:

① 在不考虑躲避障碍物的情况下,路径搜索算法即为追逐算法。

② 若考虑躲避障碍物,则随机移动;如果直线上没有障碍物,则沿直线向目标移动。这种方法在开阔的场景,且障碍物数量较少、体积较小的情况下(如只有少量的树木)是可行的。

③ 当场景中有体积较大的障碍物,则从追逐者位置向目标作投射得到一条直线,然后沿直线向目标移动。当碰到障碍物时,则沿着障碍物的边缘移动,直到又回到直线上,继续沿着直线移动。但当碰到障碍物的内转角时,不太适合采用这种方法。

④ 面包屑寻路法是记录目标走过的路径,让追逐者沿着这条路径移动。记录目标移动路径

的方法是在目标走过的地方留下玩家看不见的记号(面包屑),追逐者可以按照不同的面包屑顺序移动。采用这种方法可能会出现追逐者走回头路的现象。

⑤ A^* 搜索算法是一种启发式搜索策略,能保证在任何起点与终点之间找到最佳路径。由于 A^* 搜索算法相对比较复杂,要求 CPU 做大量的计算,因此,现在还不是游戏软件开发中最常用的路径搜索算法。特别是当 CPU 功能不太强,尤其是解决多角色游戏的路径选择问题,A^* 算法不是最佳选择,否则会影响游戏效果。由于路径的类型很多,寻求路径的方法应与路径的类型、寻径的需求有关,A^* 算法不一定适合所有场合。例如,如果起点和终点之间没有障碍物,有明确的视线,就没有必要使用 A^* 算法。

11.4.5　智能搜索引擎

在游戏开发中,核心技术是构架游戏引擎,它是决定游戏质量的关键。游戏引擎就像赛车的引擎。引擎是赛车的心脏,决定着赛车的性能和稳定性。赛车的速度、操作感这些直接与车手相关的指标都是建立在引擎的基础上的。游戏也是如此,玩家所体验到的剧情、关卡、美工、音乐、操作等内容都是由游戏引擎直接控制的。它扮演着中场发动机的角色,将游戏中所有元素捆绑在一起,在后台指挥它们同时、有序地工作。简单地说,引擎就是用于控制所有游戏功能的主程序,从计算碰撞、物理系统和物体的相对位置,到接受玩家的输入,以及按照正确的音量输出声音等。

从狭义上讲,游戏引擎就是指在对部分通用技术细节进行整理和封装的基础上,形成一个面向游戏应用的应用程序接口(API 函数),使得游戏开发人员不必再关心底层技术的实现细节,只需要调用游戏引擎中相关的 API 函数。同样在游戏里,剧情的进行、形形色色的角色衬托、各种场景的交换,也都是游戏引擎事先就约定的一种模式或大体的框架,然后依靠一些复杂的数据库来组织完成。

游戏引擎从 20 世纪 90 年代初开始,经历了诞生、发展、成熟三个阶段。在每个阶段,都有优秀的游戏引擎产生,它们组成了游戏引擎的演化史。从 2000 年开始,游戏引擎朝着两个不同的方向分化。一是通过增加更多的叙事成分和角色扮演成分以及加强游戏的人工智能来提高游戏的可玩性,如“半条命”、“神偷”和“杀出重围”等;二是向着纯粹的网络模式方向发展,如 Id Software 公司在 Quake 2 出色的图像引擎的基础上加入更多的网络成分,推出了一款完全没有单人过关模式的纯粹的网络游戏——“雷神之锤 3 竞技场”,它与 Epic 公司稍后推出的“虚幻竞技场”一同成为引擎发展史上的一个转折点。

在人工智能方面真正取得突破的游戏是 Looking Glass 工作室的“神偷:暗黑计划”。这个游戏的故事发生在中古年代。玩家扮演一名盗贼,任务是进入不同的场所,在尽量不引起别人注意的情况下窃取物品。“神偷”采用的是 Looking Glass 工作室自行开发的 Dark 引擎。Dark 引擎在图像方面比不上“雷神之锤 2”或“虚幻”,但在人工智能方面,它的水准却远远高于“雷神之锤 2”和“虚幻”。这个游戏中的角色懂得根据声音辨认敌人的方位,能够分辨出不同地面上脚步声,在不同的光照环境下有不同的视力,发现同伴的尸体后会进入警戒状态,还会针对玩家的动作做出合理的反应,玩家必须躲在暗处不被敌人发现才有可能完成任务。这些在以前那些纯粹的杀戮游戏中是根本看不到的。此后的绝大部分第一人称射击游戏都或多或少地采用了这种隐秘的风格,包括“荣誉勋章:盟军进攻”(Medal of Honor:Allied Assault)。

11.5 智能游戏开发方法与开发工具

11.5.1 智能游戏开发方法

实现游戏智能有两种不同的方式。一种是采用传统的编程技术,使系统呈现智能的效果,而不考虑所用方法是否与人或者动物所用的方法相同。这种方法称为工程学方法(engineering approach),它已经在文字识别、电脑下棋等许多领域应用。另一种是模拟法(modeling approach),它不仅要看效果,还要求实现方法也和人类或动物机体所用的方法相同或相类似,如遗传算法、神经网络等。

为了得到相同的智能效果,两种方法通常都可以使用。采用前一种方法,需要人工详细规定程序逻辑。因此,如果游戏简单,这种方法还是很方便的。但如果游戏复杂,角色数量和活动空间增加,相应的逻辑复杂度就会以指数增长,人工编程就会非常复杂,容易出错。而采用后一种方法,设计者不需要详细规定程序逻辑,而是给每个角色设置一个智能模块来进行控制。这个智能模块开始什么也不懂,就像初生婴儿那样,但他能够学习,能够渐渐地适应环境。利用这种方法来实现游戏智能,无需对角色的活动规律作详细规定,应用于复杂问题时,通常会比前一种方法更省力。

11.5.2 智能游戏开发工具

选择合适的游戏开发工具,对于游戏功能的实现、游戏的复杂度、游戏的智能性、游戏的开发效率、游戏功能的可扩充性和移植性,以及开发者使用工具的能力、开发付出的工作量、游戏软件实现的难易程度等,都起着至关重要的作用。

游戏开发工具分为创作工具类和编程语言类两大类。

创作工具类是由软件开发商开发的集各种功能在内的专业软件,主要采用“拖、拉、放”的形式进行开发。如 Virtools、PRG Maker XP、FPS Creator、Vega、Game Jackal Pro、ZD Soft Game Recorder、Game Editor 等开发工具。

编程语言类主要利用各种高级语言,通过编程实现游戏的开发,如 VC + +、C、Java、J2ME、VC. NET、Delphi、VB 等。

选择游戏开发工具应综合考虑各种因素,可以选择创作工具类,也可以选择编程语言类,或者是两者的结合。

11.6 扫雷机智能游戏开发

扫雷机工作在一个很简单的环境中,只需要若干扫雷机和随机散布的许多地雷。游戏设计目标是设计一个 BP 神经网络,能够自己演化去寻找地雷。

基本方法是用一个 BP 神经网络控制一个扫雷机,它的权值用遗传算法进行演化,使扫雷机更聪明。

可以把扫雷机看成与坦克一样,通过左右两个能够转动的履带式轮轨来行动。通过改变两

个轮轨的相对速度来控制扫雷机前进的速度以及向左或者向右转弯的角度。因此，选择左右两个履带轮的速度作为 BP 神经网络的两个输出。

这里为每个扫雷机装上触觉器，从而能够避开障碍物。这里的触觉器是从扫雷机向外辐射出来的几根线段，如图 11.1 所示。线段的长度和数目都是可以调整的。

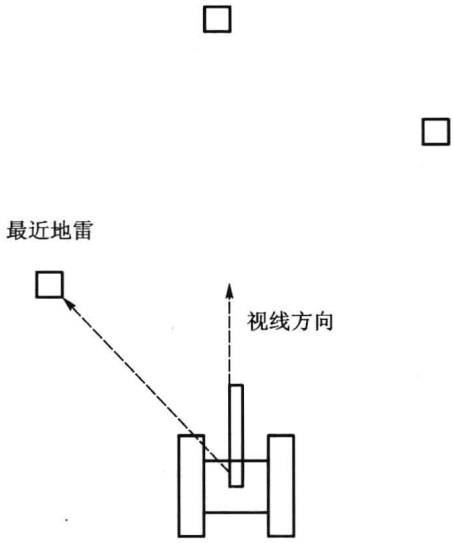


图 11.1　扫雷机的控制

在每一帧中都要调用一个函数来检测每个触觉器是否与周围世界的障碍物的边界线相交。这些触觉器检测到的数据输入扫雷机的神经网络。因此，选择控制扫雷机的输入信息是扫雷机的视线向量和扫雷机到最近地雷的向量。

选择几个（例如 10 个）神经元作为隐层，构成一个三层 BP 神经网络，如图 11.2 所示。神经元的非线性函数取 S 型函数。

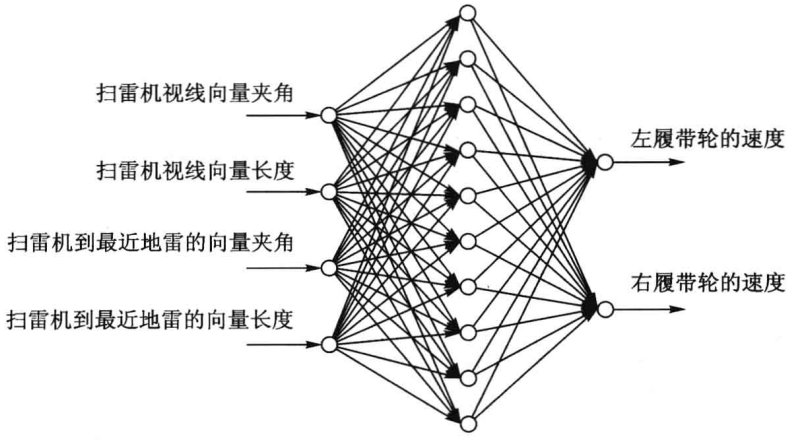


图 11.2　控制扫雷机的神经网络

可以将神经网络的所有输入进行规范化。对于扫雷机的视线向量已经是一个规范化向量，即长度等于1，分量都在0~1之间。而扫雷机到达最近地雷的向量可能很大，其中的一个分量甚至可能达到窗体宽度或者高度。如果把这个数据以它的原始状态输入到网络，将使它的影响比视线向量大很多，会使网络性能变差。因此需要把它规范化，变换到0~1之间。

首先随机设置权值。然后用遗传算法进行进化。对上面得到的神经网络从左到右依次读每一层神经元的权值以及阈值，保存到一个向量中，就实现了神经网络的编码。然后采用各种方法进行选择、交叉与变异。当产生新的一代，就用新个体表示的权值替换扫雷机神经网络的权值。当某个扫雷机找到了地雷就增加它的适应度。扫雷机对应的神经网络不断进化，扫雷机的智能就不断提高。

可以用神经网络解决许多游戏中的避障和搜索这两个人工智能问题。在扫雷机游戏中增加了一些要求扫雷机躲避的障碍物。障碍物的顶点坐标存放在一个缓冲区里。

设置适应性函数反映扫雷机与障碍物的碰撞情况，适应度值越高，表示扫雷机避障性能越好。如果扫雷机产生一次碰撞就降低这个扫雷机的适应度。

11.7 人工智能游戏的现状与未来

虽然多数游戏软件都具有一些人工智能的表现，例如，经典游戏“小精灵”里的魔鬼、第一人称射击游戏“半条命”里的虚拟队友以及许多介于两者之间的其他游戏角色。但现阶段游戏中的人工智能的开发，不是人工智能算法理论上的改进和实现，而是呈现给玩家一个看上去合理的行为。主要原因有以下几个方面：

① 游戏软件的实时性限制了计算量较大的人工智能算法的应用。

② 来自于人工智能理论本身发展的限制。虽然专家系统、模糊逻辑、人工神经网络等方法渐渐为游戏AI开发者所考虑，但由于建模的复杂性，这些工具不能很好地求解游戏中的问题。

③ 游戏中的角色并不是真正的智能体。例如，由于应用环境的相似性，游戏AI开发者大量借鉴机器人学的成果，然而即使是比较先进的机器人，它的机械复杂性也是有限的。相比之下，骨架动画的艺术创作的复杂度几乎是无限的。这些角色的基本行为更大程度上是美术工作者创造的结果的回放，甚至是通过动作捕捉(motion capture)技术所作的人类行为的记录。现有的人工智能技术水平无法与之匹配。

④ 游戏中实现人工智能的主要目的，是为了让玩家产生真实的环境的错觉，并不是为了使求解问题的性能度量最大化。例如，游戏中一个按照路径规划算法寻路的对手，玩家可能认为它的寻路能力过于完美，不符合人类真实的情况。

计算机游戏不仅是人工智能的重要应用领域，而且是人工智能的高效研究平台。利用计算机游戏进行人工智能研究相对其他应用领域比较方便而又富有乐趣。例如，在机器人等实际应用系统中研究人工智能，首先要解决复杂的底层的检测、识别与控制问题。但在虚拟的计算机游戏中，角色只要查询图像数据库，就可以解决障碍识别等问题。计算机游戏中的“玩家”能够直接观察角色的行为，及时提供反馈信息，进行人机交互，发出控制指令。这些对于人工智能研究及应用都是很重要的。许多人喜欢玩游戏，也容易验证人工智能技术的有效性。特别是许多人愿意出钱，促进了游戏产业的发展和人工智能技术的研究。游戏开发中的人工智能正处于一场

革命之中，必将迅速得到发展并推动整个游戏产业的飞速发展。

11.8　小结

应用人工智能技术设计的游戏称为人工智能游戏。

适合于游戏开发的人工智能技术称为游戏人工智能。

游戏人工智能分为定性和非定性两类。用定性技术设计的角色行为是特定的、可预测的。用非定性技术设计的角色行为具有某种程度的不确定性和不可预测性。

基本的游戏人工智能技术有：搜索技术、专家系统、遗传算法、模糊逻辑、神经网络、一阶谓词逻辑、多智能体、人工生命、基于范例的推理、有限状态机、决策树、置信网络等。

游戏中的角色根据角色的属性分为玩家和非玩家；根据运动方向与朝向的相关性，分为靠转向力改变的角色和自由运动的角色；根据角色的智能性，分为智能性角色和无智能性角色。

角色的智能性具体表现在指导与运动、追逐与躲避、群聚、路径搜索、智能搜索引擎等方面。

如何指导角色行动有两种方法：预定义行为和目标导向行为。区分预定义行为和目标导向行为的依据是：角色是否唯一地选择行动。

预定义行为方法中，角色的所有行为都是动画师和程序设计人员预先设计的。

当角色的行为仅仅取决于当前外部环境的感知时，称为反应行为。

路径搜索是智能游戏软件中最基本的问题之一。A^* 搜索算法是人工智能中的一种启发式搜索策略，用来解决最短径优化问题。A^* 算法能保证在任何起点与终点之间找到最佳路径，但要求 CPU 做大量的计算。

智能游戏不仅是人工智能的重要应用领域，而且是人工智能的高效研究平台。人工智能技术将迅速推动整个游戏产业的飞速发展。

思考题

11.1　什么是人工智能游戏？什么是游戏人工智能？

11.2　指导角色行动有哪两种方法？分析两者的优点和缺点。

11.3　设计一个简单的智能游戏。

附录　部分习题解答

第 2 章习题解答

2.1 (1) **定义谓词**:$likes(x,y)$为 x 喜欢 y。x:人;$flower1$:梅花;$flower2$:菊花。

$(\exists x)(likes(x,flower1)) \vee (\exists x)(likes(x,flower2)) \vee (\exists x)(likes(x,flower1) \wedge likes(x,flower2))$

(2) **定义谓词**:$plays(z,y,x)$为 z 在 x 时间玩 y。x:下午。

$(\forall x)(plays(he,football,x))$

(3) **定义谓词**:$have(x,y)$为 x 有 y。x:人。

$(\forall x)(have(x,rice) \wedge eat(x,rice))$

(4) **定义谓词**:$like(x,y)$为 x 喜欢 y。x:人。

$(\forall x)[like(x,play(basketball)) \rightarrow like(x,play(volleyball))]$

(5) **定义谓词**:$pass(x,y)$为 x 通过 y。$study(x,y)$为 x 到 y 学习。x:人。$exam(English)$为外语考试。

$(\forall x)[\neg pass(x,exam(English)) \rightarrow \neg (study(x,abroad))]$

2.2 (1) $(\forall x)$的辖域为$(P(x,y) \vee (\exists y)(Q(x,y) \wedge R(x,y)))$。$(\exists y)$的辖域为 $Q(x,y) \wedge R(x,y)$。谓词公式中 x 为约束变元,$P(x,y)$中的 y 为自由变元。$Q(x,y) \wedge R(x,y)$中的 y 为约束变元。

(2) $(\exists z)$、$(\forall y)$的辖域为 $P(z,y) \vee Q(z,x)$。谓词公式中 z,y 为约束变元,x,u,v 为自由变元。

(3) $(\forall x)$的辖域为$\neg P(x,f(x)) \vee (\exists z)(Q(x,z) \wedge \neg R(z,y))$。$(\exists z)$的辖域为 $Q(x,z) \wedge \neg R(z,y)$。谓词公式中 x,z 为约束变元,y 为自由变元。

(4) $(\forall z)$的辖域为$(\exists y)((\exists t)(P(z,t) \vee Q(y,t)) \wedge R(z,y))$。$\exists y$ 的辖域为$(\exists t)(P(z,t) \vee Q(y,t))$,$(\exists t)$的辖域为 $P(z,t) \vee Q(y,t)$。谓词公式中 $R(z,y)$的 y 为自由变元,z 为约束变元。

2.3 设对谓词指派的真值为:

$P(1,1)=T, P(1,2)=F, P(2,1)=T, P(2,2)=F$

$Q(1,1)=F, Q(1,2)=F, Q(2,1)=T, Q(2,2)=T$

上述指派就是对题中给出的谓词公式的一个解释。

当 $x=1, y=1$ 时,有 $P(1,1)=T, Q(1,1)=F$,则 $P(1,1) \rightarrow Q(1,1)$的真值为 F

当 $x=2,y=1$ 时，有 $P(2,1)=T,Q(2,1)=T$，则 $P(2,1)\rightarrow Q(2,1)$ 的真值为 T

当 $x=1,y=2$ 时，有 $P(1,2)=F,Q(1,2)=F$，则 $P(1,2)\rightarrow Q(1,2)$ 的真值为 T

当 $x=2,y=2$ 时，有 $P(2,2)=F,Q(2,2)=T$，则 $P(2,2)\rightarrow Q(2,2)$ 的真值为 T

即对个体域 D 中的所有 y 总存在 x 使 $P(x,y)\rightarrow Q(x,y)$ 的真值为 T。所以谓词公式 $(\exists x)(\forall y)P(x,y)\rightarrow Q(x,y)$ 在该解释下的真值为 T。

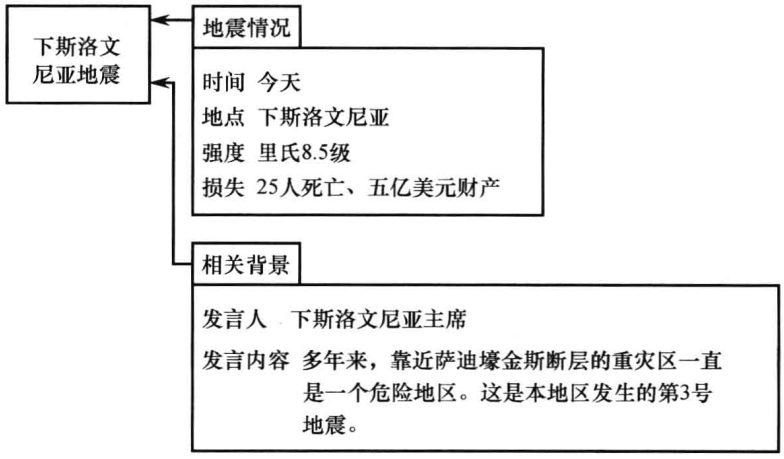
2.4 (1) $Older(x,y)$：x 比 y 大。$Older(Zhang,Li)\rightarrow\neg Older(Li,Zhang)$

(2) $Man(x)$：x 为男；$\neg Man(x)$：x 为女；$Marry(x,y)$：x 与 y 结婚。

$Marry(甲,乙)\rightarrow(Man(甲)\wedge\neg Man(乙))\vee(Man(乙)\wedge\neg Man(甲))$

(3) $Honest(x)$：x 是老实人；$Lie(x)$：x 说谎。$Honest(x)\rightarrow\neg Lie(x)$；$Lie(Zhang)\rightarrow\neg Honest(Zhang)$

2.5



第 3 章习题解答

3.1 因为 $A,A\rightarrow C\Rightarrow C$；$B,C\Rightarrow B\wedge C$；$B\wedge C,B\wedge C\rightarrow D\Rightarrow D$；$D,D\rightarrow Q\Rightarrow Q$；所以，$Q$ 为真。

3.2 $\neg P(x,f(x),z,g(x,z))$

3.3 $(\forall y)(\neg P(b,g(y))\vee R(b,y,f(a)))$

3.4 (1) $\{P(z,y),Q(u,v)\}$

(2) $\{\neg P(x,y)\vee Q(x,y)\}$

(3) $\{\neg P(x,f(x))\vee R(x,f(x)),\neg Q(y,f(y))\vee R(y,f(y))\}$

(4) $\{\neg P(x,y)\vee R(x,y),\neg Q(u,v)\vee R(u,v)\}$

(5) $\{\neg P(x,y)\vee Q(x,y)\vee R(x,f(x,y))\}$

(6) $\{P(a,b,z,g(z),v,f(z,v)),Q(a,b,z,g(z),v,f(z,v))\vee\neg R(a,z,f(z,v))\}$

(7) $\{\neg P(x,f(x))\vee Q(x,g(x)),\neg P(y,f(y))\vee\neg R(y,g(y))\}$

3.5 解：(1) $\neg P$ 与 P 归结得 NIL，故子句集不可满足。

(2) $P\vee Q$ 与 $P\vee\neg Q$ 归结得 P；$\neg P\vee Q$ 与 $\neg P\vee\neg Q$ 归结得 $\neg P$。P 与 $\neg P$ 归结得 NIL，故

子句集不可满足。

(3) 由于存在 $R(a)$,不可能归结出空子句,所以该子句集是可满足的。

(4) $P(a)$与$\neg P(y) \vee R(y)$归结得 $R(a)$;$S(a)$与$\neg S(z) \vee \neg R(z)$归结得$\neg R(a)$;$R(a)$与$\neg R(a)$归结得 NIL,故子句集不可满足。

(5) $R(b)$与$\neg R(z) \vee L(a,z)$归结得 $L(a,b)$;$P(a)$与$\neg P(x) \vee \neg Q(y) \vee \neg L(x,y)$归结得$\neg Q(y) \vee \neg L(a,y)$,与 $Q(b)$归结得$\neg L(a,b)$,再与 $L(a,b)$归结得 NIL,故子句集不可满足。

3.6 (1) F_1 的子句集为 $P(a,b)$。$\neg G$ 的子句集为$\neg P(x,b)$,归结得 NIL,故 G 为 F_1 的逻辑结论。

(2) F_1 的子句集为$\{P(x),Q(a) \vee Q(b)\}$;$\neg G$ 的子句集为$\{\neg P(x) \vee \neg Q(x)\}$;$Q(a) \vee Q(b)$与$\neg P(x) \vee \neg Q(x)$归结得$\neg P(b)$;再与 $P(x)$归结得 NIL。

(3) F_1 的子句集为① $P(f(a))$;② $Q(f(b))$;$\neg G$ 的子句集为③ $\neg P(f(a)) \vee \neg P(y) \vee \neg Q(y)$。①与③归结得④ $\neg P(y) \vee \neg Q(y)$;再与②归结得⑤ $\neg P(f(b))$;再与① 归结得 NIL。

(4) F_1 化为子句集:① $\neg P(x) \vee \neg Q(y) \vee \neg L(x,y)$;$F_2$ 化为子句集② $P(a)$,③ $\neg R(y) \vee L(a,y)$;对$\neg G$ 化为子句集:④ $R(b)$,⑤ $Q(b)$。②与①归结得⑥ $\neg Q(y) \vee \neg L(a,y)$;⑥与③归结得⑦ $\neg R(y) \vee \neg Q(y)$;⑦与④归结得⑧ $\neg Q(b)$;⑧ 与⑤进行归结得 NIL。

(5) 将 F_1,F_2,$\neg G$ 化为子句集:F_1:① $\neg P(x) \vee Q(x)$;② $\neg P(x) \vee R(x)$;F_2:③ $P(a)$;④ $S(a)$;$\neg G$:⑤ $\neg S(z) \vee \neg R(z)$。对子句进行归结:③与⑤归结得⑥ $\neg R(a)$;与①进行归结得$\neg P(a)$;与③归结得 NIL。

(6) 化为子句集:F_1:令 $y=f(z)$,① $\neg A(z) \vee B(z) \vee D(z,f(z))$,② $\neg A(u) \vee B(u) \vee C(f(u))$;$F_2$:③ $E(a)$④ $A(a)$⑤ $\neg D(a,v) \vee E(v)$;F_3:⑥ $\neg E(p) \vee \neg B(p)$;⑦ $\neg G$:$\neg E(t) \vee \neg C(t)$。③和⑥归结得⑧ $\neg B(a)$;⑧和②归结得⑨ $\neg A(a) \vee C(f(a))$;⑨和④归结得⑩ $C(f(a))$;⑩和⑦归结得⑪ $\neg E(f(a))$;⑧和①归结得⑫ $\neg A(a) \vee D(a,f(a))$;⑫ 和④归结得⑬ $D(a,f(a))$;⑬和⑤归结得⑭ $E(f(a))$;⑪和⑭归结得 NIL。

3.7 定义谓词:$R(x)$表示 x 能够阅读;$L(x)$表示 x 有文化;$D(x)$表示 x 是海豚;$I(x)$表示 x 有智能。将前提和结论表示为:$(\forall x)(R(x) \rightarrow L(x))$;$(\forall y)(D(y) \rightarrow \neg L(y))$;$(\exists z)(D(z) \wedge I(z))$;$(\exists w)(I(w) \wedge \neg R(w))$。化为子句集:① $\neg R(x) \vee L(x)$;② $\neg D(y) \vee \neg L(y)$;③ $D(a)$;④ $I(a)$;⑤ $\neg I(w) \vee R(w)$。⑤与④归结得⑥ $R(a)$;⑥与①归结得⑦ $L(a)$;⑦与②归结得⑧ $\neg D(a)$;⑧与③归结得 NIL。

3.8 定义谓词:$S(x,y)$表示 x 储蓄 y;$M(x)$表示 x 是钱;$I(x)$表示 x 是利息;$E(x,y)$表示 x 获得 y。将前提表示为谓词公式:$(\forall x)((\exists y)(S(x,y) \wedge M(y)) \rightarrow (\exists y)(I(y) \wedge E(x,y)))$;化为子句集:① $\neg S(x,y) \vee \neg M(y) \vee I(f(x))$;② $\neg S(x,y) \vee \neg M(y) \vee E(x,f(x))$。将结论表示为谓词公式:$\neg(\exists x)I(x) \rightarrow (\forall x)(\forall y)(M(y) \rightarrow \neg S(x,y))$;化为子句集:③ $\neg I(z)$;④ $S(a,b)$;⑤ $M(b)$。①与④归结得⑥ $\neg M(b) \vee I(f(a))$;⑥与⑤归结得⑦ $I(f(a))$;⑦与③归结得 NIL。

3.9 定义谓词:$U(x,y)$:x 使用 y;$E(u,v)$:u 得到 v;$I(z)$:z 是 Internet;$F(u)$:u 是信息。把已知前提表示成谓词公式:F:$(\forall x)((\exists y)(U(x,y) \wedge I(y)) \rightarrow (\exists u)(F(u) \wedge E(x,u)))$

化为子句集:① $\neg U(x,y) \vee \neg I(y) \vee F(f(x))$;② $\neg U(x,y) \vee \neg I(y) \vee E(x,f(x))$。把待求证的结论表示成谓词公式并否定得$\neg G$:$\neg(\neg(\exists u)F(u) \rightarrow (\forall x)(\forall y)(I(y) \rightarrow \neg U(x,$

$y)))$。化为子句集：③ $\neg F(u)$；④ $I(b)$；⑤ $U(a,b)$。①与④归结得⑥ $\neg U(x,b) \vee F(f(x))$；⑤与⑥归结得⑦ $F(f(a))$；⑦与③归结得⑧ NIL。

3.10 把前提表示为谓词公式：$F_1: \forall x \forall y \forall z(F(x,y) \wedge F(y,z) \rightarrow G(x,z))$；$F_2: F(Lao,Da)$；$F_3: F(Da,Xiao)$。化为子句集：① $\neg F(x,y) \vee \neg F(y,z) \vee G(x,z)$；② $F(Lao,Da)$；③ $F(Da,Xiao)$。

设求证的公式为 G：$\exists x \exists y G(x,y)$；把其否定并与答案谓词析取，化为子句集得④ $\neg G(u,v) \vee ANSWER(u,v)$。①，②归结得⑤ $\neg F(Da,z) \vee G(Lao,z)$；③，⑤归结得⑥ $G(Lao,Xiao)$；④，⑥归结得⑦ $ANSWER(Lao,Xiao)$。所以，老李是小李的祖父。

3.11 定义谓词：$AT(y,x)$：y 在 x 处。把已知前提表示成谓词公式：$F_1: (\forall x)(AT(Zhang,x) \rightarrow AT(Li,x))$；$F_2: AT(Zhang,School)$。把待求解的问题表示成谓词公式，把它否定并与答案谓词析取，得 $G: \neg(\exists x)AT(Li,x) \vee ANSWER(x)$。化为子句集：① $\neg AT(Zhang,x) \vee AT(Li,x)$；② $AT(Zhang,School)$；③ $\neg AT(Li,x) \vee ANSWER(x)$。①与②归结得④ $AT(Li,School)$；④与③归结得⑤ $ANSWER(School)$。由 $ANSWER(School)$ 得知小李在学校。

3.12 定义谓词：$A(x)$：x 是 $ALPINE$ 的一个成员；$B(x)$：x 是滑雪运动员；$C(x)$：x 是登山运动员；$L(x,y)$：x 喜欢 y。

将前提表示成谓词公式：$A(TONY)$，$A(MIKE)$，$A(JOHN)$；$(\forall x)(A(x) \rightarrow [B(x) \wedge \neg C(x)] \vee [\neg B(x) \wedge C(x)])$；$(\forall x)(C(x) \rightarrow \neg L(x,Rain))$；$(\forall x)(\neg L(x,Snow) \rightarrow \neg B(x))$；$(\forall x)(L(TONY,x) \rightarrow \neg L(MIKE,x))$；$(\forall x)(\neg L(TONY,x) \rightarrow L(MIKE,x))$；$L(TONY,Rain)$；$L(TONY,Snow)$。化为子句集：① $A(TONY)$；② $A(MIKE)$；③ $A(JOHN)$；④ $\neg A(x) \vee B(x) \vee \neg B(x)$；⑤ $\neg A(x) \vee B(x) \vee C(x)$；⑥ $\neg A(x) \vee \neg B(x) \vee \neg C(x)$；⑦ $\neg A(x) \vee \neg C(x) \vee C(x)$；⑧ $\neg C(x) \vee \neg L(x,Rain)$；⑨ $L(x,Rain) \vee \neg B(x)$；⑩ $\neg L(TONY,x) \vee \neg L(MIKE,x)$；⑪ $L(TONY,x) \vee L(MIKE,x)$；⑫ $L(TONY,Rain)$；⑬ $L(TONY,Snow)$

将待求解的问题表示成谓词公式，然后否定并与答案谓词析取：$\neg(A(x) \wedge C(x) \wedge \neg B(x)) \vee ANSWER(x)$。化为子句集：⑭ $\neg A(x) \vee \neg C(x) \vee B(x) \vee ANSWER(x)$。⑩ 与⑫ 归结得⑮ $\neg L(MIKE,Rain)$；⑮与⑨归结得⑯ $\neg B(MIKE)$；⑯ 与⑤归结得⑰ $\neg A(MIKE) \vee C(MIKE)$；⑰与②归结得⑱ $C(MIKE)$；⑱与⑭归结得⑲ $\neg A(MIKE) \vee B(MIKE) \vee ANSWER(MIKE)$；⑲与②归结得⑳ $B(MIKE) \vee ANSWER(MIKE)$；⑳与⑯归结得 $ANSWER(MIKE)$。因此，$MIKE$ 是 $ALPINE$ 俱乐部的一个成员，他是一个登山运动员，但不是一个滑雪运动员。

第 4 章习题解答

4.1 $CF(E_2) = 0.6 \times \max\{0, CF(E_1)\} = 0.6 \times \max\{0, 0.5\} = 0.3$

$$CF(E_4) = 0.8 \times \max\{0, CF(E_2 \text{ AND } E_3)\}$$
$$= 0.8 \times \max\{0, \min\{CF(E_2), CF(E_3)\}\} = 0.24$$

$CF_3(H) = 0.7 \times \max\{0, CF(E_4)\} = 0.168$

$CF_4(H) = 0.9 \times \max\{0, CF(E_5)\} = 0.36$

又因为 $CF_3(H) > 0$，$CF_4(H) > 0$，故

$CF(H) = CF_3(H) + CF_4(H) - CF_3(H)CF_4(H) = 0.47$

4.2 $CF_1(H_1)=0.7\times\max\{0,CF(E_1)\}=0.35$

$CF_2(H_1)=0.6\times\max\{0,CF(E_2)\}=0.3$

$CF_3(H_1)=0.4\times\max\{0,CF(E_3)\}=0.2$

$CF_{1,2}(H_1)=CF_1(H_1)+CF_2(H_1)-CF_1(H_1)CF_2(H_1)=0.545$

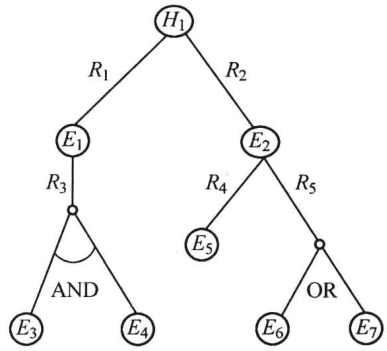
同理 $CF(H_1)=CF_{1,2,3}(H_1)=0.2+0.545-0.545\times0.2=0.636$

则 $CF_4(H_2)=0.2\times\max\{0,CF(H_1\ \text{AND}\ E_4)\}$

$=0.2\times\max\{0,\min\{CF(H_1),CF(E_4)\}=0.1$

$CF(H_2)=CF_4(H_2)+CF_0(H_2)-CF_4(H_2)CF_0(H_2)=0.37$

4.3



$CF(E_3\ \text{AND}\ E_4)=\min\{CF(E_3),CF(E_4)\}=0.8$

所以 $CF(E_1)=0.9\times\max\{0,CF(E_3\ \text{AND}\ E_4)\}=0.72$

$CF(E_6\ \text{OR}\ E_7)=\max\{CF(E_6),CF(E_7)\}=0.5$

所以 $CF_5(E_2)=(-0.3)\times\max\{0,CF(E_6\ \text{OR}\ E_7)\}=-0.15$

$CF_4(E_2)=0.7\times\max\{0,CF(E_5)\}=0.56$

$$CF(E_2)=\frac{CF_5(E_2)+CF_4(E_2)}{1-\min\{|CF_5(E_2)|,|CF_4(E_2)|\}}=\frac{0.41}{0.85}=0.48$$

$CF_1(H_1)=0.8\times\max\{0,CF(E_1)\}=0.576$

$CF_2(H_1)=0.9\times\max\{0,CF(E_2)\}=0.432$

所以，$CF(H_1)=CF_1(H_1)+CF_2(H_1)-CF_1(H_1)\times CF_2(H_1)=0.759$

4.4 因为 $K=\sum\limits_{X\cap Y\neq\varnothing}M_1(X)M_2(Y)$

$=M_1(\{b,c,d\})M_2(\{a,b\})+M_1(\{b,c,d\})M_2(\{a,b,c,d\})$

$+M_1(\{a,b,c,d\})M_2(\{a,b\})+M_1(\{a,b,c,d\})M_2(\{a,b,c,d\})$

$=0.7\times0.6+0.7\times0.4+0.3\times0.6+0.3\times0.4=1$

所以　$M(A)=K^{-1}\times\sum\limits_{X\cap Y=A}M_1(X)M_2(Y)=\sum\limits_{X\cap Y=A}M_1(X)M_2(Y)$

所以　$M(b)=M_1(\{b,c,d\})\times M_2(\{a,b\})=0.7\times0.6=0.42$

$M(b,c,d)=M_1(\{b,c,d\})\times M_2(\{a,b,c,d\})=0.7\times0.4=0.28$

$M(a,b)=M_1(\{a,b,c,d\})\times M_2(\{a,b\})=0.3\times0.6=0.18$

$$M(a,b,c,d)=M_1(\{a,b,c,d\})\times M_2(\{a,b,c,d\})=0.3\times 0.4=0.12$$

M 的其余基本概率分配函数为 0。

4.5

$A\cap B=0.5/x_1+0.65/x_2+0.8/x_3+0.9/x_4+0.7/x_5$

$A\cup B=0.85/x_1+0.7/x_2+0.9/x_3+0.98/x_4+0.77/x_5$

$\neg A=0.15/x_1+0.3/x_2+0.1/x_3+0.1/x_4+0.3/x_5$

4.6

$$\boldsymbol{A}\circ\boldsymbol{B}=\begin{bmatrix}0.7 & 0.4\\ 0.7 & 0.4\\ 0.5 & 0.4\end{bmatrix}$$

4.7

$$\boldsymbol{R}_1\circ\boldsymbol{R}_2=\begin{bmatrix}0.4 & 0.8\\ 0.4 & 0.9\\ 0.7 & 0.5\\ 0.7 & 0.6\end{bmatrix}$$

4.8

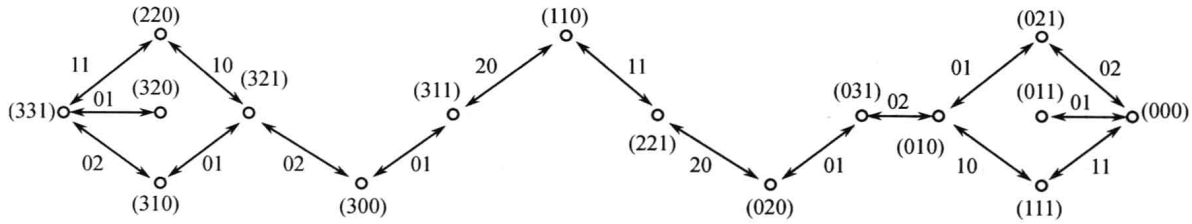
$$\boldsymbol{R}_1\circ\boldsymbol{R}_2=\begin{bmatrix}0.6 & 0.6 & 0.0\\ 0.3 & 0.2 & 0.1\\ 0.0 & 0.5 & 0.1\end{bmatrix};\boldsymbol{R}_1\circ\boldsymbol{R}_3=\begin{bmatrix}1.0 & 0.0 & 0.7\\ 0.3 & 0.2 & 0.3\\ 0.7 & 0.5 & 1.0\end{bmatrix};\boldsymbol{R}_1\circ\boldsymbol{R}_2\circ\boldsymbol{R}_3=\begin{bmatrix}0.6 & 0.6 & 0.6\\ 0.3 & 0.2 & 0.3\\ 0.1 & 0.5 & 0.1\end{bmatrix}$$

4.9

$$\boldsymbol{S}=\begin{bmatrix}0.8 & 0.3 & 0.2\\ 0.3 & 0.8 & 0.7\\ 0.3 & 0.7 & 0.8\end{bmatrix}$$

第 5 章习题解答

5.1 以变量 m 和 c 表示修道士和野人在左岸和船上的实际人数，变量 b 表示船是否在左岸，$b=1$ 表示在，$b=0$ 表示不在。问题状态用三元组 (m,c,b) 表示，则问题求解的任务为：$(3,3,1)\rightarrow(0,0,0)$。在这个问题上，状态空间可能的状态总数为 $4\times4\times2=32$，但由于遵守约束：$m+c\leqslant 2$，$m\geqslant c$，只有 20 个是合法的。例如，$(1,0,1)$，$(1,2,1)$，$(2,3,1)$ 等是不合法的。由于存在不合法的状态，导致某些合法的状态不可达，例如，$(0,0,1)$，$(0,3,1)$，所以这个问题只有 16 个可达的合法状态。渡河问题的操作算子可以定义两类：$L(m,c)$、$R(m,c)$ 分别表示船从左岸划到右岸，和船从右岸划到左岸。由于 m,c 取值的可能组合只有 5 个：10，20，11，01，02，所以总共有 10 个操作算子。可以画出渡河问题的状态空间的有向图。



5.2 该系统的状态用四元数组表示

$S=$(农夫,狐狸,鸡,米)

0:左岸　1:右岸

初始状态:$S_0=(0,0,0,0)$,$S_1=(1,0,1,0)$,$S_2=(0,0,1,0)$,$S_3=(1,1,1,0)$或$(1,0,1,1)$,$S_4=(0,1,0,0)$或$(0,0,0,1)$,$S_5=(1,1,0,1)$,$S_6=(0,1,0,1)$。

目标状态:$S_7=(1,1,1,1)$。允许操作:$F=\{f_1,f_2,f_3,f_4\}$。其中$f_1(x)$:农夫不载物到x岸;$f_2(y)$:农夫载鸡到y岸;$f_3(z)$:农夫载狐狸到z岸;$f_4(p)$:农夫载米到p岸;或$f(u,v)$。

5.3 参见例5.2。

第8章习题解答

8.1 $y=f(w_1x_1+w_2x_2)$,当$x_1=0,x_2=0$时,$y=f(0)=0$;当$x_1=0,x_2=1$时,$y=f(0.4)=0$;当$x_1=1,x_2=0$时,$y=f(0.2)=0$;当$x_1=1,x_2=1$时,$y=f(0.6)=1$。可见,符合与逻辑关系。

8.2 当$x_1=0,x_2=0$时,$x_1^1=f(0)=0,x_2^1=f(0)=1,y=f(1)=0$;当$x_1=1,x_2=1$时,$x_1^1=f(2)=1,x_2^1=f(-2)=0,y=f(1)=0$;当$x_1=0,x_2=1$时,$x_1^1=f(1)=1,x_2^1=f(-1)=1,y=f(2)=1$;当$x_1=1,x_2=0$时,$x_1^1=f(1)=1,x_2^1=f(-1)=1,y=f(2)=1$。可见,符合异或逻辑关系。

8.3 略。

8.4 略。

8.5 与例8.1类似。

8.6 略。

8.7 $E=-(w_{12}v_1v_2+w_{13}v_1v_3+w_{23}v_2v_3)+(\theta_1v_1+\theta_2v_2+\theta_3v_3)=2$

$$E=-(w_{12}v_1v_2+w_{13}v_1v_3+w_{14}v_1v_4+w_{23}v_2v_3+w_{24}v_2v_4+w_{34}v_3v_4)$$
$$+(\theta_1v_1+\theta_2v_2+\theta_3v_3+\theta_4v_4)=1.1$$

8.8 略。

8.9 $u_1(1)=-1\times\left(-\frac{2}{3}\right)+1\times\frac{2}{3}=\frac{4}{3}>0,f_1\left(\frac{4}{3}\right)=1$;$u_2(1)=1\times\left(-\frac{2}{3}\right)+1\times\left(-\frac{2}{3}\right)=-\frac{4}{3}<0,f_2\left(-\frac{4}{3}\right)=-1$;$u_3(1)=-1\times\left(-\frac{2}{3}\right)+1\times\frac{2}{3}=\frac{4}{3}>0,f_3\left(\frac{4}{3}\right)=1$。

可见,状态保持不变,表明进入稳定状态$(1,-1,1)$。

$u_1(1)=1\times\left(-\frac{2}{3}\right)+1\times\left(-\frac{2}{3}\right)=-\frac{4}{3}<0,f_1\left(-\frac{4}{3}\right)=-1$;

$$u_2(1) = -1 \times \left(-\frac{2}{3}\right) + (-1) \times \left(-\frac{2}{3}\right) = \frac{4}{3} < 0, f_2\left(\frac{4}{3}\right) = 1;$$

$$u_3(1) = -1 \times \frac{2}{3} + 1 \times \left(-\frac{2}{3}\right) = -\frac{4}{3} < 0, f_3\left(-\frac{4}{3}\right) = -1。$$

可见,状态保持不变,表明进入稳定状态$(-1,1,-1)$。

8.10 由离散 Hopfield 神经网络的计算能量函数表达式可得 $E(0)=1.0$。

第一轮迭代:$u_1(1)=-3.5<0, v_1(1)=0; u_2(1)=12.4>0, v_2(1)=1; u_3(1)=5.3<0, v_3(1)=1; u_4(1)=0.6>0, v_4(1)=1$;计算得 $E(1)=-14$。

第二轮迭代:$u_1(2)=-3.2<0, v_1(2)=0; u_2(2)=7.8>0, v_2(2)=1; u_3(2)=1.3>0, v_3(2)=1; u_4(2)=2.5>0, v_4(2)=1$;计算得 $E(1)=-14$。

继续迭代下去,总有 $E=-14$,网络稳定在 $V=\{0,1,1,1\}$,因此是一个稳定状态。

参考文献

[1] 王万良. 人工智能及其应用[M]. 2 版. 北京:高等教育出版社,2008.

[2] 王万良,吴启迪. 生产调度智能算法及其应用[M]. 北京: 科学出版社, 2007.

[3] 袁南儿,王万良,苏宏业. 计算机新型控制策略及其应用[M]. 北京:清华大学出版社, 1998.

[4] 涂序彦. 人工智能及其应用[M]. 北京:电子工业出版社,1988.

[5] 蔡自兴,徐光佑. 人工智能及其应用[M]. 3 版. 北京:清华大学出版社, 2003.

[6] 王永庆. 人工智能原理与方法[M]. 西安:西安交通大学出版社, 1998.

[7] 王士同. 人工智能教程[M]. 北京:电子工业出版社,2001.

[8] 高济,朱淼良,何钦铭. 人工智能基础[M]. 北京:高等教育出版社,2002.

[9] 廉师友. 人工智能技术导论[M]. 2 版. 西安:西安电子科技大学出版社, 2002.

[10] 马少平,朱小燕. 人工智能[M]. 北京:清华大学出版社, 2004.

[11] 王文杰,叶世伟. 人工智能原理与应用[M]. 北京:人民邮电出版社,2004.

[12] 贲可荣, 张彦铎. 人工智能[M]. 北京: 清华大学出版社, 2006.

[13] 朱福喜,汤怡群,傅建明. 人工智能原理[M]. 武汉:武汉大学出版社,2002.

[14] 张仰森,黄改娟. 人工智能实用教程[M]. 北京:北京希望电子出版社, 2002.

[15] 涂序彦,韩力群. 人工智能:回顾与展望[M]. 北京: 科学出版社, 2006.

[16] 王万森. 人工智能原理及其应用[M]. 2 版. 北京: 电子工业出版社, 2007.

[17] 刘峡壁. 人工智能导论:方法与系统[M]. 北京:国防工业出版社,2008.

[18] 李鸣华. 人工智能及其应用[M]. 北京:科学出版社,2008.

[19] Luger G F. Artificial Intelligence[M]. 4th ed. [S. l.]: Pearson Education Limited, 2002.

[20] Buckland M. 游戏编程中的人工智能技术[M]. 吴祖增等译. 北京: 清华大学出版社, 2006.

[21] Funge J D. 人工智能在计算机游戏和动画中的应用[M]. 班晓娟等译. 北京: 清华大学出版社,2004.

[22] Funge J D. 游戏人工智能——计算机游戏中的人工智能[M]. 李睿凡等译. 北京: 北京邮电大学出版社,2007.

[23] Bourg D M, Seemann G. 游戏开发中的人工智能[M]. 南京: 东南大学出版社,2006.

[24] 郭燕慧. 游戏人工智能——计算机游戏中的人工智能[M]. 北京: 北京邮电大学出版社, 2007.

郑 重 声 明

人工智能导论（第3版）

Introduction of Artificial Intelligence *Third Edition*

● 内容实用　由于人工智能正处于迅速发展阶段，教学内容繁多。本书在内容上精选一些基本理论与实用方法，引导读者学习应用理论解决工程问题的方法。

● 精选习题　本书精选了例题和习题，并在书后给出了部分习题答案，对有些难题还给出了比较详细的解答，帮助读者掌握教材的主要内容。

● 编排醒目　每章在开始位置有导读，在结束位置有总结，使读者了解学习该章目的以及该章主要内容。同时本书采用双色印刷，将重要的概念、公式、定理与方法用明显的颜色标注出来，便于读者掌握重点。

● 资源丰富　作者为使用本书教师提供电子教案、习题详细解答、实验指导书、全部讲课录像等教学资源。

ISBN 978-7-04-031503-5

9 787040 315035 >

定价 26.10元