第十七章 智能的模拟与工程化理论（一）

智能时代的人类智能进化将是一种全新意义上的进化---文化技术的进化，即一种可以按照人的意志进行设计和控制的进化。这种进化的成就之一即是“智能的工程化”。我们相信，人工智能将会逐步向人类智能逼近，机器智能与人类智能的区别将越来越多地来自于人类智能的自我设计；也相信，人类社会在数字化、信息化之后将进入智能化的时代；但是，我们更认为，智能的工程化将是一项复杂的技术革命和社会工程，其发展将受制于众多学科和相关技术的发展，智能的模拟和工程化的发展历程必定是一个漫长而曲折的历程。

智能工程化的研究和发展需要更多的理论和实践。在理论方面要求要有新的思维和行为模型；在实践方面要求要有构造与实现智能系统的理论和技术。它需要有更新的理论框架，以克服目前宏观与微观隔离、全局与局部割裂、理论和实际脱节的局面；它也需要有更成熟的技术方法，以研究和开发出通用而有效的可信赖智能系统；它需要有更好的技术集成，以集成各种智能技术和其它信息处理技术，以综合集成脑和神经科学、认知科学、心理科学、社会科学、系统科学和哲学的成果等；也需要有更多的工程系统，以创造出更多种类的机器人、人造智能体和超智能体系统。

本章将研究人类智能的“工程”拓展与人工智能系统的“构建”理论。

人类在生存和发展过程中，一直在寻求克服自身自然条件的限制，增强自己能力的方法，除集体协作[**组织起来**]之外，在认识各类客观事物的发展变化规律的基础上，发明和利用各种**辅助工具**，也是一条重要途径。各类科学知识都是人对各类客观事物的认识，各种辅助工具本质上都是人类对自己各种能力的延伸和拓展。比如，汽车是人类对自己“行走”能力的延伸，显微镜是人类对自己“视力”的延伸；畜力和动力机器是人类对自己“体力”的延伸和拓展；等等。

在人类所发明的所有工具中，**用于拓展和延伸人类“思维”能力的工具**无疑是最重要的一类。首先是**语言**的产生。作为思维的工具，它可使人类的思维更加清晰有序；而作为交流工具，它使人类可以更好地协同工作，交流和传授经验和信息。其次，是**文字**的发明。作为一类符号系统，它可用于对信息的记载，使信息和知识得以长期保存；作为一类交流工具，它可使人类知识和信息的传递和交流摆脱面对面的局限性，使信息和经验的交流得以在更广阔的时间和空间范围内进行。**计算机**的出现，更开创了人类思维解放的新篇章。这一被称为“**电脑**”的工具，可以帮助人类进行各类“计算”和信息处理，也为帮助人类“思维”创造了“系统”的条件，因为思维即“计算”，即“信息处理”。

本章，我们将从“**工程**”的角度，系统介绍将人类智能进行“**功能模拟和工程化**”的基本原理和方法，以及智能化信息处理系统构建的基本方法和技术。由于人类智能主要体现在感知、学习、思维、协作、问题解决与行为控制等方面，因此，本章的解说和研究也将主要包括：**基于知识的问题求解系统的构建；基于搜索、归约和逻辑推理等问题求解的基本方法；机器学习和知识发现；机器感知与自然语言理解；行为控制与机器人技术；群集智能及智能体技术；等等**。本章内容部分参考和选用了高济[1701]等人的研究和论述。

需要说明的是，由于本章内容主要完成于十多年前，部分内容可能有些过时；为了保留原有体

系，个别过时内容我们并未完全删除；而新技术的发展，由于发展太快，我们只能逐步补充。

17.1 人类智能“工程化”研究概述

17.1.1 人类智能“工程化”研究的主要内容

人类对智能的研究包含着不可分割的两个方面：一是对人类智能“机理”的研究和探索，希望能彻底了解人类智能的生成机制与运行机理；二是在分析人类智能特点的基础上设法将智能“赋予”（智能）机器或特定信息（处理）系统，以实现某种“机器智能”或“人工智能（Artificial Intelligence)”，或者说使智能“工程化”和“机械化”。前者主要汇集了一大批自然科学家和社会科学家为之奋斗，后者则吸引了大批工程技术专家为之努力。

在本书中，我们把将人类智能赋予（智能）机器或信息（处理）系统使之可以像人一样处理信息的研究称之为智能的“工程化”研究。作为一类学科，它包括了人工智能、神经网络和专家系统等相关“智能技术”。它们是二十世纪下半叶兴起的一类新型学科，是一类正在发展中的综合性前沿学科，是一类试图寻找替代和拓展人脑功能的技术和工具的综合性学科，是一类涉及数学、计算机科学、信息科学、心理科学、神经科学、控制理论、语言学、哲学等多个学科的交叉和边缘性学科。其研究目标，是希望用“人工”的方法和技术，模仿、延伸、扩展人类的智能，实现某种可代替人类进行思维活动的“机器智能”。目前，实现智能“工程化”的主要基础工具是计算机（高速信息处理系统，包括芯片）和以计算机为控制核心的“机器人”（智能化工程系统），即主要是利用智能计算机（智能化的高速信息处理系统或工程系统）来模拟或实现“类人”的智能。其研究包括：（1）**人类智能行为的“计算模型”研究，使辨识、思维和学习等智能行为可转化为某种可能的“计算”**。由于人类智能的核心是知识和思维，因此，关于人类智能行为的计算模型的研究中有很大一部分是研究知识的形式化表示和问题求解的思维方法的。知识的形式化表示主要包括定义符号结构和推理机制。现在提出的知识表示方法已有状态空间法、问题归约法、谓词逻辑法、**结构化表示法**（语义网络、框架）、剧本和过程表示方法、**知识图谱**等。思维则常表现为推理、搜索、辨识与联想等。目前，常用的搜索算法已有盲目搜索、启发式搜索和消解原理等；常用的推理求解技术已有规则演绎推理技术、归纳推理技术、常识推理技术、概率统计推理技术、不确定性推理（模糊推理）技术、思维链技术和非单调推理技术等。（2）**具有感知、思维、学习、辨识和决策等智能功能的信息处理系统或“计算”系统的研制**；研究如何构建一个智能化的信息处理系统来模拟人脑所从事的思维活动，去解决过去必须靠人类的智能才能处理好的各种问题，诸如辨识、决策、自然语言理解、学习、推理和问题求解等智能活动；也即用“人工”的方法和技术，研制智能机器或智能系统来模仿、延伸和扩展人的智能，以实现“机器智能”。智能的工程化研究是与智能行为的“**自动化**”有关的一类学科，是一类试图通过“**计算过程**”来理解和模仿智能行为的学科，是一类研究探索人的感觉和思维过程的规律并设法设计出各种类似人（或人脑）的某些智能行为的“自动机”的学科。其最终目标，是将人类从繁琐的脑力劳动中解放出来，最大限度地让“机器”代替人类的脑力和体力劳动。目前，人们所说的智能技术，就是研究如何用人工的方法与技术，来模拟、延伸和扩展人类的智能，研究如何让（智能）计算机来做现阶段只有人才能做得好的事情，实现某些机器思维或脑力劳动的自动化，使**人工系统**表现出**智能行为**，使之成为“像人一样可理性地思考的系统”或“像人一样可理性地行动的系统”的工程技术学科，也即是将智能“工程化”的重要学科。

尽管人类智能的“工程化”研究目前还处于“技艺”状态，但是，它却具有无比强大的生命力

和广阔的前景。它是工程和信息科学技术的前沿，其研究、应用和发展在一定程度上决定着工程和信息科学技术的发展方向。我们赞同这样的观点：

（1）**智能的“工程化”研究是研究如何开发出一类智能机器或系统，来模拟人类智能活动，延伸人类智能的学科；是希望在人类智能的不同结构和层次上，设法让一个信息处理系统能模拟并局部地（或全面地）拓展人类智能的结构和功能的学科。而智能系统则是能够在各类环境中自主地或交互地执行各种拟人任务的系统。**

（2）**智能的“工程化”研究不仅具有模拟功能，而且具有拓展人类智能的特性。**比如，其极高的计算速度和极为准确的记忆能力等，已远远超过人类。当然，我们也确信，人类所研制出的“智能系统”在整体上可以无限接近并允许其部分性能优于人类，但它目前还不能全面超越人类的智能；智能“工程系统”也许永远都是人类生活和工作中的“工具”。（**但今后会如何呢？就我们现在的认知而言，我们还无法完全预测，因为科学本身并无止境。**）

（3）**智能的“工程化”研究是多方面的**。目前的各种研究流派主要是在人类智能的不同层次和不同侧面上进行的研究，相信以后会逐步融合。

（4）**智能“工程化”研究的最终目标是实现人工智能或智能机器**，即由人工系统来实现本来由人类的智能才能实现的功能，可如费根鲍姆（Feigenbaum）所说：**只告诉机器做什么，而不告诉它怎么做，机器就能去完成指定的任务**。而其近期目标，则主要在于研究用机器来模仿和执行人脑的某些功能（有人预计很快就会达到全面模拟阶段），并研发相关理论和技术，以完成需要人类运用智能才能去完成的某些任务。

（5）用人工系统来模拟和实现人脑的某些智能化信息处理功能，在当前阶段，主要是从信息的接收、传递、储存和处理等方面进行的功能性模拟，还不是对人脑的分子生物学水平或细胞生物学水平上的真正的生物模拟（今后也许会逐步转向生物模拟）。从工程的观点看，作为一门科学技术，智能“工程化”研究的任务主要是用“机器”或“人工系统”去部分（或全面）代替人的脑力劳动。模拟研究的要点则在于学习人类怎样解决问题、发现模式、学习知识、做出决定甚至创新等。其途径主要是将人脑在进行思维或信息处理时的方法和所采取的策略、技巧或步骤等赋予“智能计算机”，让“智能机器”据此去探索解决问题的方法，使“智能计算机”（电脑）在某些方面成为“会思考的机器”。而这些方面则主要包括感知(perception)、推理(Reasoning)、学习(learning)、交互交流(communicating)和在复杂环境下的决策和动作行为(acting)控制等。

17.1.2 人类智能拓展与工程化研究的重要意义

为什么要进行智能的“工程化”研究？若简单地说，① 研究是当前信息化社会进一步发展的迫切需求；② 研究是自动化技术发展的必然趋势；③ 研究会使当前的“电脑”或“机器人”更好用，更有用；④ 研究也有助于探索人类自身的奥秘；⑤ 研究可以从根本上“扩展”和“延伸”人类智能，甚至实现“机器智能”。

在谈到智能工程化研究的重大意义时，人们往往都认同这样一个类比：**工业革命用机器替代了人类繁重的体力劳动，解放了人类的四肢，极大地促进了人类的社会进步和经济发展；以计算机和人工智能为代表的“智能革命”，解放的将是人类的大脑；其研究可实现人类脑力劳动的机械化、自动化，进而，也为人类社会生产力的发展奠定了坚实的技术基础。**

制造和使用一种能够代替人类大脑从事复杂脑力劳动的机器，是人类长期以来的一个愿望。计算机的出现，揭开了用机器替代人脑从事脑力劳动的历史序幕，尽管现在多数的计算机（电脑）还

只能替人们做一些“初级”的信息处理的工作，然而，我们研究计算机和人工智能，特别是智能机器，在本质上还是要继续用它们模拟人脑的行为和功能，使它们能成为人脑功能的延伸。

我们知道，人的智能是“人认识客观事物并运用知识解决实际问题的功能和能力，是主体有目的的行为、合理的思维、以及有效的适应环境的综合性能力，可体现在其反映客观事物深刻、正确、完全的程度上，或应用知识解决问题的速度和质量上，往往可以通过观察、记忆、想象、思考、学习、判断、决策等表现出来”。而智能机器的研究，就初衷而言，是人类运用其智能（智慧）为自己大脑运行效率的增强和思维活动质量的提高而希望创造的一个“工具”。它的性质，就如同人类为了减轻自己的体力劳动强度而创造出来的动力机械，为了扩大自己的活动范围和速度而创造出来的交通工具一样，（目前）同样都是人类智慧的产物（在都是人类智慧的创造物这一点上，智能计算机和飞机在本质上并没有任何区别），同样都是人类某一器官的功能的扩展和延伸（飞机是人腿运动功能的扩展和延伸，而智能计算机则是人脑思维功能的扩展和延伸---虽然人脑具有驾驭人腿的地位，智能电脑可具有驾驶飞机的本领，但是人脑在研制飞机时所花费的智力劳动及智力活动方式和在研制智能计算机时的情况可以说并无本质上的差别，研制智能计算机的劳动者和研制飞机的劳动者之间在智能水平的要求上也绝无高下之分）。

正是因为智能机器是人类智能和大脑功能的一种延伸，因此智能机器的作用和功能也就仅仅体现在它在本质上（在目前）只是扩大了人脑的运行效率和活动质量，而不是体现在人类大脑本质功能的升级或智能形态的提高方面（就目前的技术和我们的认知而言，可以认为，不管计算机在计算能力上有多强，在运行速度上有多快，目前，都不能改变人类智能的形态或超越大脑的功能模式去工作，甚至在模拟大脑运行方式上也可能毫无改善。一般地讲，目前，计算机可以大显身手的地方大都是要求遵循一定模式和逻辑规则去做的工作，如大规模的数据处理、严格的逻辑推理以及解决某些技能类型的问题，在这些基本上是“循规蹈矩”的工作方面，人不如计算机完成得那么快、那么好，这是毫无疑义的），更重要的是，它还不能体现在代替人类的生命活动上（**以后是否会出现硅基生命，我们还无法完全确定，此处暂不讨论**）。人类智能（目前）是有一定限度的，人不能超越自己可感知和可思维的限度去触摸不可认知的世界；可以设想，人类智能的限度就是人工智能（目前）的限度；就目前人类的认知而言，应该是这样。如此，我们也许会深刻地感受到，“智能机器”如果只是一种单纯的“机器”和“工具”（虽然它可以是能够支配自身活动的机器---如，有的智能机器人可以自如上下楼梯、能够推着小车自由行走、从事服务工作；有的医疗机器人能够做精密的心脏手术；有的智能系统可以精准识别人的表情；有的智能系统能够写出小说和歌曲等），那么，（目前的）智能机器的智能和人类固有的智能之间就有着根本的差别；也就是说，他们只是形式上和效果上的相似，但具有本质意义的区别和差异。因此，我们也就有必要进一步考虑智能机器的运行方式和人类智能的运行方式以及人工智能的结构要素和人类智能的结构要素之间的差异和区别的问题了。对此，我们会在下面作进一步的解析。

17.1.3 智能工程化研究需要解决的问题

研究智能的“工程化”有两个问题必须解决。一是，机器[计算机]可不可以模拟和替代人类智能或具有某种“类人智能”；二是，如何去构建一个“工程系统”使其具有某种“类人智能”。

17.1.3.1 “机器”可否模仿人类的智能行为从而具有“类人智能”？

对于“机器”，目前主要是“智能计算机”，究竟能否具有类人智能的问题，历来存在着两种不同的观点。一种观点认为，既然智能机器是对人脑的模拟，人工智能是对人类智能的模拟，智能机

器当然能够具有智能，否则人类就不会去研制它了。另一种观点则认为，人工智能只是对人类智能的部分功能的模拟，现有智能机器并不具备人脑思维的全部功能，二者本质上是不同的，不能因为智能机器可具有某些与人类思维功能相类似的功能，就说智能机器已经具有智能。

我们认为，要回答智能机器可不可以具有智能的问题，首先要搞清三个问题：第一，究竟什么是智能；第二，我们能否把智能赋予智能机器；第三，我们如何才能把智能赋予智能机器。

反对“计算机”可以具有智能的人认为：

（1）（人类）智能是人类特有的能力，是人类区别于其它动物的根本标志。人类因为有了智能，才能认识世界和改造世界。人类智能的表现形式十分广泛，如通过感官识别文字、图像、声音、语言等外部信息的能力，通过思维对信息进行分析、判断、演绎、归纳、推理和决策的能力，通过学习日益丰富自身知识和技能的能力，以及对变化多端的外界环境做出恰当反应的自适应能力等。而解决这类问题需要经验的积累并通过思维的作用而形成新的经验；也就是说，**它应具有自动学习、经验积累和恰当应用知识的能力**。智能的基础是思维，它不仅体现于思维的结果，而且体现于思维的过程。**思维过程具有自发性、自主性、自我性的特点**。**自发性**是指一个智能正常的人在他清醒的时候，他的思维是无时不在的，他的所有的言行都是受他的思维支配的；**自主性**是指人的思维是受其本人支配的，而不为外部的人和物所左右的，即使人受外界影响会改变自己的主意，但做出这种改变的思维也发自他自己，而不是别的人和物；**自我性**是指人对自己的思维是可了解和把握的，即自己思考了没有，思考得出的结论是什么，他是明白和确定的。而现有的电脑（智能机器）尚不能具备人类智能的一些本质特征。比如，它还不具备按自己本身的意愿积累经验即自主学习的能力；它也不具备自主优化、扩展和改变已有程序和创建新的运行程序的能力，即它还不具有**主观能动性**。根据人类智能的特点，显然可以得出结论——现有电脑，虽然具有一定的智能化信息处理能力,但尚不具备人类智能的本质特点。

（2）为了证明计算机能否具有智能，图灵曾提出了一个“图灵测试”标准。“图灵测试”是说，让人和机器同时回答问题，若我们已区分不出答案是人还是机器做出的，就说明这个机器具有智能了。这种解说方法本身就有问题。“图灵测试”只看处理问题的结果，而没有看处理问题的过程，因而并不能证明计算机(智能机器)是否具有人类的智能。对这个问题的最有力的反驳莫过于赛尔(John R.Searle)所提出的“中文屋”模型。“中文屋”模型是一个精心设计的“黑屋”，它有一个IN口和一个OUT口，屋内存有一套操作规程（一本操作者能读懂的用于指导他如何操作“中文符号”的手册，手册将详细说明当他收到什么样子的“中文符号”时，就从窗口递出另一个什么样子的“中文符号”。手册将不谈符号的意义，而只谈及符号的形式）。有一个不懂中文的人坐在屋内，从IN口接受给他送进的一些“中文字符”。懂中文的人当然知道那是一个问题，可是他却不知道那是一句表达问题的句子，他只知道去查手册，然后依照手册指示，从窗口送出另一组他完全不知其意义的“中文符号”。懂中文的人当然知道那就是一句相应回答了，但他只是按照操作规程将“输入字符”转换成了另一些“字符”，然后将这些新的“字符”从OUT口送出去而已。如果**程序设计师**[即**手册设计者**]很能干，写出了一本很周全、很理想的手册，房间里的**操作者**也能准确地按手册操作，当这个操作者正确地执行了这些操作规程后，他所送出的结果与屋外人的期望应该是完全一致的。此时，屋外的人会认为他懂中文，而事实上他对中文是“一窍不通”。毫无疑问，“中文屋”模型与计算机执行程序有惊人的相似之处，它都有以下三个步骤：① 输入，字符被送入房间；② 处理，按照操作规程，将输入的中文字符转换为另一些字符；③ 输出，新的字符被送出房间。计算机执行程序时就像中文屋里

的人按着规程执行操作一样，它并不涉及对程序本身的理解，它只是按照人预先设置好的程序处理了一些符号，并得出了正确的结果。但是计算机并不知道这些符号的意义是什么，对这个结果是怎样得出的也一无所知（深知），即计算机程序并不构成真正的“思维”。

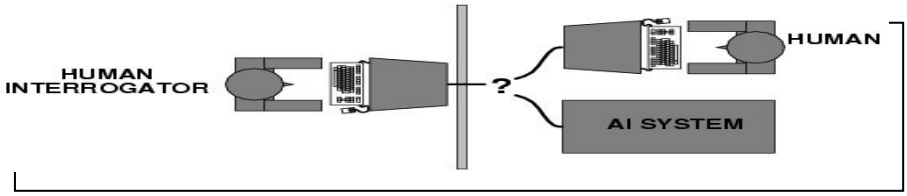

图 17.1.1 图灵测试

在塞尔的实验中，计算机是可以通过图灵测试的。因为计算机就是一个运行特定程序的系统，条件仅在于它的运行程序能被形式化地表示出来，因为计算机所能进行的都是纯形式的操作。但是，塞尔认为，即使计算机通过了图灵测验，在外显行为上和人的心理行为毫无两样，计算机仍不是在做真正的思考，或进一步说，计算机根本不能拥有与人相似的智能。**符号操作只需语法，不需要语义**。所以，计算机（加上程序）只有语法而没有语义。**没有语义就谈不上真正的理解，没有理解也就谈不上真正的智能**。

图 17.1.2　赛尔的 “中文屋”模型

（3）电脑不具有人的意识，不能做有意识的认识，甚至连最原始的生物的本能的刺激-感应能力也不具备。电脑从其物质发展水平来说，还达不到生物运动这一物质发展的高级形态。通常人们所说的“机器思维”，只不过是人类借助电脑来模拟和复制人脑的某些功能而已。人工智能只能模拟人的某些自然属性，难以模拟人的社会属性。但人的思维能力，不仅是人脑的生理机能，更是长期社会实践的产物。随着科学技术的发展，未来人们可能会通过生物工程制造出人造大脑或“生物计算机”，从这个意义上说，机器是可以模拟人的自然属性的。但从总体上看，因为电脑不是生活在社会之中，没有人那样长期的社会实践，不具备人的社会属性，因而也就不可能具有真正意义上的思维。因此，具有“机器思维”的机器智能与具有“人类思维”的人类智能，两者之间有本质的区别：

① 二者的物质载体不同。人类智能的物质载体是人的大脑，人工智能的物质载体则是“电脑”这一人脑的“仿造物”。

② 二者的活动规律不同。人脑的活动，是按照高等生物的高级神经活动规律进行的；电脑则是按照机械的、物理的和电子的活动规律进行的。

③ 人类认识世界和改造世界的活动是有目的的、能动的，在与外部环境的物质、能量和信息交换过程中，能够根据环境的变化不断调整自身，具有适应性。而人工智能则是无自主意识、无自主目的的，没有主观能动性和适应性，只能按照人为它制定的程序运行，机械地模拟人的智能行为，

却毫不理解这些行为，更不会主动提出新问题、研究新问题或解决新问题等。

④ 人类智能或人类的认识能力，只是人类意识的一部分。人是社会的人，人的认识的产生和形成不只是人的认识能力所致，还包括情感、意志、人格等因素的综合作用。人工智能则主要是对人类认识能力的一部分——理性思维的模拟，不具备其他因素。

（4）无论电脑的功能有多么强大，用途有多么广泛，它也不过是个具有超级计算能力的机器。因为严格地讲，电脑只能执行特定的指令，而人脑则是处理所有感受到的信息。所谓“特定的指令”是指电脑程序可接受的或可执行的外部输入。显然执行指令与处理信息有着本质的不同。我们并不否认电脑具有（智能化）处理信息的能力，但电脑处理信息与人脑处理信息是不同的过程。处理信息的过程本质上都是在执行外部的或给定程序中的指令。但执行指令有两种情况：一种是整个过程完全程序化，这本质上并不需要智能。现在普通电脑的工作就是这种情况。另一种是在执行过程中需要探索、发现、套用或制定程序或规则。例如，学生按老师的要求做作业，程序员按老板指令编写电脑程序等，这是需要智能的。现在，电脑的软硬件都不能自发地进化而成，电脑程序是人根据对自然规律和社会经验的归纳总结由人编制的。而人们在工作、生活和其他社会活动中不断面临大量新的情况和需要探索解决的新问题，无法完全使用现有程序来解决或不知道该用哪个程序来解决，处理这些问题才真正需要智能。这样就产生了一个令人颇难理解的结论：**凡电脑能解决的问题都不再需要人类智能发挥作用，而电脑不能解决的问题却需要人类智能来处理。**然而事实目前的确如此！这是由人类智能的本质所决定的。

（5）人类已有的智能是否可以全部形式化也是个问题。现在的计算机信息处理系统（基本上）是一个形式系统，任何一个问题要在计算机上处理必须首先进行形式化，任何不能形式化的问题，计算机根本无法处理，形式化的要求应是现在计算机处理问题的第一要求。形式化的抽象就是语言化。由于求解问题的过程就是人类思考的过程，因此，这个问题也可转化为：**人类思维是否可以全部用语言来表达？**思维科学的研究表明，抽象思维靠语言，它可以形式化；而形象思维和灵感思维不完全靠语言，因而也不能（完全）形式化。计算机在数值计算、定理证明等方面也许可以表现出优秀的性能，但现在的计算机尚不具备（完美的）形象思维和灵感思维的能力。另外，怎样实现从非形式化领域向形式化领域的转变？如果由计算机来完成这一转变，就得把这个转变形式化，那么转变的起点又在哪里？这就造成了一种回归现象。要避免这种回归，就必须假设有一种包罗万象的先验的形式化系统，然而形式化方法属于人类抽象思维范畴，先验的形式化系统是不存在的；所以，（按照现有的技术水平），人类的智能永远不可能全部被形式化。

而支持计算机（智能机器）可以具有智能的人认为：

（1）**认知即计算。**认知的计算主义理论强调，世界上所有的认知或信息处理功能，本质上都是一些形形色色的“计算”过程。我们头脑中的思维和智能，也是按照某种“计算”来理解事物的。人类智能的本质就是图灵机的“计算”，即人类智能行为在本质上跟其物质载体---人类大脑以及与大脑相互作用的物理环境没有必然的联系，只有可用图灵机来描述的、可执行的、基于抽象离散符号的操作，才是智能活动的本质。如果真是这样，那么，人工智能的最终目标就一定会实现。但问题是，计算是否真是人类的认知和智能活动的主要乃至全部的内容？支持计算机（智能机器）可以具有智能的人认为，简单的计算主义也许对认知和智能的认识是不全面的，它无法回答有关智能的所有问题；但计算主义毕竟在最抽象的层次上解释了认知和智能的本质，只不过它还需要进一步的补充----一些算法和实现层次上的补充。因为认知和智能的问题，不可能仅在最抽象的计算层次上得

到全面的解答,对认知和智能的认识还必须从算法和实现层次上给予解释。任何人类认知和智能，都是具体的和现实的，是在进化过程中为了解决问题所形成的具有适应性的本能。抽象层次上的计算是没有时间概念、进化概念和效率概念的；而现实中的认知和智能却是与时间、进化和效率相关联的。

也有人以计算机的计算是串行计算而人脑的计算是并行计算为由来反对认知的计算理论，这显然是靠不住的。也许人脑的计算是一种高度并行的计算,并且不同于序列串行式的计算机的计算，但是，预计将来会出现的神经网络计算机、DNA 计算机和量子计算机都会是高度并行的。人脑计算方式不同于当今的计算机，并不能否认人脑在深层次上也是计算的事实。串行与并行计算的问题或区别只是一个算法层次上的问题，而不是智能的深层特性，它只涉及到计算是如何进行或怎样进行的。同一计算可以由多种不同的算法来实现。在计算层次上,人脑的计算与人脑这个硬件是由什么—碳或硅—构成的确实无关。正是基于这个原因，图灵认为，可以将人的大脑看作是一台离散态的机器，冯•诺伊曼则设想了一台巨大的细胞自动机，它遵循着一定的运行规则，并证明，如果自我繁殖是生命的本质特征，那么这个特征完全可以由细胞自动机获得。当然，从现实的算法层次上讲,也许计算不再是与硬件的构成无关了，但这只涉及到一个算法或计算的效率问题。

（2）**智能模拟是功能模拟**。从电脑和人脑的比较来看，人们把计算机称之为电脑是因为人们已经认为计算机能做的事情与我们的大脑具有某种相似性。电脑和人脑的最基本行为都是对从外部输入或接收的信息和指令做出反应。若按功能分类，电脑可能会有三种类型：一是**工具型**，它可从事包括计算、信息处理、自动控制、办公事务管理等工作；二是**智能型**（**理性思维型**），它具有模式识别、知识获取、自动纠错、分析和归纳推理等能力，即具有适应环境和自我优化的能力等；三是**情感型**（**类人或类脑型**），它的输入过程可模仿人的感觉方式，它的输出过程也可模仿人的情绪反应。目前，电脑的工具功能（**纯计算的智能**）已经达到了非常高的水平，而且还在不断的高速发展和完善之中。显然它在这方面已远远超过了人脑或人的能力，在这一点上，人脑只有自愧不如。下一步，我们可以让电脑实现人的特定物理感觉方式，包括视觉、听觉、嗅觉、味觉、触觉，而且让其识别范围和精度大大超过人的自然能力也不会有问题（即，**可实现感知智能**）。人们现在关注的主要是电脑如何向智能型方向发展（即**向认知智能方向发展**）。不可否认，有一段时间，电脑在智能化方面的进展还远远比不上其在硬件方面的进步速度，因为体现智能的最重要标志之一就是系统的学习和自动纠错功能，换句话说，就是能够吸取经验教训，积极适应环境和不断完善自己的能力。对于电脑来说，这就需要它能够自发地更改其运行程序和软件系统。但问题是，现在的电脑经常不知道它的运行行为是否正确以及如何纠正，因此，它的工作还离不开人-机的交互。但支持电脑可以具有智能的人坚信，随着计算技术的飞速发展，在不远的将来，电脑可以达到人脑的学习和自动纠错水平。由此，**现在电脑在工作过程中依赖人-机交互的程度，也就直接反映了它的智能化程度**。**电脑的智能化水平，也就主要决定于其软件系统在多大程度上可以自主处理问题的水平**。智能电脑也可称为是有经验积累和有学习能力的电脑，它应能解决所有可以建立数学模型的问题甚至一些难以建立数学模型的问题。在未来的一段时期内，智能电脑的一个最重要的突破也许是自我经验积累、自动软件程序开发或者说自我运行程序开发，这虽然还不是完全的智能模拟，但这比生物克隆的意义还要大，因为这几乎就是“**智力克隆**”。

这里就有一个值得深入探讨的问题，那就是，**智力能否复制自己或被复制**。人类个体的智能无疑是靠教育和经验积累形成的。人与人的智能和性格尽管各有不同，从而形成了我们丰富多彩的社

会生活，但教育本身就具有知识或能力复制的意义，尽管只是部分意义上的，但这至少说明，**思想（思维能力）应该是可以被“复制”的**。

情感型电脑对硬件和软件都有极为特殊的要求。这方面的研究目前还仅仅是初步的。这种电脑的应用领域应该是（类人）智能机器人的开发。以前，有些所谓的本类产品，只是有一个友好的人机界面，或是用电脑荧屏显示不同的表情面孔，或是发出不同口气的言语以表达不同的“感情”，而且这种表达在很大程度上是单向的，它充其量不过是一个简单的、会取悦主人的、没有脾气的“宠物”，或是一个可模仿心理医生解答人类的某些心理问题的心理咨询机而已。**但是，ChatGPT 等大（语音）模型的成功开发，无疑使情况发生了巨大变化，使通用智能和类人智能的实现有了可能**。当然，由于情感与智能不同，它不仅与人的个性相关，而且在相当大的程度上是人的社会化的产物，它是人与人之间交流的基础；通过它，人们会产生共鸣和认同，以实现生理和心理上的平衡。情感交流也是个复杂的动态信息交流过程，受时间、地点、环境、人物对象和个人经历的影响；更需要有表情、语言或动作的配合。因此，要实现有情感的类人智能可能会更难一些。

有人认为，欲望、情感和意志是具有主体意识的人类个体所专有的，一旦它们脱离人类个体就不存在或者说变成“假”的了。情感只有是“真”的时才能起作用，不可想象一台机器会自发地产生那些根本不属于它的特性。情感是不能“制造”的，模拟永远是“假”的。具有人类情感的电脑，就像永动机一样，永远不会实现，除非它具有独立意志。对此，我们不敢妄下结论，但情感型类人智能很难实现却是不争的事实。我们认为，现在的电脑在智能和情感方面确实不如人脑，但电脑在这两个方面向人脑靠拢却是可以的，因为智能机的运行方式和人类智能的运行方式以及人工智能的结构要素和人类智能的结构要素可以不同。我们知道，人工智能理论的研究方向和人工智能机对人类智能的模拟主要是落脚在三个方面：**计算、理解和模拟**。应该说，用计算技术模拟人类智能的某些方面，的确是取得了很大的成果，比如在辅助设计、智能识别、智能控制等方面，都向智能化方向迈进了一大步；问题主要是在**理解**方面。从这里我们或许可以看到，人类智能和人工智能或许会（依照各自的优势）沿着不同的路径去运行和发展，两者之间应该允许有所不同。

（3）**人工智能和人类智能具有互补性**。主张研究和开发人工智能的人认为，人类智能的局限性正是人工智能的优越性所在，人工智能的局限性也正是人类智能的优越性，“**人在‘质’的思考方面能胜过机器，而机器则可在‘量’的方面胜过人**”，二者是可以互补互动的。人类发明计算机和研究人工智能的动因，正是基于对人脑的一些局限性的认识，以及在科学研究与生产实践中，解决用人力很难解决的问题的迫切需要。人工智能的产生和发展无疑已为人类智能的发展提供了崭新的空间和创造领域。随着信息处理技术的进步，智能计算机应用的深度和广度将不断发展，许多原本是人类思维独占的领域，都将会有人工智能的介入，除专家系统、模式识别、定理证明、问题求解、自然语言理解等等之外，知识发现、机器创新与发明等也不是不可能。**大自然也许并不能创造完美，它只能进化，但人却有可能**。人脑进化到现在，应该说是大自然赋予人类的最佳结构和配置，或者说是目前物质运动的最高实现形式，目前的生物进化很难超越；但人类可以发明电脑，可以把人脑与电脑结合起来，从而达到一种更高的境界。这种组合，是一种互补性的组合，是在更高层次上的一种“人-机交互”。

比较电脑与人脑，的确有许多相似之处。一是结构上，电脑颇类似人脑，人脑是由大约 140 亿个神经元组成的生物智能系统，电脑则是由上万或上千万个电子元件组成的物理[人工]智能系统，**如今的算力系统更是在以惊人速度增长**。二是功能上，人脑不但可以凭借感官接受信息，而且可以

通过大脑皮层存贮信息、加工信息，通过神经传导和运动神经的活动输出信息；与人脑相似，电脑也由五个部分，即输入设备、存储器、运算器、控制器和输出设备组成，分别用来接受信息、存贮信息、加工信息和输出信息。从总体上看，人脑的智能要高于（现在的）电脑，人通过自己的智慧和实践，创造了电脑（智能机器），在多个领域的多个方面应用它，并在使用它的过程中不断改进它，使之为人类服务；从电脑（智能机器）的产生、发展、应用及社会作用各方面看，（目前），它都是从属于人，是为人类服务的一种特殊工具。而从局部上看，电脑的某些功能又的确优于人脑，这也正是电脑之所以被研究和发展的必要性之所在。比如，电脑在计算上具有高度的精确性和无差错性；电脑具有人脑所难以比拟的高速度；电脑具有惊人的快速记忆力和高容量存储能力；只要电脑的元件质量完好，保证对它的能源供应，它就可以“不知疲倦”的连续工作；等等。而且，这些“局部”正一步步向“全部”迈进。

由此，我们或可以认为，电脑是人脑智力的产物，是人类智力的物化；反过来，电脑又可帮助人脑，使人脑借助电脑而得以增强其能力。电脑与人脑之间的这种相互影响、相互制约、相互促进和相互推动，在使人不断地改进和完善着自己的创造物---电脑的同时，也在不断地适应着因电脑的不断发展与进步而带来的新的环境，也在不断地提高着自己的创造力、知识水平、智能水平与实践水平。现有计算机在智能模拟方面确实还存在局限性，但这只能说明现有人工智能的理论和方法具有局限性。若我们相信人的创造力是无穷的，也就应该相信我们总有一天会让电脑具有人类希望其具有的智能水平。

人工智能归根结底是在一定背景框架下对于人类认知和思维的模拟。为此，这就需要我们首先要对人类的认知和思维的各个方面都要有十分清楚的了解，也就是对“人类究竟是如何认识客观世界的？人类认知和思维的过程和机理如何？人的知、情、意等各个方面对认知和思维的作用究竟如何？”等有比较清楚的了解。从目前的研究来看，我们在认知和思维的程序方面的认识已经比较清楚，我们已经明白了视觉、听觉、触觉等是如何产生的，已经知道了人的思维是怎样进行的。但是，在背景和信念对认知和思维的影响方面，则了解得还不太多；在知、情、意对认知的影响方面还不太清楚。一旦我们对这些方面的认识有了清楚的了解和认识，那么，人工智能的研究无疑将会有更大的进展。

社会化的认识论也给人工智能提出了一个重大的难题，即人类认识是社会化的，而不是独立的。一方面，思维是人脑的属性，也是物质运动的一种高级形式，思维过程本身是个自然过程；而另一方面，人的思维又与自然和社会相联系，伴随着不断地社会实践，思维内容会逐渐丰富，思维水平会不断提高。我们或许可以根据人类个体已表现出的行为进行程序设计，但这只能是简单的、肤浅的、低水平的行为的复制；对某些复杂的智能行为，其复杂的思维活动过程则很难规范化，也就很难模拟。要使智能机器具有这样的功能，第一，必须使机器能够在环境交互过程中像人一样能够理解；实际上，赛尔（中文屋）模型所提出的，就是一个理解能力的问题；第二，必须使机器具有像人一样的识别各种情景的能力；第三，必须使机器像人一样具有复杂的心理活动，并具有一定的意向性。如果说我们现在已经初步解决了由人的认知向机器思维转变的关键环节，即问题和思维的形式化问题。要使计算机能模拟人脑的所有功能，还必须清楚人的认识是一个复杂的心理过程。这就表明，我们只有**把人的整个“心灵”翻译成可执行的“形式化系统”**，我们才有可能对人类的心灵进行模拟（实现类人智能）。由此可见，科学认识论所从事的研究只是一项基础性的研究，它是人工智能得以的发展的前提，但仅有认识论的研究还是不够的，我们还需要对人类“心灵”的更透彻的理

解和研究。

目前，人类的有些思维活动过程，计算机是能够进行模拟的。能够进行模拟的，多是（相对）**“静态”**的过程。而目前计算机难以模拟的，则多是**“动态**（多变）”的过程。然而，即便是静态性的过程，即客观规律已经告诉了我们“有此必有彼”的关系，可以进行规范性操作的过程，其在运行过程中也不可避免地会受到动态的影响因子的影响。随着人们的认识的不断提高和技术的不断发展，“静态模拟”从低级水平逐步向高级层次过渡，动态（多变）性的模拟才有可能进行。所以，较为客观地说，**人工智能的水平所反映的实际上就是人类（认知）智能的水平；**正如迅速发展的电子技术，精密的机械加工，严密的控制程序，是“机器人”由低级智能向高级智能发展的基础；而这些，都是人类智慧的结晶。

智能计算机到底能不能具有人类智能？我们至今还没有一个完整的论证，也不可能仅仅靠理论就能来完成论证，**智能的“工程化”问题本质上也是一个实践问题**。无论人类在人工智能的研制（理论的和实践的）方面最终还能够走多远，一方面，我们不应该现在就对人工智能有过高的期望——如某些人所言，超人工智能马上就能能实现---因为人类智能是人类的专属，在可以预期的时间内，是任何一种动物都不能“超越”的，也是任何一种“机器”都无法完全“替代”的。目前，人工智能所能做的，充其量也只能是模仿人类进行“智能化信息处理”，而不能完全替代人类进行智能化的工作。另一方面，我们也不应低估人类的创造性，人类既然发明了电脑（智能机器），也一定会让电脑具有人类所期望的智能。我们坚信，随着智能理论和技术研究的逐步深入，人类最终将会在认识自己的基础上开发自己的智慧，在智能工程化的道路上走的更远。智能技术作为探索智能工程化的科学，将会进一步与认知科学紧密结合，探索出关于智能行为模拟和工程化的一系列新的概念、新的理论和新的方法，对此，我们充满信心。

17.1.3.2 人类智能模拟和工程化的主要方法与途径

智能可否工程化是一个可以自由讨论的问题，而如何实现智能的“工程化”？却是一个令人工智能的研究者头痛的问题。关于人类认知的研究只是给人类智能的功能模拟提供了一种可能性，对人类智能的“工程化”却需要解决一系列认知和技术上的难题。**自然化的认识论**把人类的认识过程完全自然化、理性化，而忽视了认知过程中的其他因素，特别是信念在认知中的作用。这在当前是必须的，因为如果我们不把认知自然化和理性化，我们就无法对认知进行模拟。但是，如果我们忽视了**信念**在认知中的作用，我们也将无法解释不同的人在实际认知过程中会产生出的不同的认识的现象，使模拟脱离现实。按照自然化的认识论的观点，分别模拟大脑的各种不同的认知功能是可能的，是迟早能够实现的。然而，要把多种认知功能的模拟有机地整合起来形成一个可解决现实问题的智能系统，却是一个十分困难的问题。尽管认知科学、脑科学和认知神经科学的研究都为此提供了大量的科学依据并且还在有所突破，但要实现智能的真实模拟和工程化、实用化，依然任重道远。

智能模拟和工程化的实现首先需要有坚实的理论基础和技术基础。智能模拟和工程化的理论研究涉及到智能科学、认知科学、心理科学、脑及神经科学、生命科学、语言学、逻辑学、行为科学、教育科学、系统科学、数理科学、工程科学，以及科学方法论、哲学甚至经济学等众多学科领域。它实际上是一门综合性的交叉学科和边缘学科。哲学可回答诸如“人的思想是如何从物质的大脑产生出来的？知识从哪里来？知识又是如何导致行动的？形式化规则能用来抽取合理的结论吗？”等问题，让我们明白智能模拟和工程化是否有可靠的思想基础。数学可回答诸如“什么是抽取合理结论的形式化规则？逻辑和计算的是否存在极限？哪些函数是可操作和可被计算的？”等问题，从而

使智能模拟和工程化成为一门规范科学，因为要实现成为一门规范科学的飞跃，就要求其完成一定程度的数学形式化。社会科学和决策理论可回答诸如“我们如何决策才可以获得最大效益？在环境不确定情况下如何决策？当行动的收益不是立即体现的，而是一些按顺序采用的行动的结果时，如何制定理性的决策？”等问题，使智能模拟和工程化的研究与实际应用相结合。智能的模拟和工程化更离不开对人类认知行为的研究。人类的认知行为具有不同的层次，研究也有多种途径。认知生理学研究认知行为的生理过程，主要是人的神经系统（神经元、中枢神经系统和大脑）的活动，是认知科学研究的底层。认知心理学研究认知行为的心理活动，主要是人的思维策略，是认知科学研究的顶层。认知信息学则研究人的认知行为在人体内的生物信息处理过程，主要是人的认知行为如何通过生物的信息处理过程，由生理活动变为心理活动或由心理活动变为生理行为的，这是认知活动研究的中间层。认知工程学则研究认知行为的工程化，即如何通过以计算机为核心的人工信息处理系统，对人的各种认知行为（如知觉、思维、记忆、语言、学习、理解、推理、识别等）进行等价信息处理和模拟。它们既是研究人类认知行为的重要手段，也是构建智能系统的理论基础。

智能模拟和工程化的研究也离不开系统论和控制论研究的支持。如何设计出一个可实现一定目标并按一定规则运行的系统？系统怎样才能在自己的控制下正常运转？作为要构建一个复杂信息系统的学科，它需要系统的理论和控制的理论。语言是思维的外壳，知识的表示和程序的编码以及运行都离不开语法和语言。语言和思维是怎样联系起来的？如何实现思维的形式化并编程实现？智能模拟和工程化的研究也就需要语言学的支持。

信息技术、计算机技术和自动化技术更是智能模拟和工程化研究的技术支柱。当前的智能模拟和工程化研究，既属于计算机科学技术的一个重要前沿领域，也属于信息处理和自动化技术的一个重要前沿领域。智能的模拟和工程化实现，需要解决一系列理论和技术上的难题。它需要一个高速的大容量的自动化信息处理系统作基础，又需要将思维和问题求解过程转换成系统的一系列数学运算，难度当然很大。这就决定了智能的模拟和工程化实现既需要多学科的综合研究，也需要多途径实现。目前已有的途径与方法就有：基于神经计算的结构模拟、基于符号演绎的功能模拟、基于控制进化的行为（生态进化）模拟、基于复杂系统行为涌现的社会模拟等。

我们知道智能的工程化是个长期的任务，智能模拟和工程化的长期研究目标，是建立一套完善的智能模拟理论并开发出可达到（或超过）人类智力水平的智能机器和智能系统，实现人类智能的工程化、自动化和社会化。具体来讲，就是要使（智能）机器不仅具有联想、推理、理解、学习等高级思维能力，具有分析问题、解决问题和发明创造的能力，而且还要具有听、说、读、写等感知和交流能力。然而，探索性的研究只能在有限的资助下由少数人去做，可服务于现有学术和工程应用的研究才应是智能技术近期研究和探索的中心。因为得不到实际应用的纯学术研究，是没有生命力的。智能模拟近期研究的重点，应集中在完善和提高现有智能技术和智能系统的理论、方法和技术方面。其实现策略应是，充分发掘现有计算机的潜力，先部分地或在一定程度上实现机器的智能，并运用智能技术解决各种实际问题特别是工程问题和管理问题，从而使现有的（智能）计算机更灵活、更好用和更有用，成为人类的智能化信息处理工具；从而可逐步扩展和不断延伸人的智能，逐步实现工程的智能化和智能的工程化进程。

尽管我们对于人类智能的本质和运行机制尚未完全清楚，但符号推理是一种高级智能行为，符号推理需要领域的专门知识却是得到公认的。所以，开发以知识和符号演绎为核心的基于知识的问题求解系统应是智能技术近期研究的一个主要方向；而提高基于知识的问题求解系统的性能（包括

解决困难问题的能力和效率）是使这些系统得到实用的关键。有些研究者认为：目前，靠提高计算机硬件速度和开发通用的问题求解方法（弱方法--如生成-测试、手段-目的、搜索算法等），难以奏效；提高智能系统性能的根本出路，在于加深对应用领域和问题求解任务的理解，以便获得能有效地用于求解特定问题的特定的方法和知识，并用合理而有效的方法加以实施。

曾经，人工智能研究所依赖的**基本哲理主要是合理性和简化论，并以物理符号系统假说作为构建智能系统的基本主导思想。**物理符号系统假说认为，一个完善的智能信息处理系统应具有下列 6 种基本功能：(1)输入符号(input)；(2)输出符号(output)；(3)存储符号(store)；(4)复制符号(copy)；(5)建立符号结构：通过找出各符号间的关系，在符号系统中形成符号结构；(6)条件性迁移(conditional transfer)：根据已有符号，继续完成活动过程。因而，这类信息处理系统也可称为符号操作系统(Symbol Operation System)或物理符号系统(Physical Symbol System)，而所谓的符号就是模式(pattern)。同样，如果一个物理符号系统，具有上述全部 6 种功能，能够完成这个全过程，那么，它也就是一个完整的物理符号系统，也就能够表现出智能。人具有上述 6 种功能，我们可以把人看成一个智能信息处理系统。若一个信息系统也是一个具有上述全部 6 种功能的物理符号系统，那它也就一定能够表现出智能。既然人是一个物理符号系统，计算机也是一个物理符号系统，那么，我们一定可以用计算机来模拟人的活动。这就是人工智能可以实现的理论基础。

人工智能的实现离不开表示、运算和搜索。试探性搜索、启发式搜索和不确定性推理方法可能更符合人类的思维过程。也就是说，在解决问题时，若有确定的算法，我们可以通过特定的表示将其转换为系统的运算；若没有确定算法或即使有可用算法但在当今计算技术下尚不能实现时，对这类问题的可行的解决方法是搜索、试探，再加上经验的启发。启发性知识多是一种来自专门领域的经验知识，在特定场合，经常能求得有关问题的满意解答。这也就促使了专家系统的诞生。专家系统曾经是智能模拟和工程化研究和实现的一条热门途径，相信，未来，它还会以智能体的形式发扬光大。

符号学派主要采取的是符号操作或符号转换的方法探索认知和智能的本质，而近期占主导地位的**联接主义学派，**则是希望从模拟大脑结构方面来探索认知和智能的本质。联接主义认为，人类的认知是从大量并行的神经元的相互作用中产生的。联接主义的目的不再是用符号和符号操作来模拟认知过程，而是希望模拟发生在神经系统中的过程。联接主义的理论在上世纪 60 年代由于受到明斯基等著名人工智能专家的质疑，曾一度处于低潮。然而，到了上世纪 80 年代，鲁梅哈特(D.E.Rumelhart)、麦克里兰(J.McClelland)等人《并行分布处理：认知的微观结构探索》一书的出版，使联接主义研究再度兴起，受到广泛的关注。**而近期 ChatGPT 等大（语言）模型的成功，更使联接主义方法走向了“巅峰”**。在联接主义学派看来，智能不是别的，智能主要是神经元网络整体“涌现”出来的特性。神经网络的涌现机制尽管跟物理符号系统不同，但神经网络的本质仍然是计算。需要说明的是，联接主义方法虽然不再把符号的操作看作是认知最重要的方面，但仍然是把每个神经元都看作是一个计算单元，并且把符号处理作为重要的基础；实际上，波拉克(J.Pollack)曾证明，人工神经网络与图灵机只有形式上的差别，在本质上它们是等价的。

17.1.4 人类智能工程化研究的简要历程

智能模拟和工程化的进程是以对智能的理论研究和对智能系统的开发为基础的，并与计算机的硬件与软件的发展密切相关的，已经历了一个漫长的发展历程。19 世纪以来，数理逻辑、自动机理论、控制论、信息论和心理学等科学技术的发展，曾为人工智能的研究，准备了思想、理论和物质

的基础。

首先是**“认知即计算”**思想的形成。最早从计算的视角审视智能问题的是关心人的认识本质的哲学家。**笛卡尔**认为，人的理解就是形成和操作恰当的表述方式。洛克认为，我们对世界的认识都要经过观念这个中介，思维事实上不过是人类大脑对这些观念进行组合或分解的过程。霍布斯更是明确提出，推理的本质就是计算。**莱布尼兹**（Leibnitz）认为，一切思维都可以看作是符号的形式操作的过程，可以建立一个符号系统和逻辑演算，来完成推理或者理解过程。布尔（George Boole）的布尔代数，将逻辑学由哲学变成了数学。将符号语言与运算相结合，即可表示任何（逻辑）事物，这也为建立严密的形式语言做了前期的准备。弗雷格、怀特海、罗素等人通过数理逻辑把人类的思维进一步形式化，形成了所谓的命题逻辑及一阶和高阶逻辑。在他们看来，逻辑和数学，都是根据特定的纯句法规则运作的。在这里，所有的意义都被清除出去而不予考虑。在弗雷格和罗素的基础上，**维特根斯坦**在他的早期哲学中把哲学史上自笛卡尔以来的原子论的理性主义传统发展到了一个新的高度。在维特根斯坦看来，**世界是逻辑上独立的原子事实的总和，而不是事物的总和**；原子事实是一些客体的结合，这些事实和它们的逻辑关系都在心灵中得到表达：我们在心灵中为自己建造了事实的形象。从某种意义上讲，人工智能一开始就是试图在机器中实现这种理性主义的理想。不过，它的实际操作还需要等到计算理论成熟之后。

在计算理论发展过程中，**阿兰·图灵**(A. Turing) 的思想可以说是最关键的。在 1936 年发表的论文中，图灵提出了著名的图灵机概念。图灵在不考虑硬件的前提下，严格描述了计算机的逻辑构造。这个理论不仅解决了纯数学基础理论问题，而且从理论上证明了研制通用数字计算机的可行性。他认为，通过建立一套精确的符号语言和演算系统，就可去解决人如何推理的问题。

然而，图灵并未止于此。他既从人类是“计算者”的观念出发提出了图灵机的原理，又反过来，希望从图灵机的概念出发，来说明大脑和计算的关系。图灵认为，**人的大脑应当被看作是一台离散态机器**。尽管大脑的物质组成与计算机的物质组成完全不同，但它们的本质应是相同的。离散态机器的行为原则上能够被写在一张行为表上；因此，与思想有关的大脑的每个特征也可以被写在一张行为表上，从而能被一台计算机所仿效。1950 年，图灵发表了《计算机器和智能》的论文，对智能问题从行为主义的角度给出了定义，设计出了著名的“图灵测验”，论证了心灵的计算本质。图灵测试的贡献，是将**“机器能思维吗?”**这样一个模糊的问题，转变为**“机器能否通过智能行为测试吗?”**这样一个可操作的问题。

早期的人工智能研究由于缺乏有效的实现手段，因而不可能有大的突破。是现代计算机的出现和计算机技术的发展，为人工智能的研究和实现提供了强有力的基础和手段，才使人工智能的研究变为了现实。同时，也是计算机的成功研制激发了人们对人工智能研究的热情，直接导致了人工智能学科的诞生。

1956 年夏季，由**麦卡锡**(John McCarthy)等年轻学者发起，一批来自数学、心理学、神经学、信息论和计算机科学等方面的专家学者，包括，哈佛大学的明斯基（M.Minsky）、贝尔实验室的申农(C.Shannon)和 IBM 公司的罗切斯特(N. Lochester),以及塞缪尔(A. Samuel)、纽厄尔(A.Newell)和西蒙(H.A.Simon) 等人参加的首次人工智能研讨会在达特茅斯(Dartmouth)大学召开，共同探讨了用机器模拟人类智能的一些基本问题，并正式提出了“人工智能” （Artificial Intelligence）这一术语。这个会议被认为是人工智能这一新兴学科诞生的标志。自此以后，人工智能作为计算机学科的一个重要分支，获得了快速的发展。其发展历程大致可划分为形成、成长、快速发展和稳步

增长等四个时期。

人工智能早期的研究成果包括：1957 年，Newell 和 Simon 提出了通用问题求解系统 GPS。他们分析了人类解决问题的一般规律：我想带儿子去幼儿园，我“已有的”和我“想要的”两者之间有什么差异？到幼儿园有一段距离，用什么方法改变这段距离？我的汽车。我的汽车坏了。修好它需要什么？一块新电池。哪儿有新电池？汽车维修店……，这就是**“手段-目标分析”方法**。GPS 就是**发现和规划从给定状态到目标状态的行动的程序**。它需要在一个大范围中进行选择性搜索。人们开发了一系列的搜索方法：搜索树、代价有限搜索、启发式搜索、双向搜索等等。1958 年，McCarthy 在 MIT 实现了 LISP。1959 年，Samuel 研制了一个跳棋程序，1962 年这个程序击败了 Connecticut 州的一个跳棋冠军。1959 年，Frank Rosenblatt 提出感知器模型(Perceptron Model)。

纽厄尔和**西蒙**认为，计算机操作的二进制数字串能代表任何东西，包括现实世界中的事物。纽厄尔和西蒙还进一步指出，人类大脑和计算机尽管在结构和机制上全然不同，但是在某一抽象的层次上则具有共同的特征：**人类大脑和恰当编程的数字计算机可被看作同一类装置的两个不同的特例，它们都通过用形式规则操作符号来生成智能行为**。1981 年，纽厄尔和西蒙把他们的观点总结为一个称作**“物理符号系统假设”**的假设：**对于一般的智能行为来说，物理符号系统具有的手段既是必要的，也是充分的。**所谓“必要”的意思是：任何表现出一般智能的系统，经过分析，都可以证明是一个物理符号系统。所谓“充分”的意思是：任何足够大的物理符号系统，都可以通过进一步组织，而表现出一般智能。这个假设的核心就是把思维看作是一种信息加工过程，把智能的本质看作是计算。

从 20 世纪 60 年代中期起，人工智能研究曾处于相对困难的阶段。主要原因是人工智能的研究并没能很快地取得突破性的进展。由于在研究的早期，人们不适当地过分强调和依赖于符号逻辑和形式推理，导致了人工智能的研究陷入基于**“弱法”（weak methods）**的纯学术研究的困境。所谓“弱法”就是通用问题求解策略，由于片面强调相应算法的通用性，忽视了问题领域特定知识的指导作用，因而很难解决所谓的“组合爆炸”问题。“组合爆炸”意指，由于复杂问题的解决涉及到大量因素，由这些因素的适当组合而构成的可能解答的数目相当庞大，以至于再高速的计算机也无法在合理的时间内通过穷尽的枚举方法来找出正确答案。其结果是，“弱法”只能解决智力游戏（过河，九宫图）或积木世界动作规划等十分简单的问题。于是，人们从刚开始的狂热中冷静下来，开始更加理智地看待人工智能。20 世纪 70 年代初，美国 Stanford 大学的 Feigenbaum 等人通过反思人工智能所遇到的问题，把目标转向解决较狭窄的问题，抓住典型事例，成功开发出 DENDRAL（从光谱仪提供的信息中推断出物质的分子结构）和 MYCIN（人类血液疾病诊断咨询系统），并进一步提出**知识工程**的思想，才使人工智能的研究从纯“弱法”的研究困境中解脱出来，赋予新的生命力，以至引起八十年代初的人工智能分支学科---**专家系统**的大发展。其关键的教训在于，早期的人工智能研究忽视了人类智能的本质在于有**“知识”**，而“知识”是可以用来合理地组织和指导问题求解，从而避免组合爆炸的。专家系统的发展使人工智能的研究摆脱了那种“高高在上”的形象，开始投入到实际问题求解的具体应用之中。知识工程的提出与专家系统的成功，确定了“知识”在智能系统中的突出地位。机器学习、智能计算、神经网络等的深入研究，更使智能工程化的研究形成了百花齐放的繁荣局面。尽管不同人工智能学派间的争论非常热烈，但它们都推动了人工智能研究的深入发展。

人工神经网络的研究从一开始就倍受关注；首先是 1943 年由 McCulloch 和 Pitts 发表的“神经

元数理逻辑模型”，他们开创了人工神经网络的方法。它的基本思想是**通过模拟大脑皮层神经网络的“结构”特征来复现智能**，因此被后人称为是“结构主义”的方法。McCulloch、Pitts、Rosenblatt、Widrow、Hopfield、Kohonen、Grossberg、Amari 等人就是这一方法的代表人物。1958 年美国心理学家罗森勃拉特（Rosenblatt）发表的著名论文：“感知器：脑的组织与信息存储的概率模型”，就提出了著名的感知器模型，成为影响巨大的多层感知器的基础。1961 年，明斯基（Minsky）证明了罗森勃拉特的感知器的缺陷：只能完成线性可分的模式分类，不能完成诸如“异或”或者非线性的分类，曾使神经网络的研究走了一段弯路。霍普菲尔特（J. Hopfield）提出的具有联想记忆能力的神经网络模型的提出和鲁梅尔哈特(D. Rumelhart)等人面向多层神经网络的 BP(Back-Propagation，反向传播）学习算法的实现，使神经网络研究的又一次兴盛。如今，人工神经网络更是被广泛而深入地研究，并开始进入实际应用。其应用的领域也包括着系统感知与辨识、模式识别（如语音识别）与自然语言理解、智能控制（如家电和工程控制）和机器学习等。而 2020 年之后，基于神经计算的 ChatGPT 等大（语言）模型的横空出世，更将人工智能的研究热潮推向前所未有的高度；各种大（语言）模型层出不穷，在使人工智能正式走入人们工作和生活的同时，也使人工智能向着通用人工智能迈进。

20 世纪 80 年代初期,也曾出现过一股被人称作是智能技术研究的“淘金热”。正像戴维斯(Davis)当时所指出的那样，“这是很有讽刺意义的：三年前（70 年代末），人工智能还被认为是不可靠的，现在却成了热门，每个人都想去利用。”当时，专家系统的初步成功和日本于 1979 年宣布的第五代机研究计划，或许对此起到了决定性的推动作用。当时，美国、西欧和英国，都曾拟定了庞大的投资计划，作为对日本人挑战的应答。在美国，DARPA（美国国防部先进科研项目管理处）提出新一代计算系统的 10 年研究计划；西欧提出 ESPRIT 计划。这一时期，人工智能研究曾被认为极具经济价值，因而冒险性投资骤然剧增。但是，由于当时人工智能技术的不成熟性，以及一些人对专家系统可产生的经济效益的期望值太高，结果不免令人失望；更糟糕的是，大部分草率上马的专家系统项目均未达到实用化的程度。于是到了 20 世纪 80 年代末期，人工智能特别是专家系统的研究热潮就大大降温。进而导致了一部分人对人工智能研究前景又开始持悲观态度，甚至有人提出人工智能的冬天已经来临。如今，人工智能的热潮在经历了“全民”的兴奋后，会不会也会出现反复呢？我们可以肯定地回答，肯定不会了，因为时代不同了，技术不同了。

尽管 20 世纪人工智能的研究跌宕起伏，一会儿兴奋，一会儿悲观，一会儿是热潮，一会儿是谷底，“悲喜交加”，但大部份的人工智能研究者还都保持着清醒的头脑。他们认为，不应过早地渲染人工智能的威力，而应脚踏实地的多做一些基础性的工作。在一些基础性的工作未能突破之前，过高的期望是没有用的。自此，才有了人工智能研究的稳健的发展时期，才使人工智能技术的实用化进程逐步进入成熟时期，才有了如今的繁荣发展。这其中，**大数据、大模型、大算力和先进的（学习、推理和生成）算法**都功不可没。

人工智能研究的主要目标，就是希望用现代科学技术的理论和方法来理解和扩展人类智能系统的能力。由于智能问题的重要性和复杂性，人们曾经分别从结构[生理]、功能[心理]和生态[社会]等多个不同侧面和不同的方法对自然智能问题展开模拟研究，并于 20 世纪分别形成了人工智能的联接[结构]主义、符号[功能]主义和生态[（行为）进化]主义等有代表性的理论体系。联接[结构]主义的基本思想是希望通过模拟大脑皮层神经网络的“生理结构”特征来复现智能，开创了人工神经网络方法。符号[功能]主义则倡导以现有计算机为硬件支撑平台、用符号逻辑描写、由软件编程来

模拟智能的方法，开创了经典人工智能（Artificial Intelligence)这一学科。它不关心系统的信息处理结构，只关注系统的功能表现，因此也被称为是“功能主义”方法。生态[（行为）进化]主义方法关注对特定环境中智能系统的输入(刺激模式)-输出(动作模式)关系生成的分析，认为智能系统中的智能应源自于在特定生态环境中的“适应性”交互学习，主张智能系统的智能应是在现实环境中交互训练和进化的结果，主张一个智能系统首先应能鉴别输入的模式，然后可根据学习到的输入-输出之间的关系去决定其输出的动作方式。目前，这些理论还没有完全统一。这表明，人们对于智能的共同本质在认识论上还存在分歧。我们认为，尽管这些研究是很有价值的，也是应该做的，但是，若只是从不同侧面分别来模拟复杂的智能系统，这在方法论上难免会残留着“盲人摸象”、“以偏概全”的弊病，因此，对智能模拟的深入研究和对通用人工智能系统的开发，需要各种方法的深入探索和融合。而联接[结构]主义、符号[功能]主义和生态[进化]主义的和谐统一，应是智能模拟和工程化理论未来发展的重要发展方向。而现今正在研发的大（语言）模型和人工智能体，本质上也是各类方法综合运用的结果。

17.1.5 智能模拟和工程化当前的研究状况—主要研究领域与方向

人类智能的产生主要基于以下能力：感知、记忆与辨识能力；思维与问题解决能力；行为与语言能力；学习及自适应能力；智力融合与交互能力等。当然这些能力是密切相关，不可截然分开的。不过，对于某一方面的特别关注，也便产生了人工智能的一些主要的研究领域和分支。比如，对感知能力和行为控制能力的模仿促进了智能机器人技术的发展；对记忆和思维能力的模仿促进了知识工程、人工神经网络、机器学习技术的发展；对语言能力的研究和模仿促进了自然语言处理技术的发展；等等。因而，智能模拟和工程化的研究内容，相应地也包括：感知与交流、记忆与联想、知识与推理、搜索与求解、学习与发现、发明与创造、系统与建造、应用与工程等。

图 17.1.3　曾经的第六代计算机研究计划的组成

人工智能对于人的智能行为和功能的模拟，主要是要模拟人的思维和认知[问题求解]过程。正是基于人工智能所追求的这一目标，它才在模拟人脑的思维和认知的过程中，无论从广度还是深度方面讲，都有了长足的发展。它的研究所包括以下领域（曾经）有：推理及搜索算法[包括状态空间图搜索、博弈树搜索、归结算法、不确定性推理等]、知识工程[包括知识获取、知识表示、知识运用等一系列知识处理技术]、基于知识的问题求解系统[是以知识为中心构建智能系统的技术，包括

专家系统(Expert Systems)等]、模式识别(Pattern Recognition)、机器学习(Machine Learning)、机器感知、自然语言理解(Natural Language Processing)、自动定理证明(Automatic Theorem Proving)、自动程序设计(Automatic Programming)、智能决策支持系统、智能机器人技术(Robert)、人工神经网络(Artificial Neural Network)、计算机博弈、数据挖掘与知识发现、分布协同与智能代理技术（Agent)、智能工程控制、智能制造和智能检索(Intelligent Information Retrieval)等。而从工程应用的角度看，目前，人工智能研究比较活跃领域则包括：机器感知（感知智能)、基于知识的问题求解系统、机器学习和知识发现、人工生命与智能机器人技术、以及基于多智能体的集成智能[多Agent技术和人机协同技术]等。下面，我们将对这些领域的研究内容作些简要的介绍。【需要说明的是，自2020年以来，**以GPT为代表的大（语言）模型**的成功开发和应用，已极大地激发了人们对人工智能的热情，研究和开发空前高涨。智能技术也逐步转向了**基于大模型和（类脑）智能体的研究和开发，目标是实现通用智能和类脑智能**。对此，我们将在后续的章节中详细介绍。】

17.1.5.1 机器感知、识别与理解（感知和认知智能）

机器感知是机器智能的基础，是机器获得外部信息的重要途径。机器感知主要包括计算机视觉和口语识别等。计算机视觉研究用计算机来模拟人和生物的视觉系统功能，使计算机具有“感知”周围视觉世界的能力；具体来说，就是让计算机具有对周围世界的物体进行传感、抽象、判断的能力，从而达到识别、理解的目的。根据其处理过程的先后及复杂程度，计算机视觉的任务可以分成下列几个方面：图像的获取、特征抽取、识别与分类、三维信息理解、景物描述和图像解释。口语（语音）识别建立在自然语言理解的基础上。书面语言的理解包括词法、句法和语义分析，口语识别需外加语音分析。因此，机器感知的技术基础是模式识别技术与自然语言理解技术。

模式识别是人类的一项基本智能。在日常生活中，人们一直在进行“模式识别”。随着计算机技术的发展和人工智能的兴起，人们当然也希望能用计算机来代替或扩展人类的识别功能。模式识别迅速发展并已成为一门独立学科。什么是模式和模式识别？广义地说，存在于时间和空间中可观察的事物，如果可以区别它们是否相同或相似，都可以称之为模式；狭义地说，模式是通过对具体的个别事物进行观测所得到的具有时间和空间分布的信息。我们把模式所属的类别或同一类中模式的总体称为模式类(或简称为类)，而“模式识别”则是在某些一定量度或观测基础上把待识模式划分到各自的模式类中去。**模式识别**技术的目标是利用计算机来模拟人的各种识别能力，包括对文字、声音、图形、图像、人物和情境等进行自动识别。模式识别的研究主要集中在两方面，即研究生物体(包括人)是如何感知对象的；以及在给定的任务下，如何用计算机实现模式识别（**感知智能**)。前者是生理学家、心理学家、生物学家、神经生理学家研究的内容，属于认知科学的范畴；而后者则通过数学家、信息学专家和计算机科学工作者近几十年来的努力，已经取得了系统性的研究成果。

一个计算机模式识别系统基本上是由三个相互关联而又有明显区别的过程组成的，即数据生成、模式分析和模式分类。数据生成是将输入模式的原始信息转换为向量，成为计算机易于处理的形式。模式分析是对数据进行加工，包括特征选择、特征提取、数据维数压缩和决定可能存在的类别等。模式分类则是利用模式分析所获得的信息，对计算机进行训练，从而制定模式判别标准，以期对待识模式进行分类。

当前，有两种基本的模式识别方法，即统计模式识别方法和结构(句法)模式识别方法。统计模式识别是对模式的统计分类方法，即结合概率统计的贝叶斯决策进行模式识别的技术，又称为决策理论识别方法。利用模式与子模式分层结构的树状信息所完成的模式识别工作，就是结构模式识别

或句法模式识别。统计模式识别的基本原理是：有相似性的样本在模式空间中会互相接近，并形成“聚集团”，即“物以类聚”。其分析方法是根据模式所测得的特征向量 $X_i=(x_{i1},x_{i2},\cdots,x_{id})^T$ $(i=1,2,\cdots,N)$，将一个给定的模式归入特定分类$\{\omega_1,\omega_2,\cdots,\omega_c\}$的某一个类别之中，然后根据模式之间的距离函数来判别分类。其中，T 表示转置；N 为样本点数；d 为样本特征数，c 为特定类别数。统计模式识别的主要方法有：判别函数法，k 近邻分类法，非线性映射法，特征分析法，主因子分析法等。在统计模式识别中，贝叶斯决策规则从理论上解决了最优分类器的设计问题，但其实施却必须首先解决更困难的概率密度估计问题。BP 神经网络可直接从观测数据(通过训练样本)进行学习，是更简便有效的方法，因而获得了广泛的应用，但它是一种启发式技术，缺乏指导工程实践的坚实理论基础。统计推断理论研究所取得的突破性成果也导致了现代统计学习理论—VC 理论的建立，该理论不仅在严格的数学基础上圆满地回答了人工神经网络中出现的理论问题，而且导出了一种新的学习方法—支撑向量机。模式识别技术发展至今，人们的一种普遍看法是不存在对所有模式识别问题都适用的单一模型和解决识别问题的单一技术，我们现在拥有的只是一个工具袋，所要做的是结合具体问题把统计的和句法的识别结合起来，把统计模式识别或句法模式识别与人工智能中的启发式搜索结合起来，把统计模式识别或句法模式识别与支持向量机的机器学习结合起来，把人工神经网络与各种已有技术以及人工智能中的专家系统、不确定推理方法结合起来，深入掌握各种工具的效能和应有的可能性，互相取长补短，才能开创模式识别应用的新局面。

模式识别技术具有近乎无限的发展潜力，它既是人工智能发展的技术基础之一，又有广阔的应用领域。模式识别的主要应用包括图像识别、文字识别、生物特征识别、语音识别和口语识别等。

文字识别、生物特征识别可看作是特定的图像识别。汉字已有数千年的历史，是世界上使用人数最多的文字，对于中华民族灿烂文化的形成和发展有着不可磨灭的功勋。机器自动识别输入技术可将文字方便、快速地输入机器。自动输入又分为汉字识别输入及语音识别输入。从识别技术的难度来说，手写体识别的难度高于印刷体识别。语音识别技术所涉及的领域包括：信号处理、模式识别、概率统计、发声机理和听觉机理、人工智能等等。语音识别的工作原理是：首先在计算机中存放所有字词的读音，建立一个样本数据库，然后通过话筒将特定用户说话的声音输入计算机；计算机将输入的声音和数据库中的所有声音样本逐一进行对照，找出最接近的声音样本，最终确定输入的声音是哪些字或词。利用基因算法训练连续隐马尔柯夫模型的语音识别方法现已成为语音识别的主流技术，该方法在语音识别时识别速度较快，也有较高的识别率。

生物特征识别包括人像识别和指纹识别等。在生物识别技术领域中，指纹识别技术以其独特的方便性、经济性和准确性等优势受到世人瞩目，并日益成为人们日常生活和工作中重要的安检验证方式。我们手掌内侧表面的皮肤凹凸不平所产生的纹路会形成各种各样的图案。而这些皮肤的纹路在图案、断点和交叉点上各不相同，是唯一的。依靠这种唯一性，就可以将一个人同他的指纹对应起来，通过比较他的指纹和预先保存的指纹，便可以验证他的真实身份。利用指纹来鉴定人的身份，可以克服证件、签字、照片、密码、钥匙、印鉴等容易假冒、丢失、遗忘的缺点。目前，指纹识别结合生物扫描技术，既可以识别指纹的平面图像特征，也已可以对指纹表皮下的毛细血管的分布特征以及手指的三维图像特征等进行识别。

自然语言理解(Natural Language Processing)系统是不仅能识别文字而且能理解文本含义的机器系统，其目标是让计算机“听懂”或“看懂”人类的自然语言。贯穿自然语言理解研究的主流技术一直是句法-语义分析，这就决定了人工智能技术，尤其是基于规则的推理技术，在自然语言理解

研究中具有不可替代的核心作用。基于自然语言理解的研究在句法-语义分析技术方面已取得了重要进展，其实用化和工程化的努力也导致一批商品化的系统出现于国际市场。但现有的实用系统多是基于语料库的自然语言理解系统。自然语言理解的另一重要应用是**机器翻译**—利用计算机把一种自然语言转变成另一种自然语言。目前已有：电子词典类翻译软件、网上在线翻译类软件和一些专业类翻译软件等。

语音（口语）识别（speech）长期以来就是人工智能要解决的困难问题。我们可以把人工智能所面临的问题按困难程度排列如下：智力游戏[如国际象棋]、定理证明、专家系统、自然语言理解、语音（口语）识别、机器视觉。智力游戏的难度相对地较小，因为推理所需要的知识和数据量有限，也不要求实时处理。而口语识别和机器学习涉及的知识和处理的数据量很大，有时还要求实时响应。两者相比，计算上的复杂度可能差好几个数量级。提供与计算机的口语接口（语音界面或语音识别系统）有许多优点：自然而流畅。语音识别曾经存在许多问题，如今大（语言）模型基本上都已解决，其基础当然是基于自然语言理解。

机器视觉的研究开始于20世纪50年代。也曾有许多困难问题，如今也大部分得到解决。机器视觉在机器人和自动驾驶系统都有迫切需要。应用中，它应具有彩色立体的视觉能力，能识别道路上的障碍物，并避开障碍物行驶。大多的自动驾驶系统通常是一个层次系统，最底层用于感觉运动和执行驾驶命令；中间层执行路面驾驶、地域分析、障碍侦察和路径规划等任务；高层的任务则包括物体识别、路标识别、基于地图的预测和长距离路线选择等。目前，具有视觉能力的机器人和自动驾驶车辆已能在崎岖的山路上行走，也可下潜到海底执行勘探任务等。

17.1.5.2 知识工程与人工智能体--基于知识的问题求解系统

基于知识的问题求解系统简称 KB(Knowledge Based)系统，是应用人类知识来解决问题（通常是困难问题）的计算机软件系统。当其表现出专家级问题求解能力时，常被称为专家系统(Expert Systems)。基于知识的问题求解系统要求系统能借助知识获取工具从领域专家或其它知识源抽取专门知识，并转变为适合于推理机制解释性使用的形式存放于知识库中，以实现类似于人的逻辑思维和问题求解能力。基于知识的问题求解系统曾被认为是智能工程化领域的最有实用前景和影响力的一个分支，在科技、工程、交通、运输、医疗、探矿、气象、商业、金融、军事、行政和管理等领域会有广泛的应用。如今，人工智能技术的发展已经超出了原有专家系统的视野，今后，重点发展的将会是人工智能体系统。

我们知道，人类智能的基础是知识，问题的解决离不开知识。但是，要构建一个基于知识的问题求解系统，需要解决许多理论和应用问题。一是知识的模型化研究。知识与信息不一样，知识是结构化的信息。知识模型是基于知识的问题求解系统的基础。由于知识必须形式化后才能被计算机存储和处理，这就需要有一套适合特定系统的信息与知识的表示理论，以及相应的“推理”系统和功能。过去，曾经提出的知识和信息表示方法有逻辑表示方法、可拓关系表示方法、语义网络表示方法和框架表示方法等；未来的发展模型可能会有：面向 Agent 的模型、面向本体(Ontology)的知识图谱模型、面向并行推理的新一代黑板模型、面向分布式推理的网络模型、能演化的模型、自组织模型和共生模型等。基于模型的（链式）推理通常是定性推理的基础，而基于模型的知识获取也将是构造知识库的有力手段。二是关于常识性知识的研究。常识性知识处理曾经是人工智能的一个难题，缺乏常识也曾是过去大多数专家系统的一大弱点。研究基于专业知识和常识的问题求解模型，需要构建常识知识库，这曾比构建专业知识库更困难，因为它的论域通常难以限定。如今，大（语

言）模型已经初步解决了这个问题，因此，基于大模型的人工智能体将会是一个重要的发展方向。三是关于非规范知识的研究。非规范知识的特点是量大（海量）、不完全（结构残缺不全）、模糊（内涵不够清楚）、不协调（含内在矛盾）、带噪音（含无用杂质）和时变（内涵不稳定）。这就需要我们要研究非单调逻辑（排除矛盾）、次协调逻辑（不排除矛盾，但限制影响）和开放逻辑（不排除矛盾，而是发展相对正确的理论）等。这也需要我们要扩充现有的认知逻辑，使其能容纳矛盾和不一致性；扩充现有知识的维护机制，使其能直接处理矛盾知识；发展基于辩论的矛盾知识处理方法，以使其能达到辩证的统一；结合时态逻辑和认知逻辑，建立反映知识增长的时态认知逻辑，以反映知识的时变特性等，并建立能反映知识集群随时间变化的形式体系等；建立可对残缺、不完全、含噪音的知识进行优化的理论，以使其可被充分利用。过去，这些都很难做到，是 GPT 等技术初步解决了此类问题，它们基于大数据、大模型、大算力，有效避开了一些复杂的逻辑难题。四是关于知识的获取理论和技术的研究。知识的获取过去一直是知识工程的瓶颈。人们曾考虑过的获取方法包括：面对面的交流获取方法、基于机器学习和数据挖掘的人工智能方法和基于知识库群的知识共享方法等。我们看好的机器自动学习（包括文本学习）和基于数据挖掘的知识发现方法。这一从数据集中识别出有效的、新颖的、潜在有用的，以及最终可理解的模式的非平凡过程，可自动将无序的信息变为有序的知识，应是知识获取，特别是领域知识获取值得关注的方法。如今，基于发达的网络和广泛的信息交互，基于大模型强大的学习能力，知识自动获取也已经变为了现实，不再是难题。五是信息与知识的转换理论。知识是信息经过加工、整理、解释、挑选和改造而形成的。由多种信息源中获取信息和知识需要信息和知识的转换理论，根据多种信息和知识形成决策更需要信息和知识的转换理论，集合多种类型的专家（知识）系统以协作解决问题也需要信息和知识的转换理论。这种转换研究不仅包括形式方面的转换研究，也包括其转换机制和转换机理的研究。因此，相关的转换机制和算法研究也许是长期的。

基于知识的问题求解系统中最重要的曾是专家系统。**专家系统**是处理源于现实世界的需要由具有专门领域的知识和能力的专家来分析和判断的复杂问题的问题求解系统。专家系统的研究曾推动了专家系统技术的发展。如今，专家系统技术的实用化已经发展为对专业智能体的开发，并越来越多地渗透到各行各业。其应用已包括：● 生产过程的监控，如化工、冶炼、发电、造纸、材料加工处理等，通过监控过程变量，可优化生产过程，保证生产质量；● 调度和生产管理，可按客户订单和工厂现状，安排生产进度和分配生产资源；● 管理和决策过程的自动化，包括库存管理和材料需求规划，市场销售规划与采购决策，保险业的风险管理，个人投资管理，政府事务的自动化等；● 工程项目管理，如建筑施工、公路维护、环境保护等，包括项目开发规划、工程进展监测、规划复审和修改等；● 数据解释，包括天气预报，地质勘探数据解释等；● 故障诊断和预测，如大型设备，生产流程的故障诊断等。今后，其应用会渗透到我们工作和生活的方方面面。

智能管理与决策支持系统可认为是一类特殊的基于知识的问题求解系统或专家系统或专业智能体系统。**决策支持系统**（DSS）是综合利用大量数据，有机组合众多模型（数学模型与数据处理模型，也包括大语言模型等），通过人机交互，辅助各级决策者（或工作人员）实现科学决策的系统。决策支持系统使人机交互系统、模型库系统、数据库系统、知识库系统有机结合起来，大大扩充了数据库的功能、知识库的功能和通用智能系统的功能。它的发展将使管理信息系统上升到了决策支持或自动决策的阶段，使那些原来不能用计算机解决的问题逐步变成能用计算机来解决。智能管理与决策支持系统把人工智能技术用于决策，可有效地解决半结构化和非结构化的（常规）问题。今后，

其应用范围会更加扩大，其决策能力会逐步提高，并最终可“替代”人。

西蒙（H.A.Simon）曾提出，**管理就是决策，决策的本质就是问题求解的过程**。西蒙曾把决策问题分成程序化决策和非程序化决策。现在，人们已把**程序化决策**的提法换成**结构化决策**。并把决策问题按结构化程度分为**结构化决策、半结构化决策和非结构化决策**。划分的标准由下面三个因素来区分：问题形式化描述的难易程度；解题方法的难易程度；解题中所需计算量的多少。结构化问题是常规的和完全可重复的，每一个问题仅有一个求解方法，可以认为结构化决策问题可以用（常规）程序来实现。由于非结构化问题尚不具备已知求解方法或存在若干求解方法，而所得到的答案也可不一致，这样，它也就难于编制程序来完成。非结构化问题解决实质上包含着创造性或直观性，常规计算机（程序）难以处理。而人则是处理非结构化问题的能手。如今，智能计算机已能有效地处理半结构化决策问题，而将智能计算机的智能管理与决策支持功能与人有机结合起来，实际上是可以有效解决半结构化和非结构化的问题的。

自动规划、调度与配置系统也可认为是一类特殊的基于知识的问题求解系统或专家系统或智能体系统。规划和配置是人类生产和社会活动的重要形式。**规划**旨在为活动实体（人、组织、机器）设计合理的行为--按时间顺序的活动序列；**配置**则为实现拟定功能的实体设计合理的部件组合结构--按空间位置的部件布局。早期经典的规划技术是机器人动作规划系统STRIPS。20世纪70年代出现的部分排序规划技术（以NOAH系统和目标回归方法为代表）使经典规划取得了突破性进展。然而，鉴于真实世界的复杂性，大多数实际规划问题（包括机器人动作规划）都不满足经典规划问题的假设（系统每个动作的执行结果是完全可预言的），于是，研究者们开始了非经典规划技术（以动态世界规划和专用目的规划为代表）的研究。作为一个实用领域，智能化的调度、规划和项目管理已得到了深入的研究。从知识工程的角度看，自动配置也是知识密集型问题求解任务，更需要领域特别的知识而非一般的解决方法。配置任务以一般的需求说明作为输入，并将选择什么部件以及如何组装它们的详细说明作为解答输出。常见的基于知识的配置问题解决过程一般遵从一种二阶段模式：解答扩展和解答精化。前者可将使用者的需求说明映射到关于配置方案的抽象说明，后者则可将这个抽象说明（解答）映射到细化的物理配置方案，以详细说明组装（配置）的安排和进一步的需求。自动规划、调度与配置系统其实就是一类专业专家系统或智能体系统，我们可以根据具体需求开发各类相关智能体。

计算机博弈曾是一类经典的人工智能程序。下棋等博弈问题也确实是很适合人工智能技术去解决的一类问题。在人工智能研究的早期，它就是人工智能研究的主要对象之一。有不少人工智能问题求解的技术，就源于对下棋等博弈程序的研究，如状态空间搜索方法等。按人工智能的术语，棋盘的一种棋子布局称为一个**状态**，所有可能的状态即构成状态空间。显然，状态空间是很大的，需有效的搜索方法去从中搜索解答。麦卡锡曾提出过alpha-beta修剪算法，可把为决定下一个走步而须对棋盘状态空间的搜索量从指数级减少为指数的平方根级，大大提高了机器下棋的水平，并曾被广泛地用于各种下棋程序中。1997年，由IBM公司研制的超级计算机“DeepBlue”曾在与国际象棋冠军卡斯帕罗夫的六盘对弈中，取得三胜二和一负的战绩。这场人机世纪之战曾向世界表明，机器智能可以在某些方面超过人类，具有划时代意义。如今，此类博弈程序早已不再新鲜，更新的阿尔法狗等程序也具有了更高级的智能。其实，我们希望人工智能和智能系统能处理现实世界中的各类博弈问题，无论是政治、军事或经济方面的。尽管会有一定难度，但我们相信，人工智能一定可以成功。

其实，基于知识的问题求解系统还应包括（高级）智能控制系统。智能控制是智能技术在工程方面最成功的应用领域。目前，它已成功应用于各类需要变结构控制的工程系统之中。

17.1.5.3 机器学习、知识发现与 AI 辅助创新

学习和创新是人类智能的最重要形式，只有让系统具有类似于人的学习和创新能力，才有可能实现智能工程化的最终目标——拓展和替代人脑的部分功能。所以，机器学习和创新一直是人工智能研究关注的核心问题。机器学习的应用首先是基于知识的问题求解系统。只有通过不断学习，不断获得解决问题的知识，基于知识的问题求解系统才能不断提高解决问题的能力。有研究认为，通过让 AI 系统阅读大量相关文献来构造大型知识库，可使知识工程的工作量大大减小；通过让 AI 系统搜索 WWW 的相关知识来构造大型知识库，可使知识工程的工作量再减少一个数量级。机器学习有多种方式，较低级的机器学习方式是机械式的学习和有教师指导下的学习。前者是通过对文献或网页的阅读理解，或通过简单地记住以往推理和解答的结果，来强化系统的知识库或将来的推理和问题求解能力；后者则是通过教师传授知识，或在教师的指导下归纳和抽取知识，来提高自己的能力。机器学习的高级阶段是知识发现(Knowledge Discovery)，即系统在没有教师的帮助的情况下，能自行发现蕴涵在客观数据中的知识。如今，机器学习可从数据仓库包含的大量数据中发现和获取隐含的知识，这为智能机器未来的自我进化带来了新的机遇。AI 辅助创新也是令人感兴趣的领域，生成式 AI 应该说已经有了部分创新功能，而 AI 辅助发明系统或 AI 辅助文艺创作系统，也正引发不少人的兴趣。

17.1.5.4 人工生命与智能机器人技术（具身智能）

机器人(Robert)是在一定的条件下能够完成人类部分的行动或功能的机械系统，它的研究通常被划分为三个阶段：程序机器人阶段、自适应机器人阶段和智能机器人阶段。智能机器人能模仿人的部分综合处理问题的能力，可以根据感觉到的信息，进行独立识别、推理，并做出判断和决策，对外界信息做出正确的反应，不用人的参与就可以完成一些复杂的工作。传统的人工智能研究和机器人制造采取的主要是自上而下的研究方法，即研究者先确定一个复杂的高层认知任务，接着把这个任务分解为一系列子任务，然后构造实现这些任务的完整系统。这种研究方法的核心概念是“思想(thought)”和“推理(reasoning)”。而新的以人工生命为主的自主进化机器人设计，则是一种自下而上的思路。这其中也包括布鲁克斯的基于行为进化的机器人设计：仅仅给机器人赋予简单的刺激-反应规则而不是复杂的知识和推理，机器人就能灵活地去适应多变的环境。未来，我们希望进化机器人是基于大模型进化的，可通过能力移植赋能于现实中的各类机器人，这无论对机器人的研究还是对人工生命的研究都将具有重要意义。

17.1.5.5 基于多智能体的（社会）集成智能系统

基于多智能体的集成智能系统研究也是近年来智能模拟领域非常流行的一个分支学科。它开始时是希望模仿自然、社会等复杂系统中的“群集”行为等来解决现实中的智能计算问题。其研究者认为，智能常表现为社会行为，即多个智能体协作的结果，无论在生物界或人类社会，均是如此。因此，分布式人工智能与多智能体系统理论的研究就具有重要的理论价值和现实意义。智能体是一种具有智能的独立“主体（Agent）”，这种“主体”具有自主性、反应性、适应性和社会性等基本特性，能依据特定心理状态（信念、期望、意向）自主工作，并具有语义互操作和行为协调合作能力。作为参与协调合作的成员之一，智能体不仅可为实施紧凑一致的协同工作提供有力的支持，也为建立面向网络计算的开放性、可重构和可伸缩的新型集成化智能计算系统奠定了基础。智能体技术所

提供的行为自控和群体协作能力，吸引了来自不同学科领域（知识工程、机器人、软件工程、信息系统等）的众多研究者，这使集成智能技术（特别是人机融合的智能集成技术）成为了集知识处理、网络通信、软件工程、社会行为认知等理论和技术于一体的综合性研究，并必将在未来的智能模拟和工程化研究中发挥重要作用。其自主特性和分布协同处理机制也为智能模拟的研究开辟了新的途径。特别是各类大模型出现之后，基于多智能体理论的人-机智能融合研究更引起人们的关注。

17.1.6 智能模拟和工程化研究---已有的方法和途径

智能模拟和工程化的研究主要包括对人类智能行为和认知机理的模拟研究和具有人类智能特性的应用系统的构建。人工智能自诞生以来，其目标就是为了制造出能像人一样进行思考的机器来，为了达到这个目标，科学家们经历了多年的努力，试图从不同的途径来达到智能模拟或工程化的目的。智能模拟已包括了结构模拟、功能模拟和行为模拟等。在智能模拟和工程化研究实践中，相应地也形成了多种模拟方法和多种实现途径，形成了多种学派。目前，智能模拟的主要学派有符号[功能]主义学派、联接[结构]主义学派和生态[进化]主义学派等。

17.1.6.1 基于符号主义的方法与途径

符号主义(Symbolicism) 学派又称为逻辑主义(Logicism) 学派、认知心理学派(Psychlogism)或计算机学派(Computerism)。其理论基础是认知心理学，特别是物理符号系统假设和有限合理性原理。他们认为，人类认知的基元是符号，认知过程即符号操作过程；认为，一个完全的物理符号系统是产生智能行为的充分而必要的条件，人是一个物理符号系统，计算机系统也是一个物理符号系统，因此，可以用计算机信息处理系统来模拟人的智能行为；认为，知识是信息的一种形式，是构成智能的基础，一个智能系统，就是一个可运用知识对信息进行“处理”的系统，这一基于知识对信息进行处理的智能系统的主要内容应是：知识的获取（knowledge acquisition）、知识的表示（knowledge representation）、知识的组织与管理、基于知识的推理和知识的运用等。它们即构成了所谓的知识工程（Knowledge Engineering, KE）或基于知识的问题求解系统的主要内容。

有限合理性[或满意性]原则是西蒙在研究人的决策时总结出的关于智能行为的一条基本原则，西蒙也因此获得了诺贝尔奖。该原则指出，人在超过其思维能力的条件下（例如，遇到所谓的 NP-完全问题—其状态空间呈现指数增长，从而需爆炸性的搜索工作量），仍然要做好决策，而不是放弃。这时，人将在一定的约束条件下（如，时间和记忆）做机遇性搜索，以制定尽可能好的决策。这样的决策制定具有一定的机遇性，往往不是最优的。当然，由于计算机系统在结构上与人脑大不一样，所以，也可能会使用与人不一样的约束条件。

物理符号系统假设认为，物理符号系统对于智能行为是必要而充分的。这是纽厄尔和西蒙于上世纪 70 年代提出的，他们也因此获得了图灵(Turing)奖。所谓“物理符号”，意指可识别的模式；任何一种模式，只要它能够和其他模式相区别，它就是一个符号；物理符号系统即是处理这些模式的系统；而所谓符号处理，则主要包括符号和符号表达式的创造、修改、复制和删除等。关于该假设的充分性已为人工智能的实践和成就所证实，但其必要性目前尚不能证明，所以只能称为假设。

符号主义曾是人工智能的主流学派，它是从心理、认知、知识和思维层面对人类智能行为所进行的功能模拟。符号主义认为，知识和思维是一切智能系统的基础，是对信息进行恰当处理的思维基础，也是构成智能的基础；认知主要就是处理符号，思维和推理就是采用启发式知识及启发式搜索对问题进行求解的过程；任何智能系统的活动过程，都是一个通过思维获取知识和运用知识的过程。因此，人工智能的核心问题应是：知识获取、知识表示和基于合理[分析与综合]思维的知识运

用。知识可用符号表示，也可在此基础上[用符号]进行[形式]推理，因而我们也就有可能建立起一个基于知识的人工智能系统。符号主义主张用用形式化的方法来描述客观世界，也就是把人类的知识和思维方法用形式化的方法表示成计算机能够理解的符号；它把人的思维和心理过程也看作是一个信息输入、存储、加工和输出的过程，认为此推理过程也可以用某种形式化的语言来描述。也可以说，它所关注的核心，就是**知识的形式化表示和形式化的推理过程**。其中，逻辑学派更是强调逻辑推理的作用，认为模拟人类的智能主要就是模拟人的逻辑思维功能，主张将问题和知识表示成某种逻辑系统，采用符号推演的方法，以实现搜索、推理、学习等功能。过去，人工智能研究中的很多成果，如定理证明、自动推理、专家系统和博弈系统等，就是采用此类方法取得的。

符号主义的重要启示之一还在于，它明确指出，一个“物理符号系统”由什么构成并不重要；一个可处理符号的系统，可以由蛋白质构成，也可以由机械部件、半导体元件或其他材料构成。计算机已具有符号处理的推算能力，这种能力本身就蕴含着演绎推理的能力，因此，可以通过运行相应的程序来体现出某种基于逻辑思维的智能行为。物理符号系统假设实际上也肯定了如下信念：（智能）计算机能够具有类人的智能。

基于符号主义所构建的智能系统，本质上也是一个“形式化系统”。一个形式系统，通常由四个部分组成：① **符号表**[模式集合]，规定系统允许使用的符号；② **形成规则**，即语法，规定符号连接成合法序列的规则；③ **初始公式**，即公理；④ **推理规则**，即符号操作规则，规定怎样将一个合法序列变成另一个合法序列。一个典型的形式系统通常具有以下特点：① 使用专门的人工符号语言；② 除初始概念以外，任何概念都必须由初始或已定义的概念来定义；③ 除初始命题即公理以外，任何断言必须是经过证明的，不许引进初始命题以外的假设作为证明的根据。其实，基于冯•诺伊曼体系结构的计算机系统，本质上就是一个形式化系统，而其程序的设计和运行，就是一种“形式逻辑”的活动。但是，要用一个形式系统来模拟人类的智能，它必将会面临以下的问题：

（1）人的全部智能行为是否都可以形式化的。由于计算机系统是一个形式系统，任何一个问题要在计算机上进行处理，都必须先进行形式化，任何不能形式化的内容，计算机都无法处理，因此，可形式化的界限应是计算机信息处理的第一界限。**形式化的抽象就是语言化**。由于求解问题的过程就是人类思考的过程，所以，这个问题也可以转化为人的思维是否可以全部用语言来表达的问题。思维科学的研究表明，抽象思维靠语言，它可以形式化，而直觉思维、形象思维和灵感思维并不一定依靠语言，因而也就不能完全形式化。目前的计算机，在数值计算、定理证明等方面可表现出优秀的性能，但它目前还很难具备直觉思维、形象思维和灵感思维的能力，就是因为直觉思维、形象思维和灵感思维目前还是很难形式化的。更进一步，如何实现从非形式化领域向形式化领域的转变？如果让计算机来完成这一转变，就得把这个转变形式化，那么，这一转变的起点又在哪里？这就造成了一种回归现象。要避免这种回归，就必须假设有一种包罗万象的先验的形式化系统。然而，形式化方法属于人类抽象思维的范畴，先验的形式化系统是不存在的。所以，我们可以断定，**人类的智能行为目前是不可能全部被形式化的**。

（2）可形式化的问题是否都是可计算的。一个可形式化问题如果是不可计算的，同样也是不能用现有计算机来求解的，可计算性应是计算机信息处理的第二界限。彭罗斯曾使用哥德尔不完备定理论述了意识的不可计算性。彭罗斯指出，每一数学体系都是不完善的；每一数学体系里必然存在的矛盾命题可以通过“直觉”把它变成一个新公理，从而构成一个新数学体系。对于一个无法在一个形式系统中用数学公理规则体系去决定其真伪的命题，人类可以由“直觉”定义它是真或者假，

然后把它作为一个公理加在原来的体系里，并形成一个新的体系。由于计算机不具有“直觉”，它也就无法实现形式系统的自我完备。可见，**即使问题是可形式化的，也不一定是可以在计算机上求解的。**

（3）可计算问题是否现有机器（计算机）可解问题。即使是可形式化的可计算问题，还须区分这个问题是不是现有计算机可解问题，这应是计算机信息处理的第三界限。通常，可计算问题可分为两类：第一类问题的求解只需要低次多项式的时间，如有序检索和分类的计算时间复杂度分别为O(logn)和O(nlogn)；第二类问题是包括那些迄今为止已知的最好算法所需时间也为非多项式时间的问题，如货郎担问题和背包问题的时间复杂度分别为$O(n^2 2^n)$和$O(2^{n/2})$。对于第二类问题，由于算法的执行所需要的时间和空间会随n的增大而急剧增加，就有可能导致即使是一个中等规模的问题现有计算机也不一定能解出。

从上面的分析我们可以看出，若一个问题在计算机上是可解的，首先必须是可形式化的，此可形式化的问题还必须是可计算的，此可计算的问题还必须是一个有合理的计算复杂度的：

{问题}=>{可形式化问题}=>{可计算问题}=>{非计算机难解问题}

因此，基于现有计算机和形式化方法的问题求解系统可以求解的问题，只能是问题世界的一部分。在所有可解的问题中，只有一部分是数学上可知的，而数学上可知的问题中，又只有一部分是用形式系统可有效处理的。

但符号主义主张，作为一门尚处于探索阶段的学科，智能模拟研究所遵循的，应是**“假设-检验”**的研究范式。计算机就是其实验室，而实验则是设计、构造、测试和评价可在计算机上运行的智能行为模型和执行机制等。尽管基于符号主义的智能模拟研究遇到了种种困难，但是这些困难是可以在探索中设法克服的。比如，对于计算机难解问题，符号主义所主张的一个基本原则是：**以搜索补偿知识的不足，以知识补偿搜索的不足**。知识工程和专家系统技术的开发，已证明了知识可以指导搜索过程，修剪不合理的搜索分支，从而可缩减问题求解的不确定性，可大幅度减少状态空间的搜索量，甚至可以完全免除搜索的需要。而当遇到从未经历过的问题又缺乏经验知识，因而不能快速解决它时，也可以和人一样采用**“尝试-错误”**方法，就是凭借人已掌握的常识性知识和领域的基本原理对问题作试探性求解，逐步解决问题，直到成功。尝试-错误的方法就是问题求解系统中基本策略的生成-测试法（generate-and-test），可用以指导在问题状态空间中搜索。由此可见，将知识运用和搜索策略运用相结合，二者相辅相成，再根据实际情况做出权衡，就可起到很好的效果。例如，国际象棋大师在下棋时很少会检查多于200个棋盘状态，但却有丰富的下棋经验知识。而一些下棋程序可检验2千万个棋盘状态，但却只拥有有限的200条规则。显然程序是以大的搜索量来弥补其经验的不足，而人却是以经验知识来弥补其搜索量的不足。

符号主义十分重视知识在智能系统和问题求解过程中的作用，这无疑是对的。但是，用基于知识的系统来模拟人的智能行为，来解决现实中的问题，也面临诸多问题。**一是，知识的获取问题。**人工智能的专家们一直希望通过构建巨大的数据库和知识库来建立智能系统。比如，莱纳特等人所筹建的CYC(CYC就取自encyclopedia的一部分)海量知识库就曾是这一思想的体现。CYC知识库是希望通过不断增加关于知识的事实来实现人类级的智能。但是，人类知识是博大精深的，是包罗万象的，包括着显性知识和隐性知识，专业知识和常识性知识。如何去获取这些知识，曾经是一个难题。另外，CYC知识库本质上是经验事实的总结，而并非可推演的理论体系。至于常识（指人们在日常生活中运用的知识，特别是那些众所周知的不言自明的知识，它是一类人们在日常生活中都在使用但又

无须明显表达出来的客观知识）因不受专业限定，数量十分庞大，人类专家的知识又是以拥有大量的常识为基础的，如何选取它们也曾是一个难题。**二是知识的形式化表示问题**。尽管人们已经提出了多种知识表示方法，但是，就目前的技术而言，要把人类的知识全部进行形式化表示，显然还是不可能的。**三是知识的灵活运用问题**。知识是智能系统的基础，知识越多系统应该越聪明。但知识多了，也将面临着推理、搜索、计算和运用时的“组合爆炸”问题。在知识系统中，随着规则和知识数目的不断增大，其组合数目将呈几何级数增长，当这种组合表达式的相互牵连达到一定规模时，用现有的计算机将很难处理。如今，随着GPT等大模型的开发，上述问题大多已不再是难题。

不过，建立在物理符号系统假设之上的符号主义方法，目前还是存在一些更深层次的问题的。比如，它还难以获取隐性知识；它还很难解决我们日常[现实]认识中因情境不同而意义不同的问题；也难以处理人所具有的情感、动机、意向性等心理活动问题。一方面，人类的知识中确实存在着一类无法用自然语言清晰描述的知识，例如与人的心理状态相关的知识；而另一方面，人类知识的运用又具有极其复杂的社会文化背景和问题情景相关性。如果人工智能的目标是要使机器能够与人进行成功的交际，那么，其研究就必须使机器能够跨越纯形式逻辑的界限而进行“类人”的自然语言的理解和自然的推理。尽管如今的GPT等大模型已经可以处理诸如自然语言理解和常识推理等，但知识运用时的社会文化相关性和问题情境相关性确实是人工智能很难处理的；因为连人类自身都无法完全把握、无法捉摸透的现象，仅靠机器更是难以办到的。单纯的形式化信息处理系统是难以应付现实世界中复杂多变的环境的。任何语言和知识都永远只是社会文化的一部分。没有了相关社会文化背景和问题情境相关性，知识也就失去了它的现实意义。显然，人类智能，所依靠的不仅仅是知识，语境、背景和文化，都是其不可或缺的一部分。也许正如明斯基所说，真正的“认知和智能活动不是由建基在公理上的数学运算所能统一描述的。因此，要在认知科学领域有实质性的突破，就应当放弃唯理主义的哲学，从生物学中去得到启示和线索。”更何况，人类认知本身就是多模态的。

为了解决上述局限性和存在的问题，人们试图用神经网络模拟来弥补物理符号系统的不足。但说到底，符号系统的根本缺陷是它无法模拟人类心理层次的（全部）意识活动，这一方面是由于心理活动的研究尚处在研究的初期，许多方面还不成熟，因而缺乏研究的基础。另一方面，也是因为心理层次的活动是不确定的、变化的。于是，人们才试图对人类生理层面的机制进行模拟，甚至试图深入到物理层次去。

假定我们把认知系统看作是具有高低不等层次的系统：一部分是高层次的意识行为，它们可以用符号系统来进行模拟；另一部分是较低层次的心理过程，它们是由那些相对不确定的心理意识所构成。从可能性来说，也许它们都可以转换成物理符号系统，但难点仅在于，它们之间存在着意识和认知方面的相对性和主观性。由于生理、心理及高层次的意识之间，是一个相互支持的统一体，它们之间是有机地联系在一起的，因此，分别对不同层次的意识状况进行模拟而忽视各层次之间的相互联系，显然是无法取得模拟的成功的。认知系统也许最终可以还原成一个物理符号系统，但是，这种还原也必然是多层次的，需要通过不同层次之间的交互才能逐步完成。因此，要想理解人的认知过程，只有符号处理的机制显然是不够的，我们还必须加上对其他层次信息处理机制的认知。

17.1.6.2 基于联接主义的方法与途径

联接主义(Connectionism)学派，又称为连接主义学派、仿生学学派(Bionicsism)或生理学学派(Physiologism)，是智能模拟研究的另一种思路，其基础主要是人脑神经网络及神经网络间的连接机制。它源于仿生学，特别是对人脑模型的研究。基于联接主义的人工神经网络(Artificial Neural

Network)就是模仿人脑的生理结构和工作机理，利用多个简单的信息处理单元彼此按照某一种方式相互连接而成的信息处理系统。因为人脑的基础就是由神经元组成的神经网络，所以，该研究途径就试图用人工神经元组成的人工神经网络模型来存储信息和知识，用神经计算的方法来实现学习、联想、识别和推理等功能。如果说符号主义的形式化方法主要是从心理活动和思维功能上来模拟人的智能，联接主义则主要是从生理结构和信息处理功能上模拟来人的智能。

在联接主义的智能模拟研究中，人们是从模拟生物大脑的生理结构入手来达到智能模拟的目的。联接主义的灵感主要来自大脑或神经系统，当然，它的目标并不是要建立一个活的人工大脑，而是以类似于脑的神经网络的系统建立认知活动的模型。联接主义不再把认知解释成**符号运算**，而是看成网络的整体活动,或称**神经计算**。对于人工神经网络来说,它的强项也不再是逻辑推理和知识表达，而是进行较低层次的认知功能研究,包括分类识别和聚类等。**联接主义所研究的，大多是非程序的、适应性的、大脑风格的信息处理，研究它们的机理、模型和算法等**。经过多年来的发展，基于联接主义的人工神经网络模型和算法不断地涌现，已给出了众多具有不同实用价值的神经网络模型。这些模型和方法的优势特征主要表现在：**以分布式的方式存储信息，以并行方式处理信息，具有自组织、自学习能力，适合于模拟人的形象思维（及抽象思维），可以比较快地得到一个近似解**。正是由于这些特点，使得神经网络已成为人们利用计算机处理信息的一个全新的方法和途径。如今，人工神经网络已经成功地应用于智能系统研究的各个方面，例如，自然语言处理、图像数据压缩、图像及声音等信息识别、宏观经济运行模拟等。由其发展而来的**计算智能**，曾包括进化计算和模糊计算等，如今，更成为各类智能系统信息处理的基础，应用于感知智能、认知智能和行为智能。

基于联接主义的智能模拟研究克服了符号主义方法要求知识和规则必须显式表达的局限性，具有多方面的优势。但是，用其来模拟人类的智能行为，也曾存在不少困难。这主要表现在：

（1）用软件或硬件来模拟人脑生理结构及其信息处理机制还比较困难。神经生理学的研究表明,人的大脑是由大约 10^{11} 个神经元组成，每个神经元约有 10^3-10^4 个突触。我们可以把一个神经回路比作一个计算网络，一个神经元比作一个集成电路，一个突触比作线路中的一个接点。但事实上，人脑的结构要远比现有计算机复杂。人脑神经网络的复杂性，不仅在于神经元和突触的数量大、组合方式复杂和联系广泛，还在于突触传递的机制复杂。比如，突触的功能既可以传递神经冲动，又可以记忆学习的影响；既可以产生兴奋，又可以产生抑制或者疲劳；远不是一个电位开关就能模拟的。在突触传递机制中，释放神经递质是实现突触传递机能的中心环节，而不同的神经递质有着不同的作用性质和特性。神经元之间的组合也很复杂,有环形组合,也有链形组合,其拓扑结构至今还很难搞清楚。从处理方式上看，大脑的每个神经元都是一个信息单元，他们的处理是并行执行的。而现有的计算机是基于冯·诺伊曼体系结构的，是典型的串行处理方式，即使是内含并行处理系统，其最基本的处理单元的执行过程也还是串行的。所以，用现有的计算机来模拟人的大脑，目前显然还很困难。有 GPU 芯片会好一些，但还不够。

（2）过去的算法和模型也制约着并行计算技术的发展。人脑是由神经元组成的并行处理网络，计算机要模仿人脑就必须解决并行处理问题。从目前计算机的发展来看，尽管并行计算已经发展到可观的速度，但并行信息处理技术仍然处于相对落后的境地，远未达到真正神经计算的需求。并行信息处理技术实用化的关键是并行程序方法学，而并行程序设计方法的核心内容是并行划分和算法映射。在算法映射中不仅要考虑进程到处理机的静态映像，还需要考虑进程调度问题，而进程调度问题已被证明是 NP 难题。

从模型方面考虑，过去的神经网络模型，无论是Hopfield模型还是B-P模型，都还不是神经计算的理想模型。比如，曾经的Hopfield模型的动力学机制过于简单，只有不动点吸引子，本质上是一种消极、被动的神经网络；目前还难以找到其通用的学习算法，其解题的能力还主要源于其连接权的精巧设计。而B-P网络也同样存在着从问题中选取典型实例组成训练集的困难；其学习过程本质上是一个对高度非线性的函数求全局最优解的过程，有可能会落入“局部最优陷阱”；另外，网络只是在学习一些事例，从中可能难以直接提取出求解问题的合理规则。

由于在软件模拟和硬件实现方面都遇到了一些实际的困难，至今，真正符合人脑信息处理特性的神经网络软、硬件模型一直未能问世，也使得神经计算的研究一直在“突破”中进展。如今，人们普遍看好基于 Transformer架构的各种神经计算方法，我们也十分赞同。我们曾提出模糊神经网络和因素神经网络，并曾看好其发展前景。对此，我们将在适当时机进行全面论述。

（3）联接主义也无法完全回避形式化问题。即使是上面所讨论的硬件和软件技术都已解决，基于联接主义的智能模拟依然无法完全回避形式化问题。神经网络模型并不能完全避开表征问题。由于目前在硬件上还不可能模拟人的大脑，所以，要想使现有计算机达到模拟人的思维的目的，就必须做并行程序设计，而程序设计就是一个形式化问题。姑且不说并行程序设计方法的瓶颈是否能够打通，软件实现说到底也还是一个形式化问题，这就又回到了形式化方法所存在的局限性。所以，联接主义也面临许多与符号主义同样需要解决的问题。

用神经网络模拟人类智能显然需要探讨人脑生理层次上的活动是如何支持人类高层次的认知功能的。有研究认为，为了支持人脑的高层次认知功能，一个神经网络系统必须具备一些最低限度的功能，如：模式联想、函数逼近、上下通讯、串并行交换等。然而，基于神经网络的模拟是否真的能具备处理日常化[或现实化]认知的能力呢？如果不具备，那么，它将会与物理符号系统一样步履维艰，因为日常化认知问题已经成为阻碍人工智能发展的最大障碍；也正是因为这一点，人们才考虑要选择神经网络模拟。而要使神经网络的行为模拟显得和我们人类一样，可对现实中的语境或行为环境加以识别并进行恰当的概括，那么，就必须使神经网络系统具有和人一样的概括能力。而要具备概括能力，网络的设计就必须设计出相关语境或相关情景，以使系统能适应某些语境或情景的常识。而要真正能像人一样具有按不同语境或情景做出不同判断的能力，系统就得具有像人一样的学习能力。

由此看来，由于对人脑的生理结构和运行机制的认识正在逐步深入，在神经网络的模拟方面，从理论上讲，我们迟早会形成比较具体而精确的认知。但是，对符号系统与神经网络系统两者之间的结合，则还有待进一步地深入研究。因为，它们任何一种独立的研究都无法取得智能模拟本质上的进一步突破，二者如今应该是相互依赖的关系。**自然化与日常化[现实化]**，也许是认知研究中的一对基本矛盾，这对矛盾的解决或许是解决其它矛盾的前提条件，也是智能系统研究取得进步的基础。因为，认知的自然化是人工智能功能模拟的基础，也是日常化的前提，而对日常化认知的模拟又是建立在自然化的基础上的。因此，只有研究和掌握了自然化的认知问题，方能进一步研究与掌握日常化的认知问题。为了同时解决这些现实问题，我们也就有必要去深入探讨在不同认知层次上的模拟问题了。

基于 Transformer 架构的大（语言）模型的成功开发，无疑让人们看到了基于联接主义的信息处理方法与基于符号主义的信息处理方法合作的途径和威力。大模型信息处理的基础无疑是“神经网络”，是基于联接主义的；但它处理的信息，包括符号，也包括亚符号，且是基于“自然语言”的。

17.1.6.3 基于生态进化主义的方法与途径

生态进化主义学派，又称进化主义(Evolutionism)学派、行为主义(Actionism)学派或控制论(Cyberneticsism)学派，其理论基础是进化论、控制论以及行为主义的认知理论。他们认为，**智能源于认知与控制，智能取决于感知和行动，智能只能在与环境的交互作用中才会表现出来，行为表达着智能**。**生态进化主义**学派更偏向于从行为、进化与交互的角度去理解智能，认为，**一个系统之所以具有智能，是因为它从行为方面来看“像是”具有智能**。因此，它主张在内在发育程序的指导下，在与环境的交互中，用控制进化的方法，使一个类似人脑的机器或系统“进化”出人在控制行为过程中所表现出来的智能行为和行为特性（自寻优，自适应，自学习）等。生态进化主义强调“现场认知”，即认为智能系统或智能机器的智能只能在与环境交互中学会或进化出，在与环境的交互中表现出智能。与符号主义不同的是，他们更注重自下而上的智能系统的设计，认为可以把智能看成是一种在复杂系统中涌现出来的现象。

我们认为，生态进化主义的思想是可取的，但是，其现有的一些提法和模拟方法，和符号系统一样，也具有明显的局限性。比如，他们认为智能行为不需要理性的知识，Brooks 等更提出“无需知识表示的智能”，“无需推理的智能”；这在许多方面显然是行为主义心理学的观点在现代智能系统研究中的反映。而行为主义尽管可以解释动物的某些行为方式，却不是可以深入阐述人类复杂智能行为的科学理论。人类的智能行为，需要学习，需要“进化”，但更需要“理性”。

我们认为，宏观而又复杂的人类认知系统是一种客观的现象，要对这样的复杂系统进行描述和模拟，一方面需要“自下而上”的对系统涌现行为的理解，另一方面也需要进行宏观方面的对复杂系统的把握和控制。两者的恰当结合应是我们认识复杂事物的可靠途径。**而将微观的涌现和宏观的控制相结合，也应是构造智能系统的恰当方法。一方面，我们可以用涌现和进化的思想来研究一个智能系统的进化，另一方面，也需要依据人类个体思维发展的机制，将“自下而上”的涌现机制与“自上而下”的控制机制相结合。**比如，可先针对某个具体问题设计多种微观的涌现机制，看看这些机制到底能涌现出什么现象，并对涌现出来的宏观现象进行相关的模式辨识及宏观的逻辑推理，从而对正确的涌现机制进行确认，而后再运用反馈控制机制，对微观中的相互作用机制进行调整，直到最后找到满意的行为规则为止。

我们知道，规则对人具有重要的指导意义，规则导致了人的“理性”行为。但是，人的行为也并不完全受规则的指导。一种行为是否遵循规则，必须要看规则的意义是否与人的心理需要相一致，是否在行为中有着必然的因果联系。实际上，人在行为过程中，有很多时候并没有完全按照规则，至少是不完全遵守规则的。“我知道人类遵守着规则，计算机也遵守着规则。但这两者之间存在着一种根本性的区别。就人类方面来说，每当我们遵守一条规则，我们就是在接受某种实际内容或规则意义的指导。就人类遵守规则而言，是意义导致了行为。”更何况，在现实中，指导人的行为的，有欲望，有信念，也有“想当然”等。

17.1.6.4 关于符号智能、计算智能与（社会）集成智能

上面，我们仅仅是从符号主义、联接主义和生态进化主义等 3 个学派的角度说明了智能模拟的方法和途径，然而，还有很多方面我们不能忽略。

从智能模拟和工程化的内容来看，智能模拟或工程化的研究可认为主要包括着下述 4 个方面的内容：

（1）**思维机制模拟与思维模拟系统的研究**。其中,对思维机制的研究包括对启发式程序、专家解

决问题的方式、知识基系统的运行机制、机器证明与机器博弈的机理等的研究。而对思维系统的研究则包括对智能机、学习机、推理机、自动机和神经网络系统的研究等。而对这些方面的研究，其核心是“**认知智能**”

（2）**感知机制模拟与感知模拟系统的研究**。对感知机理的研究包括对文字、图像、物景、声音等的识别机理的研究和对自然语言理解的研究,对计算机视觉、听觉、触觉的研究等；对感知模拟系统的研究则包括文字、图像、声音、语言的识别系统的开发，对各类感知机的开发和各种智能传感器的研制等。而对这些方面的研究，其核心是“**感知智能**”。

（3）**行为机制模拟与行为模拟系统的研究**。对行为的模拟研究包括对自适应、自寻优等智能控制的研究，对管理和决策行为的模拟研究，以及机器人在不确定的、动态的环境中的行为研究等。而对行为模拟系统的开发则包括对各类智能控制器、智能执行机构、智能机械手、智能机器人等的研制等。而对这些方面的研究，其核心是“**行为智能**”。

（4）**学习机制与学习系统的研究**。对学习机制的研究包括对符号学习、经验学习、统计学习和进化学习等的研究以及对知识发现的研究等。对学习系统的研究则渗透到各类智能系统之中，学习是智能系统具有智能的核心，无学习能力的“智能系统”最终会失去智能。而对这些方面的研究，其核心是“**进化智能**”。

从智能模拟和工程化的层次来看，智能模拟和工程化有**生理模式、心理模式、行为模式和社会模式**。生理模式的基础是对脑神经系统的生理结构和生理功能的模拟和神经计算，是通过构建各种类型的神经网络系统来实现各种信息处理功能。心理模式的基础是对人类心理进行解析的各类心智模型。若神经系统是脑的硬件，则心智可以看成是脑的软件。心智的核心是信念[知识]和思维，也包括情感，我们现在所能考虑的主要是对理性认知或称自然化的认知的模拟，其核心是知识的表达与思维功能的实现研究等。行为模式是对智能在行为层次上的模拟，其特性包括自学习、自适应、自寻优、自调整等，其主要表现是对感知信息的恰当反应。社会模式是对智能在社会层面上的模拟，它所考虑的主要是多个智能体间的交流与协作何以能产生出智慧，其典型系统是多智能体系统。智能体是某种能够理性行动的系统，可自主控制自己的操作，可感知环境，可适应变化，有能力承担其它智能体提出的任务，可通过自己的行动获得一定的信息处理结果。这里，理性的标准有着清楚而普遍的定义，它比建立在人类行为或思维基础上的方法更经得起科学发展的检验。人类的行为可以很好地适应特定的环境，且部分地取决于目前还未完全认知的社会进化过程。于是，构建理性智能体的原则是：在相当长的一段时间内，实现完美的理性在复杂的环境下是不可行的，但它应是研究问题的出发点；智能体应是有限理性的--在没有足够的计算时间的前提下，它会按照自己的信念采取正确的行动。

从智能模拟和工程化所采用的方法来看，智能模拟和工程化的研究也可归结为对**符号智能、计算智能和（社会）集成智能**等的研究。

符号智能研究的一个基本理念就是：“**智能即符号加工**”。智能产生于符号运算，智能可通过符号演算模拟，智能行为应是符号运算的结果。作为问题求解过程的人类思维，本质上就是一个“符号操作系统”或“信息加工系统”；通过构造一个与人类求解问题过程相类似的“符号操作系统”，即可模拟人类问题求解的功能。

计算智能研究的基本理念是：“**智能即计算**”。它包括着神经计算（Neural Computation，NC）、进化计算（Evolutionary Computation，EC）、免疫计算（immune computation）和自然计算（Natural

Computation，NC）等。计算智能以信息处理为基础，主要通过数字化计算，运用算法来进行问题求解。符号智能研究的停滞不前，为计算智能的蓬勃发展提供了发展的机会，它们用各种模型计算来模拟各种自然智能机制，取得了不少成功。计算智能的成功说明，**智能不仅表现在人脑的思维活动中，也广泛存在于生命活动之中**。

（社会）集成智能研究的一个基本理念是：**“智能即协同与涌现”**，智能行为可通过主体或群体的协同作用而表现出来。其典型的研究包括多智能体系统、蚁群算法和生态平衡等。多智能体系统也可看作是基于符号智能和计算智能的智能体的综合，是以符号智能和计算智能为基础的更高一级的智能系统。

从智能模拟和工程化所选用的工程系统的角度看，智能模拟和工程化研究可分**智能平台和智能应用系统**两大类。智能平台的研究包括对智能硬件平台、智能操作系统、智能网络系统等的研究。智能应用系统的研究包括基于知识的智能系统、基于算法的智能系统和兼有知识和算法的智能系统等。另外，还有分布式智能系统。

从智能模拟和工程化的基础理论的研究角度看，智能模拟和工程化研究还包括着对各种数理逻辑的研究和对各种非标准逻辑的研究。如模型论、公理集合论、递归论、证明论、模糊逻辑、粗糙集理论、贝叶斯统计决策理论、支持向量机、形式语言和自动机理论等。它们构成了智能模拟和工程化的（数学）理论基础。

我们认为，无论从哪个角度看，智能模拟和工程化的研究都是复杂而艰巨的，也是需要我们付出巨大的努力的。智能的模拟和工程化，是个长期的过程。我们不应期望会有轻而易举就可以实现的“突破性”进展，因为这是对人类智慧最直接的挑战，也是人类历史上最艰难的“长征”。

17.1.7 智能模拟和工程化研究——现状、挑战与未来

智能模拟和工程化的研究经过半个多世纪的发展，尽管其研究发展的历史曾充满了风雨和曲折，但仍然取得丰硕的成果，从而显示出了其所具有的强大的理论价值和应用潜力。智能模拟和工程化的研究已受到学术、技术、工业和商业等领域各阶层人士的关心和重视。纵观其发展的历程，其研究和发展所遵循的是典型的**“波浪式前进，螺旋式上升”**的模式。也正是由于有众多研究者的不懈努力，才开创出了智能科学和技术如今“辉煌”的成果和无限美好的未来。

智能模拟和工程化的研究是一个充满挑战性的研究领域，它企望用计算机（智能机器）来模拟人类的推理、记忆、学习、创造等智能特征，试图赋予机器以人类的智慧。随着智能技术、计算机技术和网络技术的飞速发展，许多实际系统已变得越来越复杂，智能系统研究所面临的困难也越来越多。因此，智能模拟的研究还需要有不断创新的思想，智能系统的研究开发也还需要有不断创新的模式。

目前，**符号主义**和**联接主义**已是智能技术研究中比较成熟的两种方法，但这两种方法在揭示人类智能的奥秘和模仿人类智能的方法上也都遇到过许多无法克服的困难。究其原因，是因为智能技术的根本使命是“用‘电路’复制大脑所进行的信息加工”、“由‘机器’来实现为人所独有的思维”，一句话，是用（智能）机器来模拟人脑的思维功能，而这是很艰巨而困难的。按照这种“模拟思维”或称“认知模拟”的指导理论和历史使命，势必要迫使人工智能的探索者们要同时面对两个在可以预见的未来还无法获得准确解答的世纪之谜：**人脑思维功能的宏观和微观机制究竟是什么？我们又如何用机器来模拟它？**正因为如此，“认知模拟”实现的前景曾经显得十分“渺茫”，甚至会令一些人感到“绝望”；而每一次的“突破”也会“令人感到惊喜”。比如，美国的机器翻译专家巴•希莱

尔曾断言:“全自动高质量的机器翻译是不可实现的,不是不久的将来不能实现,而是永远不能实现”。可惜它的断言已过时了。“机器翻译”，基于“大语言模型”，现在已经达到了我们可以接收的程度。但是，以“认知模拟”作为指导方针的人工智能，藉助于“建立在布尔代数之上的数字计算机”，在对人类智能本来驾轻就熟的语言翻译领域的成功,并不意味着它在日常生活领域已经达到了具有“认知智能”的程度，更别说在“探隐索微”，在从已知进入新知的发明创造领域了。由此，我们也就不难理解，智能模拟和工程化研究目前的主要任务或发展方向仍然是两个，**一个是要从生理和心理角度研究人类思维的本质，以达到对人类思维的彻底了解；另一个才是研制越来越复杂的智能化信息处理系统——（类人）智能机器，以帮助人们摆脱繁琐的信息处理工作**。至于将来会不会研制出能够真正像人类一样思维的机器（或者超越人类智慧的机器），这还不是一个现在就可以下定论的哲学问题，而是一个尚需艰苦探索的科学问题。因为智能模拟和工程化所能达到的水平，所反应的其实是人类智慧所能达到的水平。**如果我们相信人类无限的创造力，也就会对此充满信心。而人造智能系统的“无能”，也只能说明人类的“无能”**。而通过什么途径来更有效地实现智能的模拟和工程化，是符号演绎推理？知识工程？神经网络计算？还是不确定推理等？其实只是一个需要在探索的过程中去不断检验的问题。

关于智能模拟和工程化需要面对的一些基本性问题，曾引起了人们广泛的争论。比如，**知识与概念化是否是人工智能的核心？认知能力能否与载体分开来研究？认知的轨迹是否可用类自然语言来描述？学习能力能否与认知分开来研究？所有的认知是否有一种统一的结构？人在“信息加工”时是否真的像计算机那样遵循形式化原则？人类行为，无论是如何生成的，是否都能描述为一种可由机器实现的形式化系统？**等等，都曾是有争论的议题。而这些争论的结果，也曾导致了各学派的联合或部分联合，或者产生出了新的学派。即使是在人工智能蓬勃发展的今天，人工智能至今还是探索性学科，因而也会是引起争议最多的学科。它的研究还需要不断的反思。但也正因为如此，它的前途才是最光明的，才是最充满活力的。

不可否认，无论在理论方面还是在技术上，人工智能都还有许多需要突破的根本性的障碍。它们包括：

（1）人工智能体可否具有意识？人工智能希望让计算机（智能机器）在许多方面具有人脑的功能，而意识应是人脑的最主要的功能。认知科学认为，“意识是用符号形式使实在世界在我们心中内在化的产物”。人可在意识的指导下，在超越本能的制约下，按照美的规律来塑造、引导世界的进化。当前，人工智能系统只是人们利用自己的意识主导作用，将大脑的某些功能借助外在工具进行某些形式的发展，是按照人们的意愿和要求来设计和开发的。人可做自己想做的事情，而人工智能系统只能按照人们的意愿做人们要它做的事情。于是，一个不是按自己意愿做事的系统算不算有智能？如果一个程序能检查自己解决问题的行为，修改和改进它，是否等于它有意识？就成为了一个需认真思考的问题。

我们认为：就人工智能的本质来看，它是认识主体创造的，是主体的目的和需求的产物。但就人工智能在认识活动中的作用形式而言，它是人脑的延伸物，属于主体的认识手段或认识工具，是主体认识能力的表现。目前，**人工智能的客体地位是绝对的，主体作用是相对的**。从本质上讲，再高级的电脑也是人设计制造的，目前还不可能完全代替人脑而成为完全独立的认识主体。现有的电脑，不但不具有人的意识，甚至连最原始的生物的刺激-感应能力也不具备。因为现有“电脑”从其物质发展水平说，还达不到生物运动这一物质发展的高级形态。通常人们所说的“机器思维”，只不

过是人类借助电脑来模拟和复制的人脑的某些功能而已。基于机器思维的人工智能与基于人类思维的人类智能，两者之间具有本质区别：人类认识世界和改造世界的活动是有目的的、能动的，在与外部环境的物质、能量和信息交换过程中，能够根据环境的变化不断调整自身，具有适应性。而现有的人工智能则还是无意识，没有主观能动性的；只能按照人们为它制定的程序运行，机械地模拟人的智力活动，却毫不理解这些活动的意义，更不会去提出新问题、研究新问题、解决新问题等。因此，我们可以认为，尽管 IBM 的“深蓝”计算机依靠众多专家事先编制的程序以及每秒上亿次的运算能力战胜了国际象棋大师卡斯帕罗夫，但这样的胜利与其说是电脑的胜利，不如说是众多程序设计员集体智慧的胜利（当然，随着智能技术的发展，在像下棋这类智能游戏中，人脑最终还是敌不过“电脑”的）；即使现有的大模型已经“**无所不知**”，也不过是**系统开发者**和**人类智慧**的**结晶**。另外，人类智能或人类的认识能力，只是人类意识的一部分。人是社会的人，人的认识的产生和形成不只是人的认识能力所致，还包括情感、情绪、意志、性格等因素的综合作用。人工智能则只是对人的认识能力的一部分——逻辑的或理性的思维的模拟，还不具备其他因素。当然，我们在这里并没有要否定人工智能研究的作用和意义的意思。首先，人工智能与人类智能作为两种不同类型的智能，目前还具有强烈的互补性：人类智能的局限性正是人工智能的优越性所在，人工智能的局限性正是人类智能的优越性所在，二者是可以互补互动的。而人类发明智能计算机的动因，也正是基于对人脑的一些局限性的认识，以及在科学研究与生产实践中，解决用人力很难解决的问题的迫切需要。人工智能的产生和发展为人类智能提供了新的时间和空间尺度，给人类提供了一个新的可拓展自己的创新和创造领域，其意义无疑是巨大的。其次，随着技术的进步，人工智能研究的深度和广度正在不断扩展，许多原本是人类思维独占的领域，已有人工智能的介入，如专家系统、模式识别、定理证明、问题求解、自然语言理解，特别是如今的大模型等等。因此，如果人为的为人工智能的发展确定一个限度，比如认为人工智能不可能完全代替人的思维，也是没有根据的。不错，真正的智能要求软件应在没有人类干预的情况下具有优化、扩展和改变自己已有程序和创建新程序的能力，即具有我们常说的**主观能动性**，目前的技术要完全做到这一点还比较难，但也并非完全做不到，包括情感计算在内。力图使人工智能达到人类智能的目标，应是人工智能致力突破的方向，也应是人类认识自身的“最后的方式”。

（2）机器可否实现人类的记忆方式？电脑技术的高速发展并未导致电脑在智能化方面有什么重大的进展，其中的一个重要原因就是（现有）电脑的记忆方式一直停留在它的初始阶段。记忆通常包括两个部分：一是记录所接受或感受到的信息，这主要是指外部进来的信息；二是自动记录主体自身的活动过程。目前，电脑主要记忆前者，而很少记忆后者。而人脑是两者都可以做到。电脑的记忆过程是被动地执行指令，它所能记住的东西仅仅是工作所需的程序和要处理的数据。而人脑所记录的东西不仅仅是感受到的信息，而且最重要的是能够记录处理信息的过程，或者说能够记录大脑自身有意识的活动内容。记忆内容第二部分所指的过程是自动的、不受控的，而第一部分则是可以被控制的。

利用已有的经验来解决新的问题需要归纳和推理。人的这种能力是由人脑的记忆构造决定的。有人认为，人脑在发育的早期阶段，记忆过程主要是**素材和基本经验“块堆”的建立和积累**，即**机械记忆**。人脑在成熟阶段的记忆过程主要是**经验“块堆”的关联和重组，即关联记忆**。由关联记忆形成的人脑活动使人的思维模式天生具有归纳推理能力。经验的重组也使人可得到新的经验，获得进步。人脑的这种记忆构造的优点是具有模糊识别、联想回忆和记忆修补能力，缺点是老的关联成

分会因打散而消退，即产生忘却。

人记住一张脸也许不比记住一个外语单词更难，而电脑却恰恰相反，它可能宁愿去记住一个城市的电话簿也不愿去记住一张张的脸。电脑几乎完全靠机械式的精确记忆，且不能有意识地去利用归纳性的识记，因而也就无法实现智能所必须的利用自身经验之功能。**电脑虽能“识记”，但不能“经验”**。

电脑在记忆时会每次都把所有的感知信息都记录下来，其记忆数据就象人的笔记本和资料库。而人脑记忆常常是预先已有关于客观世界的某些基本素材和经验，记忆时只需把感知的信息与已有的各种相关素材和经验作**关联记忆**。这里所说的**素材**就是**人对客观世界的最基本的物理感受**。人从开始感知其生存的世界，只需不长的时期就可以得到他一生所需的关于其生存世界的基本感知素材。其他时间的记忆就是把当时感知的信息与这些素材关联，所有的关联所形成的信息网络，就构成了我们大脑的复杂记忆。关联记忆使得我们的记忆空间比机械式的记忆存储空间要小得多，并且存取速度要快得多。

动物也能进行关联记忆，它们的智能不高，是因为它们的记忆只能实现最简单的关联，因而只能进行最低层次的归纳。动物的行为受本能的支配，它们不太理会与本能无关的事物关联。而我们人类的智能，却与我们大脑的记忆特性密切相关，我们大脑有意识的活动在相当程度上也是记忆活动。探索和模拟人脑的记忆是实现人工智能的重要一环，也是电脑模拟或实现人脑智能的必经之路。

区分了两种不同的记忆将有两方面的重要意义：一是，如果我们可以把关于客观世界的基本素材预先赋予电脑，那么电脑的记忆问题就会变为如何让它在记忆中去关联这些素材。让电脑去模拟人脑的记忆功能也就解决了一半。二是，由于关联记忆并不需要记录所有的信息，或者说此时电脑并不需要重新记住已有的素材和经验，这将使记忆所需的空间得以节省，并且可以在记忆的同时实现信息的共享。

目前，是否可以让电脑实现人的记忆方式以及如何让电脑来实现类似人的记忆方式还是个问题。人工神经网络是希望实现这种功能的一种努力，但其功能的模拟还仅是外部行为的，而非内部智能机制的。我们曾希望以**因素藤网**的模式来解决，但其研究也只是开了一个头，问题还未彻底解决。

（3）**如何实现不同层次的识别？**识别是人对感觉的认知和判断能力。识别能力的高低也体现了智能水平。人类的识别由低到高分为三个层次：物理识别（仪器水平）、情景识别（模糊识别，动物水平）和有意义和情感的识别（人类智能水平）。

物理识别是对接受到的信息实现物理、化学和生物学的量化认识。视觉包括明暗、颜色、大小、形状、远近、运动状态等。听觉包括声音大小、频率、方位、波形等。触觉包括温度、导热率、硬度、粘度、大小、形状、受力、活动状态等。嗅觉和味觉包括物质的组成及化学成分等。现代科技与电脑相结合，在物理识别的范围和识别的精度方面早已大大超过人类自身的能力。几乎所有的科学仪器都是用于这类识别。这种识别的特点是识别内容分别独立互不相关，事件具有精确的重复性，无需经验和智能，完全可以程序化。所以它是最低层次的识别。

情景识别是从大量复杂的信息中识别出有用的部分，即对接收的信息与以往的记忆和经验进行关联认识，剔除无关的信息。其中，视觉识别包括在复杂的背景中辨认特定的人和物或以往经历过的人和物。听觉识别包括在嘈杂的背景中辨别出特定的声音，特别是不同人的讲话声。情景识别可解决“有谁、有什么、是谁、是什么”等问题。这种对人来说轻而易举的能力，对电脑来说目前还比较困难，因为它们大多是不精确的模糊识别，识别是需要一定“智能”的（即感知智能和初级认

知智能）。目前，电脑（智能系统）可通过对照记录的方式实现一定程度的单一识别，例如一定范围内的指纹、图形、语音识别等。由于现有的电脑（智能系统）还不具备关联记忆的能力和智能识别的能力，所以，让它在基于模糊识别的基础上实现真正的情景识别，目前还有许多困难需要突破。尽管实现这个层次的识别还只是与我们常见的动物的识别能力大致相当，但它已经让不少现有的电脑（智能系统）很不适应。

在情景识别中，语言识别（包括语音识别和文字识别）是最令人感兴趣的一种。语言是最复杂、最有系统、应用又最广的符号系统。在人类进化的过程中，语言的使用曾使人类的智能产生了突破性的发展。语言符号不仅表示具体的事物、状态或动作，而且也表示特定的意义和情感。汉语更以其独特的词法和句法体系、文字系统和语音声调系统，而显著区别于其他语言，具有**音、形、义**紧密结合的独特风格。让电脑（智能系统）能听懂（或看懂）人的话，无疑是一项重大的突破，将给我们开启人类智慧之门以极好的机会。以前，计算机系统很难做到这一点；如今，**大语言模型**已获得初步成功。

智能模拟和工程化研究中更困难的识别还在于**意义识别和情感识别**。意义识别和情感识别是最高级别的识别。它是一类**完全的“智能”识别**。这类识别首先要求机器要懂得人类的“各种语言”，包括文字、语言、歌曲、表情和动作等所表达的意义和情感，甚至包括自然现象、事件、环境和物品的人文美学含义等。这种识别已经超出了人工智能的模糊识别，达到了人工高级生命的能力。仅以单一语音识别为例：初级识别能够知道有声音，模糊识别能够知道说什么或谁在说话，而高级识别则能通过说话的内容、音调、节奏知道说话者的想法、情绪和态度等。这种识别已是要求电脑具有人类的思维和情感，显然需要**“高级（认知）智能”**。

（4）机器如何区别有意识与无意识的信息处理？人脑的信息处理可分为无意识的和有意识的两大部分。无意识就是“不知不觉”的大脑活动，而有意识就是经过了思维的心理过程。在无意识心理活动中包括着本能，它是人先天就具有的，是生命的最基本的内驱力，它决定着人类智能的生理和心理基础。而另一些无意识行为则是后天形成的。人有一个心理特性，就是不愿“浪费精力”去做重复的事情，它会自发地把重复性的工作和经验转交给“无意识”去支配，以腾出精力处理“新鲜”的事情。如果这种无意识支配的工作继续重复下去，人脑会自发地把这种支配工作转化为“准本能”。为了长期适应这个过程，他还有可能变更自己的“生理状况”，甚至将这些改变遗传给后代。这或许可以部分解释人体结构和大脑进化的过程。

人的感官无时无刻不在接收着外界的信息，每天仅视觉、听觉和触觉的接收量就非常之巨大。人脑对经感官接收到的外界的复杂信息，会自动进行分类和过滤，那些对人的生理和心理平衡不发生影响的信息，通常会被自动滤掉；对其他信息也会尽最大可能用无意识去进行处理；只有部分信息才由大脑作有意识地处理。由此可见，意识只是人脑活动露出海平面的冰山的一角。人脑的大部分活动和由其支配的行为都是与无意识有关的，无意识活动是有意识思维活动的平台。

相比之下，电脑（智能系统）的“感官”即输入端所能接受的信息不仅有限制而且量要相对少得多，它对外界信息输入的限制主要是由传感器和感知程序所确定的，因为其感知系统程序不接受对其“无用”的信息。目前，我们还没有可以模拟人的全部感觉的感知系统或可接受人的全部感受的电脑（智能系统）。由于电脑可直接感受的信息范围和精度可远远强于人类的感官能力，我们可以断定，一旦解决了信息的模拟感知和过滤问题，电脑（智能系统）的感知能力（或**感知智能**）应该能够远远超出人类。尽管认知能力的水平才决定智能的高低，但是，如果感知能力强的话，也会使得认知范围扩大，从而提高智能的水平。

有意识的对信息进行预处理对于智能模拟和工程化是必须的吗？回答应该是肯定的。由于智能电脑必须首先对接受到的由感知系统传递过来的全部信息进行过滤后，剔除此刻不影响“心理”平衡的信息，之后才能作进一步的处理，因此，它必须具有识别有意义信息和无意义信息的能力。另外，我们人类一般不能同时有两个有意识的注意中心，当人的注意力集中于某一处时，其它无关的感受就会被过滤掉。这里有个假设，那就是一个智能主体在一个时刻只能对某一事件进行有意识的注意和操作。如果高级智能系统可以同时对多个事件发生有意识的注意，那它就成了由多个“大脑”合一的“超级大脑”。无论怎样，至少意识对于过滤信息和支配有意识的行为是必要的。但现在的问题是，目前，我们还无法为电脑（智能系统）建立一个（自主）感知和识别系统，使其可模拟人类真实的认知、记忆和识别等过程；更无法为其建立一个（自主）意识系统，使其可有意识地自动做出不同种类的感知和识别等，如何实现（自主）感知智能和认知智能，目前还是一项艰巨的任务。

（5）脑-机一体化集成——是科幻还是未来的发展方向？信息加工理论的提出，使人们得以用信息的观点来认识大脑的运行和作用，从而理解思维产生的机理和过程。但是，大脑是由分子和细胞组成的，它如何使普通的分子和细胞集合体成为了一个思维着的机器（系统）？这一思维着的机器所产生的智能的本质特征是什么？如何判断机器是否具有智能？信息加工理论并未做出完美的解答。Turing 从可计算性理论出发，提出“图灵机”作为实现可计算性理论的通用性机器，认为这一具有学习或自组织能力的机器能模拟人类心灵活动的任何结果。**他认为，可计算的范围远远超过了被明确指令所包含的范围，足以包括人脑能做的具有创造性的事情**。为了说明机器具有智能，Turing 提出，对“机器能否思维”问题，可用另一种形式来提出，即从问“机器能否思维?”转为问“机器能否通过对其智能行为的测试?”这就是著名的“图灵测验”。但是，Wittgenstein 并不同意这种意见。他指出，即使机器能做出正确的行为，在屏幕上给出令人满意的结果，但是，它并不知道那些显示字符的意义，它对什么都不理解；因此，我们并不能以此认为机器有思维。John Searle 更提出了“中文屋”的假想实验，以模拟图灵测验的方式来反驳机器可以具有真正思维的论述。由此，也就引申出了“**机器能否有理解能力，才是机器是否具有智能的关键**”的看法。

Penrose 则从另一个角度来反驳（智能）机器可以具有真正智能的论断。Penrose 认为，**不能避开“意识”去讨论智能**。他指出，“Turing 在其著名的论文中没有提到‘意识’，就直接提到了‘思维’，并且在标题上用了‘智能(Intelligence)’这个词。依我自己看待事物的方法，智能问题是属于意识范围内的问题。我相信，如果没有意识相伴随，真正的智能是不会呈现的。……当我断言自己相信真正的智能需要意识时(由于我不相信强人工智能所宣称的只要制定一个算法就能召唤起意识的论点)，根据我们现在术语的意义，我的意见暗示着：智能不能用算法的方法，也就是电脑，正确地模拟智能。因为我很快就要有力的论证，在意识行为中必须有本质的非算法成分。”其后，他指出，数学洞察力具有非算法性质，“……不管数学家用什么(足够广泛的)算法去建立数学真理，或是类似真理的东西，不管他采用什么形式系统去提供真理的判据，总有一些数学命题，譬如该系统显明的哥德尔命题，这些算法不能提出答案。……这本质上就是鲁卡斯提出的论断，头脑的作用不能完全是算法的。”另外，他还指出，**灵感、洞察和创造性往往不是从算法得出，而是依据美学标准**。

上述质疑我们最终可归结为，“**机器能否有意识；意识、思维等是否可以用算法来描述，是否都具有可计算性**”的问题，这也就成为了机器是否具有智能的另一个焦点问题。

不错，图灵测验只回答了机器是否具有“智能行为”的问题，但是，即使有“智能行为”仍不等于能理解和有意识，因为这涉及到理解和意识产生的机理问题。有同样的机理才能说有同样的本质。如果大脑与机器产生智能行为的机理根本不同，则很难说（智能）机器有真正的智能。那么，

（智能）机器能具有理解、意识和思维能力吗？李衍达[1711]认为，对于机器是否能理解，是否有意识，是否可思维，我们可以提出这样的问题：**不要问机器是否有思维和意识，而应当问机器和大脑是否可有相同的作用机理**。Turing 用可计算模型来说明机器的智能，结果产生了算法能否描述意识之争。事实上，意识、理解产生的机理和可计算模型能否概括意识这两个问题，都涉及大脑的作用机理；而大脑的作用机理问题只有在理解了信息的意义，在信息处理层面和复杂系统运行机制的基础上才能回答。

意识是不是由大脑这一“物质”所产生的？Crick 在他的“惊人的假说”中曾提出了用科学的方法来解释意识产生的奥秘的问题。他认为，意识是脑神经元活动的产物，人的精神活动完全是由神经细胞、胶质细胞的行为以及构成它们的原子、离子和分子的性质所决定的。他认为，不仅仅意识，而且“自由意志”也来自神经元的活动。他并进一步认为，机器也可以具有意识，甚至具有自由意志。事实上，Crick 不仅正面地肯定了几十年前 Eddington 提出的使普通原子的集合体成为一个思维着的机器的思想，更重要的是他明确提出了要探讨意识之源的问题。他提出，“只有当我们最终真正地理解了脑的工作原理时，才可能对我们的感知、思维和行为做出近于高层次的解释。”他指出，**搞清大脑产生意识的机制，意识的神秘性会消失，模拟意识就成为可能，机器就可能具有意识或自由意志**。但是，仅从原子、离子和神经细胞也不能说清楚意识是如何产生的，如果没有信息交换、处理的概念，没有复杂系统的涌现的概念，仍然无法回答意识产生的机理问题。

近些年来，人们又提出脑-机接口(BCI)的问题。BCI 的出现表明，大脑与计算机（芯片系统）可以直接进行信息交换（交互），而且可以相互理解，计算机芯片甚至可以成为大脑（或神经系统）的一个部分(如视网膜芯片)。这表明，思维的机理在于信息处理，意识产生的机理也在于信息的交换与处理。当然，这是在复杂系统意义上的信息处理。因此，**智能系统模型与信息处理过程模型应是一致的**。这样，（智能）机器与大脑在信息处理的意义上应是一致的。在信息处理这个层次上来看，两者应该具有相同的机理，亦即，在信息处理与复杂系统这两个基础上，（智能）机器应是可以产生意识和思维的。李衍达认为[1711]，BCI、信息处理模型、复杂系统模型的发展将有助于我们理解智能。由此，我们或可以设计一种新的思维产生机理的测验，以模仿图灵测验；不过，在屋里被分隔的应是两个人，一个是普通的人，另一个是其大脑中有一部分被机器替代的人；由屋外的人分别对这两个人进行询问，如果无法区分这两个人中哪一个大脑中有一部分被机器替代，哪一个没有被替代，则说明，即使大脑中一部分被机器所替代，仍能产生相同的智能行为；如果在信息交换与处理这一层次上，机器的作用机理与大脑不同，则不可能互相交换信息，更不可能共同处理信息；如果机器与大脑中其他部分不能进行信息交换，机器与大脑中的其他部分不能合作来处理信息，则必然导致不能产生与之相应的智能行为；反之，如果所有的智能行为都能产生，惟一的可能便是机器与大脑的其他部分可以相互交换信息，可以合作处理信息。因而，在信息交换与处理这一层次上，它们具有相同的作用方式，或者说，具有相同的作用机理。

BCI 已从实验上证实，**电脑与人脑可以互相交换信息**。这在微观上给出了机器与大脑可以相互交换信息，而且相互“理解”的证据。由此，也引发了人们**“人-机一体化”，或称“脑-机一体化”的构想。----将部分的人脑、部分的电脑、部分的人体、部分的机器融为一体**。如此的人-机集成，原本是科幻小说的领域，但很可能会成为我们今后必须面对的现实问题。

在半个多世纪的时间内，智能模拟和工程化的研究曾面临过无数的憧憬、希望、奋斗、争论、困难和挑战，同时也孕育了巨大的成功和机遇，推动了其学科的迅速成长和壮大。我们坚信，在 21

世纪以信息技术和生物技术为主导的知识经济中，智能科学和技术一定会具有举足轻重的地位和影响。智能技术将渗透到社会的各行各业，导致这些行业乃至社会的革命性变革--使人类社会进入智能时代。作为一门实践性学科，智能的模拟与工程化研究，将会在多学科融合、多途径协同，多种方法协作的努力下，实现其飞跃式的发展。同时，也会给我们带来一系列新的思想和新的观念。**未来，我们不仅要用量子物理模型来解释世界，我们也要用具有复杂系统特性的信息模型来解释世界；我们不仅要考虑生物的智能和机器的智能，还要考虑人-机集成的智能。**也许，我们对生命、意识和智能的理解，会由此而发生重大的改变。**智能，不仅让我们可以认识世界和改变世界，也会最终改变着我们自己。**

17.2 基于领域显式知识的问题求解系统的研究

17.2.1 思维模拟和问题求解的基本方法

我们知道，自然、社会和人类都是物质世界的不同组成部分，它们都是物质世界由低级到高级的不同的存在形态。现代科学已经证明，人与一切生物一样，具有共同的生物学构造规律。人类社会是自然界长期发展的产物，而人类的心理意识则是伴随着人类的产生而产生的。人类心理意识的产生可简化为如图 17.2.1 所示的进程。在这一进程中，从“一般物质反应特性”到“动物心理”，是所有动物所共有的；而“人类的心理意识”则是只有人类才具有的。动物心理是动物反映的最高级形式，可以包括简单的动机、知觉、表象和情绪等，其物质基础是动物的大脑。而人类心理意识则是人脑的机能，它是人类在第一信号系统和第二信号系统基础上所进行的精神活动。从总体上看，人类心理意识是“知”、“情”和“意”的综合体。其中，“知”是指认知，它是人类对世界的知识性与理性的追求；“情”是指情感，它是人类对客观事物的感受和评价；“意”是指意志，它是人类追求某种目的或理想时所表现出来的精神状态。

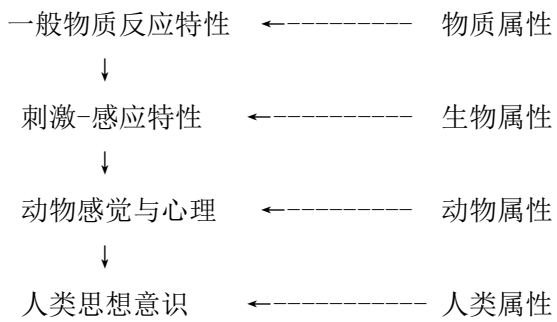

图 17.2.1 人类心理意识产生的进程

我们所说的智能的模拟应是对人类智能的模拟。而智能，作为人脑精神活动所表现出来的能力，与“知”、“情”和“意”有着密切的关系。随着心理科学、智能科学和人工智能研究的发展，人们逐渐认识到，相对于情感和意志而言，在现阶段，对人类认识和思维的理解和模拟要相对容易一些，而且人工智能的一般工程应用也不涉及情感和意志问题。因此，在现阶段，我们所说的智能模拟主要是对人类理性的认知或思维能力的模拟。

人类的认知系统，可以认为，主要由认知客体、认知主体和认知中介构成。其中，认知客体包括自然客体、社会客体和精神客体；认知中介由各种认知工具和手段组成；认知主体可以是人类个体或组织，它具有自主性和能动性，可通过认知中介同认知客体相互作用，以特定的方式对来自认知客体的信息进行有组织的加工、处理和整合，从而实现客体信息向主体认知观念的转换，并以认知（信息和知识）的形式存储于认知主体大脑之中，指导着主体的行为和实践活动。认知主体的大

脑具有复杂的生物性结构和丰富的信息性结构，其中，大脑的生物性结构是人类认识的基础，而大脑的信息性结构则决定了认识主体的认识和支配行为的能力，而且它们又都是随着主体的成长及社会实践活动的深化而不断丰富和发展的。

尽管认知主体有社会组织主体和个人主体之分，但个人主体无疑是最基本的。一方面，它的认识活动是社会主体认识活动的基本组成部分，另一方面，个人主体的认识活动也要受到其它主体认识活动的影响。

从功能角度来看，认知主体是一个“知”、“情”、“意”相统一的有机整体。但在认知主体中，直接负责认知任务的是**主体的认知结构**，它既决定着认知主体在不同的层次上对从认知客体所获得的信息的可能的加工深度，也决定着认知主体对认知客体的形态、结构、属性和含义可能做出合理解释的能力。当然，情感、意志、信念和习惯等非认知心理因素对认知过程的发展也有着激发、导向和调节等重要作用。

认知主体对认知客体的观念把握是一个有规律的辩证发展过程，过程中的每个认识周期都遵循着**“实践-认识[感性认识-理性认识]-再实践-再认识”**的逐步发展的模式进行。其中，**实践**是产生认识的基础，也是检验认识的正确性的最终手段。感性认识是人们把客观事物信息转化为主体主观认识的第一步，它所揭示的通常是事物的现象和现象之间的外部联系，其基本心理反映形式是感觉、知觉和表象等。理性认识是人借助于抽象的思维对感性认识进行加工、整理、概括而形成的关于事物的本质、整体及其内部联系的认识，它所揭示的是事物的本质和规律性。

感性认识和理性认识是人类认识过程的两个不同阶段，它们既相互区别，又相互联系。而在整个认识过程中，思维都起着非常重要的作用。认识论已经揭示：思维本质上是主体把握客体的一种认识方式，但它不是一般的认识方式，而是以揭示事物的本质和规律为目的的理性的认识方式。**思维同一定的认知结构相联系，是各种认知要素、思维方法的综合体**。人的思维方式一旦形成，就具有稳定性、定型化、时代性、地域性和民族性等特征。

对人类认知的模拟本质上就是对人类认识机理的模拟。由于思维是人类认知的核心，也是人类具有智能的核心能力，因此，**在一定程度上，对人类认识机理的模拟最根本的就是对人类思维的模拟。对人类思维的模拟是智能模拟和工程化的核心内容**。

人类的思维过程本质上是人对外来信息和感性认知进行加工、处理和信息转换的过程。在思维的过程，包含着神经元层面、认知层面和社会面层等不同层面上的信息加工过程。因此，人们可以在不同的层面上来研究人类思维的功能、结构、方式和方法，并对其模拟研究。比如，Feigenbaum 所代表的知识工程(专家系统)学派，以及 Nilsson 和 McCarthy 所代表的逻辑学派的主要工作，主要就是针对理性层面的；而 Newell 和 Simon 所代表的认知学派的工作，则主要是针对认知层面的；而 McClelland、Rumelhart 和 Hinton 等所代表的联接主义学派的工作，主要就是针对神经元层面的。

Simon 的物理符号系统(Physical Symbol System，简记作 PSS)假设认为，如果一个物理系统能对符号进行输入、输出、存储、复制、结构化和条件转换等操作，则可等价于一个智能系统。这可认为是在认知层面上对思维进行模拟的理论依据。**认知层模拟所采用的方法主要是寻找人类对问题求解的一般方法，然后用计算机来实现这些方法**。如曾经的产生-测试法、手段-目标法等。这也导致了对一般问题求解系统理论的研究。这类**通用问题求解方法**也称为**弱方法**。弱方法对一些简单的问题比较有效，但无法解决现实中复杂的具体问题。其主要原因是，一般问题求解系统缺乏解决具体问题的专门知识。为弥补这种缺陷，**用于问题求解的强方法--基于知识的问题求解系统**便应运

而生。现在，人们主张将强方法和弱方法结合起来，共同解决一般及具体问题。

基于理性层面上的（智能）模拟曾集中于对认识结构中所包含的逻辑(概念、判断和推理)和非逻辑(想象、直觉、灵感等)等机制进行模拟，其代表性的学派有知识工程学派和逻辑学派等。知识工程方法认为，所有智能活动的开展都取决于知识，知识的数量和质量是决定一个智能系统性能是否优越的主要因素。知识可为人类解决困难问题提供求解的手段和方法。知识工程方法构成的智能系统主要以专家系统、知识库系统等形式出现，它们可出色地完成如医疗诊断、地质勘探等需要某个领域专门知识的任务。但由于这类系统中的知识过于专业，缺乏通用性，因而其局限性也很大。特别是存在着诸如**知识获取“瓶颈”和知识“窄台阶”**等问题。尽管后来知识工程方法的倡导者们开始重视拓宽系统的领域知识，并加入了常识、背景知识和一般规律等，使其向认知学派靠近，但一直收效不大；直到 GPT 等模型的出现。

逻辑学方法特别注重对逻辑的研究。逻辑学方法的倡导者们认为，人类的知识大体上可分为陈述性知识、过程性知识、控制性知识和情节性知识等。而智能通常来自与上下文无关的陈述性知识和过程性知识。这些知识不同于某领域的专业知识，也不同于弱方法所用的一般规律；不过，它们都可以用各种逻辑语言(如一阶谓词逻辑、直觉逻辑)来表述和处理。通过对这些知识的逻辑表达与运用，我们即可模拟人类的思维过程。

对认知与思维的神经元层面的模拟则集中于对认识结构中包含感觉、知觉和表象等基本心理反应形式的感性认识进行模拟，其代表性的学派是联接主义学派等。联接主义学派认为，应先研究人脑的结构，然后按照这种结构来构建智能系统。现代神经生物学的研究成果表明，人脑的智能主要由大脑皮层来实施，而大脑皮层是一个大规模互连的神经元网络。因此，联接主义学派认为，智能模拟研究的重点，应是单个神经元模型的构造，以及神经网络的联结模式与其功能的关系的研究。

从行为层面和社会层面来研究和模拟人类智能也是值得考虑的方向。进化学派以人类意识的产生进程为依据，提出了智能系统的（生态）进化模拟理论：先构造具有初等智能的智能体，并将其置入真实世界环境。该初等智能体将通过连续地同环境作用来不断增长其自身的智能，从而使其逐渐地进化到高级智能，乃至人类智能的水平。此种基于生态主义的进化方法我们是赞同的，但纯粹依靠进化的做法却是不可取的，因为从动物的刺激-感应特性进化到人类意识显然经历了极其漫长的路程，固执地去模拟生物进化的整个过程并不高明。

随着信息科学技术的发展以及对人类群体认识活动研究的深化，还产生了（社会）集群智能学派。其中，关于多智能体系统的研究目前已成为一个重要热点。智能体具有三个基本要素：**知识**--智能体关于它所处的世界或它所要求解的问题的描述；**目标**--智能体一切行为都是面向目标的；**能力**--智能体所具有的推理、决策、规划、控制和通讯等能力。单智能体的目标是试图模拟单一主体的认识活动。而多智能体系统则是由多个智能体组成的一个大的（社会）智能系统，该系统不仅能丰富其中所包含的单个智能体的知识，而且还能改善其处理问题的能力；各个智能体之间的相互协调与合作，更可构成复杂的智能活动。因此，多智能体系统的目标应是对人类社会主体的模拟。

对人类思维的模拟也许需要上述多个层面、多种方法的综合。但无论采用何种方法，知识和思维永远是一个问题求解系统的核心。因为问题求解就是决策。而决策是最基本的思维操作，是知识的运用，通过思维达到所要求的目的。在问题求解系统中，知识是思维活动的基础。知识是人通过其实践认识到的客观世界的本质和规律，包括着经验、领域知识和常识等。它们是人类一切思维活动的起点，而人类思维活动的结果反过来又丰富了这些知识。当然，为了使系统能处理知识，首先

需要对知识进行表示。于是，人们曾提出多种知识表示方法，如知识工程学派所采用的一阶谓词逻辑、语义网络和框架等表示方法，联接主义学派所采用的将一切问题的特征都转换为特定联结特性(权值和闭值)的分布来加以表示和存储的方法等。知识的表示方式并不要求是唯一的，但是，为了在思维过程中能快速、准确、完全地利用知识，要求知识的表示应能便于运用。这样，系统要么得具有通用的知识表示模式，要么得具有一种灵活的知识转换机制，它能把知识从一种表示模式有效地转换到另一种表示模式。知识的运用也是如此，知识可有多种使用方式。人是最善于使用知识的，但如何将人使用知识的机理赋予机器却是一个难题。

思维方式取决于一定的思维结构，但人类的思维结构的形成是同一定的世界观和方法论相联系的，它们归根到底是受实践方式的制约和影响的。**经验思维**是一种较为简单的思维方式，它注重原始积累的经验，并把它们分为一些可用的认识模式加以记忆。经验思维的过程可分为模式匹配和模式处理，通常是先对外界获取的信息进行模式匹配，然后再按相应的样本模式进行处理。这种思维方式有很大的局限性，难以适应新环境。**逻辑思维**是以概念思维为核心的一种思维方式，其思维过程注重逻辑推理，可对事物的发展给出合理的预测。然而，由于逻辑不可能很完备，可用形式逻辑表达的思维过程并不多，而且所采集的外界信息可能还是不完全的、模糊的、甚至是错误的，从而“墨守成规”的推理方法也可能会造成推理的结论的不可信。因此，这种思维方式的运用也应受到一定的限制。**辩证思维**是一种更高层次的思维方式，它以经验思维和逻辑思维为基础，并有自己独特的思维原则和方法，可对事物多方面的本质作统一的和整体的把握。辩证思维能产生新的经验、新的知识和新的推理。但现有的智能系统仅能模拟相对简单的某一类或某几类思维方式，如联接主义机制方法擅长于模拟经验思维，知识工程方法适宜于模拟逻辑思维。要让现有的智能系统模拟辩证思维还有一定难度。

更进一步讲，人类的决策行为常常是对不同思维活动的推测进行协调的结果。思维方式的不同将直接导致人们思维活动结果的不同。人类的思维常常是在相互矛盾、相互冲突的复杂环境下进行的，根据不同的思维原则和方法得出的结论可能是完全冲突的。那么，一个可接受的结论的获得就需要在各种相互冲突的结论间进行协调。因此，为了获得一个合理的结论，可能需要多种思维方式的合作，需要有一个协调机构来控制各种思维方式的配合、中间结论的合理利用等；在进行群体思维时，也需要某种协调机制的介入，以实现对不同个体之间思维的协调与整合。在思维模拟系统中，我们同样也需要类似的**协调机制**。这一协调机制应具有如下机能：① 实现对单一主体系统内各种认知和逻辑思维结果的协调。②实现对系统内各种逻辑思维和非逻辑思维的协调。由于复杂问题的解决往往需要逻辑与非逻辑的多种思维方法的协同作用，在许多情况下，这种联合思维过程离不开一定的协调机制。③实现对多个认知主体间的认知协调。智能体模型描述的是单一主体的思维结构，而在群体认识的过程中，由于多个主体之间需要相互合作才能处理复杂认识对象，因此，多认知主体间各方面关系的协调机制应是系统整体结构中非常重要的组成部分。

17.2.2 基于知识的问题求解系统

为了实现智能模拟和工程化，人们曾提出多种“思维模拟”、“问题求解”和“行为智能控制”系统，其中典型的系统包括：基于领域显式知识的问题求解系统；基于思维模拟的通用问题求解系统；基于谓词逻辑与命题逻辑的启发式推演系统；基于神经网络和[无显式表达的]经验映射与反应系统；可变环境下基于即时反馈的适应与行为控制系统；多智能体协作系统；等等。在这些系统中，基于知识的问题求解系统是最典型也是最重要的一类智能系统。它一直是智能系统研究和开发的重

点。

从工程应用的角度，开发一个智能系统的一个主要目的就是要解决**非平凡问题**，即难以用常规（数值计算、数据库应用等）技术直接解决的问题。这些问题的求解要依赖于问题本身的描述和领域相关知识的应用。按解决问题所需的领域特有知识的多寡，问题求解系统曾被划分为两大类：知识贫乏系统和知识丰富系统。前者必须依靠搜索技术去解决问题，后者则主要依赖于识别技术或知识的运用。完全的识别系统并不需要推理，但由于问题求解系统要解决的问题往往比较复杂，且结构不良(缺乏可预先确定的从初始状态到目标状态的解答路径)，大多数系统也需要用试探性的推理去搜索解答。虽然其推理也可视为特殊的搜索，但推理主要服务于识别---通过基于领域知识的匹配检查去识别下一问题状态，而不像典型的搜索技术那样，通过设计通用的生成和评价机制去生成后继问题状态并作选择。鉴于典型的搜索技术因知识贫乏而缺乏针对性，效率往往低下，不如基于丰富知识的识别技术来得直截了当，所以，目前大多数知识工程系统（或问题求解系统）均采用包括推理的识别技术（思维链技术）来实现。

基于知识的问题求解系统亦称知识工程系统或专家系统。它本质上是对人类解决问题思维过程的模拟。人类实践的过程通常是一个获取知识和运用知识的过程，前者主要是探索和学习，后者主要是（运用知识和经验）解决实践中的问题。它通常是主体在一定的思维方式下运用已有的知识去解决问题的过程。

通过逻辑思维来解决困难和复杂的问题，是人高级智能行为的重要表现，而知识正是产生这种智能行为的基础。例如，建筑师设计漂亮的房子是因为其有建筑设计和美学方面的专门知识，而电器工程师修复出故障的电器则是因为其有关于电器的专门知识和维修电器的经验知识。所以，要让机器或系统具有像人那样的高级智能行为，就必须使机器或系统具有基于知识的问题求解能力。

基于知识的问题求解系统，当其表现出专家级问题求解能力时则被称为专家系统。专家系统是一个含有大量的某个领域专家水平的知识与经验的智能计算机程序系统，能够利用人类专家的知识和解决问题的方法来处理该领域内的问题。简而言之，专家系统是一类可模拟人类专家解决领域问题的计算机程序系统。专家系统的研究起始于上世纪中期。当时大多数人工智能的研究者曾注重于开发强有力的通用问题求解方法，如基于归结原理的定理证明方法和 GPS（**通用问题求解**）系统等。这些研究基于一个朴素的信念：只要把关于推理的法则和强有力的计算机联接起来，就会产生出超人的性能。但人们不久就发现，通用的问题求解方法具有严重的局限性，其解决实际问题能力太微弱了，几乎不能用于解决复杂的实际应用问题，这一困境导致了智能系统研究的转向，使人们开始更加注重知识在问题求解过程中的作用。

17.2.2.1 基于知识的问题求解系统的典型特点

基于知识的问题求解系统具有以下特点：

（1）具有求解问题所需的专门知识和知识库系统。专门知识分为两大类：一是应用领域的基本原理和常识；二是领域专家求解问题的经验知识。前者是构成专业知识的主体，可以精确地定义和使用，并为普通技术人员所掌握。这类知识尽管是求解问题的基础，但并不与求解的问题紧密结合，加之知识量大和推理步小，故一般不能高效地支持问题求解。后者则是领域专家根据多年的工作经验，对如何使用前者解决问题所作的高度集中、抽象和浓缩的描述。正是这类知识，使领域专家能高效高质地解决困难和复杂的问题。例如，表示状态数据和解答之间关联的启发式推理规则就是典型的经验知识，它们高度概括了遵从领域基本原理和常识的大量基础性推理操作，免除了本来应花

费大量时间去作的精细而又烦琐的运算，使问题求解过程可以大踏步地推进。然而，使用这类知识的条件比较苛刻，条件不满足时会导致不正确的解答甚至推理失败。所以，从高效地求解问题的角度，基于知识的问题求解系统更需要经验知识；而若要保证解答的正确性，系统还需要领域的基本原理和常识等。

知识库系统是把知识以一定的结构存入计算机，并支持知识的管理和问题的求解，可实现知识的共享的系统。目前，已有一些 KBMS 软件问世，这些软件的明显特色是可将推理和查询结合起来，明显改善了知识库的维护功能，可为开发具体领域的知识系统提供有用的环境。

（2）具有运用知识的自动推理机制。系统求解问题的能力主要取决于知识的表示和推理技术。最基本的知识表示技术有逻辑表示、产生式表示、知识图谱和知识的结构化表示等，如何选用它们或它们的变种，主要取决于问题求解所需知识的特征。例如，要表示应用领域中事物的结构及事物之间的关系，可以用语义网络和框架系统；而表示启发式关联则用产生式系统更合适。表示知识的特定结构要便于与其相适应的推理机制的存取和使用。推理机制的设计包括问题求解的组织和推理控制，它也取决于应用领域的特征和问题求解任务的要求。

知识系统的**推理机制**可从一个或几个已知的判断(前提)逻辑地推导出一个新的判断(结论)，它是对人类解决问题时利用以往的知识通过推理得出结论的过程的模拟。自动推理机制是系统问题求解的重要基础。其推理机制可包括演绎推理(deductive reasoning)和归纳推理(inductive reasoning)，正向推理(Forward Reasoning)和反向推理(Backward Reasoning)等。正向推理又叫数据驱动的推理，是从已知的数据/条件/中间结论出发推导出新的结论。反向推理又叫面向目标的推理，是从结论(目标)出发推导结论(目标)的前提条件是否存在的过程。

（3）具有特定的学习机制。Simon 认为，如果一个系统能够通过执行某种过程而改进它的性能，这就是学习。这一说法指明：① 学习是一个过程；② 学习是对一个系统而言；③ 学习可改变系统性能。**过程、系统与改变性能**是学习的三个要点。所有可改进系统性能的方法，都可称是学习。机器学习是必须大力开展研究的领域，因为只有机器的学习研究能取得进展，人工智能和知识系统才会取得重大突破。现在，基于知识的问题求解系统都有学习机制，但还都处于可进一步改进阶段。今后，系统学习机制研究的重点，是研究学习过程中的认知机理、学习过程中知识获取的方法、新的（自）学习算法、以及可综合多种学习方法的（多模态）学习系统的研制等。

（4）多种优势特性。知识系统或基于知识的问题求解系统所具有的优势还包括：启发性--系统能运用显式的知识与经验进行推理、判断和决策；透明性--系统能够解释本身的推理过程和回答用户提出的问题，以便让用户能够了解推理过程，提高对系统的信赖感；灵活性—系统可灵活运用已有的知识进行问题求解；成长性--系统能不断地增长知识，修改原有知识，不断更新。

17.2.2.2 基于知识的问题求解系统的基本构成

系统的结构是指系统各组成部分的构造方法和组织形式。系统结构选择恰当与否，是与系统的适用性和有效性密切相关的。选择什么结构最为恰当，要根据系统的应用环境和所执行任务的特点而定。

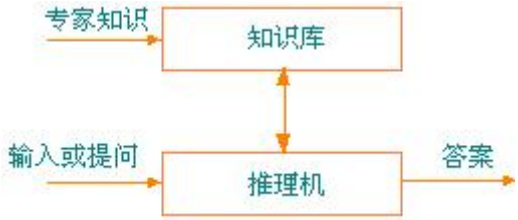

图17.2.2　**基于知识的问题求解系统的基本构成**

基于知识的问题求解系统的基本结构可视为由三个基本部分组成：**知识库、推理机和用户界面**。知识库是系统最重要的部分，它存储求解问题所需的以一定符号结构表示的知识。推理机则具有进行推理的能力，它根据输入的问题以及描述问题初始状态的数据，取用知识库中的知识去做推理，并输出最终解答。推理机也可请求用户输入推理过程中所必需的补充数据并应用户要求解释推理结果和推理过程。用户界面并非系统的主要部分，但一个使用户感到友善的界面对于系统的推广应用有重要影响。目前，自然语言交互界面、可视化图文界面和多模态界面已广泛地应用于智能系统。

知识库主要包括两部分内容。一部分是已知的同专业问题解决有关的经验性知识；另一部分是进行推理时要用到的一般性知识和领域基础知识等。

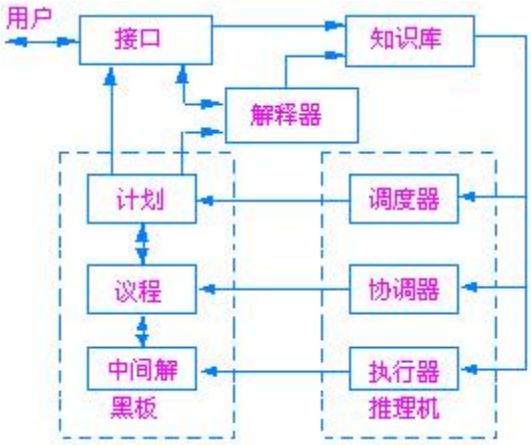

图 17.2.3 **基于知识的问题求解系统的典型构成**

更复杂一些的系统结构可如图17.2.3 所示。其中，调度器按照系统建造者所给的控制知识，从议程中选择一个项作为系统下一步要执行的动作。执行器应用知识库中的及黑板中记录的信息，执行调度器所选定的动作。协调器的主要作用就是当得到新数据或新假设时，对已得到的结果进行修正，以保持结果前后的一致性。解释器的功能是向用户解释系统的行为，包括解释结论的正确性及系统输出其它候选解的原因等。

17.2.2.3 基于知识的问题求解系统可处理的问题

基于知识的问题求解系统可处理的问题及求解任务目前主要是知识密集型的，常见的任务有解释、诊断、监控、预测、规划和设计类等。

（1）解释。通过对已知信息和数据的分析与解释，确定它们的涵义。该任务最典型的工作是分析测量和观测到的数据，以确定其含义。数据往往来自传感器和各种测量仪器。例如曾经的 DENDRAL 对质谱仪测量数据作分析，以确定被测化合物的分子结构，分子结构就是对测量数据的解释。为找出与数据相容且正确的解释，关键在于数据分析必须是完备的，即分析系统应系统化地考察所有可能的解释，删除候选解答必须具有足够的证据。测量数据因可能包含错误甚至不完全，会给正确的

解释带来困难，因此必须妥善处理。有时可能需要对数据做出某些假设，判断过程可能很复杂和很长。典型的例子还包括：语音理解、图象分析、化学结构分析和信号解释等。此外，还有各类咨询专家系统等。

（2）诊断。根据观察到的情况(被诊断事物的故障现象)来推断出某个对象机能失常(即故障)的原因。例如机器故障诊断等。为了做出正确的诊断，系统必须具有关于被诊断事物的结构、功能、因果关系和行为正常与否的知识，而关于故障现象和根源之间的启发式关联知识则可加速诊断过程。诊断任务的困难在于：多故障(一些故障被其它故障掩盖)，而且故障可能是断续出现的(这就要求强化诊断手段)；传感器和测量仪器本身可能有故障（原始数据不准或错误）；数据测量代价昂贵或危险(这就要求合理选择要测量的数据的种类)；诊断知识不精确和不完全(如人体疾病机理不清难以诊断)等等。此时，系统都需要从不确切信息中得出尽可能正确的诊断。其典型的例子还包括：医疗诊断、电子和软件系统故障诊断以及材料失效诊断等。

诊断系统的进一步发展就是自动维修专家系统，它可对发生故障的对象(系统或设备)进行处理，使其恢复正常工作。维修专家系统具有诊断、调试、计划和执行等功能。其典型例子有核电站自动维护系统和空间站自动维护修理系统等。

（3）监控。要求连续地观测和解释反映被监控对象、系统或过程的现状或行为的数据，并把观察到的行为与其应当具有的行为进行比较，发现异常时，发出警报，提出解决建议，给出异常的原因。监控的高级阶段是自动控制，即自动调节系统的各种参数，以消除异常并保持系统处于优良运行状态。监控系统一般具有解释、预报、诊断、规划和执行等多种功能。不断地把反映系统现状的数据与系统正常运行状态下的数据作比较是实现监控的主要依据。实时诊断异常的根源，避免错误警报是提高监控性能的关键。要求系统具有快速反应能力，发出的警报要有很高的准确性，能够动态地处理其输入信息。其典型的例子可包括：空中交通管制系统、商业管理系统、自主机器人控制系统、生产过程控制和质量控制等。

（4）预测。通过对过去和现在已知状况的分析，从事物的现状和发展趋势预言其未来的状态和行为。如天气预报、农作物收成预测等。预测往往要按时间顺序推理，必须处理随时间变化的事物。智能预测的特点是需要对不完全信息进行综合分析(信息完全的预测并不需智能系统)，预测必须考虑多种可能的情况(使用假设推理)。系统处理的数据随时间变化，且可能是不准确和不完全的，系统需要有适应时间变化的动态模型。其典型的例子还有：军事预测、人口预测、交通预测、经济预测和政策变化后的效果预测等。

（5）规划。任务是寻找出某个能够达到给定目标的动作序列或步骤。即根据目的制定出能满足各种约束条件的行动方案，包括生产规划、工程项目规划、军事行动规划、机器人动作规划等。规划往往涉及优化使用资源(时间、人力、物力)的问题，约束条件过严可能导致规划失败，这就需要放松某些次要的约束条件；规划往往涉及到某些预测，所以预测也是规划的依据。当规划问题大而复杂时，应注意优先考虑重要的规划目标，并妥善处理多个目标之间可能出现的交互作用。

（6）设计。任务是根据要求给出能满足各种约束条件的设计方案。如建筑设计、工程设计、电子线路设计等。设计在许多方面与规划相同，但设计更为复杂，因为它不仅需要满足各种约束条件，还要对各个设计部分（元件或零部件）的空间关系及距离、形态、外形等因素做统一安排和推断，这将消耗大量的（立体）计算资源。目前，空间关系（世界模型）的推理尚不存在较好的方法，而

如何把约束满足和设计本身的特别要求综合起来考虑，也缺乏有效的手段。要从多种约束中得到符合要求的设计，系统可能需要检索（搜索）较大的可能解空间。

（7）管理。任务是根据现有条件、现实状况和预期目标作出恰当的决策。由于管理任务十分复杂，目前让系统来完成所有管理任务还不现实，所以，现有的系统多是辅助决策系统，其最典型的例子就是智能决策支持系统。由于智能系统研究的预期目标是代替人来从事脑力劳动，而人的脑力劳动有很大一部分就是管理决策，因此，智能决策系统应是基于知识的问题求解系统的一个最重要的发展领域。

（8）教学。任务是根据学习者的特点、弱点和基础知识，以最适当的教案和教学方法对其进行教学和辅导。其特点是：具有良好的人机界面，可有针对性的教学，同时具有学习效果评估和有针对性的辅导等功能。其典型例子有：计算机辅助教学系统以及聋哑人语言训练专家系统等。

在执行上述任务时，经常碰到的困难是：大型的解空间(解释、规划、设计)、要求试探性推理(诊断、规划、设计中因缺乏时间、事实和知识而不得不作假设)、时变数据(监控和诊断)、有噪声的数据(传感器常夹带噪声)等。为克服这些困难，就需要有针对性地设计适当的推理方法，并据此去设计问题求解系统的体系结构。

17.2.3 基于知识的问题求解系统开发中需解决的一些关键问题

构造基于知识的问题求解系统的关键包括问题求解系统的体系结构设计、问题求解建模[知识的表示与获取]、推理机制的设计及系统的发展与维护等。

17.2.3.1 关于系统的体系结构的设计

体系结构研究的核心问题在于如何根据应用领域的特征和问题求解任务的要求来组织问题求解和推理控制，并由此决定表示知识的符号结构和推理机制。尽管基于知识的问题求解允许有多种多样的实施方法，但仍可以归纳出某些组织求解的原则。体系结构的合理性将影响到问题求解的效率、完备性和可靠性。鉴于问题状态空间和求解搜索方法是随应用领域和问题求解任务的不同而最易于变化的特征，因此，我们可以把这些特征作为比较和研究系统体系结构的主要线索。影响搜索方法的选择有多方面的因素，例如状态空间的规模、数据中的错误和状态空间层次抽象的有效性等。

根据问题求解任务复杂程度的不同而在体系结构设计时采用不同的推理技术是一种必要的考虑。简单的问题求解任务具有较小的状态搜索空间、可靠不变的数据和可靠的推理知识，所以问题求解的组织原则很简单，可能只需穷尽搜索和单调推理。为了提高搜索效率，可对知识库中包括的事实和规则建立索引。

然而，几乎所有的实际应用问题均不满足简单性的条件。实际应用问题常涉及不可靠的数据和知识(医疗诊断)、随时间而变化的数据(生产过程监视和控制)或状态空间很大等特征，需要在设计体系结构时作相应的考虑。在状态空间很大的情况下，系统需要根据状态空间的特征渐增地采用一些相适应的推理技术。对于超大规模的状态空间，为了提高搜索解答的有效性，还可以并行地开展多线路推理(搜索)和更有效的推理组织（如当今的大模型），或采用一些特别的知识表示技术去改进推理效率，如采用特殊的数据结构和实行知识编译等。对于相当复杂的任务，让系统直接去处理庞大的（待搜索）状态空间并不合适，此时，我们或者可以进行任务分解，将待求解问题分解为子任务序列，并分别在与子任务相应的待搜索状态空间进行搜索；在问题求解过程中，若子任务相互制约，我们或可以先做一定局部猜测然后再作进一步地分析。有时，采用多路径推理和聚集不同的知识体进行求解可能是必要的；而有时，我们可能需要更强有力的知识组织和推理控制结构，从而引

出更复杂的知识源-推理机制结构体系。

应用越广泛的问题求解系统其系统体系结构设计所需要考虑的因素也越多，也有更多需要采用的设计原则。但是，我们提倡按照应用领域特征和问题求解任务的要求来设计系统的知识表示方法和推理机制只是一个方面，只是说明某些设计原则的必要性和重要性。现实中，可能并不存在最好的设计体系结构的通用原则，再好的原则也只能适用于一定的范围。因此，越通用的问题求解系统，因末结合应用领域和问题求解任务的特征，其推理能力会显得越微弱。故而，对面向通用人工智能的大模型系统，我们也必须加以改造和综合应用，以形成可应用于专有领域的专用方法，以更有效地应用于专业问题领域，这就是未来会应用越来越广泛的基于大模型的 AI 智能体。

17.2.3.2 系统知识表示方法的选择与确定

事物常有两个方面，一个是事物的结构，另一个是其属性。

基于知识的问题求解系统的研究和开发首先要解决的问题是知识的表示问题。如何才能将知识表示成机器[系统]可“理解”、并便于存储和运用的形式？这里，我们将从知识表示的作用和功能、知识表示的基本假设和基本方法等方面来介绍知识表示的基本概念，并重点阐述曾经被深入研究并得到广泛应用的知识表示技术和方法：产生式表示和结构化表示（用产生式、语义网络和框架等知识表示方法去表示问题求解所需的知识）。并讨论与知识表示有关的一些实际问题（如，陈述性和过程性知识、表示能力和推理效率间的制约关系、回溯控制算法、将自然语言描述表示为语义网络、框架系统的特性继承和相容匹配等）。关于目前被广泛用于知识表示的知识图谱方法，我们将在后续章节论述。

1．知识表示研究的意义与功能

什么是知识？Feigenbaum 认为，知识是经过削减、塑造、解释和转换的信息。简单地说，**知识是经过加工的信息**。 Bernstein 则认为，知识是由特定领域的描述、关系和过程组成。Hayes-Roth 认为，知识是事实、信念和启发式规则，可从范围、目的、有效性三个纬度加以描述。其中知识的范围是由具体到一般，知识的目的是由说明到指定，知识的有效性是由确定到不确定。例如“为了证明 A→B，只需证明 A∧¬B 是不可满足的”这种知识是一般性、指示性、确定性的。而像“桌子有四条腿”这种知识是具体的、说明性、不确定性。而对知识的最简洁的表述是：**知识是结构化的信息**。

知识是智能的基础。这里，符合计算机处理要求的知识模式（计算机能存储、处理的知识表示模式）包括着**数据** (Data)（信息的载体和表示；可用一组符号及其组合表示信息）；**信息** (Information)（数据的语义；或数据在特定场合下的具体含义）和**知识** (Knowledge)（信息关联后所形成的信息结构，如事实与规则等；是经加工、整理、解释、挑选、改造后的信息）等。这些知识通常具有相对正确性(是在一定条件下,或在某种环境中正确;或存在认识的“中间状态”;……)、不确定性（如，随机性、模糊性、经验性、不完全性、……)、可表示性（可用语言、文字、图形、图像、视频、音频、神经网络、模型等表示）和可利用性（关键是有用--对人认识世界改造世界有帮助）等特性。按照不同的角度，这些知识常被归结为：常识性知识与领域性知识（依作用范围分）；事实性知识、过程性知识、控制知识（依作用及表示分）；确定性知识、不确定性知识（依确定性分）；逻辑性知识、形象性知识（依结构及表现形式分）；等等。

智能系统所关心的知识主要是事实、规则和控制方面的知识。认为一个问题求解系统高水平的运行需要具备与问题有关的事实知识、规则知识、控制知识和元知识。其中，事实是对问题环境中

的一些事物的认知，常以“...是...”的形式出现。如事物的分类、属性、事物间关系、科学事实、客观事实等，在知识系统中属于底层的知识。如雪是白色的、鸟有翅膀、张三和李四是好朋友。规则是有关问题中与事物的行动、动作相联系的因果关系知识，是动态的，常以“如果...那么...”形式出现。特别是一些专业性的启发式规则，通常是专家用于处理问题的专门性经验知识，这种知识可能并无严格解释却常常很有用处。控制类知识是有关问题的求解步骤和技巧性知识，告诉怎么做一件事。也包括当有多个动作同时被激活时应选哪一个动作来执行等。元知识则是有关知识的知识，是知识系统中的高层知识。它包括怎样使用规则、解释规则、校验规则、解释程序结构等。

问题求解离不开知识。里南（D.B.Lenat）和费根鲍姆（E.A.Feigenbaum）曾提出一个所谓的知识原则：一个系统可展示高级的智能理解和行为，主要是因为其拥有应用领域的特有知识。这里，**“特有”**一词很重要，因为应用领域中有效地求解问题主要靠该领域特有的知识。系统（智能体）用于解决问题的知识中只有一小部分是普通知识，如状态空间搜索方法、推理控制策略等，这些知识虽能应用于多个领域，但对具体问题作用微弱，仅靠它们不能为问题求解提供足够的约束。足够的约束主要来自特别的知识，即应用领域特有的概念、事实、表示、方法和模型等。系统所拥有的知识和其性能（问题求解能力和效率）的关系，我们或可用**“知识门槛”**作一定说明。**“使能门槛”**指知识量超过该门槛时，系统就拥有了为执行任务所需的最低限度知识。低于此门槛，系统将无足够的知识去解决问题。**“胜任门槛”**指随着系统知识量的增加，系统的性能将提高，在知识量达到此点时，系统将能成为某应用领域中求解问题的专家，胜任只有领域专家才能解决的问题求解任务。超出此门槛，附加的知识也有用，但并不再显著增加其能力，所以系统的性能随知识量的增加而呈现的提高将变得很平缓。**“全能门槛”**是指到了这个门槛，由于知识量的空前丰富，使系统能解决该应用领域内的几乎所有问题，成为全能专家。超过此门槛，附加的知识几乎不再有实质性作用（并非无用，而是原有知识已经够用了，新知识对本领域问题求解已无明显帮助）。

智能模拟研究初期，许多学者曾致力于搜索与推理方法的研究，但收效不大，导致了人工智能在黑暗中摸索，并一度使人们对人工智能的前途产生了悲观态度。正是关于知识工程和专家系统的研究工作，揭示了智能行为依赖于知识的本质，才使几乎所有的人工智能研究者都认识到知识对于智能行为的重要性。目前的专家系统有一个共同特点，就是都是由启发式知识（经验性关联知识）来指导问题求解的。相比之下，它们的推理机只包括普通的关于推理控制的知识。由此可见，系统的能力主要由知识库中包含的领域特有的知识来决定。除了专家系统，许多其它的人工智能系统的研究也开始转向基于知识的观点（今后估计也会如此）。例如，高性能的自然语言理解必须有广泛的关于相关世界的知识，仅仅依据文法知识是远远不够的。同样，一个好的视觉程序（感知智能系统）也应具有世界知识（也因此，今后，构建关于世界的模型也将是必须的）。机器学习程序（系统）也必须依赖于一个初始的、丰富的知识体，而不是从什么都不懂开始学起；并且，智能体拥有的知识越多，就学得越多越快。

知识表示在基于知识的问题求解系统中曾有着重要的作用（今后也会如此）。知识表示研究知识在知识系统中表达的可行性和有效性，即研究如何对知识进行表达才能让机器容易存储、“理解”和运用。它通常是将知识表达为一类数据结构与控制结构的统一体，即把人类知识表示成机器能处理的某种可操作数据结构。从现实意义上讲，知识表示可看作是一种描述事物及其变化规律的约定，是知识的符号化、形式化或模型化。各种不同的知识表示方法,是各种不同的形式化的知识模型。

按照符号主义的观点，智能体要有效地解决应用领域的问题，就必须拥有领域特有的知识。为此，如何表示知识就成为了人工智能系统研究的一个重要议题。尽管知识在人脑中的表示、存储和使用机理仍然是一个尚待揭开的迷，但以形式化的方式表示知识并供计算机（智能系统）自动处理，

已经发展为较成熟的技术—知识表示技术。

知识表示在人工智能系统的建造中曾起过关键的作用。可以说，正是以适当的方式表示了知识，才导致智能系统展示出智能行为。从某种意义上讲，表示可视为数据结构及其处理机制的综合：**表示 = 数据结构 + 处理机制。**其中，恰当的数据结构可用于存储和表示要解决的问题、可能的中间解答和最终解答以及待解决问题所涉及世界的描述。我们称表达这些描述的数据结构为符号结构，正是符号结构导致了知识的显式表示。然而，仅有符号结构并不能体现出系统具有知识，因为符号结构本身并不构成意义，只有可对其作适当的处理才构成意义。例如，一个汽车自动驾驶系统应有一条规则：在路口见到红灯时停车等待。显然，若该规则只是以某种形式存放于系统知识库中的字符串，则其本身并无能力产生任何有意义的作用；只有经系统的规则解释程序取用后，才会产生作用--使汽车遇红灯时停下。所以，要使系统显示出有知识，不仅要定义适当的数据结构（即符号结构），还须定义配套的处理机制去使用它们。

由此，一个智能系统中的符号结构，应满足两个主要特性：

（1）**可解释为表示知识的命题，否则，表示的就不是知识**。所以，知识表示隐含地要求符号结构能以真值理论来解释，从而可以因某些命题的存在而说明世界是什么样子的。符号结构本身不必是命题形式，但观察者必须看到智能系统能把它们“解释”为命题并能正确“理解”。例如，语义网络中的节点是以包括若干槽的符号结构来表示关于对象（个体或概念）的知识，是非命题形式，但系统必须具有可以把这种符号结构解释（转化）为命题逻辑并加以理解的能力。如节点（Node G-al Isa：Grade-assignment Student：John Course：CS100 Mark：85）可解释为命题 Grade-assignment（G-al）∧ Student（G-al，John）∧ Course（G-al，CS100）∧ Mark（G-al，85）并可进一步解释为 Grade（John，CS100，85）。

（2）**在智能系统的行为中起因果作用**。这种作用与我们将符号结构理解为表示知识的命题是一致的，从而能把系统的智能行为归因为是系统具有某种表示于符号结构中的知识。

这里必须注意，并不是系统能“意识”到它有知识，而是观察者从观察到的行为中认为系统具有某种知识。

由此，我们也就知道，设计任何智能系统均包括两个方面：**知识表示机制和领域特有的知识**。知识表示机制提供知识表示的形式和运用机制，表示的形式使我们可以以特定的符号结构去描述获取到的领域知识，而运用机制则可使系统能正确应用这些知识以表现出智能行为。

问题求解系统中，知识表示是为系统的功能服务的。对于解决复杂问题的智能系统，知识的表示必须能支持多种不同的问题求解活动。不同的活动往往需用不同方式表示的知识，所以我们面临着一种抉择：是以统一的表示方式表示所有的知识，还是以不同的表示方式表示特性不同的知识。前者以统一的符号结构来换取知识获取和知识库维护上的简易性，但会导致推理的低效性；而后者则反之。

为此，知识表示的设计是一个要根据实际应用加以权衡利弊的问题，以使其能支持系统的下列活动或功能：

（1）**机器学习和知识获取。**任何智能体（智能系统）都是逐步进化的，人工智能体（人工智能系统）也不应例外。所以，知识表示的一个重要功能就是支持智能体（智能系统）渐增地获取知识，使智能体（智能系统）的内部模型越来越精确地反映外部世界，以便有效地完成问题求解任务。这就要求智能体（智能系统）能表示外部世界中事物之间的重要差别，避免表示不必要的细节，以求

知识表示的紧凑和一致性。此外，知识表示应具有抽象和概括功能，以便在获取事物个性的同时获取和学习共性；在吸收新知识的同时应能消除可能引起的新老知识之间的矛盾，识别和处理知识的时间效应；在求解问题的过程中还应能通过自适应学习获取新知识等；这些都是知识表示应具有的品质。

（2）**感知。**感知对于智能行为有重要作用。因为只有感觉到真实世界中发生的事情，智能体（智能系统）才能去合理地行动。这也就要求智能体（智能系统）要能感知其是否正处于其拥有的知识可利用的世界状态中。为此，表示知识的符号结构和推理机制应支持智能体（智能系统）对世界的感知，支持系统对知识库的高效搜索，以便可以及时发现被感知的事物之间的关系和变化，找到对问题状态的最佳描述，处理感知信息中的错误等。

（3）**行动规划与决策。**问题求解系统的主要功能就是可处理现实问题，面对问题环境可做出相应的行动规划和决策等。它是系统所感知的外部信息与系统内部已有知识的交互作用的结果。因此，系统的知识应是可对外部世界的变化做出正确决策的经验。要使智能体（智能系统）在面对不同环境时有根据目标和期望做出规划的能力。系统如何制定规划，何时、依据什么、优先执行哪一个行动规划，如何评价、修改乃至重构规划，均应是知识表示应具备的品质。

正如我们可以用不同的方式来描述同一事物，对于同一表示模式的知识，我们也可以采用不同的表示方法。但是，在解决某一问题的时候，不同的表示方法可能会产生完全不同的效果。因此，知识表示也有一个选择的问题。为了有效地解决问题，我们必须选择一种良好的表示方法。

如何选择一个较好的知识表示方法？我们认为，对知识表示方法好坏的评价，应是根据其是否可满足该类问题求解的需要来决定。伍兹（Woods）提出，应从两个方面评价知识表示的性能，**包括表示的充分性和表示法效用**。

表示的充分性意指作重要区分和避免不必要区分的能力。一种表示方式无论怎样有效，若不能表示必要的区别，也是没有用处的。例如，一个自然语言理解系统必须能区别走路、跑动，开车和飞行的不同，但在必要时还应该能忽略它们的差别而意识到它们都是一种移动。可以说，只有具备表示的充分性，知识表示系统才能恰当地描述问题求解所涉及的事物，以及智能体对于外部世界的信念、目的和猜测等。

但表示的充分性仅是知识表示的起码要求，还不够，**好的表示方法还必须支持被表示知识的使用**。也就是说，表示知识的**元素**（**最基础符号或拟符号**）和处理这些元素的操作应能有效地支持使用知识的推理活动。这称为**表示法效用**。其又分为两个方面：

（1）**概念效率**。指知识表示方式应能有利于知识库以自然的方式吸收随意的新知识，新知识的加入和老知识的更新不会引起知识库发生大的变化。因为只有这样，才有利于知识库的逐步精化，才能使包含于知识库中的有关世界的内部模型能逐步地精化和调整到接近于正确地反映外部世界。为提高概念效率，要求表示知识的符号结构（元素及元素的组合）具有语义清晰、一致和简洁的特点，并易于修改。

（2）**计算效率**。主要指推理的有效性，如推理的速度、结论的正确性和有效性等。

要兼顾概念效率和计算效率往往是困难的，因为前者要求表示知识的符号结构与知识的获取和知识库维护相容，而后者则要求与推理机（推理机制）相容。换言之，为促进知识获取的有效性，知识应以接近于人思维的方式表示；但要提高处理效率，知识应以接近计算机目标代码的方式表示。为解决矛盾，一种可取的方法是提供两套符号结构，分别面向知识获取和机器推理，并设计自动转

换程序来实现两者间的映射。

衡量知识表示性能的这两个方面都是十分重要的。忽视任一方面都会导致关于知识表示认知的片面性。例如，有些人认为，一阶谓词逻辑的表示性能最强，而过分注重逻辑的表示，这会在强调了其表示充分性的同时，而忽视了这种表示法在概念和计算效率方面的低下。目前，多数人接受这样一种看法：一阶谓词逻辑具有最好的表示充分性，其它基本表示方式（产生式系统、语文网络和框架系统）的表示充分性仅是它的子集，但却具有更好的表示法效用。由此可见，知识表示的这两个方面的性能是相互制约的，往往提高一个方面的性能要求要以牺牲另一方面的性能为代价。所以，设计一种表示方式时，应根据应用环境和问题特征，对知识表示的这两个方面的性能作取舍权衡，以能否满足需求为最实用的评价准则。

若从实用化的角度考虑，知识表示的设计需权衡处理的是二个方面的问题，即：知识表示的程序性和陈述性，表示能力与推理效率之间的制约关系等。

由此，对于一个知识表示方法,我们通常有以下基本要求。

(1)**具备足够的表示能力**。针对特定领域,能否正确地、有效地表示出问题求解所需的各种知识就是知识表示的能力,这是一个关键的问题。选取的表示方法必须尽可能扩大表示范围并尽可能提高表示效率。同时,自然界的信息具有固有的模糊性和不确定性,因此对知识的模糊性和不确定性的支持程度也是选择时所要考虑的一个重要因素。

(2)**与推理方法（机制）的匹配**。人工智能只能处理适合推理的知识表示,因此所选用的知识表示必须适合相应推理才能完成对问题的求解。

(3)**知识和元知识的一致**。知识和元知识是属于不同层次的知识,使用统一的表示方法可以简化知识处理。在已知前提的情况下,要最快地推导出所需的结论以及解决如何才能推导出最佳结论的问题,就得在元知识中加入一些控制信息,也就是通常所说的启发信息。

(4)**清晰自然的模块结构**。由于知识库一般都需要不断地扩充和完善,具有模块性结构的表示模式有利于新知识的获取和知识库的维护、扩充与完善；表示模式是否简单、有效,便于领域问题求解策略的推理和对知识库的搜索实现,这涉及到知识使用效率；表示方法还应该具备良好定义的语义并保证推理的正确性。

(5)**说明性表示与过程性表示的协调**。一般认为，说明性的知识表示涉及的细节少,抽象程度高，因此表达自然，可靠性好，修改方便，但是执行效率低；过程性知识表示的特点恰恰相反。

实际上选取知识表示方法的过程也就是在表达的清晰自然和使用高效之间进行折衷。

2．问题求解系统中知识表示的（曾经的）基本方式和方法

经过国内外学者的共同努力,已经有许多知识表示方法得到了深入的研究,曾经研究和使用较多的知识表示方法主要有：**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、面向对象表示法、基于本体的知识表示法**等。这些表示方法也可归结为三类：

- **一阶谓词逻辑（First Order Predicate）表示法；**
- **产生式（Production）表示法；**
- **结构化表示法**[包括框架（Framework）表示法；语义网络（Semantic Network）表示法；剧本（脚本）（Script）表示法；过程（Procedure）表示法；面向对象（Object-Oriented）表示法；Petri 网（Petri Network）表示法；信念网（Belief Network）表示法等]。

可以认为，实际应用的各种知识表示方式仅是这三种基本方式的特化、变种或综合运用。如何选择合适的表示方法以充分表示领域知识，并有利于对知识的利用，便于理解和实现，便于对知识的组织、管理与维护，将是我们在构建实际系统时的考虑。而计算本质上就是将“逻辑”和“本体”映射成可计算的形式。

除了前面提到的知识表示方法以外,学者们还曾提出过许多其它的知识表示方法。例如:基于Petri 网的表示、基于关系模式的表示、基于事件关联图的表示、基于决策表的表示和问题归约表示等。这些表示方法曾各自适用于表示某种类型的知识,从而被应用于曾经的不同的领域。

一般来说,根据领域知识的特点,选择一种恰当的知识表示方法就可以较好地解决问题。但是，现实世界的复杂性造成各类专家系统（智能系统）的领域知识很难用单一的知识表示方法进行准确的表达；因此，许多智能系统的建造者们就采用了多种形式的混合知识表示方法，以此来提高知识表示的准确性以及推理效率。不但如此，有时为了开发具有较宽领域的知识系统，也需要选择多种知识表示或者采用多种表示方法相结合的办法来表示领域知识。

由于网络技术、计算机技术和智能技术发展非常迅猛，要想使智能系统也像其它软件行业那样得到快速发展，除了上面提到的混合知识表示方法以外，另外一个重要措施就是结合不断涌现的智能算法，提出和研究更有效的新的知识表示法。这样才能充分利用已有计算资源产生出功能更强的智能系统。

17.2.4 问题求解（系统）的结构化组织

17.2.4.1 知识系统开发的结构化组织需求

长期以来，知识系统的开发和维护问题就一直困扰着基于知识的问题求解系统的开发，而为了实现高性能的问题求解和系统的智能，现实又要求系统要向着大容量知识库和多种智能技术的集成的方向发展，这就更加剧了知识系统的开发和维护的难度。传统的观点曾把这种困难归因于所谓的“知识获取瓶颈”问题，并将知识获取比喻为采矿，而将知识系统视为存放知识的容器。但实际上，这种观点并未揭示造成困难的本质。知识是观察者对人的(可以是观察者自身)问题求解能力的抽象认识，而并非实际存在于人头脑中的记忆结构。因此，靠简单的知识诱导技术(如与领域专家会谈)去发掘求解复杂问题的知识是困难的。**放弃采矿观点，转而把知识获取视为建模活动，或增强智能系统自己获取知识的能力，将有助于克服（知识获取的）困难**。此时，**知识系统不再简单地作为存放知识的容器，而是模拟人的问题求解行为和应用领域世界的模型**。这就要求按系统化、结构化和功能化的方式来分析人的问题求解行为和领域世界。分析的结果将形成称为“概念模型”的框架，它可以作为知识获取和知识库维护的指导和约束。由此，作为建模活动的组成部分，知识获取(或更精确地讲，获取规则、事实和状态等详细知识条款)将不再是唯一的关键所在。**真正的瓶颈还在于建模分析---分析问题求解行为和领域世界**。

有人认为，传统的知识表示技术与人组织问题求解的思维方式常常严重失配，其面向符号级实现的固有特点，使其抽象级别太低，无法提供直观和易于人理解的方式表示问题求解的组织。为此，需要在易于人理解的更高级别去开展知识表示的研究，那就是纽厄尔提议的**知识级**。在知识级，问题求解行为是通过**目标集**和**动作集**加以刻画的；知识作为关联动作到目标的媒介，将基于简单的合理性原则。由于知识级分析不涉及符号级实现，强调按合理性原则对问题求解作功能化分析，因而十分符合人在组织问题求解时采用的思维方式。

需知识系统解决的问题往往是复杂的，且结构不良。在这种系统中，只给定推理部件的特性描述和推理部件间交互作用的法则，去（演绎）推理整个系统的行为，并不是容易的事。**为理解复杂性，层次分解是一种有效的手段，它层次地把复杂的问题分解为不太复杂的问题**。纽厄尔(Newell)把结构不良的问题视为只能通过不良（不严格）定义的目标来刻画，并缺少可预先确定的从初始状态到目标状态的决策手段。这种问题往往对解答的可接受性缺乏良好定义的准则，从而难以理解和解决。由此可见，对于复杂的且结构不良的问题，不存在简单的解决方法。从而，以前提-结论和条件-动作形式的关联知识来描述问题求解的组织，成为许多场合下的最佳选择。

表示和组合应用领域知识的最简单策略，是把问题求解所用的全部知识统统表示为规则，利用规则搜索从问题求解的初始状态到目标状态的路径。这种方式以专家系统开发工具 EMYCIN 和 OPS5 为典型代表，曾受到过人们的推崇。利用产生式规则来表示领域知识的确有许多优点，但随着要求解决的实际问题越来越复杂，规则库也越来越大，产生式系统的缺点也就逐渐显示了出来，具体表现为：

① **难以扩展**。尽管规则形式上相互独立，但在问题求解中却往往彼此相关，在推理过程中构成一定的推理网络。随着规则数目的增加，要使新加入的规则不与原有的规则发生矛盾变得越来越困难，即随着规则库的扩展，一致性维护越加困难。

② **选择规则的低效性**。由于规则是堆积在规则库里的，问题求解中的每一推理步都要对规则库作穷尽的匹配检查，以便选择合适的规则。显然，规则库越大，效率会越低。实际上，一个推理步只涉及若干特别的规则，是可以把它们构成一个规则组的。只有上一规则组用完后，才考虑下一规则组。

③ **不灵活的控制策略**。产生式系统往往采用单一的控制策略（例如按顺序考察规则库中每一规则），而实际问题的求解常常需要综合应用不同的控制技术。

④ **单一的表示形式**。尽管从理论上讲，产生式系统可以表示任何推理知识，但对于有结构的知识，或许以语义网络和框架系统等来表示更为有效。

为克服这些缺点，**将求解复杂问题的知识划分为一组相对独立的模块可能更为合适**。这与计算机软件工程强调软件设计的模块化是一致的。**模块化**的一个重要议题就是如何协调模块在问题求解过程中的相互合作。模块的划分可以是**面向动作**的或**面向对象**的。前者以推理动作的选取为核心组织问题求解所需的知识，例如在医疗诊断中，知识是以症状和疾病间的关联、疾病与治疗动作间的关联等方式表示和组织的；后者以概念和个体为单元组织问题求解所需的知识，例如可以建立关于疾病的分类体系和疾病症状的分类体系。换言之，前者面向“怎么做”进行知识的组织，后者面向“是什么”进行知识的组织。

17.2.4.2 早期的问题求解组织技术：事务表和黑板法

事务表(Agenda)是一张应由系统执行的事务的列表，也称任务表，是面向动作的问题求解组织方式之一。从组织问题求解的角度，事务表方法的意义在于它允许各个推理模块经由事务表相互通信，以求协调它们在问题求解中的合作关系。表中每个元素表示一个等待执行的任务，任务由特定的推理模块执行，按优先级大小排序。每个任务附有一张理由表，记载由其它任务提出的支持或反对执行该任务的理由。任务的优先级就是按该理由表来计算的。优先级最高的任务意味着它的执行最为紧迫和最有意义。虽然各模块可以用对执行某些任务提出支持或反对理由的方式通信，但模块之间保持着相对的独立性。

应用事务表的典型系统是 AM 系统。AM 从集合论的一些基本概念出发，能够不断地发现大量的数论新概念，从质数到哥德巴赫猜想。AM 把基本概念和其发现的新概念组织为一个概念网。概念以框架形式表示，并有启发式推理规则附加于各个表示基本概念的框架，用以指导新概念的提出和完善。AM 没有固定的推理模块，每当事务表中一任务取出执行时，根据任务的性质(建立或扩充某个新概念)以及该概念在不断向下扩展的概念网中的位置，系统就收集(沿着到顶点的路径)分散于基本概念框架中的启发式规则，并执行之。这样收集到的启发式规则就构成动态推理模块。

事务表示法可以获得大系统模块化设计的一切优点，而通过事务表把有关问题求解的各种信息汇集起来，又使独立的模块能在推理过程中合作求解问题。由于事务表中每个任务对应于一个子问题，将复杂问题分解为子问题的精细程度将是一个重要的决策。问题分解太细，必然会大幅度增加事务表中排序子任务的工作量；而太粗，又不利于发挥模块化的优点。这是一个需要权衡利弊的问题。

黑板法(Blackboard)与事务表都是面向动作的组织方式，用黑板法组织问题求解的典型方法是：系统由一组称为“知识源(Knowledge Source，KS)”的独立推理模块和黑板组成。每个 KS 包括三个部分：触发模式、直接码和体。触发模式以谓词公式表示，每当其与黑板上的内容匹配时，就产生一个激活记录，记载该 KS 的名字、触发上下文和模式中变量的约束值(因匹配而从黑板得到的值)。在产生激活记录的同时执行直接码，后者通常是一段 Lisp 代码，并参考触发上下文和模式变量的约束值加以执行，以便将该 KS 有关的调度控制信息加进激活记录。KS 的体包含为求解问题所需的专门知识，可以是一个规则组甚至一段任意的程序。KS 激活时，KS 的体通常不立即执行而是置于一排序表。一个调度程序分配和修改每个激活记录的优先级，并据此排序知识源体。具有最高优先级的 KS 体总是作为下一个要执行的，其参考激活记录中的触发上下文和模式变量的约束值。KS 的体一经执行，不能中断，直至结束。

黑板是所有 KS 可以访问的公共数据区。每个 KS 的体在执行时可改变存储于黑板中的数据，从而渐增性地形成问题的解答。黑板的内容由解答空间(状态空间)中的对象构成。这些对象可以是输入数据、部分解答和最终解答。黑板结构的设计反映了问题如何解决的骨架，以更有效地规划和组织 KS 的合作问题求解。通常，记载于黑板中的对象层次地划分到不同的分析级，每一级均设计一组相应的 KS，它们以该级对象的描述信息作为输入，并将计算(推理)结果加到该级或其它级。例如，HEARSAY-Ⅱ就将关于口语识别的假设(可能的解答，表示为对象)分为 8 个等级：从关于声音的低级假设到关于整个句子文法分析的高级假设。

17.2.4.3 问题求解建模---以问题求解建模为核心的结构化组织技术

专家系统（智能系统）开发的实践使人们逐渐认识到，要开发实用化的专家系统（智能系统），就必须对应用领域特征和问题求解任务作深入理解。这推动了问题求解结构化组织的研究向纵深方向的发展，出现了问题求解建模等研究热点。

把知识获取视为问题求解建模，并强调问题求解的结构化组织在开发和维护知识系统中的核心地位，已成为知识工程界的共识。如此，知识库的建立就不仅是知识的获取和编辑过程，更应是问题求解建模过程。知识库包含的模型可以分为两大类：问题求解行为模型和领域世界模型。前者描述问题求解过程的结构化组织，包括推理知识和推理控制知识；后者描述领域世界中的对象以及对象间的约束关系，包括结构、功能、因果和行为模型。这里，我们首先考虑问题求解行为模型的设计以及相应的知识获取方法。

知识获取过程就是把用于问题求解的专门知识从某些“知识本源”中提炼出来，转化为推理机（推理机制）可使用形式的过程。**潜在的“知识本源”包括领域专家、书本、数据库、网络信息以及普通人的经验等**。过去，曾想以知识工程师作为中间人从领域专家处获取专门知识；知识由知识工程师通过与专家的会谈抽取出来，并转变为适合于推理机使用的形式加以存放。一般来说，应用领域的基本原理和常识易于获取，较困难的是领域专家的经验知识，特别是关于问题求解组织和推理控制的知识。这些知识决定了系统的体系结构，并可指导以系统化和结构化的方式获取详细的推理知识。如今，网络技术与智能技术的发展已经打破了原有的知识获取模式，取而代之的是知识的自动获取。已有的知识获取程序模块可提供知识的获取、编辑、查错和一致性维护等功能，有效提高了知识获取的效率。

知识获取自动化取消了知识工程师的中介作用，它是让一个智能化的知识获取系统直接从文献中获取知识或直接与领域专家对话来获取知识。这些知识自动获取系统可按预先制定的面向特定问题求解方法或任务类的体系结构，以结构化交互的形式来获取知识。

如今，通过知识系统开发工程师与领域专家协作来开发一个（问题求解）智能系统，可归纳为如下几个步骤：

① 识别阶段。智能系统开发工程师和领域专家一起判别智能系统待处理问题的类型和特征，确定子问题划分、数据和专业术语、问题解答的形式、专家知识的核心和解决问题时的困难所在。另外，识别专家解决问题所采用的基本策略也很重要，这将成为组织问题求解和推理控制的依据。

② 概念化阶段。阐明重要的概念、关系和信息流特征，并用以描述问题求解的概念模型，包括问题求解方法、推理控制要求和约束条件等。

③ 形式化阶段。决定知识表示形式和推理机制，智能系统开发工程师可以根据已得到的求解问题的概念模型设计特别的知识表示语言和推理机制，或者选择适合的智能系统开发工具(工具可提供了特定的表示语言和推理机)。

④ 实现阶段。以概念模型作为框架，获取问题求解所需的详细知识，以形式化阶段决定的知识表示语言编写并存放进知识库。新建立的知识库和推理机等将构成系统的第一个原型。

⑤ 测试与调试阶段。通过各种测试手段评价原型系统的性能。用特定实例从头到尾测试原型系统是必要的，请领域专家参与评价则有助于从实用角度全面观察原型系统。原型系统往往不完善，需根据测试评价的结果返回到实现阶段，调试、修改、精化和完善原型。若发现更严重的问题，则需返回到形式化阶段，甚至概念化和问题识别阶段。

值得注意的是，上述阶段实际上并无清晰的界限，也没有严格的定义，甚至它们之间并不独立，充其量只是粗略地表征出了被称之为知识获取的复杂过程。例如，形式化和实现阶段就是紧密相关的，当实现阶段遇到困难时，往往会立即导致重新形式化。

人工知识获取是一个十分困难而又耗时的过程，随着知识系统向大型知识库（大模型）方向的发展，自动知识获取已成为构造知识系统的方向。**智能化的知识自动获取系统可以克服低级的知识表示语言引起的失配问题，因为其通过问题求解模型直接提供了组织问题求解的框架，但预先确定的体系结构也易于引起与领域专家行为的失配问题。如何克服这些失配问题成为知识工程研究的热点。也许机器学习是解决知识获取困难的最理想途径。**然而机器自动学习曾经难度太大，尽管不少人已经致力于这方面工作，但效果曾经并不理想。**如今，生成式人工智能在这方面已经有所突破，由此也引发了一场人工智能的新的革命。**

设计问题求解行为模型的最重要方面是概念模型，可以把概念模型视为描述领域专家问题求解过程的本体，它提供基本术语和术语合成法则去描述问题求解所涉及的实体、它们的属性和关系以及在求解问题中所起的作用。为此，概念模型是描述问题求解的抽象框架，也是设计建模语言的基础。概念模型往往隐含于推理机制和知识获取工具（交互式建模语言），以控制问题求解过程和指导知识获取，概念模型可归纳为三个主要范畴：基于表示、方法和任务的概念模型。

（1）基于表示的概念模型。这种概念模型直接反映与推理机关联的符号级表示，是设计符号级表示语言和推理机制的基础，传统的知识系统及开发工具的概念模型均属于这种类型。例如，曾经的 EMYCIN 概念模型就是逆向推理链和逆向推理规则集；而 KAS 的概念模型是由推理规则的互联而构成的语义网络。开发者必须理解这些概念模型，才能正确使用这些工具去构造问题求解行为模型。

基于表示的概念模型的主要弱点在于完全面向符号级建模分析，忽略了人的认识行为处于知识级这一特点。所以，在使用这些工具建模(建立知识库)时，知识级分析完全是系统开发者（知识系统开发工程师）的事情，而且开发者必须清醒地认识到他所作的知识级分析的结果能手工地转变为符合概念模型指定的符号级表示形式。若开发者不能充分理解工具隐含的概念模型，就会发生知识级分析的结果与符号级实现的严重失配。此外，知识级分析的一个重要方面是决定问题求解的方法，但这些工具不提供任何支持手段，这将严重影响知识系统开发的效率。

（2）基于方法的概念模型。这种概念模型面向知识级建模分析，提供预先定义的方法，使开发者建模的注意力集中在获取实现方法所需领域的特有知识，而不是规则和框架等符号级表示结构。这种概念模型可提供一组基本术语去描述在特别应用领域中实现方法的有关知识。例如，克拉希（Clancey）曾提出启发式的分类方法，基于这种方法的概念模型要求开发者按照解答集、数据抽象、启发式匹配和解答精化等去分析和建立知识级模型。其中，解答集包括可枚举的所有可能的解答。数据抽象将关于原始状态的信息抽象为高级的定性概念，启发式匹配要求提供定性概念和解答范畴之间的关联知识，再经精化处理，对应到解答集中的某个解答。再如，ROGET 提供的概念模型是把问题求解视为一种特别形式的启发式分类。动作、原因、问题和证据构成了描述问题求解的基本术语，而术语间的关系--数据抽象(从证据到问题)、启发式联想(从问题或证据到原因)和解答精化(从原因到动作)--的联合就构成了指导知识级分析的合成法则--启发式分类方法。只要问题求解任务适合于使用启发式分类方法，这个概念模型就可以快速地指导开发者去建立起相应的知识系统。比如，为建立医疗诊断模型(专家系统)，可以把每一种疾病对应的观测数据作为证据，然后描述疾病或观测数据与细菌之间的启发式关联，最后指出细菌和抗菌素之间的对应关系。一旦基于此概念模型的知识级分析结束，ROGET 就根据术语间关系自动把用户输入的信息组装成推理规则(符号级表示)，进而构成知识库。ROGET 曾提供了有效的手段，使开发者能在知识级描述知识系统的问题求解行为--关于如何求解问题的知识级说明，但开发者不能随意发挥，受限于概念模型的约束，不能描述超出启发式分类的行为。

基于方法的概念模型的主要缺点是应用领域与概念模型的失配问题。开发者必须认识到由概念模型预先确定的问题求解方法适合于其面对的问题，并有能力正确而一致地使用概念模型提供的基本术语；否则，就不能开发出好的问题求解模型。

(3) 基于任务的概念模型。这种概念模型不面向通用的问题求解方法，而只面向特别种类的任务；直接刻画任务结构而非执行任务的方法，因而可避免因开发者不能正确理解问题求解方法而产生的问题。此外，许多任务可综合应用多种通用的问题求解方法(弱法)，因而直接刻画任务结构更

为合适。例如，OPAL 就曾是一个包含这种概念模型的知识获取工具，用于抽象描述癌症治疗协议。尽管可以制定不同的协议，但协议的格式是类似的，仅内容不同。为此，可以抽象出描述协议的概念模型。具体的协议通过示例概念模型而生成，多个协议构成知识库；每个协议作为治疗方案的骨架，基于协议库的推理(问题求解)就是按病人的特别症状(加测试数据)选用协议并精化某个协议为具体的治疗方案。所以，概念模型隐含(而不是显式地表示)的问题求解方法又称为骨架规划精化。

OPAL 的概念模型曾提供了描述特定协议的基本术语和合成法则(如何构成一个协议)。由于概念模型可视为描述问题求解过程的本体，所以，可以从本体的观点来考虑 OPAL 提供的概念模型。本体从三个方面支持关于问题求解的知识级分析：① 静态本体。指定问题求解涉及的实体、属性以及实体间的关系。② 动态本体。定义问题求解的状态和状态间的转变以及一定状态下可做的动作。③ 认识本体。描述状态转变(或动作)的条件以及条件中涉及数据的定义。就 OPAL 的概念模型而言，静态本体提供了预先指定的规划实体和它们的关系；实体则形成一个三层结构。每个协议可以包含多种治疗方式--化疗和放射疗，每种化疗又可采用多种药物。除给定这些实体和层次关系外，各实体以哪些属性来描述也作了指定(包括属性的特点：值范围、值类型和值测试方式)，只留下属性值由系统的开发者来填充。动态本体提供描述治疗规划的手段，包括治疗状态、状态转变的图编辑手段，以便开发者指定化疗和放射疗交替使用的序列。也存在一些开发者修改治疗规划的动作，如改变化疗中某药剂的剂量、改换药物、中止或推迟化疗等，这些动作作为动态本体提供的基本术语。最后，认识本体支持描述动作发生的条件，并提供手段去定义条件中涉及的各种观察和实验室测试数据。OPAL 提供了结构化图表式人机界面，引导开发者基于本体的这三个方面去作知识级分析、填充概念模型的细节以及生成特别的治疗协议。OPAL 可将特别的治疗协议--知识级分析结果，自动转变为符号级可执行的表示形式。从建模的观点，治疗协议就是一个问题求解行为模型。

按基于特种任务的概念模型来建立问题求解模型的主要缺点在于，要解决的问题必须适合于概念模型。由于概念模型几乎完全确定了问题求解的控制流程，开发者只能填充细节内容，而不能改变流程的结构化组织，所以，这种建模方式只适合于很受限制的范围。当然，若实际问题刚好适合于概念模型，则用户不须自行设计控制流程的结构化组织，只要直接填入内容知识(模型的细节)即可。这时，领域专家只需稍加训练，就能自行设计问题求解的过程模型--在包含概念模型的知识获取工具的指导下填入内容知识，而免除知识工程师的介入。这种方式可显著提高知识获取的自动化程度，是知识获取研究追求的目标。可惜，这样的情况并不多见，除非有专门为特定求解任务而设计的这种获取工具。

总之，构造问题求解模型的一个核心工作是建立概念模型，概念模型可分为三大类：基于表示、方法、任务的概念模型。三者的主要差别在于建模所需的知识级分析(知识获取的概念化阶段)由谁来负责：基于表示的概念模型把知识级分析完全推给了建模者(用户)，容易造成知识级分析结果与隐含于推理机的概念模型失配。基于方法的概念模型以特别的问题求解方法指导建模者作知识级分析，建模者理解问题求解方法是建模的基本保证。基于任务的概念模型以应用领域的任务结构来指导建模者作知识级建模分析，建模者只须填进模型内容即可，因此这是一种最便于为非知识工程师使用的概念模型。

对于每一种任务都设计相应的知识获取(建模)工具，显然不经济，尤其是采用同样问题求解方法的不同类任务，势必要做大量重复的设计工作。于是，人们提出为知识获取工具研制开发工具的设想。例如，以包含基于方法的概念模型的工具来快速生成包含基于任务的概念模型的工具，由此

建立问题求解过程模型的步骤可表示为：建立基于方法的概念模型--建立基于任务的概念模型--建立求解手头问题的模型。例如，PROTEGE 就曾是这样的一个知识获取工具的开发工具。

17.2.4.4 智能系统开发研究中曾提出的其他一些结构化组织技术

以问题求解建模为核心的结构化组织技术的研究，曾推动了智能系统向实用化和高性能的新一代知识系统方向的发展。这些技术的共同特点，是强调对应用领域和问题求解任务作深入和系统化的理解，可视为是对基本知识系统技术的开拓型或纵深型发展。这些技术除问题求解建模外，还有定性物理、基于模型的推理、深浅层推理的综合、功能化体系结构、知识级问题求解建模等。

1．定性物理

定性物理的研究目标在于开发基于定性演算的物理系统行为模型。这些模型描述应用领域的基本原理和关于应用领域中事物的常识（结构、功能、因果和行为）。由于这些知识是应用领域的基础知识，能揭示事物的本质，我们称其为深层知识。与之相对照，领域专家的经验知识主要用于指示事物间现象上的关联，尽管因其启发式作用而使推理效率大幅度提高，但并不揭示事物的本质，所以我们称其为浅层知识。

定性物理模型属深层知识，其运用定性方程表示控制物理系统行为的法则，并通过定性演算和状态转变规则来预言和仿真物理系统的定性行为。定性物理的研究出现了三个有影响的典型方法：ENVISION、QPT 和 QSIM，它们分别以面向部件、过程和约束的方式对物理系统作分析。

相对于定量分析和定量仿真来讲，定性物理方法以牺牲对物理量描述的精确性为代价来换取对物理系统行为的推理能力，正好弥补了定量方法缺乏推理和解释能力的不足。然而定性物理方法并不能取代定量方法，为此理想的物理系统分析方法应有机地综合定性和定量方法，以取得更好的效果。

2．基于模型的推理

这里的模型指物理系统或抽象系统（如规则、思维）的结构、功能、因果和行为模型。基于模型的推理的一个主要目的是寻找事物变化的原因和结果，所以建立因果模型是达此目的之关键；结构和功能模型仅静态地描述系统状态，通常只作为因果模型的附加部分；行为模型则主要描述控制系统行为的各种约束关系。可以说定性物理模型也是因果模型，因为基于这种模型的推理结果表示为系统状态的转变序列，而状态的转变实际上描述了因果关系。但这样获得因果关系太低效了。我们需要直接表示事物间的因果关系，并在此基础上建立较高级的行为模型。

查德雷斯卡雷（Chandrasekaran）于 80 年代后期提出了基于功能的因果建模方法，旨在更有效地组织关于因果关系的知识。其分三个方面：结构、功能和行为，去描述物理系统，特别是将实现物理系统及其构件功能的行为表示为因果网片断。该方法并不建立对应于系统的整个因果网，而是将整个因果网层次地分解为片断，作为物理系统各层次构件行为的描述，并以结构、功能和行为为索引快速地收集与当前问题相关的因果网片断。基于这种模型的推理并非简单地搜索因果网中的路径，而是先将层次存放的相关因果网片断搜索到，再装配成适合于当前问题初始状态的特别因果网。这种方法与定性物理方法的不同之处在于，它不需要推导出因果关系，只是收集已存在的因果关系（可视为已事先通过定性物理方法推导出的结果），从而比定性物理方法高效得多。

3．深、浅层推理的综合

深层知识的量往往很庞大，而基于深层知识的推理步又很精细，可想而知，其推理效率是相当低下的。相比之下，浅层知识相当精炼，效率高。但具有脆弱性。所以，应综合深、浅层推理，即

以浅层推理为核心和主干，并将深层推理作为补充和后盾。另外，深层推理也应在浅层推理的指导下进行。例如，可通过浅层推理得出部分解答或将解答限制在一定的范围内再驱动深层推理，以大大提高深层推理的效率。而深层推理的结果又可为浅层推理所用。

尽管基于因果模型的推理比定性物理方法的效率高得多，但与基于经验性关联知识的浅层推理相比，效率仍然是低下的，毕竟深层推理是以牺牲效率为代价来换取其鲁棒性。显然，在实际应用的大多数场合，仅依赖于深层推理是不可取的。实际上，人类专家往往灵活地组合应用浅层知识和深层知识进行推理。通常，首先尝试更为有效的浅层推理知识，只有在其失败的情况下才求助于深层知识。深层知识本身又有层次，其中定性物理方法是较低层次的知识。在某种意义上，数值分析也是一种特别形式的推理，可以解决非数值推理的不确定性。因此．在典型的情况下，人类专家解决问题涉及到四个层次的知识经验性关联和推理：经验性关联--不精确模型--定性物理模型--定量模型。上述基于功能的因果模型属于不精确模型的范畴，因为其在描述物理系统行为时，比定性物理方程更粗略。

不精确模型知识的分层也给自适应学习创造了条件，这种学习方式是指系统在遇到新问题时，只能依赖于深层知识去推理，这时系统问题求解性能不高。但一旦问题求解成功，问题求解涉及的知识就可加以归纳、整理和编辑，形成新的浅层知识--经验性关联，或精化原有的经验性关联知识。以后再遇到类似问题时就可依赖于新增加的或精化的浅层知识高效地解决问题，从而表现出较好的自适应能力。

4．功能化体系结构

基于概念模型来组织问题求解的主要缺点在于容易导致待解决问题的特征与预先制定的概念模型的失配。对基于表示的概念模型，还存在着另一种失配--知识级分析和符号级实现之间的失配。由于概念模型直接面向符号级实现，且隐含于开发工具，如何作知识级分析，以及如何把知识级分析的结果(包括问题求解的组织和推理控制过程)转变为符号级表示形式，完全是知识系统设计者的工作。基于方法，尤其是基于任务的概念模型解决了这一失配问题，因为它以结构化的方式指导用户作知识级分析，并把分析结果自动转变为符号级可操作形式。然而，结构化的概念模型也进一步限制了其适用范围，从而往往加剧了待解决问题的特征与预先制定的概念模型的失配。其原因在于，当概念模型转变为符号级表示时，它仅仅使用常见符号级表示语言的子集。例如，启发式分类方法的概念模型可以转变为逆向链推理表示方式，但反之不一定行，因为表示启发式分类的逆向链推理仅是逆向链推理的一种特别情况。PROTEGE 在一定程度上解决了后一失配问题(根据问题特征动态生成基于任务的概念模型)，但仍存在方法(其采用骨架规划精化作为预先制定的方法)与问题特征的失配问题--问题求解方法预先固定于概念模型。

为克服基于预先固定的概念模型来设计知识系统而产生的失配问题，Chandrasekaran（查德雷斯卡雷）曾提出了以常见任务(Generic-Task)方法去实现问题求解的功能化体系结构。这种方法要求以功能单元来构造基于知识的问题求解系统，而不是以预先固定的概念模型强加于系统。为实现智能软件的重用和加速知识系统的开发，可以将每个功能单元的实现视为常见任务的示例，并为每个常见任务提供开发工具(包括知识表示语言和推理机)。鉴于复杂的问题求解任务常可以层次地分解为子任务，只要预先建立一个范围充分广阔的常见任务集，就可通过组合应用适当的常见任务来快速设计知识系统。

5．知识级问题求解建模

基于概念模型的问题求解建模方法以预先制定的概念模型方式提供固定的推理模式(问题求解的控制结构)，并提供执行这种推理模式的推理机；所以，用户只要在概念模型的指导下填充内容知识(详细的专门知识)，就可建立一个可执行的知识系统。然而，固定的概念模型往往与特定问题求解任务的要求和领域特征失配。对于复杂的任务，更难找到合适的建模工具。基于常见任务来实现问题求解的功能化体系结构，将常见任务示例为功能单元去装配复杂的知识系统，虽然在一定程度上克服了失配问题，但仍存在完备性和难以使用的问题。鉴于各种应用领域和问题求解任务的特征千差万别，不可能设计完备的常见任务集；当常见任务集较大时，理解每个常见任务的功能和适用范围(以便正确使用)本身就不是容易的事情。还有，目前尚缺乏有效的方式去综合应用选中的常见任务，而这种综合本身就是一个知识密集型的任务。随着问题求解任务的越趋复杂，深入理解应用领域和问题求解任务，就成为了提高知识系统性能的关键。从而在某种意义上，建模 shell(提供固定的概念模型)和常见任务方法往往成为开发深入理解的障碍而不是帮助。而基于知识的问题求解系统环境，是把开发对应用领域和问题求解任务的深入理解作为核心任务，也许会成为更有潜力的知识级问题求解建模方法。提供知识级建模语言去建立面向应用任务的概念模型，而不强加预先固定的概念模型，可较好地消除失配问题。一旦建立好知识级模型，就可手工或自动转变为符号级问题求解模型，再用符号级推理和控制机制加以执行。

17.2.4.5 问题求解系统中的推理解释机制

鉴于知识系统常常是面向复杂的结构不良问题，解释问题求解过程及结果的合理性也就成为了知识系统应具备的能力。简单的解释方式是规则追踪，已为大多数知识系统所应用。这些系统是基于规则的推理系统，规则追踪就是把问题求解过程中激活使用的规则按激活的次序显示给用户。更优雅的方式是将规则转变为自然语言的形式显示，以便用户理解。

规则追踪适合于审查推理过程的正确性，可作为维护和扩充知识系统时的调试手段；但因未按应用领域的基本原理和常识证实推理结果的正确性，推理结果往往不能令人信服。高级的解释功能应具有按领域基本原理和常识重构解答的能力。重构的推理路径未必是实际的推理路径，但却能令人信服地验证解答的合理性，从而表示出高级的智能行为。

当然，以哪种方式作解释要根据实际需要决定。例如，对于使用精确(非启发式的)规则作推理的演绎信息系统来讲，解答的正确性用户自明，只需提供规则追踪即可；而当系统解答的可信度不高，风险又大(如医疗诊断)时，给出令人信服的详尽解释尤为重要。此外，随着多媒体技术的发展，综合使用自然语言、表格、图形、声音和图像等去提供强大的解释功能已成为可能。

17.2.4.6 问题求解系统的评价与改进机制

系统评价有多个方面，其中最重要的是：计算、感观和性能。**计算**方面包括运行速度、存储空间、可扩展性、可移植性和是否易于与其它软件集成。**感观**方面包括易用性、可理解性、自然性和是否提供联机帮助和解释。**性能**方面包括能力范围、得出非正确解答的比率、时间和资金的节省等。实践中，知识系统与常规软件的评价常使用同样的方式。首先由知识系统的设计者用各种可能的实验测试，确保无误后再交给用户；用户以大量实际案例运行知识系统，并与原有方式执行的结果相比较；一旦发现错误就立即作修改，直到用户信服知识系统的有效性，然后才正式投入应用。

对于小型常规程序，我们常以变量的边界值来测试每个例程的每个可能分枝，以便确信程序能产生拟定的行为。然而对于大型系统，完全的测试是不可能的。软件工程的实践提议测试边界条件，在尽可能多的变化中试用新代码，用大量案例作经验性测试，但仍不能保证完全的测试。因此，大

型系统常常是一个在应用中不断完善的过程。

在知识系统中，知识库的每一个元素可以用小的单一子例程的方式来测试。不可预见的错误往往出现在元素的交互作用中，只能通过经验性测试来揭示，即收集大量的随机性问题样本(在指定的范围内)，并通过系统的执行来决定哪些样本可解决，哪些不能。在缺乏完全的逻辑分析去证实知识库和推理机正确性的情况下，必须作经验的性能分析。对于任何类型的错误，其容许出现的程度必须通过权衡错误导致的损失和正确解答带来的利益来决定。

对系统的性能的改进包括结构方面的改进和知识方面的改进。它们是一个渐进的过程。

17.2.5 基于符号-逻辑的问题解决系统（完备或不完备的逻辑推演系统）

以模拟人类逻辑推理的方式来解决问题并构建问题求解系统，是智能模拟较早的一种考虑。其基本思路是：人类问题求解的过程，本质上是一个理性思维的过程，是一个由已知的事实和经验为基础，通过理性思考对待解问题或其解决方案做出判定和论证的过程。论证即逻辑推理。

17.2.5.1（曾经的）知识的逻辑表示法

逻辑表示方法是一种基于**形式逻辑**(Formal Logic)的知识表示方法。它通过引入符号、谓词、函数等来对知识加以形式化描述，以获得有关知识的**逻辑公式**，并利用特定的逻辑公式来描述对象、性质、状况和关系等，是人工智能领域中使用最早的知识表示方法之一。其基本思想是把问题中的逻辑论证符号化，希望采用数学演绎的方式，通过证明一个新语句是否可以从那些已知正确的语句推导出来的，从而来断定这个新语句是否正确。在基于知识的逻辑表示的问题解决系统中，知识库可以看作是一组逻辑公式的集合，知识库的修改就是增加或删除逻辑公式。使用逻辑法表示知识，需将以自然语言描述的知识通过引入谓词、符号和函数等来加以形式描述，获得有关的逻辑表达式，进而可根据问题采用归结法或其它方法进行推理演算等。

最简单的符号逻辑是命题逻辑[不含量词也不含参量]。命题即可断定真假的陈述句。如“P”表示“命题 P 成立”。由联接符(→蕴涵，∧合取，∨析取，等价⇔)将命题连接在一起即构成命题公式，简称公式，例如公式：(P∧Q)→R 。不含任何连接词的命题公式称为原子公式或称原子，例如 P，Q；原子公式及其否定形式（¬或～）称为文字，例如 P，¬L；文字的析取称为子句。

命题逻辑中的归结推理方法是：

若有命题“$A_1,A_2,...A_n$”和“B”，要证明在“$A_1∧A_2∧...∧A_n$”成立的条件下有“B”成立，或说“$A_1∧A_2∧...∧A_n→B$”。

很显然“$A_1∧A_2∧...∧A_n→B$”等价于证明“$A_1∧A_2∧...∧A_n∧¬B$”是矛盾(永假)式。

归结推理的方法就是：从“$A_1∧A_2∧...∧A_n∧¬B$”出发，使用归结推理规则来寻找矛盾，从而证明“$A_1∧A_2∧...∧A_n→B$”的成立。这种方法也称为反演推理方法。

设有两个子句“$C_1=P∨C_1'$，$C_2=¬P∨C_2'$”，从中消去互补对，所得新子句“$R(C_1,C_2)=C_1'∨C_2'$”称为子句“C_1'，C_2'”的归结式。没有互补对的两子句没有归结式。归结推理规则就指的是对两子句做归结，也即求归结式。

可以证明，归结推理规则是正确的，即 C1∧C2=>R(C1,C2)，也即归结式是两子句的逻辑结论，或者说任一使 C_1，C_2为真的解释 I 下必有 $R(C_1,C_2)$也为真。特别地，当 $C_1=P$，$C_2=¬P$ 时，$R(C_1,C_2)=□$，记作为空子句。显然 C_1,C_2同时为真是矛盾，或者说空子句□体现了矛盾。

命题逻辑的归结推理过程是：

先利用逻辑蕴含式和逻辑等价式将命题公式化成合取范式；将若干个子句的合取式中的合取词

∧换成逗号，形成的集合称为子句集；从子句集 S 出发，仅对 S 的子句间使用归结推理规则（即求归结式），并将所得归结式仍放入 S 中，进而再对新子句集使用归结推理规则，重复这些步骤直到得到空子句□（说明有矛盾）,便说明 S 是不可满足的，从而与子句集 S 对应的定理是成立的。

例如,欲证(P→Q)∧¬Q==>¬P ,先将已知以及结论公式化成合取范式

(¬P∨Q)∧¬Q∧(¬¬P)==>(¬P∨Q)∧¬Q∧P)

得到子句集　S= {¬P∨Q，¬Q，P}；

对 S 作归结：(1) ¬P∨Q ，(2) ¬Q，(3) P；(4) ¬P[对(1)(2)求归结式]；(5)□[对(3)(4)求归结式]；得证。

最常用的符号逻辑是谓词逻辑，谓词逻辑是一种表达力很强的形式语言，是人工智能中最常用的逻辑表示方法。谓词逻辑中的替换合一技术，也是符号推理中模式匹配的基本技术。具有以下优点：

① 对如何由简单命题构造关于复杂事物的知识有明确、统一的规定，并且有效地分离了知识和处理知识的程序，结构清晰；

② 谓词逻辑与数据库，特别是与关系数据库有密切的关系；

③ 一阶谓词逻辑具有完备的逻辑推理算法；

④ 逻辑推理可以保证知识库中新旧知识在逻辑上的一致性和演绎所得结论的正确性；

⑤ 逻辑推理作为一种形式推理方法，不依赖于任何具体领域，具有较大的通用性。

但是，谓词逻辑表示法也存在着一些缺点：

① 难于表示过程和启发式知识；

② 由于缺乏组织原则,使得知识库难于管理；

③ 由于是弱证明过程，当事实和逻辑公式的数目增大时,在推理过程中有可能会产生组合爆炸；

④由于表示的内容与推理过程的分离，推理按形式逻辑进行，内容所包含的大量有用信息被抛弃，这就使得处理过程加长、工作效率降低；

⑤ 谓词逻辑适合表示事物的状态、属性、概念等事实性的知识，以及事物间确定的因果关系，但是难以表示不确定性的知识。

一阶谓词逻辑的基本表示元素为谓词公式、联接符（与、或、非、蕴涵）和量词（全称和存在）。谓词公式由谓词符号及若干参数项组成，参数可以是常量、谓词演算变量和函数。一阶谓词逻辑限定谓词符号和联接符不能是变量。这种表示方式的优势在于表示元素具有良好定义的语义，具有自然性（接近自然语言，容易接受）、精确性（可表示精确知识）、严密性（有严格的形式定义和推理规则）和易实现性（易于转换为计算机内部形式）的特点，便于自然地表示知识，准确而灵活，具有很好的表示充分性，从而使得基于这种表示方式的归结定理证明方法能适用于各种应用领域。然而，归结原理不能应用启发式知识控制推理，因而它有在知识库较大时推理效率较低的缺点，使其不能适用于多种实时领域。为增进实用性，人们对一阶谓词逻辑作了种种限制。例如，引入封闭世界假设，限制知识库只能由两类符号结构--事实和规则组成，对事实和规则的表示方式作进一步限定（如事实只能表示为不带量词和连接词并不含变量的谓词公式；限定规则的左部只能是单文字，规则的右部只能是单文字的与、或组合等）。但这些仍未提供有效的手段控制推理过程。另外，其表示知识的符号结构过于简单，无法有效地描述结构复杂的事物，也无法表示不确定性知识，从而使得其所能表示的知识范围太狭窄；它难以表示启发性知识及元知识，以及推理与知识的语义完全

割裂的特点，也使其不能充分利用与问题本身特性有关的知识，造成效率低下。

一阶谓词逻辑表示方法中，谓词表示个体对象之间的关系、属性或状态，其表示形式为：

$P(x_1, x_2, x_3, \ldots x_n)$，

其中：P 是谓词符号，表示 $x_1, x_2, x_3, \ldots x_n$ 等个体对象之间的某种属性、状态或关系。$x_1, x_2, x_3, \ldots x_n$ 是谓词的参量（或称项），一般表示个体，它可以是个体常量、个体变量或是个体函数。个体变量的变化范围称为个体域（或论域）。

例如，FRIEND(a,b)：表示“a 和 b 是朋友”；father(b)表示“个体 b 的父亲”，FRIEND(a,father(b))表示“a 和 b 的父亲是朋友”。

通常约定：用大写字母表示谓词符号，如 P，Q，R，S 等；用小写字母 x,y,z 等作为个体变元符号；用小写字母 a,b,c 等作为个体常元符号；用小写字母 f,g,h 表示个体函数符号。

当一谓词公式中不含变量，或变量值均取定时，谓词公式所表示的事物间的关系也就唯一确定。若这种关系在应用域确实存在，则谓词公式取值为真（记为 T），否则为假（记为 F）。在这个意义上，每个谓词公式均有一个确定的真值：T 或 F。

谓词公式是谓词演算的基本单元，也称为原子公式。通过引入连词和量词，可以把原子公式组合为复合谓词公式。通常将复合谓词公式称为逻辑语句，所以谓词演算也称为谓词逻辑。

在谓词公式前面加连词“¬或～”，称为否定，也叫做取反；用连词“∧”连接谓词公式称为合取，产生的逻辑语句称为合取式，其每个成份称为合取项；类似地，用连词“∨”连接谓词公式称为析取，产生的逻辑语句称为析取式，其每个成份称为析取项；用连词“→或⇒”连接谓词公式产生蕴涵式，连词“⇒”左部称为前项，右部称为后项；类似地，用连词“⇔或<->”连接二个谓词公式产生等价式，等价式可以视为正、逆向二个蕴涵式的合取。连词的应用可以扩展到连接任意复杂的逻辑语句。

谓词逻辑中量词包括：**全称量词**--表示“所有”，“一切”，“任一”，“凡是”等，记为∀x。即以符号（∀x）P(x)来表示对于某个论域中的所有（任意一个）个体 x 都令 P(x) 真值为 T 。**存在量词**--表示“存在”、“有些”、“至少有一个”、“有的”等，记为∃x，即以符号（∃x）P(x)来表示某个论域中至少存在一个个体 x 使 P(x)真值为 T。这里，P(x)可以是任意逻辑语句，除 P(x)是原子公式外，P(x)应用括号括起来。

引入量词和联接符（→蕴涵，∧合取，∨析取，否定词¬或～）之后，谓词的表达能力可大大扩充。例如，

若 M(x)表示“x 是人”，N(x)表示“x 有名字”，则

（∀x）(M(x)→N(x))表示：“凡是人都有名字”。

（∀x）[Road(x)⇒ Lead(x, Roma)]表示：“条条大路通罗马”；

而式（∀x）(∃y）[Person(x)∧Book(y)∧Give(Mary,x,y)]表示：“Mary 给每个人至少一本书”。

不含有任何联接符以及量词的谓词公式称为原子谓词公式。如 P(x)。原子谓词公式或其否定形式统称为文字。如 P(x)，¬P(x) ；若干个文字的一个析取式称为子句，如 P(x)∨¬R(x,y)；紧接于量词之后被量词作用（即说明）的谓词公式称为该量词的辖域。量词后的变元如 ∀(x,y)中的 x,y 称为指导变元，其取值仅在量词的辖域内有效；而在一个量词的辖域中与该量词的指导变元相同的变元称为约束变元；其它变元成为自由变元。例如： 在（∀x）[A(x)∧B(y)]中，x 是指导变元，x

为约束变元，y 为自由变元。自由变量是相对的，例如：（∀y）[P(y) ∨（∀x）Q(x，y)]，y 在全称量词（∀x）辖域内是自由变量，但在全称量词（∀y）辖域内成为约束变量。

由于一个变元既可以约束出现,又可以自由出现，为了避免混淆，可实行改名原则，使得一个公式中一个变元仅以一种形式出现。改名原则如下：（1）应同时更改变元在量词及其辖域中的所有出现；（2）新变元符号必须是量词辖域内原先没有的，最好是公式中也未出现过的。

若谓词公式 A 有如下形式：$B_1 \wedge B_2 \wedge \ldots \wedge B_n$,其中 $B_i(i=1,2,\ldots n)$形如 $L_1 \vee L_2 \vee \ldots \vee L_n$，并且 $L_1, L_2, \ldots L_n$都是文字,则称其为谓词公式A合取范式,如,$(P(x) \vee \neg Q(y)) \wedge (\neg P(x) \vee R(x,y)) \wedge (\neg Q(y) \vee \neg R(x,y))$。应用逻辑等价式可以将任一谓词公式化成合取范式。若谓词公式 A 有如下形式：$B_1 \vee B_2 \vee \ldots \vee B_n$,其中 $B_i(i=1,2,\ldots n)$形如 $L_1 \wedge L_2 \wedge \ldots \wedge L_n$，并且 $L_1, L_2, \ldots L_n$都是文字，则称其为谓词公式 A 的析取范式，如，$(P(x) \wedge \neg Q(y) \wedge R(x,y)) \vee (\neg P(x) \wedge R(x,y))$。

应用逻辑等价式和逻辑蕴含式可以将任一谓词公式化成析取范式。若 P 是谓词公式，D 为论域，对于 D 中任一解释 I：若 P 恒为真，则称 P 在 D 上是永真式。若 P 恒为假，则称 P 在 D 上是永假式。若至少有一个解释，可使 P 为真，则称 P 在 D 上是可满足或是 D 上的可满足式。

要想通过一个算法证明一个谓词公式的永真性是不可能的，因为谓词公式的值与论域和论域上的解释有关。但我们可以证明一个谓词是可满足的或不可满足的（这称为半可判定性），这正是我们后面所学的归结演绎推理的基础。

将所有的量词都放在谓词公式的前面的范式称前束范式，如，∀(x,y)(P(x,y)∧Q(a)∧R(z))就是前束范式。∀x(N(x)∧∀yD(y,x))就不是前束范式。将一个谓词公式中所有存在量词消去之后，所得到的范式称为该谓词公式的 Skölem 标准型。

将一个谓词公式转化为子句集的步骤如下：

（1）消去蕴含式和等价式“→，⇔”：A→B<=>¬A∨B ；A<->B<=>(¬A∨B)∧(¬B∨A)；

（2）缩小否定词的作用范围，直到其作用于原子公式：¬(¬A)=A ；¬(A∧B)<=> ¬A∨¬B ；¬(A∨B)<=> ¬A∧¬B ；¬xA(x)<=>x¬A(x) ；¬xA(x)<=>x¬A(x)；

（3）适当改名，使量词间不含同名指导变元和约束变元；

（4）消去存在量词（形成 Skölem 标准型）。消取存在量词时，还要进行变元替换。变元替换分两种情况：①若该存在量词在某些全称量词的辖域内，则用这些全称量词指导变元的一个函数代替该存在量词辖域中的相应约束变元，这样的函数称为 Skolem 函数；②若该存在量词不在任何全称量词的辖域内，则用一个常量符号代替该存在量词辖域中相应约束变元，这样的常量符号称为 Skölem 常量；

（5）消去所有全称量词；

（6）化成合取范式；

（7）适当改名，使子句间无同名变元；

（8）消去合取词∧，用逗号代替，以子句为元素组成一个集合 S，即为与原谓词公式相对应的子句集。

例：为了机器证明“梯形的对角线与上下底构成的内错角相等”，给出逻辑描述，建立子句集合。

解：设梯形顶点依次为 a,b,c,d,定义谓词：

T(x,y,u,v):表示以 xy 为上底,uv 为下底的梯形.

P(x,y,u,v):表示 xy//uv

E(x,y,z,u,v,w)表示∠xyz=∠uvw，

问题的描述和相应的子句集为

∀(x,y,u,v)[T(x,y,u,v)→P(x,y,u,v)]...梯形上下底平行

子句：¬T(x,y,u,v)∨P(x,y,u,v)

∀(x,y,u,v)[P(x,y,u,v)→E(x,y,v,u,v,y)]...平行则内错交相等

子句：¬P(x,y,u,v)∨E(x,y,v,u,v,y)

T(a,b,c,d)...已知

子句：T(x,y,u,v)

∀(x,y,u,v)[T(a,b,c,d) →E(a,b,d,c,d,b)]...要证明的结论。

子句：T(x,y,u,v)∨¬E(a,b,d,c,d,b)

子句集 S 为：{¬T(x,y,u,v)∨P(x,y,u,v)，¬P(x,y,u,v)∨E(x,y,v,u,v,y)，T(a,b,c,d)，T(x,y,u,v)∨¬E(a,b,d,c,d,b)}

关于谓词公式与其子句集有如下定理：

定理 1：谓词公式 G 与其子句集(或 G 的 skolem 标准型)不等价。

例：G=∃xP(x)，消去存在量词后得 G′ =P(a)，若论域是 D={0,1}，取 D 下的一个解释 I：a=0；P(0)=F；P(1)=T；

G|I=P(0)∨P(1)=F∨T=T

G′ |I=P(a)=P(0)=F，显然 G 与 G′ 不等价。

总之，G 与 S 不等价，但在不可满足的意义上是一致的。

定理 2：谓词公式 G 不可满足当且仅当其子句集 S 不可满足。

证明从略，因为谓词公式 G 与其 Skolem 标准型是一般与其示例的关系，从而把证明一个公式 G 的不可满足性转化成证明其子句集 S 的不可满足性。

定理 3：子句集 S 是不可满足的当且仅当其全部子句的合取式是不可满足的。

17.2.5.2 基于谓词逻辑的问题求解系统中（曾经的）基于归结的演绎推理方法

人的问题求解行为有很大一部分是一个解答识别过程。识别解答或部分解答要依赖于应用领域特有的知识，此时，符号推理可成为基于知识来求解问题的主要手段。符号推理的重要方式是演绎推理，它的基础是谓词演算。命题公式是不包含变量和量词的谓词公式，命题演算应是谓词演算的子集，尽管它可以简化符号推理，但因缺乏有效的表达能力去表示一般性概念，故谓词演算（或更广义地，形式逻辑）是智能系统最常用的知识表示方法，被广泛地应用于各种智能系统的设计。

1．H 域与归结原理

让计算机来实现定理的自动证明是人工智能研究的早期目标之一。定理证明的一般表示形式为：

$\{F_1, F_2, \cdots, F_n\} \models W$

其中，$F_1, F_2, \cdots, F_n$ 都是合适公式，公式间隐含合取关系，可构成一个公式集（也称公理集或事实集），W 是待证明的一个合适公式（也称为定理或目标公式）。定理证明的实质就是要证明：$(F_1 \wedge F_2 \wedge \cdots \wedge F_n) \rightarrow W$ 是永真的；从而，当公式集为真时，定理 W 成立。

证明的方法可分二大类：直接证明 $(F_1 \wedge F_2 \wedge \cdots \wedge F_n) \rightarrow W$ 为永真，即演绎法；间接证明 $\neg((F_1 \wedge F_2 \wedge \cdots \wedge F_n) \rightarrow W)$ 为永假，即反证法。

证明（$F_1 \wedge F_2 \wedge \cdots \wedge F_n$）∧（¬W）的永假性，实际上就是要证明{ F_1, F_2,… ,F_n }∪{¬W}是一个矛盾集。海伯伦（Herbrand）和鲁宾逊（Robinson）在这方面做了卓越的研究工作。由海伯伦提出的 H 域（海伯伦域）和海伯伦定理，为自动定理证明奠定了理论基础；而由鲁宾逊所提出的归结原理则使机器定理证明成为可能。

海伯伦在将合适公式标准化为子句集的基础上，通过引入 H 域（即海伯伦域），从理论上给出了证明子句集(从而合适公式)永假(即不可满足)的可行性及方法。

我们把仅由文字的析取构成的合适公式称为**子句**，这里文字只能是原子谓词公式或其取反。如此，合取范式就可定义为子句的合取。进而，可以把合取范式表示为子句集，只要隐含着子句间具有合取关系。

鉴于子句集隐含地受到全称量词的约束，而全称量词又可分配到有合取关系的各个子句，所以各子句中的变量实际上都是全称量词的约束变量，且作用域只在子句范围内。为消除子句间不必要的交互作用（即保持子句间的相互独立性），需要作变量换名，使各子句都使用不同的变量。

当一个合适公式 F 化简为标准化的子句集 S 时，有一个重要性质，即 S 的不可满足(对于任意论域上的任意解释，S 中都至少有一个子句真值为 F)成为 F 永假的充分必要条件。当然，这并不意味着 F 和 S 间的等价。由于在合适公式 F 的化简过程中，为消除存在量词而引入了 Skölem 函数，从而使子句集 S 实际上只是 F 的一个特例。然而，可以证明 F 和 S 在永假性上是等价的，这成为建立海伯伦定理的重要基础。

证明子句集的不可满足性与证明合适公式的永真性是类似的，由于个体论域的任意性和解释个数的无限性，使得证明工作十分困难。若能建造一个较为简单的特殊论域，使得只要证明子句集在该域不可满足，就可确保子句集在任何可能的论域上不可满足，将是十分有意义的。海伯伦建立的特殊域 H 就具有这样的性质。

设 S 为子句集，D 为 S 的某个论域。我们可以这样来构成 H 域：

(1) 令 H_0 是 S 中出现的所有常量的集合，若 S 中未出现常量，就任取常量 a∈D,并令 H_0 ={a}；

(2) 令 $H_{i+1}=H_i \cup$ {出现于 S 中的函数在 H_i 上的所有实例},i=1,2,…；形如 $f(x_1, x_2, \cdots, x_n)$ 的函数的实例通过让 $x_j=k_j \in H_i$ 来形成（j=1,2,…,n）。

显然，H_i 可以迭代扩展到 H_∞，我们称 H_∞ 为海伯伦域，简称 H 域。一般情况下，H 是一个可数无穷集。

为研究子句集的永假性，引入 H 域上的原子谓词公式实例集 A：A={所有出现于 S 中原子谓词公式的实例}，若原子公式是命题（不包含变量）则其实例就是其本身；若原子公式形如 $P(t_1, t_2, \cdots, t_m)$， t_i 是变量（i=1，2，…m），则其实例通过让 $t_i=k_i \in H$(即 H_∞)来形成（i=1，2，…，m）。我们称 A 中的元素为基原子，进而 A 也称为基原子集。鉴于这些元素都是原子命题，只要给它们每个指派一个真值（T 或 F），就可建立子句集在 H 域上的一个解释，记为 I^*。可以证明，对于子句集 S 的任一可能论域 D 上的任一解释 I，总能在 S 的 H 域上构造一个相应的解释 I^*，使子句集具有相同的真值。进而，只要能够确定子句集对 H 域上所有解释都不满足，就可确定，对于任意可能论域上的所有解释，子句集都不满足，即子句集是不可满足的。

可以用语义树来形象地描述子句集在 H 域上的可能解释。当子句集包含的原子公式均为命题时，基原子集是有限集，容易画出完整的语义树。当子句集包含的原子公式均为命题时，容易画出完整的语义树。设一个这样的子句集包含原子命题公式 P、Q 和 R，则其基原子集 A={P、Q、R}。由于每

个基原子只可能有二个真值（T 或 F），所以很容易以二叉树的形式来建立语义树。显然，从树根节点 n_0 到叶子节点 n 的路径就指示了一个解释，记为 I(n)，其表示为路经上标记的集合，每个标记是一个文字。可以对于语义树指示的每个解释，判别子句集的真假性，进而判别子句集是永真，可满足，还是不可满足。

对于一般的子句集，H 是可数无穷集，从而相应的语义树也可能成为一棵无穷树。不过，若某个子句集不可满足，则不必无限地扩展语义树，就可以确定语义树上的所有路径都分别对应一个导致子句集不满足的解释，这样的语义树称为封闭语义树。例如，对于子句集 S ={P(x)∨Q(f(x)), ¬P(a), ¬Q(y)},可以有：

H={a,f(a),f(f(a)),…}

A={P(a),Q(a),P(f(a)), Q(f(a)),…}

当子句集中一子句包含的变量用 H 域中元素取代时(即该元素作为变量值)，我们称这样产生的子句为基子句。实际上基子句就是基文字（基原子或其取反）或基文字的析取。通常，语义树中每一导致子句集 S 不满足的路径（相应于 H 域上一解释）都至少引起一个基子句真值为 F。当建立起一棵封闭语义树时，实际上也建立了一个由有限个不可同时满足的基子句构成的集合 S′ 。

在上述研究工作的基础上，海伯伦提出了后来以他名字命名的著名的海伯伦定理，如下：子句集 S 不可满足的充要条件是存在一个有限的不可满足的基子句集 S′ 。

从封闭语义树的角度讲，不可满足的子句集具有封闭的语义树，进而可依据该语义树，建立一个有限的不可满足的基子句集。反之，只要能设法建立起封闭语义树，也就建立了一个有限的不可满足的基子句集，进而可以确定子句集在 H 域上不可满足，即子句集不可满足（永假）。

根据海伯伦定理和借助于语义树手段，从理论上讲，可以建立计算机程序去实现自动定理证明。但设计这样的程序存在二个主要的困难：一是子句集的不可满足性是不可判的，即子句集的不可满足性不能确保在有限的计算步范围内判定。因为在许多情况下，H 域和在其上的解释是无穷的，当子句集是永真或可满足时，判定过程将无休止地进行下去。二是即使对于不可满足的子句集，能在有限的计算步范围内加以证明，但依据 H 域上的每个解释判别不满足的基子句，计算量也往往很大。为提高判定子句集不可满足的有效性，鲁宾逊于 1965 年提出了归结(Resolution)原理，也称为消解原理。归结原理简单易行，便于计算机实现和执行，从而使定理的机器自动证明成为现实，也成为人工智能技术实用化的一次重要突破。归结原理的基本思路是通过归结方法不断扩充待判定的子句集，并设法使其包含进指示矛盾的空子句。空子句是不可满足（即永假）的子句，既然子句集中子句间隐含着合取关系，空子句的出现实际上判定了子句集不可满足。

设有二个子句：C_1=L∨C_1' ，C_2 =¬L∨C_2' ；从 C_1 和 C_2 中消去互补文字 L 和¬L，并通过析取将 C_1 和 C_2 的剩余部分组成新的子句：C= C_1' ∨C_2' ,则称 C 为 C_1 和 C_2 的归结式。例如有子句，P(A)∨Q(x)∨R(f(x))和 ¬P(A)∨Q(y)∨R(y)，则可以消去互补文字 P(A)和 ¬P(A)，生成归结式：Q(x)∨R(f(x))∨Q(y)∨R(y)。

可以证明，二个子句 C_1 和 C_2 的归结式 C 是 C_1 和 C_2 的逻辑推论。该结论意指，在任一使子句 C_1 和 C_2 为真的解释 I 下，必有归结式 C 为真。这是容易证明的，因为 L 和¬L 的互补性，只能同时有其中一个为真；若 L 为真，则为使 C_2 为真，C_2' 必须为真；若 L 为假，则为使 C_1 为真，C_1' 必须为真；既然 C 为 C_1' 和 C_2' 的析取式，当然 C 必定为真。

由此也可推论：设 C_1 和 C_2 是子句集 S 中的两个子句，并以 C 作为它们的归结式，则通过往 S

中加入C而产生的扩展子句集S′与子句集S在不可满足的意义上是等价的,即 S′的不可满足 <=>S的不可满足。这个推论确保了用归结原理来判定子句集不可满足的可行性。

特别地，当 C_1= L 和 C_2 = ¬L 时，归结式为空；我们以口指示为空的归结式，并称 C = 口为空子句。显然，此时 C_1和 C_2是一对矛盾子句--无论为子句集指派什么解释，C_1和 C_2不可同时满足，所以空子句实际上是不可满足的子句，进而导致子句集不可满足。换言之，空子句成为用归结原理判定子句集不可满足的成功标志。

2．归结推理过程

下面分别讨论基于命题逻辑和谓词逻辑的归结推理过程。

（1）命题逻辑中的归结推理过程。在命题逻辑情况下，子句中文字只是原子命题公式或其取反，由于不带变量，易于判别哪些子句对包含互补文字。归结过程很简单，并可用演绎树表示。

例如，若 S={¬P∨Q, P∨R, ¬Q, ¬R}，首先，子句¬P∨Q 和 P∨R 有互补文字，产生归结式 Q∨R；Q∨R 和子句¬Q 有互补文字，产生归结式 R；R 与子句¬R 是互补文字，从而归结出空子句。归结过程中归结式 Q∨R，R 和口作为新子句加入 S，使 S 扩展为 S′。鉴于 S′包含了空子句，其是不可满足的；再由前述关于归结式的推论，就判定了子句集 S 是不可满足的。

（2）谓词逻辑中的归结推理过程。在谓词逻辑情况下，由于子句中含有变量，不能像命题逻辑那样直接发现和消去互补文字，往往需要对潜在的互补文字先作变量置换和合一处理，才能用于归结。

对潜在的互补文字作合一处理，就是通过变量置换，使相应于这二个文字的原子谓词公式同一化的过程。所谓变量置换就是用置换项取代公式中的变量，置换项可以是变量、常量或函数。服务于合一处理的一个置换表示为包含若干置换元素的集合，每个元素形如：t/v，v 是公式中的变量，而 t 是置换项。

例如，有潜在的互补文字如下：P(x, y, x, g(x))，¬P(A, B, A, z)，我们可以为它们建立多个置换：

S_1 = {A/x, B/y, g(x)/z}；S_2 = {f(w)/x, z/y, C/z}；S_3 = {B/x, f(w)/y, y/z}

置换结果为：

{P(x, y, x, g(x)), ¬P(A, B, A, z)}={P(A, B, A, g(A)), ¬P(A, B, A, g(A))}|S_1

{P(x, y, x, g(x)), ¬P(A, B, A, z)}={P(f(w)), z, f(w), g(f(w)), ¬P(A, B, A, C)}|S_2

{P(x, y, x, g(x)), ¬P(A, B, A, z)}={P(B, f(w), B, g(B)), ¬P(A, B, A, y)}|S_3

显然，只有 S_1使这对潜在的互补文字中的原子谓词公式变为同一，进而确认互补性，并用于归结。研究者们已经提供了健全的面向任意表达式的合一算法；不过，归结演绎过程中只须对原子谓词公式作合一处理，所以实际上只需通过一个匹配过程去检查两个原子谓词公式的可合一性，并同时建立用于实现合一的置换。匹配过程可归纳如下：

（1）二个公式必须具有相同的谓词和参数项个数；

（2）从左到右逐个检查参数项的可同一性：

- 若一对参数项中有一个变量 v（不必关注另一个是否变量），并初次出现，则这对参数项可同一，并以另一参数 t 为置换项，与该变量一起构成一个置换元素 t/v；
- 若该变量出现过，则已建立相应的置换元素，就取其置换项，代替该变量，检查是否与另一参数同一；若不能同一，则合一处理失败。

• 若一对参数项中没有一个是变量（往往都是常量），则它们必须相同，否则合一处理失败。

（3）若每对参数项都可同一，则合一处理成功，并构成用于实现合一的置换。

示例 1：用该匹配过程对二个原子谓词公式 P(x, y, x, g(x)), P(A, B, A, z) 作合一处理。

首先，第一对参数项是可同一的，并建立置换元素 A/x；接着，第二对参数项也是可同一的，并建立置换元素 B/y；在第三对参数项中变量 x 已出现过，就取其置换项 A 与另一参数项（也是 A）作比较，发现同一；最后一对参数项中变量 z 初次出现，与另一参数项 g(x)一起构成置换元素 g(x)/z。从而，这对原子公式可合一，且建立起相应的置换 S_1={A/x, B/y, g(x)/z}。

示例 2：作为谓词逻辑中归结的例子，观察子句：

C_1 = P(x, y)∨Q(x, f(x))∨R(x, f(y))；C_2 = ¬P(A, B)∨¬Q(z, f(z))∨R(z, g(z))

令 L_{11}= P(x, y)，L_{21} = ¬P(A, B)，显然 L_{11}和 L_{21}是潜在的互补文字，用上述匹配过程可以确定 L_{11}和 ¬L_{21}是可合一的，并建立置换 S_1= {A/x, B/y}。注意，变量的置换必须在整个子句对范围内进行，所以消去互补文字，得归结式：Q(A, f(A))∨R(A, f(B))∨¬Q(z, f(z))∨R(z, g(z))

在谓词演算的情况下，往往两个子句可以有多于一对的互补文字。该例中就有另一对。令 L_{12} = Q(x, f(x))，L_{22} = ¬Q(z, f(z)),可以确定 L_{12}和¬L_{22}是可合一的，并建立置换 S_2 = {z/x},消去互补文字，得归结式：

P(z, y)∨R(z, f(y))∨¬P(A, B)∨R(z, g(z))

最后，通过一个实例来说明谓词逻辑中的**归结推理过程**。给定子句集：

（1）¬R(x, y)∨¬Q(y)∨P(f(x))

(2) ¬R(z, y)∨¬Q(y)∨W(x, f(x))

(3) ¬P(z)

(4) R(A, B)

(5) Q(B)

对这些子句进行归结：

（6）¬R(x,y∨¬Q(y)，（1）与（3）归结，{f(x)/z}

(7) ¬Q(B)　（6）与（4）归结，{A/x,B/y}

(8) 口，　（7）与（5）归结。

基于归结的演绎方法是完备的，即若子句集 S 不可满足，就必定存在一个从 S 到空子句的归结演绎；反之，若存在一个从 S 到空子句的归结演绎，则 S 必定是不可满足的。关于归结演绎的完备性可用海伯伦定理进行证明，从这个意义上讲，归结原理是建立在海伯伦定理之上的。不过，归结原理并不能用于解决子句集不可满足性的不可判问题，即对于永真和可满足的子句集，判定过程将无休止地进行下去，得不到任何结果。

3．归结反演

归结演绎方法为采用间接法（即反证法）证明定理提供了有效手段，我们称应用归结演绎方法的定理证明为归结反演。

归结反演的基本思路是：要从作为事实的公式集 F 证明目标公式 W 为真，可以先将 W 取反，加入公式集 F,标准化 F 为子句集 S，再通过归结演绎证明 S 不可满足，并由此得出 W 为真的结论。因此，一个归结反演系统应由二个部分组成：标准化部件和归结演绎部件。前者将每条事实和取反的目标公式分别标准化为子句集，再合并为子句集 S；后者遵从归结演绎方法，控制定理证明的全过

程。

作为归结反演的真实应用，观察下面的例子。

已知：下列事实为真：

{ 王（Wang）喜欢（Like）所有种类的食物（Food）；

苹果（Apples）是食物。

任何一个东西，若任何人吃了（Eat）它都不会被害死（Killed），则该东西是食物。

李（Li）吃花生（Peanuts）且仍然活着（Alive）。

张（zhang）吃任何李吃的东西。}

证明：王喜欢花生。

（1）形式化。为应用归结反演进行证明，首先要把以自然语言表示的事实和要证明的目标形式化地表示为合适公式。确定 Like、Food、Eat、Killed 和 Alive 为谓词；鉴于 Killed 和 Alive 是反义词，可以 Alive 取代 Killed。上述事实的形式化表示如下：

（∀x）[Food(x) => Like(Wang,x)]

Food(Apples)

（∀x）（∀y）[Eat(y, x)∧Alive(y) => Food(x)]

Eat(Li, Peanuts)∧Alive(Li)

（∀x）[Eat(Li,x) => Eat(Zhang,x)

目标形式化表示为：Like(Wang, Peanuts)

（2）标准化。应用合适公式的性质，将事实公式和取反的目标公式分别标准化为子句，再收集进子句集 S：

(1) $\neg$Food(x_1)$\vee$Like(Wang,x_1)

(2) Food(Apples)

(3) $\neg$Eat(y,x_2)$\vee\neg$Alive(y)$\vee$Food(x_2)

(4) Eat(Li,Peanuts)

(5) Alive(Li)

(6) $\neg$Eat(Li,x_3)$\vee$Eat(Zhang,x_3)

(7) $\neg$Like(Wang, Peanuts)

（3）归结演绎。应用归结演绎方法不断生成归结式以扩展子句集 S，直到生成空子句。此时目标公式得以证明，即王确实喜欢花生。归结演绎过程：

(8) $\neg$Eat(Li, x_2)$\vee$Food(x_2),(3)与（5）归结，{y/Li}；

(9) $\neg$Eat(Li, x_2)$\vee$Like(Wang, x_2),(1)与（8）归结，{x_1/x_2}；

(10) Like(Wang, Peanuts)，(9)与（4）归结，{x_2/Peanuts}；

(11) 口，(10)与（7）归结。

实用上，归结反演系统面临着大子句集而引起的演绎效率问题。解决问题的关键在于选择有利于导致快速产生空子句的子句对进行归结。若盲目地随机选择子句对进行归结，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过分扩张了子句集，从而浪费了时空，并降低了效率。为此，研究归结策略成为促进归结演绎技术实用化的重点。归结策略主要分为二大类：删除策略和限制策略。前者通过删除某些无用的子句来缩小归结的范围；后者则通过设置选用条件对参与

归结的子句进行种种限制，减少归结的盲目性，如支持集、线性输入、单文字子句优先、祖先过滤等策略。

归结反演可以用于实现问题回答系统。这种情况下目标公式往往是受存在量词约束的，形如(∃x)W(x)。既然是问答系统，当然我们不会满足于证明目标公式为真，还要求给出使W(x)为真的个体实例，即x的某个取值。这种要求称为回答提取。为实现回答提取，问答系统的运作分为二个阶段：归结反演和回答提取。

回答提取又分二个步骤：（1）对于标准化取反的目标公式而产生的子句（称为目标子句）G,建立其重言式G∨¬G；（2）以G∨¬G取代G,重复已进行过的归结演绎过程。结果，归结反演树的树根不再是空子句，而是¬G，且G中的变量已为其置换项取代，实现了回答提取。

在上述关于食物的例子中，若问张吃什么食物，就可用这种回答提取方法获取回答。首先形式化该问题为目标公式：(∃x)[Food(x)]∧Eat(Zhang, x)；取反并标准化为子句：

（12）¬Food(x_4)∨¬Eat(Zhang,x_4)

进行归结反演并生成归结反演树后。接着再进行回答提取，即建立（12）式的重言式：

¬Food(x_4)∨¬Eat(Zhang, x_4)∨[Food(x_4)∧Eat(Zhang, x_4)]

用其取代（12）式重复已进行过的归结演绎过程，产生修正的归结反演树，称为修改证明树。

从修改证明树可以看出，目标子句G的重言式G∨¬G中的¬G并不真正参与归结演绎；实际上用重言式取代G重复已进行过的归结演绎过程，只是为了使¬G中的变量随着G中出现的变量置换而同时得到置换。所以归结反演和回答提取可以合为一体进行，不需先生成归结反演树，而是一次性直接生成修改证明树。为了避免在归结演绎中误用¬G参与归结。可以用特定的符号替代¬G。

应用修改证明树提取问题的回答是可行的，因为依据归结演绎过程，树根中¬G实际上是作为事实的公式集与重言式G∨¬G的逻辑推论，由于重言式永真，只要公式集为真，则¬G为真。

17.2.5.3 基于谓词逻辑的问题求解系统中（曾经的）基于规则的演绎推理

基于归结的演绎推理提供了一种简单易行的方法去实现问题求解，只要将问题求解的依据（事实或公式）和目标以合适公式加以形式化描述，就可交由归结反演系统和问答系统执行。然而，归结演绎技术存在严重的缺点：

（1）必须将合适公式标准化为高度统一的子句集，从而丢失了隐含于合适公式的启发式知识。例如蕴涵式P∧Q => R隐含着启发式知识“P∧Q为真时导致R为真”，但一旦其简化为子句¬P∨¬Q∨R，就丢失了这一启发式知识。正因为如此，归结反演系统解决问题的效率低下，难以适应于复杂的应用域。

（2）归结演绎并非人类的自然思维方式，不利于人们从自然思维的角度组织问题的求解和提供问题求解所需的知识，从而难以建立高水平的问题求解系统，如专家系统。

保留蕴涵式，将其作为推理规则，用于直接推导目标公式，不仅符合人的自然思维方式，也能通过规则（作为启发式知识）更有效地引导演绎推理过程。所以，基于规则的演绎推理成为比归结反演更为有效的技术，广泛地应用于许多问题求解任务中。

规则演绎将求解问题所需的知识分为二类：规则和事实。规则表示为蕴涵式，作为启发式知识（表示应用域中存在的规律和法则），引导演绎推理过程。事实则表示为非蕴涵形式的合适公式，作为应用规则进行推理时参考的有关问题状态和环境的知识。规则演绎的任务就是从给定的事实（即问题的状态和环境知识）和规则，证明某个目标公式成立。

基于规则的演绎推理可以区分为二大类：正向演绎和逆向演绎。前者从事实出发，应用规则不断推导出中间结果作为新的事实，直至推导出目标公式。后者则从目标公式出发，逆向应用规则不断推导出子目标，直至所有子目标就是给定的事实为止。换言之，目标公式通过逆向推理找到了支持其成立的所有依据。

1．基于规则的正向演绎推理

正向演绎推理要求将问题求解的描述分为三个部分：事实、规则集和目标。为便于演绎推理，需将表示它们的合适公式简化为下述标准形式，并加以适当限制。

（1）问题求解的规范表示

（a）**事实。**为支持演绎推理，事实表达式不必化简为子句集，只需规范地表示为不含蕴涵符号的文字与或形。例如有事实表达式：$(\exists x)(\forall y)(\forall z)\{Q(x, y)\vee[(R(y)\vee P(z))\wedge S(x, z)]\}$，先化简成：$\neg Q(A, y)\wedge[(\neg R(y)\wedge\neg P(f(y)))\vee\neg S(A, f(y))]$，再对顶层合取式的各项作换名处理，使它们不含同名变量：$\neg Q(A,w)\wedge[(\neg R(y)\wedge P(f(y))\vee\neg S(A, f(y))]$。换名的可行性和意义类似于对子句集中各子句的换名处理。由于未进一步化简为子句集，文字与或形更接近原始表达式。

事实表达式的文字与或形可以用与或图表示，两者间的对应关系规定如下：① 若母式（化简得到的文字与或形）为析取式：$E_1\vee E_2\vee\cdots\vee E_k$，则以一个 K-连接指向各析取项 E_i（$i = 1, 2, \cdots, k$）。② 若母式为合取式：$E_1\wedge E_2\wedge\cdots\wedge E_k$，则以 K 个 1-连接指向各合取项 E_i($i = 1, 2, \cdots, k$)。这样，与或图的根节点就是整个事实表达式，叶节点则对应于表达式中的每一个文字。注意，这里仅借用与或图表示事实表达式，语义上与表示搜索过程的与或图有本质的区别。

（b）**规则。**正向演绎推理以正向方式使用规则(称为 F 规则)，要求规则化简为以下形式：$L => W$。其中，L 为单文字，W 是与或形。之所以限制规则左部为单文字，是为了在演绎推理过程中便于通过 L 和与或图叶节点的匹配来激活规则。

设表示规则的原始蕴涵式为：

$(\exists x)\{(\exists y)(\forall z)P(x, y, z) => (\forall u)[Q(x, u)\vee R(x, u)]\}$，

则先暂时消去蕴涵符号并化简为文字与或形：

$P(x, y, f(x, y))\vee Q(x, u)\vee R(x, u)$，

再恢复蕴涵式表示：

$P(x, y, F(x,y)) => Q(x, u)\vee R(x, u)$。

有时化简的结果会出现形如 $L_1\vee L_2 => W$ 的情况，可以进一步化简为与其等价的两条规则

$L_1 => W$, $L_2 => W$

单文字的限制对知识的表示有所限定，例如，$L_1\wedge L_2 => W$ 这样的规则就不允许出现，只有转变为

$L_1 => \neg L_2\vee W$ 或 $L_2 => \neg L_1\vee W$

才能使用，丢失了启发式知识（L_1或 L_2同时为真导致 W 为真）。不过，这种限制对演绎推理方法本身并不产生影响。

（c）目标。目标公式化简后限定表示为文字的析取式，即子句。化简时量词消去方法取事实表达式的对偶形式，即将全称量词的约束变量以 Skölem 函数或常量取代，并使子句隐含地受存在量词约束。

例如，目标公式：

（∀x）(∃y)（∀z）[P(x, y, z)∨Q(x, y)]，

则先以 Skölem 常量 A 和 Skölem 函数 g(y)分别取代约束变量 x 和 z，再消去全称量词，于是得:

(∃y)[P(A,y,g(y))∨Q(A,y)]，

然后消去存在量词，并隐含着 y 是存在量词的约束变量。

关于为何需用对偶方式消去量词，这里仅通过与归结反演方法作对比来加以直观说明。在归结反演中，需将目标公式取反，自然上例中 x 和 z 成为存在量词约束变量，而 y 成为全称量词约束变量。最后子句中变量须作换名处理，使各文字不含同名变量。所以化简结果为：

P(A,y,g(y))∨Q(A,w)。

鉴于存在量词可分配到析取式的各析取项，而每个量词的约束变量又是相互独立的，所以换名不仅可行，而且是必要的。

（2）正向演绎推理的实现

借助于与或图表示方式，可以认为，正向演绎推理是从事实表达式出发，不断用激活（左部单文字和与或图叶节点匹配）的 F 规则对与或图进行变换（从而扩展了与或图），直至得到一个将目标表达式（子句）包含的所有文字都作为叶节点的一致解图的过程。

（a）命题逻辑的情况

如上所述，正向演绎推理的基本方法就是选用激活的规则对与或图进行变换。由于命题逻辑情况下所有公式都不含变量，所以寻找左部单文字和与或图中叶节点匹配的规则比较容易。

作为例子，设事实表达式为：（P∧Q)∨(R∧S)∨T；给定规则集：P => C_1∧C_2，S => C_3∨C_4，T => C_5；目标公式为：C_1∨C_3∨C_5。正向演绎推理的实现如下：

首先建立相应于事实表达式的初始与或图（树），然后选用激活的规则变换与或图。例如，规则 P=> C_1∧C_2的左部单文字与初始与或图叶节点匹配（相同），则被激活。对与或图的变换实际上就是将该规则加入与或图：将规则左部单文字作为节点插入与或图，并通过匹配弧（以双箭头表示）和与或图相应叶节点连接起来；再将规则右部以与或图子图的方式插入。如此，也可将另二条规则用于变换与或图，并使与或图不断扩展。当目标公式中的某个文字（如 C_1）和与或图叶节点匹配时，目标文字作为目标节点（以方框指示）插入与或图，并通过匹配弧（类似于规则左部单文字的插入）和与或图相应叶节点连接。当目标公式中的所有文字（本例中的 C_1、C_3和 C_5）全部作为目标节点进入同一个解图时，演绎推理成功结束。

解图的概念借用了与或图搜索中解图的语义，即解图只有“与”分支（不过在正向演绎推理中“与”对应于析取关系），并把目标节点视为解图终节点。因此，可以把正向演绎推理的结束条件表示为：目标公式中的所有文字进入同一解图（作为终节点）。不过，并不要求解图的所有叶节点都是目标节点。

（b）谓词逻辑的情况

由于公式含有变量，谓词逻辑情况下的演绎推理增加了复杂性。首先在判断规则能否激活时，要对规则左部的单文字和与或图中的相应叶节点作合一处理；其次，演绎推理过程可能会对同一变量进行不一致的置换，从而导致不一致解图的生成。

作为应用实例，下面观察一个简单的问题求解任务。给定事实、规则和目标如下：

事实：Fido 会吠叫（Barks）和咬人（Bites），否则 Fido 就不是狗（Dog）。

规则：所有 Terrier（一种小猎狗）都是狗。

所有会吠叫的东西都是吵人的（Noisy）。

目标：存在某个东西，除非它不是 Terrier，否则是吵人的。

首先作形式化表示：

事实表达式：[Barks(Fido) ∧Bites(Fido)]∨¬Dog（Fido）

规则:R_1: （∀x）[Terrier(x) => Dog(x)]

R_2: （∀y）[Barks(y) => Noisy(y)]

目标公式：（∃z）[¬Terrier(z)∨Noisy(z)]

再把规则和目标公式标准化为：

规则：R_1: ¬Dog(x) => ¬Terrier(x)，应用逆否律

R_2: Barks(y) => Noisy(y)

目标公式: ¬Terrier(z)∨Noisy(z1)

然后从相应于事实表达式的与或图开始，先后用 R_1 和 R_2 对与或图进行变换，再把目标公式中的二个文字作为目标节点插入与或图，就产生了以这二个目标节点为终结点的解图。

将解图中所有变量的置换都作为置换元素，收集于一个置换中，称为解图置换，并对其作合一复合处理。处理的方法如下：

（a） 设解图置换中元素形如 t_i/v_i(i = 1, 2, …, n)，t_i 指示置换项，而 v_i 指示变量。

（b） 建立两个分别由 t_i 和 v_i 构成的表达式（i = 1, 2, …, n）：

$U_1 = (v_1, v_2, …, v_n)$　　$U_2 = (t_1, t_2, …, t_n)$

（c） 检查 U_1 和 U_2 能否合一，若不能合一，则该置换是不一致的，进而解图是不一致的；若能合一，则建立起使 U_1 和 U_2 合一的置换，称为解图置换 S 的合一复合，进而解图是一致的。

表 17.2.1 给出若干解图置换的合一复合例。该例子中的解图置换是：S = {Fido/z_1, Fido/y, Fido/z, Fido/x}。显然，其合一复合就是其自身。至此，已得到一致解图，演绎推理成功结束，只要将目标公式中的变量 z、z_1 分别以置换项 Fido 取代，就可得到解答：¬Terrier(Fido)∨Noisy(Fido)。在解图不一致的情况下，并不意味着演绎推理失败，因为可能存在另一解图并且其是一致的。只有在找不到一致解图的情况下，演绎推理失败。

表 17.2.1　解图置换的合一复合

S	U_1 和 U_2	合一复合 S′
{A/x,B/x}	U_1=(x,x) U_2=(A,B)	不一致
{x/y,y/z,z/w}	U_1=(y,z,w) U_2=(x,y,z)	{x/y,x/z,x/w}
{g(y)/x,g(A)/x}	U_1=(x,x) U_2=(g(y),g(A))	{g(A)/x,A/y}
{g(y)/x,f(x)/y}	U_1=(x, y) U_2=(g(y),f(x))	不一致

有时演绎过程会多次调用同一条规则，应注意每次使用规则都要将变量改名，以免给变量置换附加不必要的约束。类似地，当同一目标文字建立多个目标节点时，也要改换变量名。

2．基于规则的逆向演绎推理

逆向演绎推理与正向演绎推理形成对偶关系，其从目标公式出发，逆向应用规则对相应于目标公式的原始与或图进行变换，直至得到一个所有叶节点都是事实节点（以事实表达式中的文字作为节点内容的节点）的一致解图为止。逆向演绎推理情况下问题求解的描述也分为三部分：目标公式、规则和事实，它们的标准化表示形式与正向演绎呈现对偶关系。

（1）问题求解的规范表示

（a）目标公式。逆向演绎推理情况下，目标公式的表示不受限制，化简为不含蕴涵符号的文字与或形。例如有目标公式：（∃x）(∀y）{P(x) => [Q(x, y)∧(R(x) => S(x, y))]}，先化简为：（∃x）(∀y）{¬P(x)∨[Q(x, y)∧(¬R(x)∨S(x, y))]}，将全称量词的约束变量 y 以 Skölem 函数取代，消去全称量词和存在量词，隐含着变量 x 受存在量词的约束。于是得：¬P(x)∨[Q(x,f(x))∧(¬R(x)∨S(x, f(x)))]；再通过换名，使顶层析取式的各析取项不含同名变量：¬P(z)∧[(Q(x,f(x))∧(¬R(x)∨S(x,f(x))))。

目标公式的文字与或形也可以用与或图表示，两者间的对应关系规定如下：（1）若母式（化简得到的文字与或形）为合取式：$E_1∧E_2∧\cdots∧E_k$则以一个 K-连接指向各合取项 E_i(i = 1, 2, …, k)。（2）若母式为析取式：$E_1∨E_2∨\cdots∨E_k$则以 K 个 1-连接指向各析取项 E_i(i = 1, 2, …, k)。这样，与或图的根节点就是整个目标公式，叶节点则对应于目标公式中的每一个文字。

（b）规则。逆向演绎推理以逆向方式使用规则（称为 B 规则），要求规则化简为以下形式：W => L。其中，L 为单文字，W 是与或形；规则的激活取决于 L 和与或图叶节点的匹配。设表示规则的原始蕴涵式为：（∀x）{(∃y)（∀z）[P(x, y, z)∧Q(y, z)] =>（∀u）R(x, u)，则先暂时消去蕴涵符号并化简为与或形：¬P(x, y, f(x, y))∨¬Q(y, f(x, y))∨R(x, u)，再恢复蕴涵式表示：P(x, y, f(x, y)∧Q(y, f(x, y)) => R(x, u)。

有时化简的结果会出现形如 W => $L_1∧L_2$的情况，可以进一步化简为与其等价的两条规则：W => L_1, W => L_2

（c）事实表达式。事实表达式化简后限定表示为文字的合取式，消去量词时将存在量词的约束变量以 Skölem 函数或常量取代，并使合取式隐含地受全称量词约束。

（2）逆向演绎推理的实现

借助于与或图表示方式，逆向演绎推理不断用激活（右部单文字和与或图叶节点匹配）的 B 规则对与或图进行变换，使与或图不断扩展。被激活规则的右部单文字通过匹配弧插入，而规则左部则以与或图子图的方式插入。当事实表达式包含的文字和与或图叶节点匹配时，事实文字作为事实节点插入与或图（通过匹配弧）。当获得一个叶节点全是事实节点的一致解图时，演绎推理成功结束。逆向演绎推理实际是给目标公式寻找原始证据（事实文字）的过程，当推理的各个分支（解图中的分支）都到达事实节点时，目标公式也就有了支持它成立的足够证据。

3．关于演绎推理应用中的几个问题

（1）正、逆向两种演绎推理技术具有共同的特点，即都以与或形和相应的与或图来表示和支持推理过程，推理控制的基本策略都是通过不断激活规则去变换与或图，直至搜索到某个一致解图为止。两者关于问题求解的描述都分三个部分：事实、规则和目标，并采用同样的方式标准化这三个

部分的表示。一个必须注意的共同之处，就是事实和规则的标准化过程对量词及其约束变元的处理遵从合取范式的化简要求，而目标的标准化过程采用对偶原则作处理，即将全称量词的约束变量以 SkÖlem 函数或常量取代，并在量词消去后隐含着所有变量均受存在量词的约束。正、逆向演绎推理的特点对比见表 17.2.2。　尽管正、逆向演绎推理都对问题求解的描述加以了一定的限制，但在许多实际应用领域均可适用。

（2）在实际应用中，可以把正向和逆向演绎推理结合起来，建立基于规则的双向演绎推理系统。这种系统可以免除因单向推理而引入的限制，并实现优势互补。然而，处理结束条件的复杂性以及需分别支持正、逆向推理，增加了系统设计和知识获取的困难。更实用化的方式是将复杂的问题求解任务划分为相对简单的若干子任务，然后根据子任务的特点选用正向或逆向演绎推理方式，以便充分发挥两种方式各自的优势。

表 17.2.2 正、逆向演绎推理的特点比较

	正向演绎推理		逆向演绎推理	
问题求解的描述	事实	文字与或形	事实	文字合取式
	规则	L => W	规则	W => L
	目标	文字析取形	目标	文字与或形
初始与或图	相应于事实表达式 ∨-K-连接		相应于目标公式 ∧-K-连接	
演绎推理	F 规则 事实--->目标		B 规则 目标--->事实	
结束条件	包含所有目标节点的一致解图		以事实节点作为所有终节点的一致解图	

（3）在基于规则的演绎推理过程中，基本的解图搜索策略是先找到（生成）一个任意解图，检索其一致性；若否，则搜索（生成）另一个，直至找到一致解图为止。由与或图中同一叶节点激活的多条规则隐含“或”关系，正是由于这种“或”关系，才导致了多个解图存在的可能性。直到解图生成后才检查其一致性，往往是太晚了；实际上，许多不一致的解图在其构建的早期就可发现其不一致。所以，建立解图搜索修剪策略，及早修剪掉不一致的局部解图，可作为一项重要考虑。所谓修剪策略，就是每当以规则扩展局部解图时，就及时检查解图置换的一致性，并随即修剪掉不一致的局部解图，以免除对这种局部解图的进一步徒劳搜索。

17.2.5.4 （曾经的）逻辑语言 Prolog 与基于规则的逆向演绎推理【已过时的内容，只供参考】

Prolog(Programming in logic)是一种面向演绎推理的逻辑型程序设计语言，最早于 1972 年由柯尔麦伦纳（Colmeraner）及其研究小组在法国马赛大学提出。Prolog 以处理一阶谓词演算为背景，由于其简单的文法、丰富的表达力和独特的非过程语言的特点，很适合用来表示人类的思维和推理规则，从而一问世就赢得了人工智能研究和应用开发者的广泛兴趣。Prolog 语言已推广应用于许多应用领域，如关系数据库、数理逻辑、抽象问题求解、自然语言理解、专家系统等。

Prolog 实际上就是一种基于逆向规则的演绎推理技术，只不过对规则和目标的表示有严格的限

制，再加上演绎推理控制机制自身的简单性，较难适用于比较复杂的应用领域。

Prolog 语言的基本成分是 Horn 子句，其表示形式为：$P_1 \wedge P_2 \wedge \cdots \wedge P_n \Rightarrow P$ 其中，P 和 P_i（i = 1, 2, …, n）均是原子谓词公式，且 P 可以缺省。Horn 子句可以转变为子句：$P_1 \vee \neg P_2 \vee \cdots \vee \neg P_n \vee \neg P$，可见，Horn 子句实际上只是合取范式子句的特别受限形式，即子句中最多只能包含一个不取反的原子公式。

在 Prolog 语言中，Horn 子句以三种形式分别表示逆向演绎推理中问题求解描述的三个部分：目标、规则和事实。其中，目标表示为只有左部的 Horn 子句：$P_1 \wedge P_2 \wedge \cdots \wedge P_n$，即在 Prolog 语言中，目标公式只限于表示为原子公式的合取。规则表示为典型的 Horn 子句：$P_1 \wedge P_2 \wedge \cdots \wedge P_n \Rightarrow P$，只是规则左部限定为原子公式的合取。事实表示为事实元素的集合，每个事实元素表示为只有右部的 Horn 子句，即单一原子公式 P。事实元素间隐含合取关系。

用 Prolog 语言将问题作上述描述后，支持 Prolog 语句的推理机制就能自动执行基于规则的逆向演绎推理，并最终给出解答。由于目标和规则的左部都限定为原子公式的合取，而这些原子公式又可激活多条规则（这些规则间隐“或”关系）。所以，Prolog 演绎推理过程所形成的演绎路径常呈现“与”、“或”关系逐层交替的局面。随着推理沿“与”“或” 演绎途径向下扩展，最终会形成一致演绎路径解图，从而演绎推理成功；否则推理失败。

若把目标中包含的各原子公式视为子目标，每当由子目标所激活的规则被引入“与”“或” 演绎路径图时，规则左部的各原子公式又会成为新的子目标，再用于激活新的规则；如此，随着规则的激活和新子目标的产生，“与”“或”演绎路径图将逐步扩展，直到子目标直接与事实元素（原子公式）匹配为止。

Prolog 是一种会话式语言，其程序语法和执行控制很简单，只要将演绎推理所需的事实和规则存入演绎数据库（也可称为知识库），就可以目标公式作为询问语句，让 Prolog 系统给出解答。由于事实和规则均已被表示为 Horn 子句，Prolog 程序实际上就是这些 Horn 子句的集合。故可以将 Prolog 程序以 BNF 表示如下：

<程序>:={<子句>}+

<子句>:=<事实元素>|<规则>

<事实元素>:=<句首>

<规则>:=<句首>:- <句体>

<句首>:= <原子谓词公式>

<句体>:= <原子谓词公式> {，<原子谓词公式>}*

<原子谓词公式>:= <谓词>（<项> {，<项>}*）| <谓词>

<项>:=<常量>|<变量>|<函数>

<函数>:=<函数名>（<项>{，<项>}*）

询问语句的 BNF 表示如下：<询问> :=> <句体>

在 Prolog 语言中，规定变量的命名以大写字母开始，其它项的命名用小写字母。另外，原子谓词公式间的“，”指示合取关系。注意，若把“:-”视为反写的蕴涵符号“=>”，则 BNF 表示的规则形如“句首=>句体”，再改写为规范形式“句体=>句首”，就是规范 Horn 子句。事实元素则是只有句首的 Horn 子句，而询问表示为只有句体的 Horn 子句。

Prolog 程序的总控流程是不断取用子目标，并从演绎数据库（知识库）寻找可激活规则和事实

的过程。子目标（包括相应于询问语句的顶层目标和由被激活规则的句体提供的子目标）按建立的先后次序引入栈表。对于每个当前处理的子目标，都按在演绎数据库中自上而下的排列次序，检查规则和事实的可激活性。这意味着若演绎推理过程处理过 n 个子目标，就须对演绎数据库作 n 遍匹配检查。遵从基于规则的逆向演绎推理机制，Prolog 程序的执行控制主要依赖于二个基本处理：匹配和回溯。

匹配是激活规则和事实的依据。从 Prolog 程序的角度，匹配处理自询问语句（目标）出发，检查目标包含的所有子目标是否能满足，并遵从从深度优先的原则。即首先检查目标中排在最左边的子目标（原子公式）与哪些规则的句首匹配，并将这些激活的规则记载于一个表 L 中；然后递归调用匹配处理，将 L 表中第一条规则的句体中的原子公式作为新子目标，进行匹配检查，以发现新的可激活规则，直到所有子目标均与事实元素匹配为止。以这种方式，其“解图”可以自顶向下并从左到右地生成。在“解图”生成过程中的任何时候，只要发现某一子目标不能满足（即找不到与之匹配的规则或事实元素），或相应的局部“解图”不一致时，即表示此问题的“解图”搜索失败。

由于每个子目标可以激活多条规则或事实元素，且每个对应于一个不同的局部“解图”，所以每当目前搜索所生成的局部“解图”失败时（即当前激活规则对应的推理分支失败），系统可按时序（子目标匹配检查的顺序）回溯到最近（即上一次）处理的子目标，从演绎数据库中该子目标激活的上一个 Horn 子句（规则或事实）处，继续搜索可激活的规则或事实，并由此建立新的局部“解图”（该“解图”的其余部分与失败的“解图”相同），进行搜索；若发现该“解图”又失败，则再回溯，直至找到一致“解图”为止。在所有子目标激活的所以规则都导致失败“解图”的情况下，演绎推理失败，并给出解答失败信息。

Prolog 语言提供不少内部谓词。内部谓词是由 Prolog 系统预先提供，而非由程序设计者自定义的谓词。内部谓词实际上就是 Prolog 语言系统的预定义函数，用以完成无法由演绎推理机制自身提供的计算功能，如输入输出操作、算术运算、程序控制、表处理和程序调试等。应用关于内部谓词的表达式可以如同普通的程序员自定义的原子公式那样出现于 Horn 子句的句体中，但对这些表达式不作一般的匹配处理，只是直接加以执行。执行的结果取真值 T 或 F，分别对应于子目标满足或不满足。

Prolog 提供的内部谓词包括算术谓词、“is”谓词、比较谓词和控制谓词等。Prolog 语言提供“+、-、*、/ ”和“mod（取模）”作为内部谓词，以支持算术运算，并形成简单的算术表达式或复合的算术表达式，如：X + Y/(A - B) + A * (B mod C)。“is”也是内部谓词，其应用表达式形如：X is Y ，其中 X 是变量，而 Y 可以是任何算术表达式。当 X 尚未实例化时，就将 Y 的计算结果赋值给 X，使其实例化，且 is 表达式的执行结果取真值 T；若 X 已经实例化，则判别 X 的束缚值与 Y 的计算结果是否相等，以决定 is 表达式执行结果的真值（T 或 F）。Prolog 的比较谓词有“=、\=（不等）、＞、＜、=＜、＞=”等，例如：X = Y，X \= Y，X ＜ Y，X =＜ Y，X ＞= Y，其中，X、Y 均可以是常数、变量或算术表达式。Porlog 语言提供多种输入输出谓词，其中最主要的是 read(X)和 write(X)。write(X)的执行结果总有真值 T。当变量 X 已实例化时，就向输出设备（屏幕、打印机等）或文件输出 X 的束缚值；否则输出一个出错信息。若 X 尚未实例化，read(X)则从输入设备或文件输入下一数据项，并将其赋值给 X；否则，X 已有束缚值（已赋值），则依据输入的数据项能否与 X 的束缚值匹配来决定 read(X)的真值。控制谓词的作用是修改演绎推理的控制行为，最常用的是 cut（截断）谓词。 缺省情况下，Prolog 的演绎推理机制将遍历整个搜索域，以找出所有可能的

一致“解图”。引入 cut 谓词，可以改变这种缺省行为。cut 是无参数项的特殊谓词，以“！”表示。其作为子目标出现于规则句体时，若在演绎推理中首次遇见，总是满足的；但其不能被重复满足，即推理回溯时再遇到同一个 cut 谓词时，就总是失败。换言之，Prolog 程序设计者可以使用 cut 谓词来切断不必要的回溯，提高程序运行效率。

内部谓词工作时涉及一个重要的概念是变量的实例化。任何时候，演绎推理过程对每条 Horn 子句操作之初，子句中的变量都是不受束缚的，即没有赋值。变量值只能通过匹配检查（子目标与规则句首或事实元素作合一处理）时建立的置换获得。以这样的方式取得变量值，称为变量的实例化。

17.2.6 基于产生式的知识表示与问题求解系统--智能系统开发的基础

基于产生式的问题求解系统是一类基于知识的“**自然推理系统**”。也曾是应用最为广泛的一类问题求解系统。人们之所以采用产生式系统作为问题求解系统的主要考虑，主要是基于两点理由：一是，用产生式系统结构求解问题的过程和人类求解问题时的思维过程很相象，因而可以用它来模拟人类求解问题时的思维过程；二是，我们可以把产生式系统作为复杂智能系统中的基本结构单元或基本模式，就好象是积木世界中的积木块一样，因而，研究产生式系统的基本问题也就具有了其最一般的意义。

17.2.6.1 知识的产生式表示方法

产生式知识表示法是常用的知识表示方式之一。它是依据人类大脑记忆模式中的各种知识之间的大量存在的因果关系,并以“IF-THEN”的形式,即产生式规则来表示的。这种形式的规则捕获了人类求解问题的行为特征,并通过认识-行动的循环过程求解问题。由于便于模拟人求解问题的思维方式，系统模块性强，易于修改扩充，产生式系统已得到广泛应用；可以说目前大多数专家系统（尤其是中小型系统）都是采用产生式系统的结构方式来建立。

（1）事实的表示

事实可看成是断言一个语言变量的值或是多个语言变量间的关系的陈述句。语言变量的值或语言变量间的关系可以是一个词，也可以是数字。一般使用三元组（对象，属性，值）或（关系，对象 1，对象 2）来表示事实，比如，事实“老李年龄是 55 岁”，可表达为（Lee，age，55） 事实“老李、老张是朋友”，可表达为（friend，Lee，Zhang）。其中，对象就是语言变量，若考虑不确定性就成了四元组表示（增加可信度）。这种表示的机器内部实现就是一个表。

（2）规则的表示

产生式也称为产生式规则，或简称规则。通常用于表示事物间的启发式关联，其基本形式为：P → Q（condition→action；或 IF P THEN Q）。其中，P是产生式的前提，也称为前件，它给出了该产生式可否使用的先决条件（为规则激活使用的条件），由事实的逻辑组合来构成，即是由逻辑运算符AND、OR、NOT组成的表达式，常是一些事实A_i（i=1,2,…,m）的合取；Q是一组结论或操作，也称为产生式的后件，它指出当前题P满足时，应该推出的结论或应该执行的动作[指示规则激活时（即规则条件部分满足时）应该执行的动作（或应该得出的结论）]；产生式规则的语义是：如果前提P满足，则可推出结论Q或执行Q所规定的操作。如考虑不确定性，需另附可信度度量值，如CF=[0,1]；CF(Certainty Factor)为确定性因子，也称置信度。

用 BNF（Backus Normal Form）表示的产生式形式描述及语义：

〈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>, ...)]

由于后件由前件来触发，所以，前件是规则的执行条件，后件是规则体。产生式还包括各种操作、规则、变换、算子、函数等等。产生式描述了事物之间的一种对应关系（包括因果关系和蕴含关系）。一条产生式规则就是一条知识，用产生式可以实现推理和操作。

依据规则右部的表示方式，可以把规则分为条件-动作型和前提-结论型。前提-结论型规则更接近于演绎推理规则，例如，“若 某动物是哺乳动物，且 吃肉；则 这种动物是食肉动物”。也可表示为：(Mammal ?x) ∧ (Eat ?x Meat)) => (Carnivore ?x)，其中，“?x”称为模式变量；可以把“?x”视为隐含地受全称量词约束，从而该规则实际上就是一条正向演绎推理规则。不过，前提-结论型规则并不要求遵从一阶谓词演算的表示形式，实际上规则的前提可以是匹配模式（即谓词公式）、关系表达式和真值函数的任意“与、或、非”组合，而规则的结论则可以是任意数据结构（向量、数组、表格等）。条件-动作型规则除左部类同于前提-结论型规则外，规则右部可以是任意操作函数，不仅用于操作综合数据库，也可是屏幕、图像、文件操作和用于执行各种预定的计算功能。

鉴于规则的结论可视为对综合数据库的增加操作，为表述方便起见，除特别需要场合外，我们不再区分这二种规则，即将规则的前提或条件统称规则的左部（或前件），规则的结论和动作也不加区分地统称为规则的右部（或后件）。

谓词逻辑中的蕴涵式与产生式的基本形式相似，事实上，蕴涵式只是产生式的一种特殊情况。因为，（1）蕴涵式只能表示精确知识，其值非“真”即“假”，而产生式不仅可以表示精确知识，而且还可以表示不精确知识。例如，MYCIN中有如下产生式：**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 CF=0.6** 。这里，CF表示知识的强度，谓词逻辑中的蕴涵式不可以这样做。(2)用产生式表示知识的系统中，“事实”与产生式的“前提”中所规定的条件进行匹配时，可以是“精确匹配”，也可以是基于相似度的“**不精确匹配**”，只要相似度落入某个预先设定的范围内，即可认为匹配。但对谓词逻辑的蕴涵式而言，其匹配必须是精确的。

产生式规则表示法具有非常明显的优点：①自然性好，产生式表示法用“If-THEN”的形式表示知识,这种表示形式与人类的判断性知识基本一致，直观、自然，便于推理；②除了对系统的总体结构、各部分相互作用的方式及规则的表示形式有明确规定以外,对系统的其它实现细节都没有具体规定，这使设计者们在开发实用系统时具有较大灵活性，可以根据需要采用适当的实现技术，特别是可以把对求解问题有意义的各种启发式知识引入到系统中；③表示的格式固定，形式单一，规则间相互独立，整个过程只是前件匹配，后件动作，匹配提供的信息只有成功与失败，匹配一般无递归，没有复杂的计算，这使得知识库和系统的建立较为容易；特别是知识库与推理机是分离的，这种结构给知识的修改带来方便，无须修改程序，对系统的推理路径也容易作出解释；④由于规则库中的知识具有相同的格式，并且全局数据库可以被所有的规则访问，因此规则可以被统一处理；⑤模块性好,产生式规则是规则中最基本的知识单元，各规则之间只能通过全局数据库发生联系，不能互相调用，增加了规则的模块性,有利于对知识的增加、删除和修改；⑥产生式表示法既可以表示确定的知识单元，又可以表示不确定性知识；既有利于表示启发式知识，又可方便地表示过程性知识；既

可表示领域知识，又可表示元知识。所以，产生式表示知识常作为构造专家系统的第一选择。但是，产生式规则表示法也存在着下列缺点：①推理效率低下。由于规则库中的知识都有统一格式，并且规则之间的联系必须以全局数据库为媒介，推理过程是一种反复进行的“匹配-冲突消除-执行”的过程。而且在每个推理周期，都要不断地对全部规则的条件部分进行搜索和模式匹配，从原理上讲，这种做法必然会降低推理效率，而且随着规则数量的增加，效率低的缺点会越来越突出，甚至会出现组合爆炸问题。②不直观。数据库中存放的是一条条相互独立的规则，相互之间的关系很难通过直观的方式查看。③缺乏灵活性。产生式表示的知识有一定的格式，规则之间不能直接调用，因此较难表示那些具有结构关系或层次关系的知识，也不能提供灵活的解释。

17.2.6.2 产生式系统

产生式系统是应用产生式表示的知识和问题求解系统，是智能系统的一种最典型最普遍的结构形式。从体系结构看，目前大多数的智能系统（如专家系统）都是产生式系统。一般来讲，产生式系统（Production System）由三个基本部分组成：产生式规则库、综合数据库（动态数据库）和控制系统（推理机）。前二者构成产生式系统的知识和问题状态表示（描述），后者则控制应用规则推出解答的全过程。

产生式规则库亦称产生式规则集，由领域规则组成，是产生式规则的集合，用于描述应用领域的常识和启发式知识，所以规则库就是产生式系统的知识库。在规则库中，规则是以产生式表示的，在机器中则以某种动态数据结构进行组织。规则集蕴涵着将问题从初始状态转换到解状态的那些变换规则，是专家系统的核心。规则可表示成与或树形式，事实对这与或树的求值过程就是推理过程。在建立规则库时，一要注意有效地表达领域内的过程性知识：包括规则的建立、不确定性知识的表示、推理链的形成、知识的完整性等。二要对知识进行合理的组织与管理：目的是使得推理避免访问与所求解的问题无关的知识，以提高问题求解效率。

综合数据库是一个用于存放问题求解过程中各种当前信息（例如，问题的初始状态、原始证据、推理中得到的中间结论、最终结论等）的动态数据库。当规则库中某条产生式的前提可与综合数据库中的某些已知事实匹配时，该产生式就被激活，系统将把用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，是动态的。 综合数据库中的已知事实通常用字符串、向量、集合、矩阵、表等数据结构来表示。综合数据库可视为推理过程中间结果的存贮池。随着中间结果的不断加入，将使综合数据库描述的问题状态逐步转变为目标状态。

控制系统又称推理机构（推理机），由一组程序组成，负责驱动和控制整个产生式系统的运行，特别是控制协调规则库与数据库的运行，包含推理方式和控制策略，实现对问题的求解。控制系统的主要工作：（1）按一定的策略从规则库中选择规则，并与综合数据库中的已知事实进行匹配。（2）当发生冲突（即匹配成功的规则不止一条）时，调用相应的冲突解决策略予以消解。（3）在执行某条规则时，若该规则的右部是一个或多个结论，则把这些结论加到综合数据库中；若规则的右部是一个或多个操作，则执行这些操作。（4）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。（5）随时掌握结束产生式系统运行的时机，以便在适当的时候停止系统的运行。

控制系统基本的控制流程是：识别-行动循环，并在每一循环中选择激活（条件部分满足）的规则和执行规则右部拟定的动作。在每个循环的识别阶段，控制系统在规则库中识别条件为真的规则，使这些规则激活；然后在该循环的行动阶段，执行激活的规则，即执行规则的右部指定的对于综合

数据库的操作和任何其它合适的操作。

由于规则的条件部分不限于谓词公式（可以是关系表达式和真值函数），且动作可以是任何操作（不像演绎推理系统中规则的右部只能是推理结论），因而增加了表示法的概念效率。另外，通过设置控制元素于综合数据库和采用多规则激活情况下的冲突解法，可以有效地控制推理过程，增加表示法的计算效率。

在一个循环的识别阶段，若有多于一条的规则激活，就称引起了一个冲突，所谓冲突解决就是基于某种控制策略去选定需要执行的规则。冲突解决的策略可以分为三大类：

• First--选用首条激活的规则加以执行；

• Best--选用已激活规则中最好的加以执行，这里“最好”的评价依赖于系统制定的标尺；

• All--执行所有激活的规则。

例如，在关于动物世界的产生式系统中，已建立规则库，其包括上述形式化表示的规则；也建立了如上所述的综合数据库；则因该规则的左部二个匹配模式（Mammal ?x）和（Eat ?x，Meat）分别与综合数据库中的事实元素（Mammal Dog）和（Eat Dog Meat）匹配（能合一），从而得以激活，并将规则右部的结论（Carnivore Dog）插入综合数据库。

实际上，产生式系统的控制机制就是不断地挑选可激活的规则对综合数据库进行操作，直至得到解答（综合数据库内容转变为描述了目标状态），或失败结束。产生式系统的三大组成部分的相互关系如下：

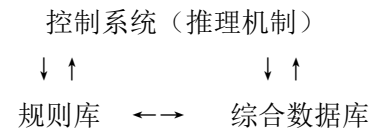

图 17.2.1 产生式系统基本结构

17.2.6.3 产生式系统求解问题的一般步骤

产生式系统运行时，除了需要规则库以外，还需要有初始事实(或数据)和目标条件。目标条件是系统正常结束的条件，也是系统的求解目标。产生式系统的运行过程就是从初始事实出发，寻求到达目标条件的通路的过程。所以，产生式系统的运行过程也是一个搜索的过程，但一般把产生式系统的运行过程称为推理。产生式系统启动后，推理机就开始推理，按所给的目标进行问题求解。一个实际的产生式系统，其目标条件一般不会只经一步推理就可满足，往往要经过多步推理才能满足或者证明问题无解。产生式系统求解问题的一般步骤是：

（1）初始化综合数据库，把问题的初始已知事实送入综合数据库中；

（2）若规则库中存在尚未使用过的规则，而且它的前提可与综合数据库中的已知事实匹配，则继续；若不存在这样的事实，则转第（5）步；

（3）执行当前选中的规则，并对该规则做上标记，把该规则执行后得到的结论送入综合数据库中；若该规则的结论部分指出的是某些操作，则执行这些操作；

（4）检查综合数据库中是否已包含了问题的解，若已包含，则终止问题的求解过程；否则，转第（2）步；

（5）要求用户提供进一步的关于问题的已知事实，若能提供，则转第（2）步；否则，终止问题求解过程；

（6）若规则中不再有未使用过的规则，则终止问题的求解过程。

17.2.6.4 产生式系统的推理与控制策略

1．产生式系统的推理模式。

利用产生式规则可以实现有前提条件的指令性操作，也可以实现逻辑推理。实现操作的方法是当测试到一条规则的前提条件满足时，就执行其后部的动作。这称为规则被触发或点燃。利用产生式规则实现逻辑推理的方法是当有事实能与某规则的前提匹配（即规则的前提成立）时，就得到该规则后部的结论（即结论也成立）。实际上，这种基于产生式规则的逻辑推理模式，就是逻辑上所说的假言推理（对常量规则而言）和三段论推理（对变量规则而言），即：

$$\frac{A \rightarrow B \quad A}{B} \qquad \frac{A \rightarrow B \quad B \rightarrow C}{A \rightarrow C}$$

这里的大前提就是一个产生式规则，小前提就是证据事实。

我们把有前提条件的操作和逻辑推理统称为推理，可认为产生式系统中的推理是更广义的推理。

产生式系统的推理方式有正向推理、逆向推理和双向推理。

正向推理或称数据驱动方式，就是从初始的事实数据出发，正向使用规则进行推理，并推动问题求解从初始状态向目标状态逼近，最终求得问题的解。以正向推理方式使用的规则称为正向规则，或F规则（Forward rule）。推理过程是规则集中的规则前件与数据库中的事实进行匹配，或用动态数据库中的数据测试规则的前提条件是否满足，得到匹配的规则集合。从匹配规则集合中选择一条规则作为使用规则。执行使用规则的后件。将该使用规则的后件送入数据库中。重复这个过程直至达到目标。具体说如数据库中含有事实A，而规则库中有规则A->B，那么这条规则便是匹配规则，进而将后件B送入数据库中。这样可不断扩大数据库直至包含目标便成功结束。如有多条匹配规则需从中选一条作为使用规则，不同的选择方法直接影响着求解效率，选规则的问题称作控制策略。

正向推理具体算法是：（1）将初始事实/数据置入动态数据库；（2）用动态数据库中的事实/数据，匹配/测试目标条件，若目标条件满足，则推理成功，结束；（3）用规则库中各规则的前提匹配动态数据库中的事实/数据，将匹配成功的规则组成待用规则集；（4）若待用规则集为空，则运行失败，退出；（5）将待用规则集中各规则的结论加入动态数据库，或者执行其动作；转（2）。

可以看出，随着推理的进行，动态数据库的内容或者状态在不断变化。如果把动态数据库的每一个状态作为一个节点的话，则上述推理过程就是一个从初始状态（初始事实或数据）到目标状态（目标条件）的状态图搜索过程。

逆向推理也称为反向推理、目标驱动推理。就是从目标（作为假设）出发，反向使用规则进行推理，朝初始事实或数据方向前进，证实由已知事实可以推出目标。其推理过程是：将规则集中的规则后件与目标进行匹配，得匹配的规则集合；从匹配的规则集合中选择一条规则作为使用规则；将使用规则的前件作为子目标；重复这个过程直至各子目标均为已知事实成功结束。以逆向推理方式使用的规则称为逆向规则，或B规则（Backward rule）。

逆向推理的具体算法是：（1）将初始事实/数据置入动态数据库，将目标条件置入目标链；（2）若目标链为空，则推理成功，结束；（3）取出目标链中第一个目标，用动态数据库中的事实/数据同其匹配，若匹配成功，转步（2）；（4）用规则集中的各规则的结论同该目标匹配，若匹配成功，则将第一个匹配成功且未用过的规则的前提作为新的目标，并取代原来的父目标而加入目标链，转

步（3）；（5）若该目标是初始目标，则推理失败，退出；（6）将该目标的父目标移回目标链，取代该目标及其兄弟目标，转（3）。

可以看出，与正向推理不同，逆向推理的推理树是从上而下扩展而成的，而且推理过程中还会发生回溯。其推理的过程也是一个图搜索过程，而且一般是一个与或树搜索。

除了正向推理和逆向推理外，产生式系统还可进行**双向推理**。双向推理是同时使用正向推理和逆向推理，就是同时从初始数据和目标条件出发进行推理去求解问题，如果在中间某处相遇，则推理搜索成功。从上面的分析可以看出，正向推理是自底向上的综合过程，而逆向推理则是自顶向下的分析过程。双向推理时的综合数据库必须有两套数据结构，分别描述从初始状态出发推得的中间状态—正向状态，和从目标状态出发推得的中间状态—逆向状态。换言之，综合数据库 = 正向状态描述 +逆向状态描述，以便于F、B规则分别作用于不同的状态描述。

选用那种推理模式，主要取决于问题的特征，也取决于推理的分支因素，即每个识别-行动循环激活的规则数。显然规则数越大，分支越多，规则的激活检查和启发式评价的工作量也就越大。所以，应选择分支因素小的推理方向。在正、逆向分支因素接近的情况下，双向可提高效率，但由于需求解的问题本身的复杂性和关于规则选用的启发式知识往往不完善，也易于发生双向推理不相交的情况，导致比单向更低的推理效率。

2. **产生式系统的控制策略**

在正向推理算法中，若对所有匹配成功的规则都同时触发启用，所实现的搜索实际上是穷举式的树式盲目搜索。此时，正向推理的启发式搜索算法为：（1）将初始事实/数据置入动态数据库；（2）用动态数据库中的事实/数据，匹配/测试目标条件，若目标条件满足，则推理成功，结束；（3）用规则库中各规则的前提匹配动态数据库中的事实/数据，将匹配成功的规则组成待用规则集；（4）若待用规则集为空，则运行失败，退出；（5）用某种策略，从待用规则集中选取一条规则，将其结论加入动态数据库，或者执行其动作，撤消待用规则集；转（2）。

可以看出，该算法与前面的算法仅在步骤5有所差别。规则的选择策略称为冲突消解策略。产生式系统的推理方式、搜索策略及冲突消解策略等，一般统称为推理控制策略，简称控制策略。控制策略常体现在推理机的算法描述中。

从产生式系统的推理控制机制可知，选用合适的规则推进问题求解是控制策略的主要内容。鉴于智能系统通常面向的是困难问题，系统设计时获取的关于求解问题的知识往往不足以确定最合适的规则；从而使问题求解在一定程度上常表现为一种搜索过程，即仅对有希望导致最好解答的规则作尝试性使用。

一般图搜索策略同样适用于作为产生式系统的控制策略，只是产生式系统以激活的规则（而非问题状态）作为试探性选择的对象，从而需作某些特别的处理。一个最突出的不同于一般图搜索的处理就是问题状态的恢复。在一般图搜索过程中，问题状态（节点）是基本处理对象，并记载了其在该状态下的综合数据库内容，不存在搜索（推理）失败回溯时的问题状态恢复需求。然而，产生式系统处理的基本对象是规则，综合数据库只反映某条规则执行（即其右部执行）后的当前问题状态，规则执行前的问题状态未加保留；所以当推理（搜索）失败回溯时，必须设法恢复到规则执行前的状态，才能再选择另一激活的规则去推进问题求解。这就要求，一旦发现试用失败，就需撤销试用规则对问题状态产生的影响，使综合数据库的内容恢复到执行该规则之前，再选用别的可激活规则作试探性推理；或者作某些弥补工作（即设法对已得到的部分解答进行修正），再继续进行推理；

以期最终搜索到正确地解答。

类似于搜索策略，可以把产生式系统的控制策略划分为二大类：无信息控制和优化控制，后者又可划分为不可回溯和可回溯二种方式。

（1）无信息控制

这种控制策略实际上就是简单地应用“First”冲突解法去控制整个问题求解过程，并相当于深度优先的盲目搜索方式。这种控制策略类似于Prolog的推理机制。推理过程的每个识别-行动循环的识别阶段按排列顺序自上而下地对规则库中的规则作激活检查，第一条激活的规则就用于推进问题求解。然后基于新的问题状态（由内容改变了的综合数据库指示），新一轮的识别-行动循环开始。依次进行，直到推理成功或失败。

（2）不可回溯的优化控制

优化控制实际上就是应用“Best”冲突解法去控制整个问题求解过程，相当于应用启发式知识的搜索方式。应用启发式知识选择“最好”的规则去推进问题求解，意味着这些服务于推理控制的知识是关于如何选取问题求解基础知识（产生式规则）的知识，所以称之为“元知识”。换言之，服务于控制的元知识是产生式系统优化控制推理过程的关键。

在推理过程不可回溯的优化控制方式下，识别-行动循环一个接一个地推动问题求解向目标状态前进，即使有些循环中选用的规则在执行后发现不合适（甚至错误），也不撤销它们，而是选择新的更合适的规则去弥补它们的不良影响。尽管需要从应用领域获取服务于控制的元知识或直接将元知识隐含地设置于推理机（就像将h(n)函数作为搜索控制器的一部分那样），总起来讲，这种控制方式简单易行，对于注重找到解答而不留意解答步骤（识别-行动循环个数）多少的应用问题较为适合，如八数码游戏，文法分析等问题。有些应用问题本身不允许回溯，即已执行的规则右部操作函数不可撤销，例如化学反应监控系统，一旦操作已做，无法撤销和还原，只能做某些补救工作。再如下棋程序，一旦落子，即使“臭着”，也不能悔棋。实现不可回溯优化控制的一种方式就是爬山法，控制系统只需以应用领域相关的启发式知识指导Best冲突解法的执行，一旦选定某条激活的规则，其余激活的规则全部丢弃，不再予以考虑。但爬山法只适合于单一极值问题，即确保能推出解答而不会陷入失败的问题。对于推理过程可能会陷入失败点的问题，不宜采用爬山法，因为推理过程一旦陷入失败（非最大值的极值点）就不能自拔。

（3）可回溯的优化控制

这种控制方式允许在推理进入失败点时返回到按时序最接近的推理分支点，让推理过程沿另一分支方向前进。

如前已述，递归过程是实现回溯控制策略的有效途径，可以将递归过程算法改写为服务于产生式系统的可回溯优化控制，并取名为 PS-BACKTRACK。

PS-BACKTRACK(db)：

(1) 若 db 指示了目标状态，则输出（显示）db 作为解答,算法成功结束；

(2) 若 db 指示了失败状态，则返回真值 F；

(3) rs:=RULE-ACTIVATE(db)，并用启发式知识对 rs 中的规则按从优到劣的次序排列；

(4) 若 rs 为空，则返回真值 F；

(5) r:=MOVE-FIRST(rs)；

(6) PS-BACKTRACK(TRANSFORM(db,r))；

(7) 返回语句（4）。

在上述算法中，db 设置为综合数据库的内容，调用算法进行问题求解时，db 指示问题求解的初始状态。算法的每次调用执行一个识别-行动循环。算法前 3 句对应于识别阶段，第（1）句识别解答是否成功，第（2）句识别算法是否进入失败状态，否则进一步通过函数 RULE-ACTIVATE 识别可激活的规则，把它们收集于列表 rs 中，并用启发式知识排序。算法后 4 句进入循环的行动阶段。首先通过函数 MOVE-FIRST 取出 rs 表首的规则，记为 r；然后通过函数 TRANSFORM,用 r 对 db 进行操作，并生成综合数据库的新内容 db′（db 仍保留原内容）；再基于 db′ 递归调用算法去推进问题求解。

若递归调用进入失败状态，则必须返回到算法的本次调用，并依据语句（7）的指示，返回语句（4），从 rs 取用表首规则（即剩余规则中最优的一条），从另一推理分支方向前进。这就实现了推理失败时的回溯，由于采用递归算法，从失败状态回溯到上一状态时，上一状态的描述（即 db）仍然保留。实际上每一次算法的递归调用，就通过参数 db 保留了对于当前状态的“快照”。所以不需作任何状态恢复工作，就可顺利实现回溯控制策略。在 rs 为空的情况下，推理过程继续往上回溯。若回溯到算法的顶层调用都找不到解答，则整个算法失败结束。

（4）推理修补技术

尽管可回溯的优化控制不失为一种较好的控制策略，但若启发式知识还不健全的话，可能会引起太多的失败回溯，以至严重影响推理效率。再说，这种按时序的回溯往往只是盲目地回溯到上一个推理分支点，并不能准确地直接返回到推理失败的根源相应的推理分支点，使许多回溯工作白白浪费。另一方面，有些问题求解任务不允许回溯或不需要回溯。所以，作为回溯控制方式的对立策略，可以开发推理修补技术，免除回溯的需要。

在简单情况下。可以通过设置修补规则来实现推理修补，以便在推理失败的情况下激活这些规则，对推理的中间结果（综合数据库的内容）作某些变换，进而使失败的推理能继续下去。例如对于下面的英语句子作文法分析：Have the Students who Missed the exam take it today.句首单词“Have”可以作为助动词或主动词。若文法分析时先试探性地将其作为助动词，则名词短语“the student who missed the exam”被分析为主语。由于随后出现“take”而非“taken”，则推理（文法分析）失败。但只要把“have”改作主动词，其后的名词短语改作宾语，就可继续文法分析直到成功，不必回溯。这种推理修补可以通过设立相应的产生式规则来实现，该规则依据现有的分析结果（置于综合数据库的部分解答），即“have”是助动词、“students”是宾语和“take”是动词原形，被激活而进行预定的修补操作。

一种称为“从属导向”的回溯技术也有类似的推理修补效果。当推理到达失败点时，并不按时序简单地回溯到上一推理分支点，而是分析失败的原因，直接确定导致失败的推理分支点。另外，推理系统并不撤销失败点到失败根源之间的所有推理结果，而是依据记载的推理前提和结果间的从属关系信息，仅撤销那些与失败相应的结果，即相当于作了推理修正工作，从而比盲目的时序回溯有效得多。严格地说，“从属导向”的回溯并非回溯技术，因为其不撤销推理路径上已产生的正确结果。

（5）产生式系统与图搜索

分析前面给出的两个正向推理算法，可以看出，它们只能用于解决逻辑推理性问题。如果要用正向推理来求解规划性问题，则上述算法中至少还需要增加以下功能：(1)记录动态数据库状态变化的历史，这就需要增设一个CLOSED表。(2)若要回溯，则还需保存与每个动态数据库状态对应的可用

规则集。因为动态数据库状态与可用规则集实际是一一对应的。(3)要进行树式搜索，还需设置一个OPEN表，以进行新生动态数据库的状态保存和当前动态数据库状态的切换。(4)还要考虑一条规则是否只允许执行一次。若是，则要对已执行了的规则进行标记。

这样一来，产生式系统的推理算法与图搜索算法就相差无几了。事实上，产生式系统和图搜索在对问题的表示和解法实际是等价的。要说差别的话，图搜索技术描述了问题求解的方法，而产生式系统则给出了实施这种方法的一种计算机程序系统的结构模式。这样，问题求解、图搜索和产生式系统三者的关系是：问题求解是目的，图搜索是方法，产生式系统是形式。

能用问题归约方式解决的问题也可以通过设计可分解的产生式系统来加以求解。可分解的产生式系统的基本思想是：把一个规模较大且比较复杂的问题（初始数据库）分解为分别若干个规模较小且比较简单的子问题，然后对每个子问题进行求解。鉴于综合数据库的内容反映了问题求解状态，将该内容分解为若干独立的部分去对应于子问题求解，且子问题解答能联合构成整个问题的解答，成为设计可分解产生式系统的必要条件。对应可分解的产生式系统，解答成功的关键在于每个子问题（直到本原问题）都必须解答成功。推理过程可以表示为类似的与或图，所有推理的成功实际上就意味着搜索到了一个一致解图。由于问题归约所具有的优势，可分解产生式系统能解决某些复杂的问题求解任务，而且具有较高的效率。

17.2.6.5　产生式系统的开发与程序实现【仅供参考】

1．产生式规则的程序语言实现

一般来讲，产生式规则的前提和结论部分可以是一个复杂的逻辑表达式，但为了使表达简单规范，且便于推理，在实践中人们往往把规则的前提部分整理成形如：（（条件1）AND（条件2）AND … AND（条件n））或（（条件1）OR（条件2）OR … OR（条件m））的形式（其中的条件可以带否定词）；把规则结论部分整理成形如：（（断言1/动作1）AND（断言2/动作2）AND … AND（断言k/动作k））或（（断言1/动作1）OR（断言2/动作2）OR … OR（断言k/动作k））的形式，或者进一步简化成（断言/动作）即仅有一项的形式。由于含OR关系的规则也可以分解为几个不含OR关系的规则，所以，产生式规则也可仅取下面的一种形式：（（条件1）AND（条件2）AND … AND（条件n））→（断言/动作）。即前件是若干与关系的条件，后件仅有一个断言或动作。

2．规则库的程序实现

规则库的程序实现分为内存和外存两个方面。在内存中规则库可用链表实现，在外存则就是以规则为基本单位的数据文件。但若用PROLOG程序，对于用PROLOG的规则表示的产生式规则，规则库就是程序的一部分；对于PROLOG事实表示的规则，则规则库在内存就是动态数据库，在外存就是数据库文件。还需说明的是，对于规则库实际上还需配一个管理程序，即知识库管理系统，专门负责规则及规则库的各项管理工作。知识库管理系统的设计也与规则的表示形式密切相关。

3．动态数据库的程序实现

动态数据库中事实数据的具体表示方法与上面所述的规则条件与结论的语言表示方法基本一样，区别就是动态数据库中的事实数据中不能含有变量。动态数据库在内存可由（若干）链表实现并组成。在PROLOG程序中实现动态数据库，则可不必编写链表程序，而利用PROLOG提供的动态数据库直接实现。

4．推理机的程序实现

推理机的程序实现，除了依据某一控制策略和算法编程外，一般来说，程序中还应具有模式匹配与变量的替换合一机制。因为模式匹配是推理的第一步，同时规则中一般都含有变量，而变量的匹配必须有替换合一机制的支持。当然，要实现合一，就要用合一算法。那么，前面归结推理中的合一算法，对产生式系统也是适用的。

17.2.6.6 对产生式系统的几点看法

产生式系统具有许多优点：（1）自然性。由于产生式系统采用了人类常用的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。（2）模块性。产生式是规则库中的最基本的知识单元，形式相同，易于模块化管理。规则（知识单位）间相互较为独立，使知识库的建立较为容易。（3）有效性。能表示确定性知识、不确定性知识、启发性知识、过程性知识等。（4）清晰性。产生式有固定的格式，既便于规则设计，又易于对规则库中的知识进行一致性、完整性检测。（5）简便性。另外推理方式单纯，也没有复杂计算。特别是知识库与推理机是分离的，这种结构给知识的修改带来方便，无须修改程序，对系统的推理路径也容易作出解释。因此，产生式规则最适合于表示各种启发式知识以指示事物间的经验性关联，可广泛地应用于专家系统的设计中。产生式表示知识应是构造专家系统的第一选择。

产生式系统的缺点在于：（1）效率不高。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。由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。此外，在求解复杂问题时容易引起组合爆炸。（2）规则的堆积存储，缺乏组织；冲突解法单一，不能自然地适应于许多场合；而通过设置控制元素来设计控制策略，在复杂的情况下也往往不可行。（3）不能表达具有结构性的知识。产生式系统对具有结构关系的知识无能为力，它不能把具有结构关系的事物间的区别与联系表示出来，因此，人们经常将它与其它知识表示方法（如框架表示法、语义网络表示法）相结合。

由产生式系统与图搜索的关系可见，产生式系统完全可以作为问题求解的表示模型和求解模型，而且可作为人工智能问题求解系统的通用模型。用产生式系统也可实现基于谓词逻辑的演绎推理和证明。事实上，当一个产生式系统中的规则是逻辑蕴含式时，其运行过程就是演绎推理（假言推理或三段论）的过程。这时目标值已知时就是证明，当目标值未知时就是推理求值。还需要说明的是，归结演绎系统也可以看作是一种特殊的产生式系统，只是这时规则只有一条，那就是归结原理。由于产生式系统既可用于操作性问题求解，也可用于推理性问题求解。因此，产生式系统也是专家系统的基本结构形式。用它既可实现规划型专家系统，也可实现结论型专家系统。总之，产生式系统在人工智能技术中占有重要地位。

产生式系统最适用领域是：（1）由许多相对独立的知识元组成的领域知识，彼此之间关系不密切，不存在结构关系。（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。

17.2.7 问题求解系统中（曾经的）基于搜索的通用问题求解方法

作为一种形式化表示，一个通用问题可表示为：

问题 ={问题领域；问题背景；问题状态[初始状态 可能状态 目标状态]}

而相应地，一个通用问题求解系统可表示为：

问题求解系统 ={问题；操作算子；搜索引擎[搜索策略]；评价函数[路径搜索代价]；解集合}

对上述问题的求解本质上是寻求问题的解集合或最优解。其求解的关键则通常是合适的搜索技术的研究和开发。对搜索技术的研究曾经是人工智能早期研究致力于寻求的目标。而最适合于设计一种搜索技术来实现求解的问题是基于一个**操作算子集**的问题求解任务，其每个操作算子的执行均可使问题求解更接近于目标状态，此时，问题的搜索路径将由实际选用的操作算子的序列构成。本

节，我们将主要介绍基于问题状态空间的一般图搜索算法和启发式搜索算法，然后介绍考虑问题规约的与或图搜索算法。

17.2.7.1 问题状态空间与搜索

用搜索技术来求解问题的系统均定义一个状态空间，并通过适当的搜索算法在状态空间中搜索解答或解答路径。状态空间搜索的研究焦点在于设计高效的搜索算法，以降低搜索代价并解决组合爆炸问题。

状态空间 SP 可表示为一个二元组：SP =（S,O)。S 是在问题求解（即搜索）过程中所有可达的合法状态构成的集合；O 是操作算子的集合，操作算子的执行导致状态的变迁。所以，状态空间又可描述为一个有向图，其节点指示状态，节点间的有向弧表示状态变迁，弧上的标签则指示导致状态变迁的操作算子。状态可通过定义某种数据结构来描述，用于记载问题求解（即搜索）过程中某一时刻问题现状的快照。

状态空间的搜索 SE 可表示为一个五元组：SE = (S,O,E,I,G)。其中，E--搜索引擎；I--问题的初始状态，I ∈ S；G--问题的目标状态集，G∈S。状态空间搜索的基本思想就是通过搜索引擎寻找一个操作算子的调用序列，使问题从初始状态变迁到目标状态之一，而变迁过程中的状态序列或相应的操作算子调用序列称为从初始状态到目标状态的解答路径。搜索引擎可以设计为任意实现搜索算法的控制系统。

通常，状态空间的解答路径有多条，但最短的只有一条或少数几条。由于一个状态可以有多个可供选择的操作算子，导致了多个待搜索的解答路径。这种选择在逻辑上称为“或”关系，意指只要其中有一条路径通往目标状态，就能获得成功解答。由此，这样的有向图称为“或图”，常见的状态空间一般都表示为或图，因而也称“一般图”。除了少数简单问题外，描述状态空间的一般图都很大，无法直观地画出，只能将其视为隐含图，在搜索解答路径的过程中只画出搜索时直接涉及到的节点和弧线，构成所谓的“搜索图”。搜索图小得多，一般可以图示。如此，尽管状态空间可以很大，但只要确保搜索空间足够小，就能在合理的时间范围内搜索到问题解答。

搜索空间的压缩程度主要取决于搜索引擎采用的搜索算法。换言之，当问题有解时，使用不同的搜索策略，找到解答路径时，搜索图的大小是有区别的。一般来说，对于状态空间很大的问题，设计搜索策略的关键考虑是解决组合爆炸问题。复杂的问题求解任务往往涉及许多解题因素，问题状态就可以通过解题因素的特别组合来加以表示（解题因素可设计为状态变量）。所谓组合爆炸意指解题因素多时，因素的可能组合个数会爆炸性（指数级）增长，引起状态空间的急剧膨胀。例如某问题有 4 个因素，且每个因素有 3 个可选值，则因素的组合（即问题状态）有 3^4 =81 个；但若因素增加到 10 个，则组合的个数达 3^{10} =3^4 ×3^6 =81×729，即状态空间扩大到 729 倍。解决组合爆炸问题的方法实际上就是选用好的搜索策略，使得只要搜索状态空间的很小部分就能找到解答。

17.2.7.2 一般图搜索策略

如前所述，一般图搜索是在状态空间中搜索从初始状态到目标状态解答路径的过程。这里涉及到几个与图论相关的术语：节点深度--搜索图是一种有根图，根节点指示初始状态，令其节点深度为 0，则搜索图中的其它节点的深度 d_n就可递归地定义为其父节点深度 d_{n-1}加 1： $d_n=d_{n-1}+1$。路径--从节点 n_i到 n_k的路径是由相邻节点间的弧线构成的折线，通常要求路径是无环的，否则会导致搜索过程进入死循环。节点扩展--应用操作算子将上一状态（节点 n_i）变迁到下一状态（节点 n_j），n_i称被扩展节点，n_j 即是 n_i的子节点。路径代价--相邻节点 n_i和 n_j间路径的代价记为 $C(n_i, n_j)$，它

由两部分组成：路径本身代价和路径搜索代价。路径本身代价即操作算子的执行代价。路径搜索代价又分为二部分：路径建立代价和路径选择代价。前者是搜索过程中从节点 n_i 扩展出节点 n_j 所付出的代价；后者则是选择这条路径作为搜索方向（即选择 n_j 作为下一步搜索起点）所付出的代价。在许多情况下，任意相邻节点（即父-子节点）间路径的搜索代价大致同等。为简化处理起见，令任意相邻节点间的路径代价相同，并以路径长度 1 指示。记目标状态相应的节点为 n_g，则从 n_i 到 n_g 的路径代价可递归地定义为 $C(n_i, n_g) = C(n_i, n_j) + C(n_j, n_g)$。

令：s—为初始状态节点；G—为搜索图；OPEN--作为存放待扩展节点的表；CLOSE--作为存放已被扩展节点的表；MOVE-FIRST(OPEN)—表示取 OPEN 表首的节点作为当前要被扩展的节点 n，同时将节点 n 移至 CLOSE 表；SNS—为子节点集合；一般图搜索算法的一般过程如下：

1） G := s； / 即算法开始时搜索图只包括初始状态节点；

2） OPEN := (s), CLOSE := ( )； / 即此时仅有 s 作为待扩展节点，而 CLOSE 表为空；

3） 若 OPEN 是空表，则算法以失败结束； / 因为此时未搜索到解答（目标状态），又无法继续搜索；

4） n := MOVE-FIRST(OPEN)；

5） 若 n 是目标状态节点，则搜索成功结束，并给出解答路径；

6） 扩展节点 n，将非节点 n 祖先的子节点置于子节点集合 SNS 中，并插入搜索图 G 中；

7） 标记和修改指针：

- 把 SNS 中的子节点分为三类：（1）全新节点，（2）已出现于 OPEN 表的节点，（3）已出现于 CLOSE 表的节点；/后二类子节点实际上意味着具有新老二个父节点；
- 加第 1 类子节点于 OPEN 表，并建立从子节点到父节点 n 的指针；
- 比较第 2 类子节点经由新、老父节点到达初始状态节点 s 的路径代价，若经由新父节点的代价较小，则移动子节点指向；
- 对于第 3 类子节点作与第 2 类同样的处理，并把这些子节点从 CLOSE 表中移出，重新加入 OPEN 表；

8） 按某种原则重新排序 OPEN 表中的节点；

9） 返回语句 3)；

通过循环地执行该算法，搜索图会因不断有新节点加入而逐步长大，直到搜索到目标节点。实际上 OPEN 表中的节点都是该搜索图的叶子节点，它们联合起来作用为一个多路开关，并由排序原则决定选择哪一个作为下一次被扩展的节点。在搜索图中标记从子节点到父节点的指针，方便了在搜索到目标状态时快速返回解答路径：自初始状态 s 到目标状态的一个节点序列。

在一般图搜索算法中，提高搜索效率的关键在于优化 OPEN 表中节点的排序方式，若每次排在表首的节点都在最终搜索到的解答路径上，则算法不会扩展任何多余的节点就可快速结束搜索。所以排序方式成为研究搜索算法的焦点，并由此形成了多种搜索策略。

一种简单的排序策略就是按预先确定的顺序或随机地排序新加入到 OPEN 表中的节点，常用的方式是深度优先和宽度优先。深度优先规定，扩展当前节点后生成的子节点总是置于 OPEN 表的前端，即 OPEN 表作为栈表使用，后进先出，使搜索优先向纵深方向发展。宽度优先规定扩展当前节点后生成的子节点总是置于 OPEN 表的后端，即 OPEN 表作为排队表使用，先进先出，使搜索优先向横广方向发展。当一个问题有多个解答或多条解答路径，且只须找到其中一个时，深度优先法十分适用。

不过有些问题的状态空间搜索或许会无限延伸，而又存在较短的解答路径，则可以对搜索深度加以限制，以求提高搜索效率并确保寻找到较短的解答路径。相对地，先进先出排序策略使宽度优先法能确保搜索到最短的解答路径。

上述二种搜索方法可直接应用于一般图搜索算法，且只要问题有解就一定能搜索到解答。由于不需要设计特别的节点排序方法，从而简单易行，很适合于许多复杂度不高的问题求解任务。另外，由于这二种搜索方法实际上都是按某种顺序遍历搜索图中涉及的节点，故在一般图搜索算法的第(7)步，只需将第 1 类子节点加入 OPEN 表即可，另外 2 类子节点不必处理。

上述两种搜索方法的共同缺点是节点排序的盲目性，由于不采用领域专门知识去指导排序，往往会在白白搜索了大量无关的状态节点后才碰到解答，所以也被称为盲目搜索。

若引入与应用领域相关的启发式知识来指导 OPEN 表中节点的排序，使最有希望出现在较短解答路径上的节点优先考察，就能显著提高搜索的有效性。用启发式知识指导排序可有二种方式：局部优化排序--这是对上述深度优先法的改进，仅对新扩展出来的子节点排序，使这些节点中最有希望者能优先取出考察和扩展。全局优化排序--对 OPEN 表中的所有节点排序，使最有希望的节点排在表首。

17.2.7.3 一般图搜索的启发式搜索算法

搜索是一种试探性的查寻过程，引入启发式知识可以减少搜索的盲目性，增加试探的准确性，这种应用了启发式知识的搜索就是启发式搜索。**启发式知识对于搜索的指导作用**可归纳为三方面：① 选择下一个要被扩展的节点，排序 OPEN 表中的待扩展节点是一种常用的方法。②在扩展一个节点时，仅仅有选择性地生成部分有用的节点，而非所有可能的子节点。③ 修剪掉某些估计不可能导致成功的子节点。

我们只讨论如何应用第一方面的启发式知识于一般图搜索。一种有效的方法就是设计体现启发式知识的所谓评价函数来计算每个节点的得分，以便用于排列它们在 OPEN 表中的位置。应用这种评价函数来实现启发式搜索的典型是算法 A，其将评价函数 f 设计为：

$$f(n) = g(n) + h(n)$$

其中，n--搜索图中的某个当前被扩展的节点；f(n)--从初始状态节点 s，经由节点 n 到达目标状节点 o，估计的最小路径代价； g(n)--从 s 到 n ,估计的最小路径代价； h(n)--从 n 到 o,估计的最小路径代价。

通常我们可以用至今已发现的自 s 到 n 的最短路径作为 g(n)的值，但 h(n)则要依赖于启发式知识来加以估算，故而 h(n)称为启发式函数。算法 A 的设计与前述一般图算法类同，主要的不同是在算法第（6）句中增加了子节点函数 f 的计算，在第（7）句中依赖于 f 值确定子节点指向父节点指针的修改，并在第（8）句中按 f 值从小到大排序 OPEN 表中的节点。算法 A 第（6）和第（7）句的修改如下：

(6) 扩展节点 n，将非节点 n 祖先的子节点置于子节点集合 SNS 中，并插入搜索图 G 中，对于 SNS 中每个子节点 n_i,计算 $f(n, n_i) = g(n, n_i) + h(n_i)$；

(7) 标记和修改指针

- 把 SNS 中的子节点分为三类（同一般图搜索算法）；
- 加第 1 类子节点于 OPEN 表（同一般图搜索算法）；
- 比较第 2 类子节点 n_i 经由新、老父节点的评价函数值 $f(n, n_i)$与 $f(n_i)$；若 $f(n, n_i) < f(n_i)$，

则令 $f(n_i) = f(n, n_i)$，并移动子节点指向老父节点的指针，改为指向新父节点。

• 对第 3 类子节点作与第 2 类同样的处理，并把这些子节点从 CLOSE 表中移出，重新加入 OPEN 表。

启发式算法 A 按 f(n)排序 OPEN 表中的节点，f(n)值最小者排在首位，优先加以扩展，体现了最佳优先(best-first)搜索策略的思想。

鉴于启发式搜索在提高搜索效率和解决组合爆炸问题中的作用，相关的研究成为人工智能形成和成长期的重要议题之一，也产生了许多成熟的研究成果，并且至今启发式搜索仍是一个活跃的研究领域。下面仅就实现启发式搜索应考虑的关键因素作一些讨论。

1．搜索算法的可采纳性(Admissibility)。

在搜索图存在从初始状态节点到目标状态节点解答路径的情况下，若一个搜索法总能找到最短（代价最小）的解答路径，则称算法具有可采纳性。例如，宽度优先的搜索算法就是可采纳的，只是其搜索效率不高。为考察启发式搜索算法 A 的可纳性，首先引入评价函数 f^*：$f^*(n)= g^*(n) + h^*(n)$。其中，$f^*(n)$、$g^*(n)$、$h^*(n)$分别指示当经由节点 n 的最短（代价最小）解答路径找到时，实际的路经代价（长度）、该路径前段（自初始状态节点到节点 n）的代价和后段（自节点 n 到目标状态节点）的代价。在存在多个目标状态的情况下，$h^*(n)$取 $h^*(n, o_i)$中最小者（i=1，2，…）。

将评价函数 f 与 f^*相比较，实际上，f(n)、g(n)和 h(n)分别是 $f^*(n)$、$g^*(n)$和 $h^*(n)$的近似值。在理想的情况下，设计评价函数 f 时可以让 $g(n) = g^*(n)$,$h(n) = h^*(n)$,则应用该评价函数的算法 A 就能在搜索过程中，每次都正确地选择下一个从 OPEN 表中取出加以扩展的节点，从而不会扩展任何无关的节点，就可顺利地获取解答路径。然而 $g^*(n)$和 $h^*(n)$在最短解答路径找到前是未知的，故而几乎不可能设计出这种理想的评价函数；而且对于复杂的应用领域，即便是要设计接近于 f^*的 f 往往也是困难的。一般来讲，g(n)的值容易从迄今已生成的搜索树中计算出来，不必专门定义计算公式。例如就以节点深度 d(n)作为 g(n),并有 $g(n)\geqslant g^*(n)$。然而，h(n)的设计要依赖于启发式知识的应用，所以如何挖掘贴切的启发式知识是设计评价函数乃至算法 A 的关键。

可以证明，若确保对于搜索图中的节点 n，总是有 $h(n) \leqslant h^*(n)$，则算法 A 具有可采纳性，即总能搜索到最短（代价最小）的解答路径。宽度优先算法可视为采用了 $h(n) \equiv 0$ 的评价函数：$f(n) = g(n) = d(n)$，鉴于 $h(n) \equiv 0 < h^*(n)$，必定能搜索到最短路径。事实上，正是先进先出的 OPEN 表排序策略，使得宽度优先算法能确保搜索到最短路径。我们称满足 $h(n) \leqslant h^*(n)$的算法 A 为算法 A^*,显然，A^*算法也是可采纳的。

2．启发式函数的强弱及其影响

可以用 h(n)接近 $h^*(n)$的程度去衡量启发式函数的强弱。当 $h(n)<h^*(n)$且两者差距较大时，h(n)过弱，从而导致 OPEN 表中节点排序的误差较大，易于产生较大的搜索图；反之，当 $h(n)>h^*(n)$,则 h(n)过强，使算法 A 失去可采纳性，从而不能确保找到最短解答路径。显然，设计恒等于 $h^*(n)$的 h(n)是最为理想的，它可确保产生最小的搜索图（因为 OPEN 表中节点的排序正确），且搜索到的解答路径是最短的。

对于复杂的问题求解任务，设计恒等于 $h^*(n)$的 h(n)是不可能的。为此，取消恒等约束，设计接近、又总是小于 $h^*(n)$的 h(n)就成为了应用 A^*算法搜索问题解答的关键。可以证明，对于解决同一问题的两个算法 A_1和 A_2，若总有 $h_1(n) \leqslant h_2(n) \leqslant h^*(n)$，则 $t(A_1) \geqslant t(A_2)$。其中，h_1、h_2分别是算法 A_1、A_2的启发式函数，t 指示相应算法达到目标状态时搜索图含的节点总数。

3．设计 h(n)的实用考虑

随着问题求解任务复杂程度的增加，即便是设计接近、又总是小于 $h^*(n)$的 h(n)也变得更困难，而且往往会导致在 h(n)上的繁重计算工作量。若 h(n)的计算开销过大，即使最短路径找到，实际的搜索代价也会高居不下，因为路径选择代价随 h(n)的计算开销而大增。删除 $h(n) \leqslant h^*(n)$的约束，将会使 h(n)的设计容易得多，但却由此也丢失了可采纳性（可能丢失最短解答路径）。不过在许多实用场合，人们并不要求找到最优解答（最短解答路径），通过牺牲可采纳性来换取 h(n)设计的简化和减少计算 h(n)的工作量还是可行的。

从评价函数 f(n) = g(n) + h(n)可以看出，若 $h(n) \equiv 0$, 则意味着先进入 OPEN 表的节点会优先被考察和扩展，因为即使不以 d(n)作为 g(n), 通常先进入 OPEN 表的节点 n 也具有较小的 g(n)值，从而使搜索过程接近于宽度优先的搜索策略；反之若 $g(n) \equiv 0$,则导致后进入 OPEN 表的节点会优先被考察和扩展，因为后进入 OPEN 表的节点 n 往往更接近于目标状态，即 h(n)值较小，从而使搜索过程接近于深度优先的搜索策略。

为更有效地搜索解答，可使用评价函数 f(n) = g(n) + wh(n)，w 用作加权。在搜索图的浅层（上部），可让 w 取较大值，以使 g(n)所占比例很小，从而突出启发式函数的作用，加速向纵深方向搜索；一旦搜索到较深的层次，又让 w 取较小值，以使 g(n)所占比例很大，并确保 $wh(n) \leqslant h^*(n)$,从而引导搜索向广度方向发展，寻找到较短的解答路径。

17.2.7.4 一般图搜索的回溯策略和爬山法

在 $g(n) \equiv 0$ 的情况下，若限制只用评价函数 f(n)= h(n)去排序新扩展出来的子节点，即局部优化排序，就可实现较为简单的搜索策略：回溯策略和爬山法。由于简单易行，在不要求最优解答的问题求解任务中，回溯策略得到广泛的应用。爬山法则适用于能逐步求精的问题。

爬山法是实现启发式搜索的最简单方法。人们在登山时，总是设法快速登上顶峰，所以，若不考虑好不好爬，总是会选取最陡处，以求快速登顶。爬山实际上就是求函数极大值问题，不过这里不是用数值解法，而是依赖于启发式知识，试探性地逐步向顶峰逼近（广义地，逐步求精），直到登上顶峰。

在爬山法中，限制只能向山顶爬去，即向目标状态逼近，不准后退，从而简化了搜索算法；即不需设置 OPEN 和 CLOSE 表,因为没有必要保存任何待扩展节点;仅从当前状态节点扩展出子节点(相当于找到上爬的路径），并将 h(n)最小的子节点（对应于到顶峰最近的上爬路径）作为下一次考察和扩展的节点，其余子节点全部丢弃。爬山法对于单一极值问题十分有效而又简便，但对于具有多极值的问题就无能为力了，因为很可能会因错登上“次高峰”而失败--不能到达“最高峰”。

回溯策略可以有效地克服爬山法面临的困难，其保存了每次扩展出的子节点，并按 h(n)值从小到大排列。如此，相当于爬山的过程中记住了途经的岔路口，只要当前路径搜索失败就回溯（退回）到时序上最近的岔路口，向另一路径方向搜索，从而可以确保最后到达最高峰（即目标状态）。

实现回溯策略的有效方式是应用递归过程去支持搜索和回溯。令 PATH、SNL、n、n′ 为局部变量：PATH 为节点列表，指示解答路径；SNL 是当前节点扩展出的子节点列表；MOVE-FIRST(SNL)把 SNL 表首的节点移出，作为下一次要加以扩展的节点；n、n′ --分别指示当前考察和下一次考察的节点。递归过程的算法取名为 BACKTRACK(n),参数 n 为当前被扩展的节点，算法的初次调用式是 BACKTRACK(s)，s 即为初始状态节点。算法的步骤如下：（1） 若 n 是目标状态节点，则算法的本次调用成功结束，返回空表；（2）若 n 是失败状态，则算法的本次调用失败结束，返回“FAIL”；

（3）扩展节点 n，将生成的子节点置于列表 SNL，并按评价函数 f(k) =h(k)的值从小到大排序（k 指示子节点）;(4)若 SNL 为空，则算法的本次调用失败结束，返回“FAIL”;(5)n′ = MOVE-FIRST(SNL)；（6）PATH = BACKTRACK(n′)；（7）若 PATH =“FAIL”，返回到语句（4）；（8）将 n′ 加到 PATH 表首，算法的本次调用成功结束，返回 PATH。

在上述递归回溯算法中，失败状态通常意指三种情况：不合法状态；旧状态重现；或状态节点深度超过预定限度。失败状态实际上定义了搜索过程回溯的条件；另一种回溯条件是搜索进入“死胡同”，由该算法的第(4)句定义。由于回溯是递归算法，解答路径的生成是从算法到达目标状态后逆向进行的，首先产生空表，然后每回到算法的上一次调用就在表首加入节点 n′，直到顶层调用返回不包含初始状态节点 s 的解答路径。

影响回溯算法效率的关键因素是回溯次数。鉴于回溯是搜索到失败状态时的一种弥补行为，只要能准确地选择下一步搜索考察的节点，就能大幅度减少甚至避免回溯。所以，设计好的启发式函数 h(n)是至关重要的。

17.2.7.5 一般图搜索的状态空间抽象和生成-测试法

为支持大型状态空间的有效搜索，状态空间**抽象策略**和**生成-测试法**应运而生。前者更适合寻找令人满意的解答，后者则用于最优化问题。

状态空间抽象（变搜索空间）是减少搜索量的重要技术。常用的方式是将状态空间按子问题进行划分，子问题构成抽象空间。显然，抽象空间中的一个搜索与步(一个子问题)对应于实际状态空间中的许多步，因而抽象空间小得多，使搜索大幅度加快。例如，驾驶汽车长途旅行的问题就适合于用状态空间抽象策略去解决。旅行者可先用只描述主要公路的全国交通图决定大致的行走路线(对应于抽象空间中的搜索)，然后决定每一段大致路线的详细行程，这时他需要沿途城市的交通详图。状态空间抽象的本质是将搜索的注意力集中于问题求解的重要因素，复杂的问题还可采用多级抽象。

状态空间搜索有时能表示为**生成-测试法**。搜索过程由两个部件合作完成：可能解的生成器和修剪不正确解答的测试器。生成器的性能取决于完备性和无冗余性，即能生成所有可能解并只生成一次。在搜索过程中，生成器和测试器的工作往往需交替进行。使用生成-测试法应考虑的关键问题是如何在生成器和测试器之间分配知识。经验证明，知识丰富的生成器会导致较高的搜索效率。实际应用的生成-测试法常是分层的，整个问题求解过程可视为不断搜索部分解答的过程，每一层的生成-测试法服务于搜索一个部分解答。搜索部分解答相当于解答分类，因此，搜索树实际上就是分类树。分层的目的在于设计有力的测试器，以尽量早期地(在搜索树上层)删除不合适的候选解，从而大幅度减少搜索量。

17.2.7.6 关于一般图搜索中启发式搜索的适用性讨论

启发式搜索在人工智能研究的早期是很重要的议题，甚至有人说人工智能就是启发式搜索。但随着知识工程的兴起，启发式搜索已较少用于作为人工智能系统的顶层控制结构，尤其是使用全局评价器的启发式搜索方法，因为其不适合知识密集型的问题求解。但启发式搜索仍不失为重要技术用于解决某些适合于它的子问题，且搜索技术的不少思想方法也可用于知识系统的设计。

纵观启发式搜索技术的应用，其面临三个关键选择：

- 确定性或非确定性搜索方式；
- 搜索目标状态本身还是达到目标状态的解答路径(往往表示为状态或操作算子序列)；
- 搜索一个还是全部或最优解答。

可以说，后两者在一定程度上决定了前者。对于复杂问题的状态空间，搜索路径上有许多分枝点。所谓确定性方式，就是利用启发式信息选取最好的分枝，而舍弃所有其余分枝不再予以考虑。这常用在能保证所选路径可通向成功的场合。在仅靠启发式知识不足以保证所选路径能导致搜索的最终成功或需要搜索所有解答的场合，非确定性方式无疑是明智的选择。尤其是要求获得最优解答的场合，更需要采用非确定方式，因为只有穷尽搜索到全部成功路径，才能判断最佳者。

解答为目标状态或解答路径也影响到确定性或非确定方式的选择。通常要求解答为路径无非是要寻找较好的操作算子序列，确定性搜索方式显然难以奏效；但若只要求到达目标状态，而又有较多路径可达，则确定性方式易于奏效。常用的非确定性搜索方式又可分为两类：最佳优先和深度优先加回溯。前者将评价器作为加在各搜索分枝上的通断开关，使搜索过程总是在总体上最有希望的分枝上进展；后者则是局部最优的搜索方法，系统总是在当前状态直接产生的后继状态中利用启发式知识选取最有希望的状态推进搜索过程。由于容许评价不精确，甚至知识贫乏，搜索有可能失败，这就需要通过回溯去纠正错误，以使搜索能在新的路径上展开。由于只考虑局部最优，易于开发领域特有的启发式知识，深度优先加回溯方法更适用于基于知识的智能系统的设计。

然而无论是最佳优先或深度优先加回溯，都是十分低效率的，尤其是后者更招致了严厉的批评。避免回溯成为智能系统研究的一个重要课题，研究明显地向确定性方式倾斜，概括如下：

(1)有些应用领域只能使用确定性搜索方式。如下棋程序，一旦落子，即使是“臭着”，也不能悔棋；化学反应控制也是如此。

(2)迂回路径不一定是缺点。若搜索的是目标状态而非解答路径，且快速路径难以找到，则可以作深度优先的搜索，但在搜索失败时不是简单地回溯到上一分枝点(也称时序回溯)，而是作适当的修补后继续向前搜索。例如，自然语言理解中常使用“带修补的最佳路径”法。该法只沿单一路径搜索，当发现错误时，不是回溯，而是对已作的分析结果作适当修正(迂回效果)，并继续向前搜索。下面是待分析的英语句子：

Have the students who missed the exam take it today?

句首单词“Have”可作为助动词或主动词。若句法分析时将其作为助动词，则名词短语“the students Who missed the exam”将分析为主语。由于随后出现“take”而非“taken”，搜索失败。但只要把“have”改作主动词，其后的名词短语改作宾语，就可继续句法分析直到成功，不必回溯。一种称为“从属导向”的回溯方法也有类似的搜索修补效果。当发现搜索失败时，其并不按时序简单地回溯到上一搜索分枝点，而是分析失败的原因，直接确定导致失败的搜索步，并对已作的推理过程作适当修改，从而比盲目的时序回溯有效得多。严格地说，这种方式不是回溯技术，因为其不撤消已有搜索路径上做的正确操作，仅改正导致搜索失败的错误操作。

(3)改变状态空间的定义。在无法找到好的启发式评价函数的情况下，往往是搜索空间定义得不好。鉴于人的问题求解行为并不涉及很多搜索，所以通过寻找确定性搜索策略去改变状态空间的定义是明智的。

相关的搜索算法还有手段-目的(means-ends)法。作为启发式搜索的变种，手段-目的法分析操作算子对减少当前状态和目标状态之间差别的贡献，以决定操作算子的选用。无论是手段-目的法的分析程序或上述启发式搜索的评价函数均可设计为全局或局部的评价器。复杂的问题倾向于设计局部的评价器，因为全局的评价器难以寻找。评价器可以是知识贫乏的，从而作用微弱，例如选用最早生成的未考察状态，或选用最新产生的状态优先考察。全局的启发式函数包含了应用领域特有的

知识，但仍然是知识贫乏的，因为无法容纳每个搜索步特有的知识。设计服务于每个搜索步的局部评价器可以使搜索过程依赖于丰富的领域知识，这种局部评价器往往可设计为规则组去识别合适的操作算子，以推进搜索过程到下一状态。实际上，这样的问题求解系统更像一个识别系统而非搜索系统。

17.2.7.7 考虑问题归约的与或图描述

问题归约（变问题空间）是人求解问题常用的策略，先把一个复杂的问题变换为若干需要同时处理的较为简单的子问题后，再分别加以求解；当这些子问题全部解决时，问题也就解决，问题的解答就由子问题的解答联合构成。问题归约可以递归地进行，直到把问题变换为本原问题的集合。所谓本原问题就是不可或不需再通过变换化简的“原子”问题，本原问题的解可以直接得到或通过一个“黑箱”操作得到。

问题归约的描述。问题归约是一种广义的状态空间搜索技术，其状态空间同样可表示为一个二元组：SP = (S,O)，只不过问题状态可以通过子问题状态的联合加以表示，而操作算子的执行则导致问题的变换。变换可区分为以下三种情况：

（1） 状态变迁--导致问题从上一状态变迁到下一状态，这就是一般图搜索技术中操作算子的作用。

（2） 问题分解--分解问题为需同时处理的子问题，但不改变问题状态。

（3） 基于状态变迁的问题分解--先导致状态变迁，再实现问题分解，实际上 就是前二个操作的联合执行。

应用问题归约策略得到的状态空间图，通常是“与或图”，可用一条圆弧将几条节点间的关联弧连接在一起去指示问题分解为子问题，这些子问题的求解具有逻辑“与”关系，只有全部解决才会导致问题的解决。换言之，相应于这些子问题状态的节点具有“与”关系，它们各自的状态描述联合起来即构成问题状态的描述。同时，问题的分解也可以有多种方式，问题（子问题）也可能激活多条规则；显然，只要有一种分解方式或有一条激活的规则能导致最终的解答成功即可，这就意味着逻辑“或”的关系。

由于存在“与”的分支和“或”的分支，对于复杂问题，可以把面向状态变迁的操作和面向问题求解的操作合并为基于状态变迁的问题分解操作，以获得更为紧凑的问题归约描述。在问题归约的过程中，有些子问题相互独立，子问题的进一步归约和本原问题的求解无交互作用，可按任意次序进行。然而，对于许多复杂问题，各子问题仅相对独立，它们之间仍存在一定的交互作用。在这种情况下，正确按排子问题求解的先后次序，甚至子子问题求解的次序也是重要的。

鉴于应用问题归约策略求解问题的过程可以表示为与或图，考虑问题归约策略的求解问题，将转化为一个问题的与或图的抽象表示及在与或图上的搜索策略问题。

我们可以把与或图视为对一般图的扩展，或把一般图视为与或图的特例，即一般图不允许节点间具有“与”的关系，所以又可把一般图称为“或图”。与一般图类似，与或图也有根节点，用于指示初始状态。由于同父子节点间可以存在“与”关系，父、子节点间不能简单地以弧线关联，需要引入超连接概念。同样原因，在典型的与或图中，解答路径往往也不复存在，代之以广义的解路径--解图。

（1）**K-连接**--用于表示从父节点到子节点间的连接，也称为父节点的外向连接，并以园弧指示同父子节点间的“与”关系，K 为这些子节点的个数。一个父节点可以有多个外向的 K-连接。K 大

于 1 的连接也称为超连接，K 等于 1 时超连接蜕化为普通连接，而当所有超连接的 K 都等于 1 时，与或图蜕化为一般图。

（2）**根、叶、终节点**--无父节点的节点称为根节点，用于指示问题的初始状态；无子节点的节点称为叶节点。由于问题归约伴随着问题分解，所以目标状态不再由单一节点表示，而是应由一组节点联合表示。能用于联合表示目标状态的节点称为终节点；终节点必定是叶节点，反之不然；非终节点的叶节点往往指示了解答搜索的失败。

（3）**解图的生成**--在与或图搜索过程中，可以这样建立解图：自根节点开始选一外向连接，并从该连接指向的每个子节点出发，再选一外向连接，如此反复进行，直到所有外向连接都指向终节点为止。如此，可生成一个解图。注意，解图是遵从问题归约策略而搜索到的，解图中不存在节点或节点组之间的“或”关系；换言之，解图纯粹是一种“与”图。另外，正因为与或图中存在“或”关系，所以往往会搜索到多个解图。为确保在与或图中搜索解图的有效性，要求解图是无环的，即任何节点的外向连接均不得指向自己或自己的先辈，否则会使搜索陷入死循环。换言之，会导致解图有环的外向连接不能选用。这里，我们给出关于解图、解图代价、能解节点和不能解节点的释义如下：

解图：与或图（记为 G）任一节点（记为 n）到终节点集合的解图（记为 G_E）是 G 的子图：（1）若 n 是终节点，则 G_E 就由单一节点 n 构成；（2）若 n 有一外向 K-连接指向子节点 n_1，n_2，…n_k，且这些子节点每个都有到终节点集合的解图，则 G 由该 k-连接和所有这些相应于子节点的解图构成；（3）否则不存在 n 到终节点集合的解图。

解图代价：以 C(n)指示节点 n 到终节点集合解图的代价，并令 K-连接的代价就为 K，则有：（1）若 n 是终节点，则 C(n) = 0；（2）若 n 有一外向 K-连接指向子节点 n_1，n_2，…n_k，且这些子节点每个都有到终节点集合的解图，则 $C(n) = K + C(n_1) + C(n_2) + \cdots + C(n_k)$。

能解节点：（1）终节点是能解节点；（2）若节点 n 有一外向 K-连接指向子节点 n_1，n_2，…n_k，且这些子节点都是能解节点，则 n 是能解节点。

不能解节点：（1） 非终节点的叶节点是不能解节点；（2）若节点 n 的每一个外向连接都至少指向一个不能解节点，则 n 是不能解节点。

17.2.7.8 与或图的启发式搜索算法

与一般图（或图）的搜索过程类似，引入应用领域的启发式知识去引导搜索过程，可以显著提高搜索的有效性，加速搜索算法的收敛。考虑到与或图中搜索的是解图，非由相邻节点间路径连接成的解路径，所以估算评价函数 f(n)的第 1 分量 g(n)没有意义，只须估算第 2 分量 h(n)。注意，h(n)也非对于最小路径代价的估计，而是对于最小解图代价的估计。另外，由于与或图中子节点或子节点组间可以存在“或”关系，所在搜索过程中会同时出现多个候选的待扩展局部解图，应估计所有这些局部解图的可能代价，并从中选择一个可能代价最小的用于下一步搜索。由于解图以递归方式生成，解图的代价也以递归方式计算，所以一旦某父节点 n 的由外向 K-连接指向的子节点（n_1，n_2，…n_k）每个都估算了其 $h(n_i)$的值，（i = 1，2，…,k），则从父节点 n 到终节点集合解图的可能代价 f(n)可以用公式：　$f(n) = K + h(n_1) + h(n_2) + \cdots + h(n_k)$

计算，并用于取代原先在扩展出节点 n 时直接基于 h(n)估算而得出的 f(n)值。显然，基于子节点 $h(n_i)$算出的 f(n)更为准确。如此递归计算，可以计算出更为准确的 $f(n_0)$，即从初始状态节点到终节点集合的解图的可能代价。

下面就给出实现与或图启发式搜索的算法 A_0^*。

设 G 为搜索图，G′ 为被选中的待扩展局部解图，LGS 为候选的待扩展局部解图集，n_0 指示根节点，即初始状态节点，n 为被选中的待扩展节点，$f_i(n_0)$ 为第 i 个候选的待扩展局部解图的可能代价。

AO*算法的实现过程如下：

(1) G := n_0，LGS 为空集；

(2) 若 n_0 是终节点，则标记 n_0 为能解节点；否则计算 $f(n_0) = h(n_0)$，并把 G 作为 0 号候选局部解图加进 LGS；

(3) 若 n_0 标记为能解节点，则算法成功返回；

(4) 若 LGS 为空集，则搜索失败返回；否则从 LGS 选择 $f_i(n_0)$ 最小的待扩展局部解图作为 G′；

(5) G′ 中选择一个非终节点的外端节点（尚未用于扩展出子节点的节点）作为 n；

(6) 扩展 n，生成其子节点集，并从中删去导致有环的子节点以及和它们有“与”关系的子节点；若子节点集为空，则 n 是不能解节点，从 LGS 删去 G′ (因为 G′ 不可能再扩展为解图)；否则，计算每个子节点 n_i 的 $f(n_i)$,并通过建立外向 K-连接将所有子节点加到 G 中；

(7) 若存在 j 个（j>1）外向 K-连接，则从 LGS 删去 G′，并将 j 个新局部解图加进 LGS；

(8) G′ 中或在取代 G′ 的 j 个新局部解图中用公式 $f(n) = K + h(n_1) + h(n_2) + \cdots + h(n_k)$ 的计算结果取代原先的 f(n)，并传递这种精化的作用到 $f_i(n_0)$（i = 1, 2, …, j)；同时将作为终节点的子节点标记为能解节点，并传递节点的能解性。

(9) 返回语句（3)。

17.2.8 基于结构化知识表示的问题求解系统及其推理机制

真实世界是复杂的，不仅是因为世界充满了万事万物，更是由于万事万物间存在着千丝万缕的联系。这就需要从多方面、多角度描述各种事物所具备的属性和事物间存在的各种关系。尽管产生式规则和一阶谓词逻辑可用以有效地表示事物和事物间的关系，但它们将对于各种事物的描述混杂在一起，未能提供结构化的手段去充分或集中地描述特定的事物和事物间的关系。为克服产生式表示法和一阶谓词逻辑表示法的不足，便于结构化地表示世界和事物，语义网络、框架和面向对象等知识表示方式应运而生，并曾获得过迅速地发展，我们统称之为结构化知识表示方法。

17.2.8.1 语义网络知识表示方法及基于语义网络的问题求解系统

语义网络最早由奎廉（J.R.Quillian）于 1968 年提出。作为描述人类联想记忆的一种显式心理学模型，当时语义网络主要应用于自然语言理解系统中。由于其强大和直观自然的表示能力，不久就被广泛应用于人工智能研究和应用开发的许多领域，成为知识表示中表达能力较强且较灵活的重要的方法之一。

1．语义网络表示知识的方法

语义网络的基本思想是用“节点”和带标记的边构成的有向图来描述事物、事件、概念、状况、动作及事物之间的关系。其中，“节点”表示各种事物、概念、事件、情况、属性、动作、状态等；带标记的边（节点间的“连接弧”）通过指明它所连接的节点间的某种语义关系来描述事物之间的各种关系。它在形式上是一个带标识的有向图，节点和弧都必须带有标识，以区分各种不同对象以及对象间各种不同的语义联系。

最简单的语义网络是一个三元组：**（节点 1，弧，节点 2）。**例如，**“海浪把战舰轻轻地摇”** 显然可表示成如下的语义网络：

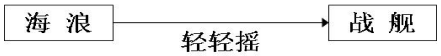

图 17.2.2 “海浪把战舰轻轻地摇”的语义网络

亦即，**(海浪，轻轻摇，战舰)。**

从逻辑表示法来看，一个语义网络完全可以转而表示为一阶逻辑表示形式，只需用一元谓词指示个体的类属并用二元渭词描述个体间的关系。因为三元组（结点 1，弧，结点 2）可写成 P（个体 1，个体 2)，其中个体 1、个体 2 对应于结点 1、结点 2，而弧及其上标注的结点 1 与结点 2 的关系由谓词 P 来体现。例如，**轻轻摇（海浪，战舰)。**

显然，关系弧只能表示二元关系；但自然语言理解或其它问题求解活动中常会遇到多元关系，例如，“John 给 Mary 一个礼物”就是一个 3 元关系：Gives(John, Mary, Gift)。解决的办法是先将多元关系转变为多个 2 元关系的合取，再建立相应的语义网络表示。对于这个例子，可以先将整个描述表示为一个给出事件 G1，使其作为事件类 Giving-Event 的一个例子，再说明 G1 中的 John 是给出者（Giver），Mary 是接受者（Receiver），而 Gift 则是给出的东西（Thing）。从而整个描述可以谓词公式的合取方式表示为：

Isa(G1,Givig-Event) ∧ Giver(G1,John) ∧ Receiver(G1,Mary) ∧ Thing(G1,Gift)

可以看出，只要将谓词作为关系弧标签，很容易把按这种方式表示的谓词公式合取式转变为语义网络表示。

对于**（海浪，轻轻摇，战舰）也可如此更深度的**表达。若把理解这一语句时所使用的语法知识加进去，即可将原来的一个谓词拆成如下三个谓词：[**动作主体（海浪，摇） 动作对象（战舰，摇） 动作方式（轻轻，摇）]，则**改进后的较为详细的语义网络可表示为：

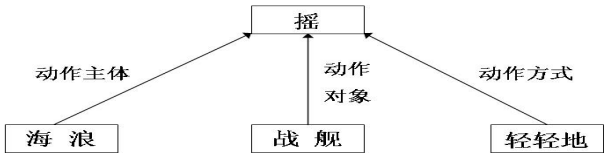

图 17.2.3 “海浪把战舰轻轻地摇”改进后的语义网络

在此改进后的语义网络中，海浪、战舰、摇、轻轻等概念之间的相互关系均已给出，且这种关系是命题本身所包括的。可以说，到此为止，我们也已经穷尽了命题本身包含的知识。

但是，海浪、战舰、摇、轻轻等概念本身究竟有什么含义，在这个语义网络中并未体现出来，因为命题中也没有这样的知识。为了对其作进一步地描述，我们可以加入该命题以外的知识，包括我们对客观世界上各类事物的范畴及其属性的认识，甚至《军港之夜》这首歌曲的上下文信息等。如此，在添加上我们对海港中各种事物的范畴及其属性的认知之后，我们也可得如下更为详细的语义网络：

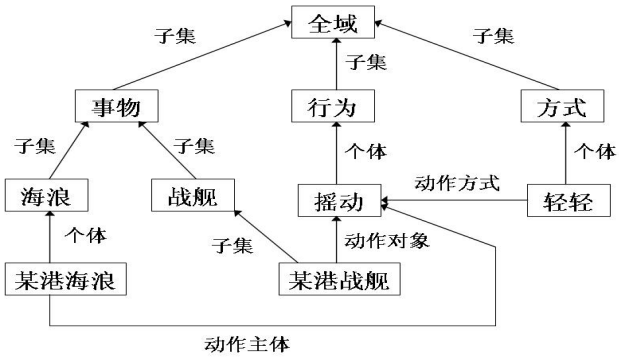

图 17.2.4 “海浪把战舰轻轻地摇”改进后的语义网络

如此，我们用语义网络可以表示关于事物和事物之间关系的一般性描述，而要达到此种描述，我们就需要引入相关的变量和量词。通常地做法是，将语义网络节点中出现的变量视为隐含地受全称量词的约束，而所有存在量词约束的变量都用 SkÖlem 函数或常量取代，这样，就可免除量词在语义网络中的显式表示。例如“John 给每个人一个礼物”，可以先表示为如下合适公式：

（∀x）（∃y）(∃z) [Isa(z,Giving-Event)∧Giver(z,John)∧Receiver(z,x)∧Thing(z,y)]

然后消去量词得：

Isa(f(x),Giving-Event)∧Giver(f(x),John)∧Receiver(f(x),x)∧Thing(f(x),g(x))。

另一种表示一般性描述的方法是显示全称量词及其辖域。该辖域在语义网络中形成一个子网，用于提供一般性描述的内容称为构架（Form）。令节点 GS 指示所有一般性描述的集合，则每个一般性描述就是 GS 的一个实例，以包含一常量（例如 G）的节点指示。从实例节点（例如 G）引出二种关系弧：带标签 Form 的弧指向相应于全称量词辖域的子网，带标签的弧指向全称量词的约束变量。若同一辖域受多个全称量词约束，则需要引出多个带标签的弧指向这些全称量词的约束变量。辖域内其它变量都隐含受存在量词的约束（不必 SkÖlem 化）。

因此，语义网络可视作一种知识的单位。人脑的记忆就是由存储了大量的语义网络来体现的。语义网络可用 BNF 描述为：

<语义网络> ::= <基本网元>|Merge(<基本网元>,...)

<基本网元> ::= <节点><语义联系><节点>

<节点> ::= （<属性-值对>,...）

<属性-值对> ::= <属性名>:<属性值>

<语义联系> ::= <系统预定义的语义联系> | <用户自定义的语义联系>

具体而言，通过语义网络来表示知识的方法如下：

（1）**事物间的[语义]关系描述**

语义网络描述事物间的关系采用的是带有标识的关系弧。常用的语义联系[关系弧]有：

（a）**ISA 链**：用来表示具体-抽象关系，或说表示一种隶属关系，可体现某种层次分类。特点是具体层结点可继承抽象层结点的属性。

（b）**A-Member-of 链**：表示个体与集体（类或集合）之间的关系，它们之间有属性继承性和属性更改权。例如，“张三是工会会员”。

（c）**a-part-of 链**：用来表示部分-全体关系，或说表示包含关系。特点是 part-of 关系下各层结点的属性可能是很不相同的。

（d）**is 链**：用于表示一个结点是另一个结点的属性。

（e）Composed-of 链：表示“构成”联系，是一种一对多的联系，被它联系的节点间不具有属性继承性。例如，“整数由正整数、0 及负整数组成”。

（f）Have 链：表示属性或事物的“占有”关系。例如，“鸟有翅膀”。

（g）Before, After, At 链：表示事件之间的时间先后关系。其中，Before 表示一个事件在另一个事件之前发生；After 表示一个事件在另一个事件之后发生；At 表示一事件发生的时间。

（h）Located-on(-at, -under, -inside, -outside) 链：表示事物之间的位置关系。例如，“书放在桌子上”。

（i）Similar-to, Near-to 链：表示事物之间的相似和接近关系。例如，“猫与虎相似”。

（2）事物间逻辑关系的表示

事物之间不仅存在可以直接用关系弧表示的语义关系，也存在与、或、非、蕴涵等逻辑关系。语义网络可以通过附加一些特殊的标记来指示逻辑关系。其中，语义网络中由关系弧指示的二元关系之间具有隐含的逻辑“与”关系，所以不必作附加处理。这种“与”关系的隐含可以从多元谓词公式转变为多个二元谓词公式的过程中观察到。当 2 条或多条关系弧有逻辑“或”关系时，考虑到语义网络中所有二元关系均用关系弧指示，一种常用的方法就是用虚线框将具有某种逻辑关系的关系弧围起来，并在虚线框上加标记 DIS（disjunction）。逻辑“非”可以用加标签 NEG(negtive)的虚线框围起 2 条或多条关系弧，以指示对这些弧联合语义的取反。若有嵌套在“或”关系内的“与”关系，这就需要将描述每一事件的多条关系弧用加标签 CONJ(conjunction)的虚线框围起来。

两个事件之间的蕴涵关系可以这样表示：以加标签 ANTE(antecedent)的虚线框围住描述蕴涵前项的关系弧；以加标签 CONSE(consequent)的虚线框围住描述蕴涵后项的关系弧；然后再用一条虚线将这两个虚线框连接起来，以表示它们属于同一个蕴涵关系。这样，事实与规则的表示是相同的，区别仅是弧上的标注有别。

（3）语义网络的存储表示

以语义网络方式表示的知识必须存储于知识库，才能加以使用。存储表示方式可以分为二类：节点集，节点集加关系弧集。前一类方式较为简单，且适用面广，是存储表示的基础。

语义网络中的节点可表示为具有若干槽（slot）的数据结构，以 BNF 定义如下：

<节点> :=（Node <节点名> {<槽名>：<槽内容>}+)

其中，槽就取名为关系弧的标签，槽内容即语义网络中关系弧指向的节点。槽内容是节点时，实际上槽内容并非节点名，而是以计算机内存中的指针指向相应的节点，不过语义网络的设计者不必关心这一点，有关处理由语义网络的表示机制自动解决。

鉴于关系弧已转而表示为节点数据结构中的槽，所以语义网络简单地就表示为这些数据结构的集合，此外只要把逻辑关系均视为事件间的逻辑关系，每个事件类似于上述那样加以表示，再把每个具体的逻辑关系视为逻辑关系的实例，则不难建立逻辑关系的存储表示。

逻辑表示法和产生式表示法常用于表示有关论域中各个不同状态间的关系，然而用于表示一个事物同其各个部分间的分类知识就不方便了。用槽（slot）与填槽表示方法就很便于表示这种分类知识。因此，语义网络和框架表示方法就在这方面比逻辑表示法和产生式表示法具有优势。

语义网络知识库的修改通常是通过插入和删除客体及其相关的关系来实现的。

采用网络表示法比较合适的领域大多数是根据非常复杂的分类进行推理的领域以及需要表示事

件状况、性质以及动作之间的关系的领域。相应地，也就有了各种不同类型的语义网络。如我们上述例子所表达的命题型语义网络；语言语义网络--用于自然语言的分析和理解的语义网络；结构网络--用于描述客观事物的结构的语义网络，常见于模式识别、机器学习等应用领域；分类网络--用于于描述事物，并对它们按层次分类；推理网络--它本质上是一种命题网络，只是在某种程度上规范化，更适于进行推理。推理网络的基本节点是事实或概念，而节点间的关系则表示推理规则；框架网络--是语义网络和框架的联合使用，包含两方面的含义：第一，网络中的节点是框架，相当于基本事实或假设，利用节点之间的关系可由某些框架推论出另一些框架；第二，网络中的节点既可代表框架，也可代表框架中的槽，每条弧的一头联着某个框架的一个槽，另一头联着另一个框架，表示后面的框架是前面的槽所代表的子框架。以此方式就可实现框架的任意深度的嵌套调用。

2．语义网络系统求解问题的基本过程

语义网络系统是指用语义网络表示知识的问题求解系统。语义网络系统的两大组成部分：由语义网络构成的知识库；以及语义网络推理机，即用于求解问题的解释程序。在语义网络系统中，问题的求解一般是通过匹配实现的。语义网络系统求解问题的主要过程是：（1）根据待求解问题的要求构造一个网络片断，其中有些节点或弧的标识是空的，反映待求解的问题；（2）依此网络片断到知识库中去寻找可匹配的网络，以找出所需要的信息；当然，这种匹配一般不是完全的，具有不确定性，因此需要解决不确定性匹配问题；（3）当问题的语义网络片断与知识库中的某语义网络片断匹配时，则与询问处匹配的事实就是问题的解。

语义网络表示下的推理方法不像逻辑表示法和产生式表示法的推理方法那样明了。语义网络表示法是依匹配和继承来进行推理的。最简单的 isa 关系下的推理是直接继承，当然，我们也可以将语义网络引入逻辑含义，表示出“∧，∨，～”关系，便可以使用归结推理法。但应用语义网络的主要推理方式还是以关系弧作为索引，快速地在网络中搜索到所需的信息。支持搜索的最重要的技术基础是层次分类和特性继承。

（1）层次分类。世界万物都遵从层次分类法则。例如鸟是一类动物，鸟类又有许多子类，例如会学舌的鹦鹉和不会飞的鸵鸟等。用关系弧 Ako 很容易建立起面向层次分类的语义网络。再通过关系弧 Isa 就可把个体事物关联到分布于层次分类网络中的各个概念节点（视类名为概念）。

设置层次分类网络的显著优点是可以分别存储个体事物的共性，进而大幅度提高信息的存储效率。广义上，节点间关系弧指示事物的属性（特性）；可以把个体事物的公共属性值和典型属性值存放于个体事物所属的类或超类节点中，而个体事物节点本身只存放其特有的信息。例如在关于鸟的层次分类网络中，鸟类的公共特性“有较高体温”和典型特性“能飞翔”可存放于“鸟类”节点中，鹦鹉的典型特性“会学舌”存放于鹦鹉节点中，鸵鸟的典型特性“不会飞”存放于鸵鸟节点中，鹦鹉和鸵鸟各个体的特性（如体重，年龄等）则存放于各个体节点中。显然，以这种方式表示知识，可以大幅度压缩信息的重复存储，紧凑一致，易于维护。

（2）特性继承。与建立层次分类网络紧密相关的是特性继承技术，可以说正是由于有特性继承技术的支持，才使层次分类网络得以实用化。特性继承原理可非形式地阐述如下：当取用不记载于个体事物节点的公共或典型特性值时，可以沿 Isa 和 Ako 关系弧（也称为链）追溯到存放这些特性值的类或超类节点。例如，当查询鹦鹉[E_1]能否学舌时，由于鹦鹉[E_1]未记载相应的属性值，就沿关系弧 Isa 追溯到鹦鹉节点，查到“会学舌”的属性值作为鹦鹉[E_1]的缺省（默认）特性。类似地，若问鹦鹉[E_1]和鸵鸟[T_1]是否会飞，则可分别追溯到鸟类节点和鸵鸟节点，进而正确回答鹦鹉[E_1]会

飞，但鸵鸟[T_1]不会飞。特性继承原理不仅支持对于缺省值（即公共或典型值）的继承，也支持对于操作功能的继承。实现缺省推理。

（3）**逻辑推理。**基于逻辑关系的表示，语义网络也可以支持逻辑推理，但不总如直接用谓词逻辑和产生式表示法来得直观和有效。语义网络是通过限制表示能力来换取了其有效的推理方式--图搜索，但其既不便于表示启发式推理规则（如 Prolog 语言或产生式系统那样），也难以开展演绎推理。实际上，较好的处理方式是综合应用不同的表示方式：以语义网络支持结构化信息的存取，而谓词逻辑或产生式表示法则用于支持逻辑推理。

语义网络的节点通过 Isa 链组织成了层次分类体系。其类则定义了个体应具有的属性。这些均为语义网络特有的隐含知识。正因为语义网络有以上特点，它才允许用图搜索技术作为特别种类的推理形式。由于连接弧的索引功能，寻求个体属性的推理可以有效地进行，而特性继承功能则可用以解答标准一阶逻辑系统难以回答的问题。由此，我们可以认为：（1）语义网络等价于数据库加上特性继承机制。（2）语义网络不能表示特性继承机制无法处理的不完全知识。（3）语义网络提供了集结信息的结构化描述手段。

3．分块（复合）语义网络

与其它知识表示方法相比，语义网络表示法具有以下明显的优势：（1）**结构性**：因为语义网络是一种结构化的知识表示方法，它能把事物的属性以及事物间的各种语义显式地表示出来。（2）**联想性**：它最初是作为人类联想记忆模型提出来的。着重强调事物间的语义联系,体现了人类思维的联想过程。（3）**自然性**：直观地把事物的属性及其语义联系以明确、简洁的方式表示出来,是一种直观的知识表示方法，符合人们表达事物间关系的习惯，把自然语言转换成语义网络较为容易，因而语义网络表示法在自然语言理解系统中的应用最为广泛。（4）**广泛性**：具有广泛的表示范围和强大的表示能力,用其它形式的表示方法能表达的知识几乎都可以用语义网络来表示。

语义网络表示法的不足是：（1）非严格性：与一阶谓词逻辑相比，语义网络没有公认的形式表示体系。一个给定的语义网络所表达的含义完全依赖于处理程序如何对它进行解释。通过推理网络而实现的推理不能保证其正确性。此外，目前采用的表示量词（包括全称量词和存在量词）的语义网络表示法在逻辑上是不充分的，不能保证不存在二义性。**（2）处理上的复杂性**：语义网络表示知识的手段多种多样，虽然灵活性很高，但同时也由于表示形式的不一致使得对其处理的复杂性提高，对知识的检索也就相对复杂，要求对网络的搜索要有强有力的组织原则。（3）**不便于推理**：其推理规则不十分明了，不能充分保证网络操作所得推论的严格性和有效性；一旦节点个数太多，网络结构复杂，推理就难以进行；它也不便于表达判断性知识与深层知识。

为了能处理一般的谓词公式，即允许命题中包含变量和量词，Hendrix 在 1975 年曾提出了“**网络分块化技术**”。

网络分块化技术的基本原理是：当用语义网络表示一个复杂命题时，可将其拆成许多子命题，每个子命题用一个小的语义网络表示，称为一个“**空间**”。复杂命题构成大空间，子命题构成子空间，它本身又可看作是大空间中的一个节点。子空间可以层层嵌套，也可以用弧相互连接。实践证明,这种复合网络可使语义网络的表达能力进一步增强。例如,对于命题“每个学生都读过一本书”，其谓词公式为：(∀s) (∃b) [读过（学生(s)，书(b)）]，其分块语义网络可表达为：

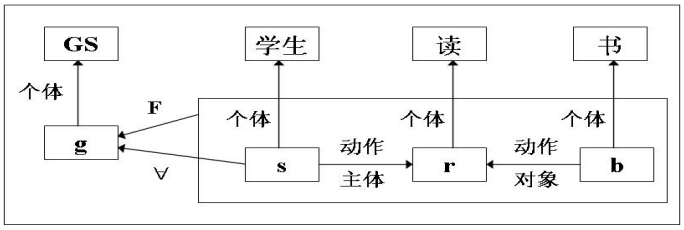

图 17.2.5　“每个学生都读过一本书”的**分块语义网络**

这里，命题“学生读书”构成一个空间，节点 g 是其代表，节点 GS 是全体命题的集合；通向 g 的两条弧，**F 弧**指示它代表的命题是什么，**∀弧**指示 s 是一个全称变量，若有多个全称变量，就要有多个∀弧。上图表示，只有 s 是全称变量，r,b 都是存在变量，它们都是全称变量 s 的函数。

从上图也可知，语义网络表示法和谓词表示法不同。在谓词公式中，“读”作为一个谓词出现，而在这里，它是作为一个事件（动作 r）和“学生”、“书”等统一处理。

分块语义网络表示法要求，子空间中的所有非全称变量的节点都是全称变量节点的函数，那些不是全称变量节点函数的其它节点，应该拉到空间之外去。例如，命题“每个学生都读过《红楼梦》”，就应该表示成如下形式的语义网络：

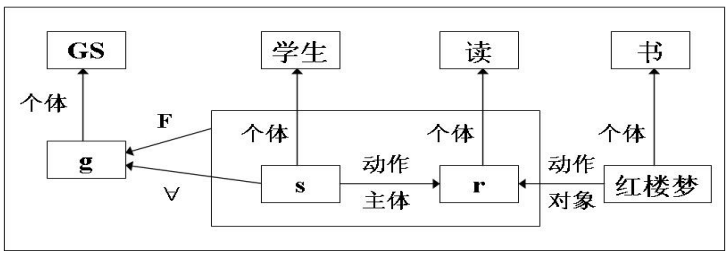

图 17.2.6　“每个学生都读过《红楼梦》”的**分块语义网络**

命题“每个学生都读过所有的书”，其谓词公式为：**(∀s) (∀b)[读过 ((学生(s)，书(b))]**，其分块语义网络则为：

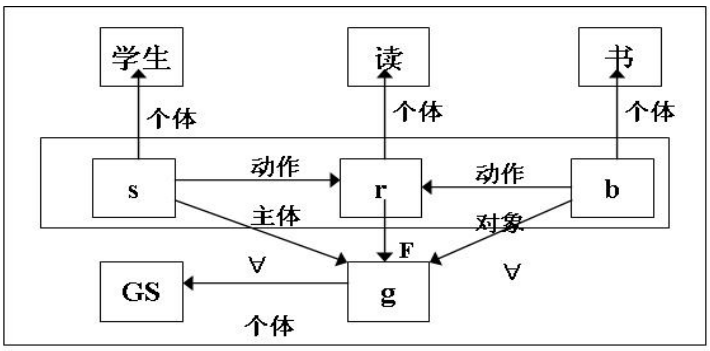

图 17.2.7　“每个学生都读过所有的书”的**分块语义网络**

17.2.8.2 框架表示法与基于框架表示的问题求解系统

1．框架表示法

框架表示法是一种关于事物内部结构化描述的表示法，最早由明斯基于 1975 年提出，作为理解视觉、自然语言和其它复杂行为的一种基础，现已发展为广泛应用的知识表示方法。框架理论的基本思想是：认为人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，人脑存储有大量典型情景，当人面临新的情景时，就从记忆中选择一个合适的框架，这个框架是以

前记忆的一个知识架构，而其具体内容可依新的情景而改变，即对这一架构的细节加工修改和补充，从而形成对当前事物的认识。框架表示法可以较好地反映人观察事物的思维方式，即人对自己熟悉的事物早在头脑中形成了抽象模型。

框架表示法可用以方便地表示上述人脑中关于事物的抽象模型。其最突出的特点是善于表示结构性知识，能够把知识的内部结构关系以及知识之间的特殊关系表示出来，并把与某个实体或实体集的相关特性都集中在一起。框架理论将框架视作知识单位，将一组有关的框架连接起来便形成框架系统。系统中不同框架可以有共同结点，系统的行为由系统内框架的变化来表现。框架表示法是一种适应性强、概括性高、结构化良好、推理方式灵活、可把陈述性知识与过程性知识相结合的知识表示方法。

框架是一种描述对象（事物、事件或概念等）属性的数据结构，在框架理论中，框架是知识表示的基本单位。 一个框架由若干个“槽”（Slot）结构组成，每个槽又可分为若干个“侧面”。 槽用于描述所论对象某一方面的属性；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称为槽值和侧面值。

框架的一般可表示为：

<框架名>

槽名 1：侧面名 1　　值 1，值 2，...，值 p_1

侧面名 2　值 1，值 2，...，值 p_2

……　　……

侧面名 m_1　　值 1，值 2，...，值 p_{m1}

…　　……　　……

槽名 n：侧面名 1　值 1，值 2，...，值 r_1

…　　……　　……

约束：　约束条件 1

……

约束条件 n

若用 BNF 描述，则为：

<框架> ::= <框架头><槽部分>[<约束部分>]

<框架头> ::= 框架名<框架名的值>

<槽部分> ::= <槽>,[<槽>]

<约束部分> ::= 约束<约束条件>,[<约束条件>]

<框架名的值> ::= <符号名>|<符号名>(<参数>,[<参数>])

<槽> ::= <槽名><槽值>|<侧面部分>

<槽名> ::= <系统预定义槽名>|<用户自定义槽名>

<槽值> ::= <静态描述>|<过程>|<谓词>|<框架名的值>|<空>

<侧面部分> ::= <侧面>,[<侧面>]

<侧面> ::= <侧面名><侧面值>

<侧面名> ::= <系统预定义侧面名>|<用户自定义侧面名>

<侧面值> ::= <静态描述>|<过程>|<谓词>|<框架名的值>|<空>

<静态描述> ::= <数值>|<字符串>|<布尔值>|<其它值>

<过程> ::= <动作>|<动作>,[<动作>]

<参数> ::= <符号名>

在框架的上述 BNF 描述中，（1）框架名的值允许带参数，当别的框架调用它时需要提供相应的实在参数。（2）当槽值或侧面值是一个过程时，它既可以是一个<动作>串，也可以是对某个过程的调用。（3）当槽值或侧面值是谓词时，其真值由当时谓词中变元的取值确定。（4）槽值或侧面值为<空>时，表示当时未能确定该值，待以后填入。（5）<约束条件>是任选的，当不指出时，表示无约束。

上述描述也表明，一个框架的典型特征是：（1）每个框架有一个框架名（可带参数）；（2）每个框架有一组属性，每个属性称为一个槽，里面可以存放属性值；（3）每个属性对它的值有一定的类型要求，不同属性的类型要求可以不一样；（4）有些属性值可以是子框架调用，子框架调用可以带参数；（5）有些属性值是事先确定的，而有些属性值需在生成实例时代入；（6）有些属性值在代入时需满足一定的条件。有时，在不同属性的属性值之间有一些条件需要满足。

例： framework: <大学教师>

类属：<教师>

学历：（学士，硕士，博士）

专业：<学科专业>

职称：（助教，讲师，副教授，教授）

外语： 范围：（英，法，德，...） 默认：英　　水平：（优、良、中、差）默认：良

框架是一种集事物各方面属性的描述为一体，并反映相关事物间各种关系的数据结构。在此结构中，“槽”起着至关重要的作用：描述事物各有关方面的属性并指出相关事物间的复杂关系。因此，在用框架作为知识的表示模式时，对槽的设置与组织应给以足够的重视。在设置与组织框架中“槽”的时候，应注意：（1）充分表达事物各有关方面的属性；（2）充分表达相关事物间的各种关系；（3）对槽及侧面进行合理的组织；（4）有利于进行框架推理。

作为描述事物状态的一种数据结构，我们可以把框架看成是一个由节点和关系组成的网络，是语义网络一般化、形式化的一种结构。将语义网络中结点间弧上的标注放入槽内，就成了框架表示法。显然，框架的表示结构与语义网络的表示结构是很接近的，只是由于前者引入了侧面描述，使框架比节点具有更丰富的表示结构。另外，这两种表示法在使用上还是存在语义差别的，即框架表示法更强调表示事物的内部结构（利用框架的丰富表示结构），而语义网络更强调表示事物间的关系。由于槽值可以是逻辑的、数字的，也可以是程序、条件、默认值或是一个子框架。因而，一个框架中可以包含各种信息：描述事物的信息，如何使用框架的信息，关于下一步将发生什么情况的期望及如果期望的事件没有发生应该怎么办的信息等等，这些信息包含在框架的各个槽或侧面中。

框架内部结构的丰富程度取决于事物描述本身的需要。一般来讲，表示概念（例如类概念）的框架结构复杂，而表示个体事物的框架就很简单。

事物的抽象模型可以通过表示概念的框架来加以定义。由于每个事物有多个属性（特性），而每个属性又需从多个侧面加以描述，所以表示概念的框架往往有复杂的表示结构。

框架表示法具有以下优点：① 框架系统的数据结构和问题求解过程与人类的思维和问题求解过程相似；② 框架结构表达能力强，层次结构丰富，提供了有效的组织知识的手段，只要对其中某些细节作进一步描述，就可以将其扩充为另外一些框架；③ 可以利用过去获得的知识对未来的情况进

行预测，而实际上这种预测非常接近人的认识规律，因此可以通过框架来认识某一类事物，也可以通过一系列实例来修正框架对某些事物的不完整描述(填充空的框架,修改默认值)。

框架表示法与语义网络表示法一样，也存在着相似的不足：① 缺乏形式理论,没有明确的推理机制保证问题求解的可行性和推理过程的严密性；② 由于许多实际情况与原型存在较大的差异，因此适应能力不强；③ 框架系统中各个子框架的数据结构如果不一致会影响整个系统的清晰性，造成推理的困难。④ 不善于表达过程性的知识。故框架表示法常与产生式表示法结合起来使用，以取得互补的效果。

2．框架系统求解问题的基本思路

应用领域问题的求解往往涉及到相互关联的许多框架，这些框架联合起来就构成一个基于框架的问题求解系统。与语义网络类似，一个具体事物可由槽中已填入值的框架来描述,具有不同的槽值的框架可以反映某一类事物中的各个具体事物。框架的某些槽的侧面值可以是其它框架，从而能建立起节点是框架的网络。相关的框架链接在一起就形成了一个框架系统。**框架网络**是指具有横向联系及纵向联系的一组框架。横向联系是指一个框架中的槽值或侧面值可以是另一个框架的名字，即通过一个框架可以找到另一个框架，这样就在两个框架之间建立起横向联系。纵向联系是指，由于属性的继承性，故可以把共同属性抽取出来，构成一个上层框架，然后再对专有属性分别构成下层框架，并在下层设立一个专用的槽（一般称为“继承”槽）。这样不仅在框架间建立了纵向联系，而且建立了上下层框架的属性及值的继承关系。较常用的框架系统有二类：事物组成和分类体系。事物组成类框架系统主要用于描述复杂事物的层次组成。事物的组成是广泛存在的概念，例如，人体就由头部、躯干和四肢组成，也可视为由皮、肉、骨、血等构成。每个组成部分或组件均可用一个框架加以描述；而整个事物即构成一个框架系统。层次分类体系型框架系统和语义网络类似，用之也能描述事物的层次分类体系。而且，由于框架可以具有丰富的内部结构，能更有效地表示分类体系和支持结构化信息存取。

一个基于框架的问题求解系统一般由两大部分构成:(1)由框架及其相互关联构成的知识库;(2)由一组解释程序构成的框架推理机。利用框架系统求解问题的基本过程与人类求解问题的思维过程有许多相似之处：当人们对某事物不完全了解时，往往先根据当前已掌握的情况着手工作，然后在工作过程中不断发现、掌握新情况、新线索，使工作向纵深发展，直到达到最终目标。框架系统在求解问题时，系统首先根据当前已知条件对知识库中的框架进行部分匹配，找出预选框架，并且由这些框架中其它槽的内容以及框架间的联系得到启发，提出进一步的要求，使问题的求解向前推进一步。重复上述过程，直至问题得到解决为止。框架系统中由一个框架到另一个框架的转换，可以表示状态的变化、推理或其它活动。

3．基于框架系统的推理过程

框架表示法没有固定的推理机理。但框架系统的推理和语义网络一样遵循匹配和继承的原则，而且框架中如 if-needed、if-added 等槽的槽值是附加过程，在推理过程中也起重要作用。

框架推理是一个反复进行框架匹配的过程，且大部分匹配都具有不确定性，为了推理得以进行，通常都需要设置相应的槽来配合。

设计框架系统的主要目的是支持结构化信息的存取，并由此支持问题求解系统应用从框架系统取到的信息（知识）去进行推理。支持结构化信息存取的技术主要是特性继承和相容匹配。

（1）**特性继承**。由于框架的槽包含多个侧面，框架系统可以提供功能强大的特性继承技术。在典型情况下特性继承可由描述事物类（概念）的框架中设置的三个侧面：Value、If-Needed 和 If-Added 所提供的缺省推理功能来组合实现。这三个侧面的作用如下：

• Value--记载类的个体相应属性的公共值或典型值，作为缺省值；

• If-Needed--在不可能提供统一缺省值的情况下，提供计算函数或推理知识去产生相应属性的一个值，简称执行了 If-Needed 操作；

• If-Added--当给类的某个体的一个属性赋值或修改时，提供计算函数或推理知识去作必要的后继处理，包括对其它相关槽的赋值和修改处理，以及任何需要的附加处理；简称执行了 If-Added 操作。

特性继承的实现过程由对个体框架槽的操作来激活。若查询一个体的某属性，且描述该个体的框架未提供属性值（槽值）时，就沿 Isa 和 Ako 链（Isa 和 Ako 槽）追溯到具有同名槽（属性）的类或超类框架。在该槽提供 Value 侧面值的情况下，就继承该值（缺省值）作为查询结果返回；否则，该槽应提供 If-Needed 侧面供继承，可执行 If-Needed 操作去产生一个值作为查询结果。若对一个体的某属性进行了赋值或修改工作，则系统自动沿 Isa 和 Ako 链追溯相应的类和超类，只要发现类或超类框架中的同名槽具有 If-Added 侧面，就可执行 If-Added 操作，作拟定的后继处理工作。

If-Needed 操作和 If-Added 操作的区别在于它们激活的时机和操作目的不同。前者在系统试图查询个体框架中未记载的属性值时激活，并应查询需要，被动地即时产生所需属性值；后者则在系统对个体框架的属性作赋值和修改工作后激活，目的在于通过后继处理主动做好配套操作和消除可能的不良影响（如不一致问题）。

（2）**相容匹配。**匹配是实现推理技术的重要环节，无论是规则演绎推理、或是产生式系统的推理，都涉及到对应表达式的匹配检查。框架系统可以给采用这些推理技术的问题求解系统提供需参照的结构化事实；特别是，可以就把框架系统作为结构化的综合数据库。

由于框架系统支持层次分类体系和特性继承，两个框架的匹配检查往往非严格意义上的相等比较，而是所谓的相容匹配。相容匹配的方法如下：

• 检查这二个框架是否存在祖先-子孙关系，即若从一个框架出发，经由 Isa 和 Ako 链可以追溯到另一框架，则认为两者是相容匹配的。

• 分别将这二个框架各自直接记载的属性值和可继承到的缺省值都取到，把属性区分为关键和非关键二类，只有关键属性都匹配的二个框架才是相容的。

• 若一个框架能搜索到多个相容的框架，则可进一步依据非关键属性的匹配程度（个数）挑选出最相容的匹配框架。

17.2.8.3 其它的一些结构化知识表示方法简介【曾经的研究，可供参考】

1．剧本（脚本）表示法

剧本表示法是 1975 年 R.C. Schank 依据他的**概念依赖理论**而提出的一种知识表示方法。脚本与框架类似，由一组槽组成，用来表示特定领域内一些事件的发生序列。

概念依赖理论认为，在人类的各种知识中，常识性知识是数量最多、涉及面最宽、关系最复杂的知识，很难把它们形式化地表示出来交给计算机处理。概念依赖理论的基本思想是：把人类生活中各类故事情节的基本概念抽取出来，构成一组原子概念，确定这些原子概念间的相互依赖关系，然后把所有故事情节都用这组原子概念及其依赖关系表示出来。

抽取原子概念应遵循的基本原则是：（1）所有原子概念表示的意思必须是无二义性的。即使用来表示此原子概念的词原来就有二义性，在使用时也必须把这种二义性除掉。如“运动”一词。（2）所有相同意思的概念必须用同一个原子概念来表示，即表示的唯一性。否则，本来是相同的概念会导致不同的理解。（3）各原子概念之间，它们的表达范围不应该重复，即原子概念的正交性。（4）各原子概念之间应该互相独立。一个原子概念不应该用另一个原子概念来定义，即原子概念的不可再分性。（5）原子概念的数目要尽量少。数量少而表达的东西多，说明其概括性强。

Schank 对 11 种动作（ACT）的原子化如下：（1）PROPEL:应用物理力量（推、拉、打等）于一对象。（2）GRASP：一个演员抓起一个物理对象。（3）MOVE：演员身体的一部分变换空间位置，如抬手等。（4）PTRANS：物理对象变换位置，如走进、跑出等。（5）ATRANS:抽象关系的改变，如传递、赠送、革命等。（6）ATTEND:用某个感官获取信息，如用目光搜索等。（7）INGEST:演员把某个东西吸入体内，如吃、喝等。（8）EXPEL:演员把某个东西送出体外，如呕吐、落泪等。（9）SPEAK:演员产生一种声音，如唱歌、尖叫等。（10）MTRANS:信息的传递，如读报、看信、看电视等。（11）MBUILD:由旧信息形成新信息，如“眉头一皱，计上心来”。

剧本是描述特定范围内原型事件的结构。剧本的组成包括：（1）**进入条件**：指出剧本所描述的事件可能发生的先决条件，即事件发生的**前提条件**。（2）**角色**：描述事件中可能出现的人物。（3）**道具**：描述事件中可能出现的有关物体。（4）**场景**：描述事件序列，可以有多个场景。（5）**结局**：给出剧本所描述的事件发生以后必须满足的条件。下面就是一个剧本的例子。

Schank 餐馆剧本

剧本：就餐

进入条件：顾客饿了，品尝佳肴，招待亲友，顾客有钱

角色：顾客、服务员、厨师、经理、老板

道具：食品、桌子、菜单、钱

场景：

第一幕：进入餐馆

PTRANS：步入餐馆

ATTEND：用目光寻找空桌

MBUILD：选定桌子

MOVE：坐下

第二幕：定菜

ATRANS：服务员送菜单

MTRANS：读菜单

MBUILD：选定所要的菜

MTRANS：告诉服务员

ATRANS：付钱

第三幕：吃饭

ATRANS：服务员上菜

INGEST：吃饭

第四幕：离开

MOVE：站起身来

PTRANS：步出餐馆

结局：顾客吃了饭；顾客花了钱；老板挣了钱；餐厅食品减少了。

由上述剧本实例可知，剧本就像电影剧本一样，一场一场地表示一些特定事件的序列。

实际上，现实问题和剧本完全一致的情况是很少的，故每个剧本应该有适应临时出现的新情况的能力。这些能力包括：（1）**子剧本调用**。这往往是由意外情况引起。例如：“老王正在吃烤鸭时，忽然在鸭肉里咬到一根铁钉”。此时，正常的剧情就不能演下去了。可能需要调用名为“交涉”的子剧本。（2）**排除故障**。由于剧本的“进入条件”往往是隐含的，有时可能因某个条件不成而 ACT 无法进行，这时要求剧本备有变通动作来排除这种障碍。例如：老王找不到空位置，今日烤鸭已售完，身上带的钱不够等，这些都是障碍。相应的变通动作可以是：站在别人桌旁，甚至换一家烤鸭店等。（3）**调用剧本以外的知识。**应该有许多备用知识（即额外知识）存在库中供剧本调用。此外，额外知识还可以用来排除障碍和调用子框架。（4）**提炼、忘却和想象**。为理解故事的核心，一个剧本系统应该能提炼出最主要的情节，为此就要忘掉许多次要的情节，即故事情节收缩。还应该能通过想象、子剧本调用、插入障碍、设置变通等方式使得故事情节膨胀。

与框架表示法相比，剧本比较呆板，能力也有限。另外，人类日常的行为千变万化，很难用一个剧本就能理解各种各样的故事情节。目前，剧本表示法已用于自然语言理解方面。

2．过程表示法

在 AI 界关于知识表示方法有两种不同的观点：第一种观点认为，知识主要是陈述性的，其表示方法应着重将其静态特性（即事物的属性及事物间的关系）表示出来。人们称以这种观点表示知识的方法为陈述式或说明性表示方法。第二种观点认为，知识主要是过程性的，其表示方法应将知识及如何使用这些知识的控制策略均表示为求解问题的过程，以这种观点表示知识的方法被称为过程性表示方法。

过程表示法的基本思想是：注重于对知识的利用，把与问题有关的知识以及如何运用这些知识求解问题的控制策略都表述为一个或多个求解问题的过程，每个过程就是一段程序，用于完成对一个具体事件或情况的处理。在问题求解过程中，当需要使用某个过程时就调用相应的程序并执行之。在以这种方法表示知识的系统中，知识库是一组过程的集合，当需要对知识库进行增、删、改时，则相应地增加、删除或修改有关的过程。

一般地，一个过程规则包括如下四个部分：（1）**激发条件**：由两部分组成： a）**推理方向**：指出是前向推理还是后向推理。b）**调用模式**。（2）**演绎操作**：由一系列的子目标构成。（3）**状态转换**：用于对数据库进行增、删、改。（4）**返回**：过程规则的最后一个语句是 RETURN，用于指出将控制权返回到调用该过程规则的上级过程规则去。

过程表示法的优点是：效率较高；控制系统容易设计。缺点是不易修改、添加新的知识，而且当对某一过程进行修改时，有可能影响到其它过程，对系统的维护带来诸多不便。

3．Petri 网表示法

Petri 网的概念是德国学者 Cah Abam Petri 首先提出的，用于构造系统模型及进行动态特性分析，后来逐渐被用作知识表示方法。Pertri 网主要有三个基本元素：位置、转换、标记。一个典型的 Pertri 网单元如下图所示。

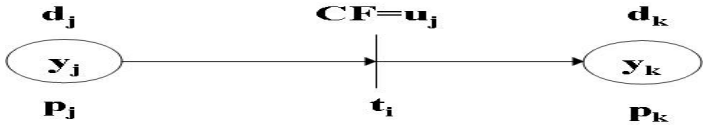

图 17.2.8　Petri 网表示法

上述 Petri 网用于表达与“IF d_j THEN d_k (CF = u_i)”类似的含义。其中，P_j与 P_k分别代表第 j 个和第 k 个位置；y_j与 y_k分别是这两个位置的标记；t_i是某个转换；产生式规则的前提 d_j和结论 d_k分别对应于 P_j与 P_k；u_i是规则强度。对于比较复杂的知识，Pertri 网通常用一个八元组来表示知识间的因果关系，具体形式是：

$$Pnet = \{ P, T, D, I, O, f, \alpha, \beta \}$$

其中，P 是位置的有限集，记为 $P=\{p_1, p_2, \ldots, p_n\}$；T 是转换的有限集，记为 $T=\{t_1, t_2, \ldots, t_n\}$；D 是命题的有限集，记为 $D=\{d_1, d_2, \ldots, d_n\}$；I 为输入函数，表示从位置到转换的映射；O 为输出函数，表示从转换到位置的映射；f 为相关函数，表示规则强度，取值范围[0, 1]；α为相关函数，表示位置对应命题的可信度[0, 1]；β为相关函数，表示从位置到命题的映射。

Petri 网表示法的优点是便于描述系统状态的变化及对系统特性进行分析，可在不同层次上变换描述，而不必注意细节及相应的物理表示，这样就可把注意力集中到某个层次的研究上。

4．面向对象的知识表示方法

面向对象的表示法起源于 SMALLTALK 的研究工作。1980 年，SMALLTALK-80 作为面向对象编程语言的推出，推动了各种不同风格和不同用途的面向对象语言的研究和相继问世，尤其是 C++,已发展为应用最为广泛的主流编程语言。因而，面向对象（Object-oriented，简称 OO）方法和 C++编程技术已经成为广为使用的、成熟的关键技术。

面向对象的知识表示方法的基本出发点是：客观世界是由一些实体组成的。这些实体有自己的状态,可以执行一定的动作。相似的实体抽象为较高层的实体,实体之间能以某种方式发生联系。所谓对象就是对这些实体的映象。对象中封装了数据成员(或者叫实例成员)和成员函数(方法)。数据成员可以用来描述对象的各种属性,这些属性是对外隐蔽的。外界可以且仅可以通过成员函数访问对象的私有成员,数据成员可以被初始化,可以通过成员函数被改变,因此对象可以动态地保存当前自己的状态。由于对象中还包含了操作(成员函数),因此可以把求解机制封装于对象之中。这样对象既是信息的存储单元,又是信息处理的独立单位,它具有一定的内部结构和处理能力。各种类型的求解机制分布于各个对象,通过对象之间消息的传递完成整个问题求解过程。用对象表示的知识与客观情况更为接近，这种表示方案比较自然,易于理解。

面向对象表示法具有如下优点：①“继承”带来了天然的层次性和结构性。在高层次,对象能封装复杂的行为,使具体细节对该层知识使用保持透明，从而降低问题描述和计算推理的复杂度；通过继承可以减少知识表达上的冗余,知识库的修改、增加、删减以及使用和维护都十分方便；对一个知识单元进行修改不会影响其它单元,每一知识单元中所包含的知识规则有限,推理空间小,提高了推理效率；②对象本身的定义产生了良好的兼容性和灵活性,它可以是数据,也可以是方法；可以是事实,也可以是过程；可以是一个框架,也可以是一个语义子网络；③用几何语言来描述的话，面向对象的抽象机制实际上是将对象看成了客观世界及其映射系统的分形元，因而事物都可以由这些分形元堆垒而成。分形的特征首先是不断的细分，这和知识结构的不断扩展是一致的。其次是“比例自

相似性”,使得我们有可能“从简单的原则衍生出复杂的系统”。

从本质上讲，面向对象的表示法与框架表示法有许多相似之处，如层次分类体系和特性继承机制等。但由于应用目标不同，实现和使用方式有较大的差别。框架表示法旨在支持知识的陈述性表示，强调事物的结构化描述和对人思维方式的模拟，注重清晰、灵活地表示思维；而面向对象的表示法则强调信息（数据）的结构化处理，注重信息和信息处理的封装和程序设计的模块化，注重安全有效的信息处理和程序的易维护性。

5．基于本体的知识表示方法

本体是对领域实体存在本质的抽象，它强调实体间的关联，并通过多种知识表示元素将这些关联表达和反映出来。这些知识表示元素也被称为元本体,主要包括：①概念—表示领域知识元,包括一般意义上的概念以及任务、功能、策略、行为、过程等等,在本体的实现中,概念通常用类(class)来定义,而且通常具有一定的分类层次关系；②属性—描述概念的性质,是一个概念区别于其他概念的特征,通常用槽(slot)或者类的属性(Properties)来定义；③关系—表示概念之间的关联,例如一些常用的关联：父关系、子关系、相等关系；④函数—表示一类特殊的关系,即由前 n-1 个要素来唯一决定第 n 个要素；⑤公理—表示永真式,在本体中,对于属性、关系和函数都具有一定的关联和约束,这些约束就是公理,公理一般用槽的侧面(facet)来定义；⑥实例—表示属于某个概念类的具体实体。

本体的每一个知识表示元素也可以被看作一个知识片,每一个知识片都包含名称、定义和文档说明。

总的来说,构造本体的目的都是为了实现某种程度的知识共享和重用。从广义来讲,本体的作用主要有以下两方面:①本体的分析澄清了领域知识的结构,从而为知识表示打好基础。本体可以重用,从而避免重复的领域知识分析；②统一的术语和概念使知识共享成为可能。较为具体的讲，本体的作用包括三个方面：即交流(communication)、互操作(inter-operability)和系统工程(systems engineering)。其中，交流主要为人与人之间或组织与组织之间的交流提供共同的词汇；互操作在不同的建模方法、范式、语言和软件工具之间进行翻译和映射，以实现不同系统之间的互操作和集成。本体分析能够为系统工程提供以下方面的好处：① 重用(reusability)。本体是领域内重要实体、属性、过程及其相互关系形式化描述的基础，这种形式化描述可成为软件系统中可重用和共享的组件；② 知识获取(knowledge acquisition)。当构造基于知识的系统时,用已有的本体作为起点和基础来指导知识的获取,可以提高其速度和可靠性；③ 可靠性(reliability)。形式化的表达使得自动的一致性检查成为可能,从而提高了软件的可靠性；④ 规范描述(specification)。本体分析有助于确定 IT 系统(如知识库)的需求和规范。

本体作为一种知识表示方法,与谓词逻辑、框架(frame)等其他方法的区别在于它们属于不同层次的知识表示方法。本体表达了概念的结构、概念之间的关系等领域中实体的固有特征,即“共享概念化”,而其他的知识表示方法如语义网络等,可以表达某个体对领域中实体的认识,不一定是实体的固有特征。这正是本体层与其他层次的知识表示方法的本质区别。

6．贝叶斯信念网（信念网络）

贝叶斯信念网（Bayesian Belief Networks）也称贝叶斯网络（Bayesian Networks）、信念网络（Belief Networks）。它是在贝叶斯分类器的基础上发展而成的一种知识表示方法。贝叶斯信念网的引人之处，是提供了一种方便的途径以表示因果知识。

朴素贝叶斯分类器要求所有变量在给定目标变量值时是条件独立的，当此条件成立时，朴素贝叶斯分类器可得到最优贝叶斯分类。然而，在许多情况下，这一条件独立假定明显过于严格。贝叶斯信念网的基本思路是：它描述的是一组变量所遵从的概率分布，它通过一组条件概率来指定一组条件独立性假定。与朴素贝叶斯分类器不同的是，贝叶斯网络可表述变量的一个子集上的条件独立性假定。因此，贝叶斯信念网提供了一种中间的方法，它比朴素贝叶斯分类器中条件独立性的全局假定的限制更少，又比在所有变量中计算条件依赖更可行。

贝叶斯网表示联合概率分布的方法是：指定一组条件独立性假定（它表示为一有向无环图）以及一组局部条件概率集合。联合空间中的每个变量在贝叶斯网中表示为一个“节点”。对每一个变量需要两种类型的信息：（1）网络中的“弧”表示断言“此变量在给定其直接前驱时条件独立于其非后继”。（2）对每个变量有一个条件概率表，它描述了该变量在给定其直接前驱时的概率分布。

对于网络变量的元组$<Y_1,\ldots,Y_n>$赋以所希望的值$(y_1, \ldots, y_n)$的联合概率的计算公式为：

$$P(y_1,\cdots,y_n) = \prod_{i=1}^{n} P(yi|Parent(Yi))$$

其中，$Parents(Y_i)$表示网络中 Y_i的直接前驱的集合；$P(y_i \mid Parents(Y_i))$的值等于与节点 Y_i关联的条件概率表中的值。

在实际应用中，由于客观条件的限制，人们不可能预先知道贝叶斯网络中所有变量的概率分布，而希望在知道一部分变量的值时获得某变量（如 ForestFire）的概率分布。一般来说，贝叶斯网络可用于在知道某些变量的值或分布时计算网络中另一部分变量的概率分布。由于我们所要处理的是随机变量，所以一般不会赋予目标变量一个确切的值，事实上，真正需要推理的是目标变量的概率分布，它指定了在给出其它变量的观察值条件下，目标变量取每一个可能值的概率。

对贝叶斯网络的研究主要有：（1）贝叶斯网络中的近似推理方法研究。由于对任意贝叶斯网络的概率的确切推理已经证明是一个 NP 完全问题，比较流行的研究策略是：“牺牲精度换取效率”的近似推理方法。当然，从理论上说，即使是贝叶斯网络中的近似推理也可能是 NP 难题，好在实践中的很多情况下这类近似推理方法还是非常有效的。（2）从训练数据（样本）中学到贝叶斯信念网络的有效算法研究。解决“学习贝叶斯信念网络”问题的两个基本思路是： a) 既可预先给出贝叶斯网络结构，也可由训练数据（样本）推得。特别是当网络结构无法预先知道时，学习贝叶斯网络就显得十分困难。Cooper 和 Herskovits 提出一个贝叶斯评分尺度(Bayesian scoring metric)，以便从不同的网络结构中进行选择。Spirtes 提出基于约束的学习贝叶斯网络结构的研究途径，这种方法能从数据中推导出“独立”和“相关”的关系，然后用这些关系来构造贝叶斯网络。b) 网络中的所有变量（概率分布）可以直接从每个训练样本中观察到，或者某些变量（概率分布）不能观察到，未能观察到的变量需要由已知的变量推得。（3）贝叶斯网络的梯度上升训练（Boosting）方法研究。为了定义贝叶斯网络的条件概率表中的参数，Russell 等人提出了梯度上升（Boosting）规则，它通过计算 LnP(D|h)的梯度使得 P(D|h)最大化。（4）EM 算法研究。该算法的目的是：用观察到的变量实例去预测未观察到的实例中的变量的值。该算法是 Dempster 等人在 1977 年提出的。

17.2.8.4 关于知识表示方法的进一步讨论

1．关于知识表示的充分性与效用

从一定意义上讲，大多数知识表示方式都是以一阶逻辑为基础的，均有可转变为等价的一阶逻辑表示方式的手段。所以，逻辑是知识表示的基本手段，进而也就构成了智能系统研究的基础。从

衡量知识表示性能的角度考虑，可以说一阶逻辑具有充分的表示能力，但其表示法效用（包括概念效率和计算效率）较差。由此，人们希望通过产生式系统去提高推理的效率，通过开发语义网络和框架系统去提高概念效率（实现结构化表示）。然而表示法效用的提高是以牺牲表示的充分性为代价的。在许多情况下，牺牲表示的充分性是允许的，也是合理的，因为实用的智能系统往往只涉及到某些方面的表示充分性，其它方面可以忽略。例如，Prolog 语言只允许以特别形式的符号结构--Horn 子句表示知识，丢失了表示不完全知识的能力，如否定、或、存在等；语义网络限制了与其等价的一阶逻辑表示到只允许一元或二元谓词公式，并易于引起节点和节点间关系的二意性解释。这些均减弱了表示的充分性，但却提高了表示法效用，使这些表示方式能应用于许多实际问题的解决。另外，正因为语义网络表示的不充分性以及在表示法效用方面的加强很不相同，才使它们适用于明显不同的场合。

语义网络注重表示对象间的关系，而**框架系统**更强调对象的内部结构。由于节点（框架）集中了概念或个体的所有属性描述和关系描述，又可用槽作为索引，故而这两种方式在知识库中进行检索时具有较高的效率。同时，由于特性（属性和关系）继承和程序附加功能（在槽中附加程序），使有效地组织和处理知识成为可能。它们的缺点在于这两种结构化的表示方式在刻画真值理论方面过于自由化，容易引起二意性甚至严重的错误。例如，一个框架的语义就有作严格定义（定义一类个体必须具备的特性）和典型范例（说明一类个体应有的典型特性）这两种不相容的解释；Isa 链既用于表示集合-子集关系又用于表示集合-成员关系；表示集合概念的框架既定义了集合本身的特性，又记载了其成员的共性。诸如此类的二意性因不存在通用的法则去克服，有可能会成为开发知识系统的陷阱。另外，由于结构化表示的复杂性，知识库维护需付出比前两种方式高得多的代价。

2．关于程序性和陈述性知识的表示

智能系统所拥有的知识可分为两大类；表达或归类为程序性的知识和表达或归类为陈述性的知识。在智能系统中，被表达或归类为陈述性的知识常以相互独立的陈述语句的形式出现，例如谓词公式、语义网络节点和框架等，甚至直接以受限的自然语言语句的形式出现。这些陈述性知识的共同特点是；能清晰地存储于独立定义的符号结构中；可由应用该知识的程序存取。与此相对照，系统内被表达或归类为程序性的知识，如搜索算法控制策略等，则隐含于推理机中。

陈述性知识表达在智能系统构建中处于突出而重要的地位，关于知识表示的各种研究，也主要是针对陈述性知识表达的，原因在于它有着许多程序性知识表达无法比拟的优点；（1）易于修改。由于各陈述性语句相互独立，对于知识的修改不会产生副作用；而程序性知识的修改直接涉及程序的变动，显然困难得多。（2）可应用于多重目标。例如 MYCIN 知识库中存放的知识就可以为推理、解释和教学等几个程序共享，甚至能服务于设计知识库时未曾想到的目的。该知识库在设计时仅考虑到应用于推理和推理结果的解释，后来才发现可直接询问知识库包含的内科疾病诊断和治疗知识，进而开发出教学程序。与此相对照，由于程序性知识表达隐含于特定的程序，只能为该程序所专用；若要用于其它目标，就需重复表示这些知识，很不方便。（3）易于扩展。由于陈述性语句的相互独立性，知识库不仅便于静态扩展，也便于动态吸收在推理过程中衍生的附加知识，以支持智能系统的自适应学习。（4）支持自我意识。可以设计所谓的内省程序去查询系统接受的问题求解任务是否超越系统自身拥有知识的范围和方面，以便表现出一定的自我意识能力。知识独立于应用程序而存储，使这种查询成为可能。

当然，为获得以上优点而使用陈述性知识表达是要付出代价的，那就是导致计算开销的增大和效率的降低。因为陈述性知识表达要求应用程序对其作解释性执行，显然效率比直接用程序性知识表达（相当于编译执行）低得多。换言之，陈述性知识表达是以牺牲效率来换取灵活性。归纳起来，这两种知识应用的倾向如下：（1）鉴于高级的智能行为（如人的思维）似乎强烈地依赖于陈述性知识，智能系统的研究应注重陈述性知识的开发。当然，这绝对不是意味着程序性知识不重要。例如，当陈述性知识被反复应用于某一特别目的时，将其嵌入实现该目的之应用程序是十分可取的。(2) 程序性知识的陈述化表示。用规则表示的启发式知识属于程序性知识，因为它们描述推理过程且上下文相关。但这些知识从推理机分离出来并由推理机作解释执行，具有陈述性特点。这种表示方式可以促进推理和控制的透明化，有利于智能系统的维护和进化。（3）程序性知识和陈述性知识的综合。以适当的方式综合应用这两种知识可提高智能系统的性能。框架表示法为综合提供了有效的手段，每个框架陈述性地表示对象的属性和对象间的关系，并以附加程序（If-Needed，If-Added 等）的方式表示程序性知识。

3．关于表示能力和推理效率之间的制约关系

如前所述，衡量知识表示性能的两个方面--表示的充分性和表示法效用之间存在着相互制约关系，即往往强调了表示的充分性就会牺牲表示法效用，反之亦然。可以说，任何知识系统的设计都是权衡这两个方面利弊的结果，以便提供具有足够表达能力的知识表示语言，并使以这种语言表示的知识能有效地用于推理。这里，我们从一阶谓词逻辑表示不完全知识的能力出发，进一步阐述关系数据库、逻辑程序设计和语义网络是如何通过全部或部分消除表示不完全知识的能力来提高推理效率的，以说明表示能力和推理效率之间制约关系的存在，和如何作权衡处理。

（1）一阶逻辑的不完全知识表示。一阶逻辑用于知识表示的目的不同于数学应用。后者从研究数学基础理论的角度，强调无限集合的形式化表示。知识表示系统刻画的应用领域却往往是有限的，所以一阶逻辑的应用并不注重处理无限集合，而是侧重于不完全知识，因为描述领域世界时常遇到它们。例如：

¬Student(John)　（指出 John 不是学生，但未说明他是什么人）

Parent(Sue，Bill) ∨ Parent(Sue，George)（指出 Sue 的长辈是 Bill 或 George，但未说明究竟是哪一个人）

（∃x）Cousin(Bill，x) ∧ Male(x)（指出 Bill 至少有一个表兄弟，但未说明他是谁）

从这些例子可以看出，一阶逻辑并不用于描述事物的复杂细节；相反，它们是通过取反、析取和存在量词去避免表示不必或无法说明的细节。正是这些不完全知识构成一阶逻辑表示能力的重要方面。然而，实现处理不完全知识的能力是要付出代价的，即增加了不易处理性。在实际应用中，我们往往不需或只需有限的不完全知识表示能力，因此，可以通过限制表示不完全知识的能力去换取推理的有效性。

（2）关系数据库。完全消除表示不完全知识的能力，就是让知识库退化为关系数据库。让我们来观察一个简单的数据库，其只包含一个关系 COURSES 如表**表 17.2.3**。

表 17.2.3　关系表

标识	课程名	开课系	人数	教师

CS248	程序设计	计算机	42	王芳
MAT100	数学分析	数学	137	李进
CSC373	人工智能	计算机	53	周文
⋮	⋮	⋮	⋮	⋮

这个关系表完全可以用一阶逻辑来表示如下；

Course(CSC248)，Dept(CSC248，计算机)，Student(CSC248，　42)，…

Course(MAT100)，Dept(MAT100，　数学)，Student(MAT100，137)，…

显然，这仅仅涉及到一阶逻辑的部分表示能力，即只使用谓词表达式并隐含地指出谓词表达式之间存在合取关系，不涉及量词、析取和取反，从而消除了表示不完全知识的能力。例如，数据库不能表示如下信息：

Dept(MAT100，数学) ∨ Dept(MAT100，计算机)

然而，包含于数据库的某些信息并没有体现于一阶逻辑的表示中。例如，我们或许会问，计算机系开设了几门课？以一阶逻辑表示的知识不足以用来回答这个问题，因为没有足够的知识说明至少有两门课（CSC373 和 CSC248 可以是同一门课的两个名字），也没有说明最多有两门课（假设谓词表达式的列表中只显式指出这两门课由计算机系开设，但有可能还有该系开设的其它课程未列于表中）。作为对照，若把该问题转变为询问语句去操作数据库，就可以得到解答：两门课。原因在于已有隐含的知识包含于数据库管理系统：

- 同一数据场的不同可取值指示不同的个体。
- 除显式列举的可取值外不存在其它可取值。

为了使知识库能用于回答上述问题，只要把这些隐含的知识显式地表示于知识库即可：

$C_i \neq C_j$ for distinct constants C_i and C_j

$\forall x[\text{Course}(x) \Rightarrow x=\text{CSC249} \vee \cdots\cdots \vee x=\text{MAT100}]$

由此可以得出结论：

（a）数据库可以等价地表示为基于一阶逻辑的知识库。为此，不仅需要用一阶逻辑表示直接记载于数据库关系表中的信息，也应显式表示隐含于数据库管理系统的信息。

（b）数据库不能表示不完全知识，这使其表示能力很受限制。例如，无法表示"CSC148 或 CSC149 是计算机系开设的课程"这样的事实。显然，用一阶逻辑表示直接记载于关系表的信息时，不需使用量词、析取和取反。

（c） 数据库系统中，推理行为退化为演算，因为不存在非显式表示的个体，只需检索和演算， 不需推理。

尽管从知识表示的角度，数据库表示能力十分受限，但因免除了推理，大幅度地提高了处理效率，所以，在许多不需推理的事务处理中，数据库是表示领域世界的最佳选择，可删除不必要的表示能力，以换取高效的信息处理服务。

（3）逻辑程序设计。逻辑程序是另一种一阶逻辑的受限表示形式，典型的例子为 Prolog 语言。用该语言编写的知识库也分为显式和隐含两个部分。显式部分是称为 Horn 子句的受限形式的一阶逻辑语句，形如：

$(\forall x_1)\wedge(\forall x_n)[P_1\wedge\cdots\cdots\wedge P_m\Rightarrow P_{m+1}]$

其中，$m\geqslant 0$，P_i是谓词表达式。当 $m=0$ 且 P_i中不含变量时，子句集退化为等价于数据库的知识库。隐含的知识包含在 Prolog 解释程序中：

·存在于子句集中的任何两个不同的项（常量、变量和函数）均指示不同的个体。

·不能由显式表示于子句集的知识推导出的事实（以谓词表达式表示）均设想为假--称为封闭世界假设。

由此可见，逻辑程序与数据库一样，只允许表示关于领域世界的完全知识，不同之处仅在于逻辑程序允许经由推理回答问题。例如，以下是显式表示于 Prolog 知识库的知识；

```
Parent（John， Bill）.
Parent（John， Mary）.
Father（x，y）:- Parent（x，y），Male（y）.
Male（Bill）.
```

Prolog 解释程序能准确回答 John 的父亲是 Bill，但只是在推理成功地执行之后。

由此，我们可以得出结论：(1）逻辑程序是一阶逻辑的受限表示形式。（2）不表示不完全知识，体现在不使用量词、析取和取反。（3）允许通过使用变量和规则推断隐含的信息。（4）通过限制表示能力（只能以 Horn 子句表示知识），与经典一阶逻辑相比，可显著提高推理效率。可见，逻辑程序设计也是通过限制表示能力（不表示不完全知识）来换取处理效率的。由于允许推理，它显然比数据库具有更强的问题求解能力；但因需要表示推理规则和允许使用变量，效率要比数据库低得多。

（4）几点结论。从标准的一阶逻辑、逻辑程序和数据库表示能力的递减和解答问题效率的递增，已经很好地说明了知识表示的充分性和推理的有效性之间的制约关系。

·表示能力和易处理性之间存在制约关系。

·知识表示的性能无绝对衡量标准，应依据手头实际的问题求解任务权衡所需的表示能力和易处理性。可以说，一阶逻辑具有充分的表示能力，但不易处理；数据库处理效率最高（只需检索，不要推理），但表示能力差，只能适用于不要求推理的应用问题。

·删除不必要的表示能力、将会明显提高推理的有效性。例如，除了智力游戏和特别的定理证明问题需要处理"析取"表示外，大部分实用问题都不涉及，所以目前实用的专家系统几乎都不处理"析取'，从而大幅度提高了问题求解的有效性。

·现有知识表示语言均可视为一阶逻辑的受限情况。由于受限方面不同，以及隐含信息的作用，这些语言在表示能力和易处理性上各有优缺点。例如，逻辑程序语言便于作演绎推理，而语义网络则便于图搜索和特性继承。

·混合系统促进表示性能。适当地混合不同的表示语言，可以让它们相互间取长补短，实现优良的表示性能。例如 Brachman 等人通过混合使用一阶逻辑和框架语言，设计了 KRYPTON 表示系统。框架语言用于定义术语（概念）以及术语间的包含关系，一阶逻辑则以这些术语作为谓词来描述领域涉及的信息，并作归结定理证明。结果，KRYPTON 既有充分的表示能力（基于一阶逻辑），又有高效的推理能力（基于框架的图搜索和特性继承）。此外，产生式表示和框架表示的混合也广泛应用于各种专家系统，以便综合两者的优点：产生式的演绎推理、框架 语言的对象结构化描述、图搜索

和特性继承。当然，实现良好的综合是一种艺术，既要使两者协调合作，取长补短，又不能使它们相互干扰。

17.2.9　问题求解系统中的信念维持

人们在解决实际问题时，常常会因时间、事实和知识等资源的缺乏而不得不依赖于个体对认识作某些假设(不确保正确的信念)，并以此为基础进行某种尝试性推理。一旦推理失败或发现矛盾，就撤消推理结果和导致矛盾的假设。显然，尝试性推理具有非单调性，这与传统的逻辑系统支持的单调推理不同。后者基于永真的公理集，仅是把该公理集隐含的事实显式化，从而使推理结果单调地增加，不存在需撤消的推理结果。所以，传统的逻辑推理技术不能适用于非单调推理的情况，必须研究适合于非单调推理新的推理方法和技术。

在问题求解过程中，人的信念常常是不确定的或不精确的。不确定的信念不能以简单的真假逻辑加以表示。例如，我们约定于某个时刻与一个朋友共进午餐，但我们并不能完全肯定他能如期赴约，因为他有碰到意外事件的可能性。这种不确定性取决于城市交通状况和朋友的工作环境。而不精确的信念意指人的认识具有模糊性和近似性。例如，我们知道某人是个高个子，显然“个子高”是确定的，但高的程度却是模糊的。基于传统逻辑的“硬”计算无法解决上述问题，因为其要求使用确定的和精确的数据及知识。为此，也需要突破传统逻辑的限制，发展能模仿人做近似而非严格推理的“软”计算技术。

本节,我们将考虑问题求解系统中信念的维持与不确定性的处理方法，介绍非单调推理技术和可实现“软”计算的几种主要方法。

17.2.9.1 传统逻辑推理系统的局限性

传统的逻辑推理系统是将其拥有的关于问题领域的事实和知识表示为一阶谓词演算公式的一个有限集Δ，也称为系统的基本信念集。为解答问题或启动适当的行为，系统必须首先决定某个公式φ是否可从Δ推导出来。若采用归结反演方法，则只需能从$\Delta \wedge \neg\varphi$ 归结出空子句。尽管从理论上讲，这类逻辑系统具有最强的表示充分性，但在表示真实世界时仍存在局限性，这可归纳为以下三个方面。

（1）**系统的有限信念集仅是真实世界的近似描述**。因为真实世界是千姿百态的，必定会存在一些违反常识的甚至穷举不尽的例外。例如，“鸟会飞”是人们的常识，可以表示为：$\forall x$（Bird（x）$\Rightarrow$Flies（x)），但鸵鸟例外，故常识应修改为；$\forall x$（Bird（x）$\wedge \neg$Ostrich（x）$\Rightarrow$Flies（x））。然而，实际上还存在其它不会飞的鸟，如幼鸟、死鸟、伤残鸟等，若逐个列举，将有无穷多个例外。这类问题称为资格限定（qualification）问题，需要通过非传统逻辑推理来解决。

（2）**系统要求其基本信念集必须是一致的**。系统要求其基本信念集中已有的信念不会改变，任何新的信念的加入也不应引起矛盾。但实际上，真实世界是不断变化的，人的信念也并不总是正确的，会随着认识的提高而不断发生变化。如何表示暂时的试探性的信念是传统逻辑系统解决不了的问题，因为这些信念并不能保证永远正确，以后可能会发现错误而不得不加以修改或删除。

（3）**系统基本信念集中的信念只有非真即假这两种选择**。而事实上，真实世界中许多事情并不能以这种走极端的方式表示，常常是模糊的和不确定的。例如，“今天天气很冷”，“很冷”是一个模糊的概念，涉及到冷的相对程度问题；“聪明的人往往头发少”，“往往”也意味着不确定，其中有个如何表示其确定性程度的问题。

对上述三个局限性中的第一个，我们或可以通过语义网络的特性继承原理来解决。特性继承是缺省推理的一种实现方式，其基本思想可归纳如下：语义网络中某节点（指示个体或子类）之某特性缺省，则隐含着继承其父节点（指示类或超类）的相应特性。而第二和第三个局限性，则需要非单调推理技术和软计算技术。

17.2.9.2 信念维持与非单调推理技术

如上所述，传统的逻辑推理系统实际上是严格的单调推理系统，要求新加进系统的知识（信念）

必须与已有的知识（信念）相一致，不引起矛盾。所以，随着运行时间的推移，系统内所包含的知识有增无减，这就是所谓的单调性。单调性的优点在于：① 加入新命题时不需审查与系统原有知识的相容性，因为这些新命题只能是已有知识的逻辑推理结果，不可能引起矛盾。换言之，加入的新命题必定是永真的。② 不需要记忆推导过程。因为推导出的结论永远不会失败，不存在需要事后审查推导过程的问题。这就使得基于单调逻辑推理的问题求解技术能简单而有效地得到应用。

但是，众所周知，真实世界充斥着不完全可信的信息和不断变化的状况，在解决复杂问题的过程中，也需要应用一些并不能保证完全正确的假设。即使对于一个较简单的问题求解任务，也常常难以找到一组一致性的逻辑公式来准确表达，就是找得到，也不能保证它们在变化的世界中能永远保持一致性。所以，放宽传统逻辑推论系统的限制到允许包含一定程度的“假设”是必要的。这些“假设”可作为推理的依据，但在推理过程中，随着新事物的出现，可能到头来会发现这些“假设”并不正确，应予删除，从而造成推理过程的非单调性；即新知识（事实）的加入会引起已有知识（“假设”以及基于“假设”的推理结果）的删除。由此，传统的逻辑演绎技术就不再适用，而必须开拓适合非单调推理的推理方法和技术。

最先想到的非单调推理的推理方法应是缺省推理。由于很少有这样完美的信息系统--其在处理过程中拥有所需的一切信息，因此，在缺乏信息时，一个有效的作法就是根据已有信息和经验先做一些有益的“猜测”，只要不发现反面的证据，就认为它们是正确的。构造这些猜测的过程就被称为缺省推理。例如，在美国第一次应邀去朋友家作客，按常理应带点礼物，但你并不了解朋友喜欢什么样的礼物，这时就可以根据常识进行猜测--鲜花总是受欢迎的。实践证明，根据常识行事往往正确。但常识并不等于真理，有可能在某些特殊情况下出错，尽管可能性较小。例如，当你带着鲜花到朋友家，可能发现主人对花过敏，见了直打喷嚏，显然，这时常识出了错，若你带着备用礼品的话，就应立即改送备用礼品。

上述例子属于一种常见的缺省推理--最有可能的选择。意指，若已知在一组可选事物中必有一个为真，则在缺乏完全信息的情况下，选择最有可能的一个。比如，因为大多数人喜欢花，所以会朋友一般首选送花。另一种重要类型的缺省推理是“限制(Circumscription)”推理，它是将满足某特性 p 的对象“限制”为仅是从基本信念集Δ（永真）可以推导出满足 p 的那些对象。例如，若当前的任务是划船渡河，可能会有许多不利的条件：有风有雨、水深浪急、缺少船桨、船漏、船可能在过河时搁浅在泥沙中等，都会阻止划船渡河的成功。如果我们要全部仔细考虑实际上几乎不可能发生的所有不利因素和其解决办法，可能永远也不能实施划船渡河。正确的做法应是只须考虑实际上明显存在的不利条件即可，其余的只当不存在。只有这样，我们才可能快速决定是否利用船和怎样利用船。例如，我们发现船漏，但不严重，就会决定采用一定的措施堵漏，并立即用其渡河。总之，“限制”推理是把应考虑的因素限制在易于清晰证明的范围内。在上例中，若没有直接的证据指示船不能用，就视为船可用。

缺省推理的一个精确的形式化定义是：设 x 为某个信念（可表示为一阶逻辑语句），y 为 x 为假时才为真的某结论；则可以将缺省推理定义为以下三种：若不知道存在 x，则认为有结论 y；若不能证明 x 为真，则认为有结论 y；若不能在某个给定的时间期限内证明 x 为真，则认为有结论 y。这三个定义汇总起来意指：若无 x 为真的迹象，则认为 y 是真。注意，这三个定义一个比一个更放松其适用范围。从计算机实现的角度，所谓“知道”意指信念 x 显式存储于知识库（信念数据库），这很难；因为知识库只能存储世界描述的极小部分，其余必须从已知部分推出。第二个定义将适用范围放松到 x 能否被证明；遗憾的是，x 能否被证明并不是逻辑系统本身保证可判定的。为了防止证明无休止地进行下去，第三个定义给证明花费的时间规定了允许的期限。

我们可以把需要非单调推理的理由归结为以下几个方面：① 不完全信息的出现要求缺省推理。正如上面所述，缺省推理是非单调推理的典型表现。② 一个不断变化的世界必须用变化的知识库加以描述。世界总是不断变化的，即使能获得关于特定问题求解的全部知识，也不能持久。当然，变

化会仅涉及局部事物，而其它的不变，这就是所谓的“框架问题”。解决的办法是取消那些已经变得不精确的知识，而代之以另一些更精确的知识。这也就是说，在增加逻辑语句到知识库的同时也删除原有的语句，从而导致非单调推理。③ 产生一个问题的完全解答或许要求关于部分解答的暂时性假设。有些问题求解系统虽然不存在上述两方面问题，但为了促进求解，往往也需要加进一些假设作为试探性的部分解答。这些假设可能不正确，需要在以后发现时加以修改或删除，从而形成非单调推理。

作为第三类需求的一个例子，现在观察一个“为三个忙人安排会议时间”的问题求解任务。解决此问题十分简单，可以先假设会议在某个具体日期举行，比如星期三，并将此假设作为部分解答存储于知识库；然后再查这一天三人是否有相容的会议参与时间，若出现冲突，则取消此假设，改用另一天作为假设的会议日期，这显然是一个非单调推理，可用带回溯的树搜索来处理。整个搜索过程可视为一个约束满足问题求解，需要满足的约束有两个：会议举行时三个忙人都有空，会议举行时具有可用的会议室。问题的求解可从决定会议举行日期开始，由于不存在某天最好的理由，可随机决定星期三。然后检查前一个约束能否满足，结果发现三个忙人每天下午两点均有时间参加会议。再检查后一个约束能否满足，不巧得很，星期三下午两点无会议室可供使用。由此，推理失败，回溯到选日期的决策点，并撤消星期三举行会议的假设和部分解答“下午两点有时间”。继而作新假设“会议于星期二举行”，再重新推理，直至结果成功。

仔细观察这个非单调推理中的回溯过程，不难发现存在不合理性，即每天下午两点三个忙人均有时间并不受日期选择的约束，但简单的推理回溯却把这个本应保留的部分解答取消了，使得选定星期二为会议举行日期后不得不重量新推出该部分解答，造成浪费。显然，若能区分出依赖于假设（该例中的星期三）的推理结果和不依赖于假设的推理结果（该例中的下午两点）就行了，这样推理回溯时只需撤消假设，而保留会议可在下午两点举行的部分解答。通过从属制导或称从属导向（Dependency-directed）回溯，可以实现这种合理性。

非单调推理比单调推理难以处理得多。因为当一个假设被发现错误而撤消时，一系列基于它的推理结果都要撤消。所以，设计非单调推理系统的另一个关键问题在于防止系统花费过多的时间在这种处理上。

对非单调推理，人们已开发了多种形式化方法。其中，最基本的也易于实用化的方法有：封闭世界假设、谓词完备和缺省推理。

（1）**封闭世界假设。**封闭世界假设（CWA，Closed World Assumption）是一种对由某基本信念集Δ定义的一个理论τ(Δ)作完备化的方法。我们说理论τ(Δ)是完备的，意指其（显式或隐式）包含了每一个基础原子公式或其取反。作为一种简化，可以限定基础原子公式为只包含常量项的谓词公式，且常量必须在该理论中出现过。例如理论|P(A)，P(A)⇒Q(A)，P(B)|是不完备的，因为对于常量B，无论是Q(B)或¬Q(B)均不在该理论中，只有将这两者之一加入才使该理论完备化。封闭世界假设仅通过增加基础原子公式的取反来实现理论的完备化。换言之，若一个基础原子公式不能经由逻辑推理从基本信念集Δ导出，就将其取反作为Δ的扩充。显然，封闭世界假设是非单调的，因为一旦以后有新的基础原子公式加进Δ，则为完备τ(Δ）而生成的扩充集就必须收缩（删除该基础原子公式的取反）。

由于为完备τ(Δ)而生成的扩充集中的每个基础原子公式的取反均是假设的暂时信念，故记该扩充集为Δasm。对于一个基础原子公式 P（省略其常量项），封闭世界假设定义：¬P ∈ Δasm，当且仅当 P ∉ τ(Δ)。记经由封闭世界假设方法完备的理论为CWA(Δ)，其扩大了τ(Δ)的推理能力，允许不能从Δ导出的结论φ可从Δ∪Δasm导出。上例中Δasm={¬Q(B)}，CWA(Δ) = τ(Δ∪Δasm)。

然而，封闭世界假设方法并不确保被完备的理论CWA(Δ)是一致的。作为典型例，令Δ = {P(A)∨P(B)}，则Δasm = {¬P(A)，¬P(B)}；显然Δ∪Δasm，进而CWA(Δ)是不一致的，其导致P(A)∨P(B)为假。解决不一致性是非单调推理的重要议题。下述定理试图解决其不一致性。

定理 17.2.9.1：CWA（Δ）是一致的，当且仅当对于每个可由Δ推导出的子句 $L_1 \vee L_2 \vee \cdots \vee L_n$，都至少存在一个 L_i 可从Δ推导出；其中 L_i 均为正文字（对应于无取反符的基础原子公式）。

然而测试上述定理的条件是困难的，因此，引入下述推论很有意义。

推论 17.2.9.1：若限制 D 包含的子句为 Horn 形且是一致的，则 CWA(Δ)必定一致。

Horn 子句定义为文字 A_i 的析取形式：$A_1 \vee A_2 \vee \cdots \vee A_n$，且最多只能有一个 A_i 是正文字（对应于无取反符的基础原子公式）。然而，这对于许多应用来讲,限制太强，如果实用上仅对某一特殊谓词 P 感兴趣，则可减弱定理 1 的强度，只将谓词为 P 的基础原子公式的取反加入Δasm，而 Horn 子句也减弱到仅谓词为 P 的文字满足 Horn 子句定义。例如，Δ = {P(A)∨Q(A)，P(A)∨R(A)}，Δ包含的两个子句均非 Horn 子句，但都是关于谓词 P 的广义 Horn 子句；令Δasm = {¬P(A)}，则可得到关于谓词 P 的扩充理论 CWA′（Δ），并有Δ∪Δasm |= Q(A)，Δ∪Δasm|= R(A)。我们同样可以得到关于一个谓词集Π的 CWA′（Δ），此时Δ包含的每个子句均为关于Π的广义 Horn 子句，即每个子句中最多只能出现一个其谓词属于Π的正文字。

遗憾的是,关于谓词集Π的 CWA(Δ)不能确保一致性。例如,令Π = {P},Δ = { P(A)∨Q,P(B)∨¬Q}；则有Δasm ={¬P(A)，¬P(B)}。由于Δ |= P(A)∨P(B)，故 CWA′（Δ）是不一致的。为确保 CWA′（Δ）的一致性，我们需要更为严格的关于谓词集Π的 CWA′(Δ)定义。

（2）**谓词完备。**谓词完备方法是将满足一谓词的对象限制为仅仅由Δ说明必须满足该谓词的那些。最简单的情况下，P(A)是Δ包含的唯一公式，P(A)等价于以下表达式

$$\forall x\ (x=A \Rightarrow P(x))$$

意指若 x=A，必有 P(A)，这可视为 P(x)的充分条件。谓词完备方法则假设 P{A}的必要条件也成立：

$$\forall x\ (P(x) \Rightarrow x=A)$$

这称为谓词 P 的完备公式，其使Δ中 P 的定义完备化（即 P 具有充分和必要条件）

Δ和谓词 P 的完备公式的合取称为Δ中 P 的完备，记为 COMP [Δ，P]，我们有：

$$COMP[\Delta, P] \equiv \forall x(P(x) \Rightarrow x=A) \wedge \Delta \equiv \forall x(P(x) \Leftrightarrow x=A)$$

若Δ包含两个关于谓词 P 的公式，即Δ＝{P(A)，P(B)},则有：

$$COMP[\Delta,P] \equiv \forall x(P(x) \Rightarrow x=A \vee x=B) \wedge \Delta \equiv \forall x(P(x) \Leftrightarrow x=A \vee x=B)$$

然而，当Δ是一阶谓词公式的任意集时，谓词完备十分复杂，我们只就易于处理的所谓对于谓词 P 是单一的子句集作进一步讨论。

一个子句集对于谓词 P 是单一的，如果集中每个含有关于谓词 P 的正文字的子句仅有 P 的一次出现。注意，对于 P 是单一的子句集必定是关于 P 的广义 Horn 子句集，但反向并不成立。例如，Q(A)∨¬ P(A)∨P(B）是关于 P 的广义 Horn 子句，但并非对于 P 是单一的。

设Δ是对于 P 单一的子句集，可将Δ中含有关于 P 的正文字的每个子句写成以下形式；

$$\forall y\ (Q_1 \wedge \cdots \wedge Q_m \Rightarrow P(t))$$

其中，t 指示参数项的元组：$(t_1, t_2, \cdots t_n)$，而 Q_i 是谓词非 P 的正文字；Q_i 和 t 均可包含变量，并以 y 指示这些变量的元组。

上述子句等价于：$\forall x \forall y((x=t) \wedge Q_1 \wedge \cdots \wedge Q_m \Rightarrow P(x))$，其中，x=t 是（$x_1=t_1 \wedge \cdots \wedge x_n=t_n$）的缩写．且 x_i 不与 t_j 同名（i,j=1,2,…,n)。至此，由 y 指示的变量不再出现于该子句中蕴涵式的右端，从而可将子句改写为（以∃y 取代∀y，仍保持原子句的真值）：$\forall x \exists y((x=t) \wedge Q_1 \wedge \cdots \wedge Q_n \Rightarrow P(x))$ 这样形式的子句称为正则形，并记其中蕴涵式的左端为 E_i。设想Δ包含 k 个对于 P 是单一的子句，则可表示为以下正则形：

$$\forall x(E_1 \Rightarrow P(x))$$
$$\forall x(E_2 \Rightarrow P(x))$$
$$\cdots$$
$$\forall x(E_k \Rightarrow P(x))$$

将这些正则形组合为单一蕴涵式如下：$\forall x(E_1 \vee E_2 \vee \cdots \vee E_k \Rightarrow P(x))$，建立谓词 P 的完备公式：

$\forall x(P(x) \Rightarrow E_1 \vee E_2 \vee \cdots \vee E_k)$。既然 E_i 不包含 P，可以有

$$COMP[\Delta, P] \equiv \forall x(P(x) \Rightarrow E_1 \vee E_2 \vee \cdots \vee E_k) \wedge \Delta$$
$$\equiv \forall x(P(x) \Leftrightarrow E_1 \vee E_2 \vee \cdots \vee E_k) \wedge \Delta$$

下面观察关于谓词完备的一个简例。

设Δ为：　$\forall x(Ostrich(x) \Rightarrow Bird(x)$

　　$Bird(Tweety)$

　$\neg Ostrich(Sam)$

显然，Δ对于谓词 Bird 是单一的，可以构造 Bird 的完备公式，并有

$$COMP[\Delta, P] \equiv \forall x(Bird(x) \Leftrightarrow Ostrich(x) \vee x=Tweety) \wedge \Delta$$

从 $COMP[\Delta, P]$，可以证明$\neg Bird(Sam)$。

此例通过谓词完备，建立了除 Tweety 和 Ostrich 是鸟外，不存在其它的东西是鸟的信念。既然 Sam 不是 Ostrich，就可以断定其不是鸟。

尽管在简单的情况下，谓词完备和封闭世界假设具有同样的效果，但在一般情况下它们是不同的。例如，$\Delta = \{P(A), Q(B)\}$，用封闭世界假设方法扩展出的假设$\Delta asm = \{\neg P(B))$，而谓词完备方法则扩展出假设$\forall x(P(x) \Rightarrow x=A)$。显然，两者不等价。

谓词完备方法与封闭世界假设方法一样是非单调的。如果一个新的对于谓词 P 是单一的子句加入Δ，P 的完备公式将发生变化。通常，对于满足 P 的对象的限制将减弱，因为有更多的对象满足 P。由此，某些$\neg P$ 形式的证明就不再可以获得。例如，上述关于鸟的简例中，加 $Penguin(x) \Rightarrow Bird(x)$ 到Δ，则 Bird 的完备公式变成：

$$Bird(x) \Rightarrow Ostrich(x) \vee Penguin(x) \vee x=Tweety$$

并且原先可证明的$\neg Bird(Sam)$再也不可获得了。

我们也能并行地为一个谓词集Ⅱ中的每个谓词分别建立完备公式，每个完备公式都直接导自Δ中的原始公式，互不影响。这种平行的谓词完备使我们能限制满足Ⅱ中任何谓词的对象仅是由Δ说明必须满足的那些。为了表示的方便起见，我们将前述关于 P 的正则形组合

$$\forall x(E1 \vee E2 \vee \cdots \vee Ek \Rightarrow P(x))$$

改写为　　$\forall x(M \Rightarrow P(x))$，

其中 $M \equiv E_1 \vee E_2 \vee \cdots \vee E_k$，不包含谓词 P 的出现。

现在建立关于$\Pi = \{P_1, P_2, \cdots, P_n\}$的并行谓词完备，首先为每个 P_i 建立其正则形组合：

$$\forall x(M_1 \Rightarrow P_1(x))$$
$$\forall x(M_2 \Rightarrow P_2(x))$$
$$\cdots$$
$$\forall x(M_n \Rightarrow P_n(x))$$

然后建立每个 P_i 的完备公式$\forall x(P_i(x) \Rightarrow M_i)$，再加上Δ，就可形成Δ中Ⅱ的完备 COMP（Δ，Ⅱ）。

（3）**分类层次体系和缺省推理。**事物可以按其共性分类，使得属于某类的事物拥有该类共有的典型特性，成为人之常识，除非已知该事物在某特性上异常。例如，人们都会假设随意的一只鸟会飞，除非已知它不会飞。这就是所谓的缺省推理。下面将从分类层次体系的角度介绍如何通过并行谓词完备方法的一个变种来实现缺省推理。

分类层次体系是语义网络的主干部分，通过层次分类，个体和子类可以继承类和超类的特性，除非另加特别说明。仍以鸟类为例，人们的信念集可以包括以下定义分类层次体系的公式：

$$Thing(Tweety)$$
$$Bird(x) \Rightarrow Thing(x)$$
$$Ostrich(x) \Rightarrow Bird(x)$$
$$Flying\text{-}Ostrich(x) \Rightarrow Ostrich(x)$$

这仅是整个信念集的一部分，以子集ΔH 指示。Δ的另一部分将描述该分类层次中对象的特性，以子集ΔΠ指示。例如，人们会说除了鸟以外无东西会飞，除鸵鸟以外的鸟都会飞，除会飞的鸵鸟外所有鸵鸟均不能飞。于是ΔΠ将包括以下公式：

Thing(x)∧¬Bird(x) ⇒ ¬Flies(x)

Bird(x)∧¬Ostrich(x) ⇒ Flies(x)

Ostrich(x)∧¬Flying-Ostrich(x) ⇒ ¬Flies(x)

Flying-Ostrich(x) ⇒ Flies(x)

ΔΠ中的前三条演绎规则都在左端插入了子类否定文字来指示规则中的例外，如 ¬Bird(x)，¬Ostrich(x)和 ¬Flying-Ostrich(x)。如资格限定问题一样，实际上这些规则均可包含任意多的例外。插入否定文字的方式不便于计算机操作，因为每当发现新的例外时都要修改规则。一个较好的方式是设立指示异常的谓词。例如，可以用谓词 Abl；指示对于一般东西都不会飞的异常。于是ΔP 中的第一条规则可改写为.

Thing(x)∧¬Abl(x) ⇒ ¬Flies(x)

为了表示鸟类是例外，只须说明其满足 Abl： Bixd(x) ⇒ Abl(x)。这称为继承取消规则，其与分类规则 Bird(x) ⇒thing(x)联合使用，意指尽管 Bird 是 Thing 的子类，但因其是异常子类而不继承“不会飞”的特性。显然，人们可以增加任意多的继承取消规则，而不必修改上述改写后的第一条规则。

继承取消规则通过说明子类异常来封锁特性继承，不仅能以不同的谓词名指示不同的异常，也促进例外在分类层次体系的不同部位的处理。我们可以改写ΔP 的第二、三条规则：

Bird(x)∧¬Ab2(x) ⇒ Flies(x)

Ostrich(x)∧Ab3(x) ⇒ ¬Flies(x）

并增加相应的继承取消规则：

Ostrich(x) ⇒ Ab2(x)

Flying-Ostrich(x) ⇒ Ab3(x)

现在，我们可以在ΔH 中包括以下规则来定义分类层次体系：

Flying-Ostrich(x) ⇒Ostrich(x)

Flying-Ostrich(x)⇒Ab3(x)

Ostrich(x)⇒Bird(x)

Ostrich(x)⇒Ab2(x)

Bird(x)⇒Thing(x)

Bird(x)⇒Ab1(x)

Thing(Tweety）

将并行谓词完备方法作用于ΔH的谓词集：

{Ab1，Ab2，Ab3，Flying－Ostrich，Ostrich，Bird，Thing}，可获得以下完备公式：

Thing(x) ⇒Bird(x)∨x=Tweety

Bird(x) ⇒ Ostrich(x)

Ostrich(x) ⇒ Flying-Ostrich(x)

¬Flying-Ostrich(x)

Abl(x) ⇒ Bird(x)

Ab2(x) ⇒ Ostrich(x)

Ab3(x) ⇒ Flying-Ostrich(x)

其中，¬Flying-Ostrich(x)来自特别的谓词完备处理。鉴于ΔH 中不存在对于谓词 Flying-Ostrich 是单一的子句（两个相关的子句中 Flying-Ostrich(x)均为否定文字），谓词完备方法假设无任何

对象满足该谓词，并表示为∀x(F ⇒Flying-Ostrich(x))，相应的完备公式则为∀x(Flying-Ostrich(x) ⇒ F)，　　即　∀x(¬Flying-Ostrich(x))。

在这个分类层次体系中，仅有对象 Tweety 被提及是个东西。由此，通过并行的谓词完备，我们设想除了 Tweety 外没有其它东西，更没有个体的鸟、鸵鸟和会飞的鸵鸟存在，也不存在任何具体的异常。从而，我们可以得出结论¬Flies(Tweety），其在首先证明¬Flying-Ostrich(Tweety)，继而¬Ostrich(Tweety)，¬Bird(Tweet)和¬Abl(Tweety)后得出。

如果加新的信念 Bird(Tweety)进 ΔH，则上述第二个完备公式将改变为：

Bird(x) ⇒ Ostrich(x)∨x=Tweety

这时，我们不再可能证明¬Abl(Tweety)，但仍能证明¬Ab2(Tweety)，从而得出结论 Flies(Tweety)。可见，只要推理系统获悉关系到异常子类的任何对象，就会使适当谓词的完备公式重新建立，进而导致推理结论的改变，如从¬Flies(Tweety）改变为 Flies(Tweety）。

由于这里所用的谓词完备方法仅面向Δ的子集，我们称其为有界完备，以区分对整个Δ的谓词完备。有界完备典型地产生比对整个Δ的谓词完备更强的假设（已知的知识越少，谓词完备产生的限制就越强），往往强的假设更有利于促进推理的效率。不足之处在于，有界完备往往会引起不一致性，需谨慎使用。这使解决不一致性和设计更好缺省推理方法的研究成为非单调推理需要进一步研究的一个方面。

17.2.9.3 真值维持系统

实现非单调推理系统的核心技术之一是维持推理的一致性，并在出现矛盾（不一致）时及时加以解决。我们可以把一个非单调推理系统的信念集（常识集）分为两个部分，即 D = Δ∪A，Δ为前述的基本信念集，而 A 为假设集，其被视为对Δ的尝试性扩充。鉴于推理系统视Δ为永真，故而推理中产生的不一致仅由引入不适合的假设而引起。尽管关于非单调推理形式方法的研究已对确保 A 与Δ的一致性作了许多探索，但大多仅适用于特别的应用场合，尚不存在适用于一般应用域的简便方法。所以，现行的实用化非单调推理系统，主要依赖于应用的特点和有关知识，提出不保证与Δ一致的试探性假设（实际上Δ往往也是问题求解过程中逐步积累起来的，即使用形式化方法也不可能确保提出的假设不与以后才加入Δ的信念冲突）。真值维持系统（TMS，Truth Maintenance System）正是服务于维持推理一致性的有效技术。

真值维持系统最早由多伊尔（J.Doyle）提出，从此，有大量的研究工作集中在了真值维持领域。真值维持系统的作用在于协助问题求解系统维持推理过程的正确性，而非自身产生新的推理。从而，一个非单调推理系统应由两个部件组成：问题求解器和真值维持系统。前者基于应用领域的知识进行推理和计算，后者则通过真值维持确保推理上下文的一致性。实际上，可视真值维持系统为从属于问题求解器的子系统，它执行两个主要功能：检查推理上下文的一致性；通过从属制导回溯消除发现的不一致性。

麦克阿伦斯特（D.McAllester）对真值维持技术进行了革新，将真值维持视为建立数据从属（Data-dependency）网并维持其一致性。数据从属意指一个命题的真假依赖于其它命题的真假。麦克阿伦斯特称基于数据从属网的真值维持系统为理由维持系统（RMS，Reason Maintenance System)。

1．数据从属和理由维持

所谓理由维持，就是要保持推理结论和推理依据的一致性，并以推理依据作为得出结论的理由。在非单调推理情况下，理由维持十分必要。例如，演绎数据库（知识库）中有正向规则

（ISA ?x DOG）⇒（ISA ?x MAMMAL）

设问题求解器提出一假设（ISA FIDO DOG），则经由正向链推理，可得到（ISA FIDO MAMMAL)；这两个命题都加入数据库。然而，在以后的推理中，问题求解器发现（ISA FIDO DOG）非真，则必须撤消。但仅撤消它仍不能维持数据库的一致性，还必须将依据该失败假设导出的结论（ISA FIDO MAMMAL）也撤消才行。这就是理由维持。数据从属可以作为实现理由维持的一个重要技术。

数据从属记载演绎数据库中元素（规则、事实、假设、结论等）之间的相互依存（即从属）关系--一些元素的真值取决于其它元素的真值。比如上例中，（ISA FIDO MAMMAL）的真值就取决于一规则和一假设。由于记载了从属关系，当假设（ISA FIDO DOG）撤消时，易于找到推理结论（IS FIDO MAMMAL）将其撤消。反之，若后者在推理过程中引起矛盾，发现 FIDO 非 MAMMAL，则也可由从属关系找到矛盾根源--该假设非真，从而一并撤消此二元素，以维持演绎数据库的一致性。

（1）**数据从属子句** 保持数据从属轨迹的基本数据结构称为数据从属子句（DDC，Data Dependency Clause），其表示为文字 x_i 的析取形式：DDC := $x_1 \vee x_2 \vee \cdots \vee x_n$

$x_i = p_i | \neg p_i$，$i=1, 2, \cdots, n$

其中 p_i 是命题。子句相互关联，形成数据从属网，记为 DDNET：

DDNET := DDC1∧DDC2∧…∧DDCm；

引入图表示法将有助于理解和设计数据从属算法。令方块形节点指示数据从属子句；圆形或椭圆形节点指示命题，称为 DDN；数据从属子句节点通过称为 DDL 的链与子句包含的每个 DDN 关联。DDL 带有标签（T，NIL）指示关联的命题是否带取反符。DDN 本身也有标签，但意义不同于 DDL，用于指示命题的当前真值：T 为真，NIL 为假。

（2）**标记数据从属网** 显然，DDL 和其关联的 DDN 标签可以不一致，而且不一致时，相应的数据从属子句仍能保持为真。只有当一个数据从属子句之所有 DDL 均与关联的 DDN 不一致时，数据从属子句才为假，即数据从属子句成为矛盾子句。

一个 DDL 若其标签与关联的 DDN 标签相同，则是调准的（Aligned），若标签相反，则是失调的；若 DDN 尚未加标签，则未调准。由于 DDNET 是数据从属子句的合取，所以只有在每个数据从属子句均非矛盾的情况下，网络才是一致的。

(a) ¬(ISA FIDO DOG) ∨(ISA FIDO MAMMAL)

(b) (A∨¬B)∧(¬A∨B)

(c) 由单一命题 A(ISA FIDO MAMMAL) 构成的数据从属子句

只有一个 DDN 的数据从属子句称为前提子句，该 DDN 的标签必须与 DDL 标签一致（DDL 是调准或未调准的），才不会引起矛盾。因此，对于这类子句的处理成为数据从属算法的关键。显然，在子句众多的情况下，人为维持 DDNET 一致性是很困难的。这可交由理由维持系统 RMS 去执行，DDN 标签的设置均由 RMS 负责，RMS 的职责就是在增删 DDC 时维持 DDNET 的一致性，尽可能避免矛盾，并在矛盾不可避免发生时记载矛盾，以供问题求解器处理。

（3）**从新子句传递标签。**当一个新的数据从属子句插进网络时，或许会迫使某些 DDN 的标签确定为 T 或 NIL（DDN 初始建立时以 OUT 指示未确定真值），特别是一个前提子句插进时，其包含的唯一 DDN 必定强制为关联 DDL 的标签（T 或 NIL）。注意：每个命题（即 DDN）在 DDNET 中是唯一的，可为多个数据从属子句所共享；相反，DDL 则是数据从属子句专用的，不能为其它子句所共享。由此，DDNET 中随着子句的增加，对各 DDN 标签的约束渐增，直至迫使某些 DDN 确定其标签，甚至引起矛盾。

标签传递实际上是指 DDN 标签的连锁确定过程。显然，只有在一个数据从属子句中仅剩下一条非失调 DDL 时，才会发生标签传递。我们称该 DDL 为数据从属子句的活动链，关联的 DDN 为活动节点，而这样的数据从属子句为活动子句。当加一新数据从属子句 C 到网络时，标签传递算法如下：

① 若该数据从属子句的所有 DDL 都失调，则其是矛盾子句，置于矛盾子句表，中止算法并转入矛盾处理；

② 若该数据从属子句有多于一条 DDL 非失调，则不可能产生标签传递，中止算法；

③ 否则以该数据从属子句和活动链为参数调用链调准算法。

链调准算法如下：

① 若活动链已调准，则中止算法；

② 否则，将与活动链关联的 DDN（其一定是未调准的）调准到具有与活动链相同的标签；

③ 对于有 DDL 与该 DDN 关联的每一个其它数据从属子句，调用标签传递算法。

注意，RMS 本身不处理矛盾，仅把矛盾子句置于专门的表中，交由问题求解器处理。该算法存在的缺点是不完备。从算法可见，只有前提子句加进网络时才会迫使其唯一命题节点确定标签，继而引起标签传递。其它子句加进网时，可能不会强迫标签的确定和传递，从而即使引起矛盾，也不会当即发现。不过这种不完备是有意留下的，否则算法将十分昂贵。再说，随着前提子句的逐个加入，会很快发现矛盾。

不过，不能认为矛盾子句就是网络不一致的根源，应进一步寻找矛盾的根源。作为应用数据从属技术的建议，应保证除前提子句外的所有子句组成的网络保持一致性。这样，本算法就完全够用了，而且许多实际应用域常满足这一限制。

（4）**撤消子句和解除节点标签。**一旦发现矛盾，应设法把引起矛盾的根源（某子句，常是前提子句）从网络中删除；而子句的删除将会放宽对 DDN 上标签的约束，从而消除矛盾，甚至解除强加于某些 DDN 的标签。与标签传递算法对偶，解除节点标签也应以活动链为线索，这也是一个连锁反应过程。

当撤消一个数据从属子句时，标签解除传递算法如下：

① 若数据从属子句的所有链均失调，则其是矛盾子句（已由上述标签传递算法置于矛盾子句表），将其从矛盾子句表删除；

② 若该数据从属子句有一活动链，则调用节点标签解除算法对该活动链关联的 DDN 作处理，然后对于返回结果--一个子句集 S 中的每个数据从属子句调用标签传递算法，作节点标签恢复工作。

节点标签解除算法；

① 若该 DDN 已解除标签，则中止算法；

② 否则，将具有失调链与该 DDN 关联的所有活动子句的活动节点（每个活动子句都有一活动节点）收集到表 L 中；

③ 解除该 DDN 的标签；

④ 对于 L 中的每个节点递归调用节点标签解除算法，收集返回的子句集，并把它们合并为集合 S；

⑤ 将具有活动链与该 DDN 关联的所有子句加入子句集 S，并将 S 作为返回结果。

以上算法中提到要作节点标签恢复工作，原因在于往往存在多个活动子句同时对同一 DDN 强加同样标签的情况。这样，其中一子句的删除并不会解除该节点的标签。但本算法未考虑这种情况，不加区分地一律解除节点标签，所以有必要将其它关联到该 DDN 的活动子句收集起来，调用标签传递算法对它们中的每一个作处理，以恢复不该解除的节点标签。

2．从属制导回溯

从属制导回溯意指，无论何时理由维持系统发现矛盾，都可依据数据从属记载寻找到引起该矛盾的根源，以便推理过程直接回溯到矛盾点，而非按时序盲目回溯到上一个推理的（解答搜索）分支点。若我们确保除前提子句外，所有其它子句不会引起系统（DDNET）矛盾（许多实际问题满足这一限制），则矛盾的根源就是某些前提子句的集合；所以，不需作时序回溯，只要撤消这些前提子句中的一个或多个，就可解决矛盾。

为实现从属制导回溯，一个非单调推理系统应由三部分组成：理由维持子系统（RMS）、回溯器、问题求解器。前两部分联合起来可实现典型的真值维持系统的真值维持功能。

需以从属制导回溯解决的问题往往可以表示为可选集的集合。每个可选集提供一个部分解答，即可选集中每个元素（表示为命题）作为部分解答的候选；所以问题的解答由从每个可选集选中的单一元素（部分解答）联合组成。我们可以将可选集表示为数据从属子句，而数据从属子句包含的文字（xi）指示可选集中的元素。由此，这些数据从属子句的集合构成初始的 DDNET，表示问题求

解的初始状态。然后，问题求解器依据领域启发式知识，逐个选择各可选集中的元素，并调用理由维持子系统将指示被选中元素的前提子句加入 DDNET。由于选择元素的决策是试探性的，这些前提子句就是可以被撤消的假设。随着前提子句的逐个加入，可能会引起 DDNET 矛盾。一旦理由维持子系统发现矛盾，就调用回溯器寻找引起矛盾的根源--某些前提子句的集合，并建立所谓 Nogood 子句来指示不可能由这些前提子句指向的元素（DDN）构成问题解答。

归纳起来，上述非单调推理系统三个部分的功能如下：理由维持子系统--维持 DDNET 的一致性和记载发现的矛盾子句。回溯器--依据理由维持子系统提供的矛盾子句，寻找矛盾的根源（一组前提子句），建立 Nogood 子句并将其插入 DDNET。问题求解器--向理由维持子系统提供应用领域公理和表示可选集的子句，以建立 DDNET（它们永真，不可撤消；但可在问题求解过程中按需逐步插入 DDNET，不必在问题求解初始化时一次性提供）；决定可选集中元素的取用（以建立前提子句）；从引起矛盾的前提子句中选择适当的一个加以撤消。

下面仍以为三个忙人安排会议为例，进行基于从属制导回溯的非单调推理。建立三个可选集（以 DDCs 表示）：

日期可选集：$A_1 \vee A_2 \vee \cdots \vee A_7$（$A_i$指示日期选定星期 i）；

时间可选集：$T_1 \vee T_2 \vee T_3 \vee T_4$（$T_i$指示上午 8～10，10～12 点和下午 2～4，4～6 点这 4 个时段）；

房间可选集：$R_1 \vee R_2 \vee R_3$（指示只有 3 个房间可选用）。

先前已安排的其它会议应作为当前会议安排的约束。例如$\neg A_3 \vee \neg T_3 \vee \neg R_3$，指示星期三下午 2～4 点房间 R_3已安排了会议，其它人不得在这个时间占用房间 R_3。设想已有以下约束：

$\neg A_2 \vee \neg T_3 \vee \neg R_1$

$\neg A_3 \vee \neg T_3 \vee \neg R_1$

$\neg A_2 \vee \neg T_3 \vee \neg R_2$

$\neg A_3 \vee \neg T_3 \vee \neg R_2$

$\neg A_3 \vee \neg T_3 \vee \neg R_3$

这些约束和上述三个指示可选集的子句构成初始的 DDNET。由于不存在直观的理由选定某特别日期，问题求解器就随意选一个，如星期三，即 A_3；随即指示 A_3为真的前提子句由理由维持子系统插入 DDNET。接着，问题求解器发现，恰巧每天下午 2～4 点三个忙人都有空，于是指示 T_3为真的前提子句插入 DDNET。该子句的插入引起在上述表示约束的子句中的标签传递，并导致命题节点 R_1，R_2，R_3的标签确定为 NIL（以指示房间 R_1，R_2，R_3均不能占用）。此时，理由维持子系统发现子句 $R_1 \vee R_2 \vee R_3$成为矛盾子句，并将其反馈给问题求解器。问题求解器由此得知无房间可供会议使用，推理失败，并启动回溯器寻找矛盾（推理失败）的根源。鉴于 DDNET 中只有两个前提子句，就把这两个子句视为矛盾的根源，并建立 Nogood 子句：$\neg A_3 \vee \neg T_3$，以指示部分解答 A_3和 T_3不能同时取用。由于三个忙人每天下午 2～4 点有空，而日期不过是随意选定的，改变日期更有利于问题的解决。于是问题求解器决定撤消假设 A_3，改用假设 A_2。A_3的撤消导致 DDNET 中命题节点（R_1，R_2，R_3）标签的取消。指示 A_2为真的子句插入 DDNET 后，导致命题节点 R_1和 R_2的标签再次确定为 NIL，但 R_3未确定。接下来，R_3可以选为部分解，从而得出最后解答：星期二下午 2～4 点在 R_3房间举行三人会议。

从这个例子可见，从属制导回溯有三点不同于时序回溯：① 直接消除矛盾，不做无用的工作。假定星期三整天所有房间都已安排了会议，按时序只能回溯到时间选择，但重新选择时间就会白费劲；应直接回溯到日期选择。② 不撤消有效的工作，上例中保留了时间选择 T_3，问题求解器在重新选定日期后，不必重复推导出选择 T_3的决策。③ 决不重新尝试引起矛盾的假设。上例中由于插进了 Nogood 子句 $\neg A_3 \vee \neg T_3$，命题节点 T_3的保留导致命题节点 A_3确定为 NIL，从而修剪了可选集中的元素；使得只要 T_3为真，A_3就不可能被选，避免了不合理的失败重演。

显然，从属制导回溯的优势是以维持数据从属网为代价换取的，只有当通过时序回溯搜索解答

的工作量大于维持数据从属网的工作量时，采用从属制导回溯技术才有意义。从属制导回溯应付出的另一个代价是寻找矛盾的根源。简单情况下，把所有进入 DDNET 的前提子句的集合视为矛盾的根源，就像上面例子中处理的那样。但当前提子句集较大时，将会给问题求解器解决矛盾（决策撤消其中一子句）造成很大困难。一个较好的方法是记住命题节点被强迫确定标签的轨迹，以便精确认定究竟是哪些前提子句引起矛盾，从而得到包含文字最少的 Nogood 子句（注意，Nogood 子句是命题取反后的析取，且每个命题是导致矛盾的前提子句包含的命题）。

为了简化处理，我们限制了矛盾只能由前提子句引起，即只有它们才是假设，其它子句均为永真的公式。实际上，其它子句也可以作为假设，但这将大大增加系统设计的复杂性，因为每一子句（不仅是前提子句）加入 DDNET 都有可能引起矛盾。一个有效的解决办法是对于非前提子句形式的假设，可以附加一个额外的前提子句去断言其为真。

从属制导回溯是否在许多问题求解任务中有效，是有争议的；因为从属制导回溯需要维持数据从属网的一致性，而在网络较大的情况下，维持一致性的开销（标签传递和撤消，发现矛盾）很大。1986 年，迪克勒（J.De Kleer）提出了基于假设的真值维持系统（ATMS，Assumption Based TMS），取消了维持演绎数据库（和 DDNET）一致性的需求，并提出清晰地记载与推出某结论相关的所有假设的集合。只要相关的假设均为真，就确保推理结论为真；至于这些假设是否与其它假设矛盾，并不影响结论的真假。由此，ATMS 消除了数据从属网一致性维护的必要性，也完全免除了回溯。

17.2.10 问题求解过程中的不确定性处理

不确定推理方法大多是在确定性推理方法的基础上发展起来的，或者说是对确定性推理的扩展和改进。既然人的信念经常是不确定的，就存在关于信念强度的问题，即确定性程度到底是多少的问题。最常见的方式是把指示确定性程度的数据附加到推理规则，并由此发展了主观 Bayes 方法、确定性方法和 D－S 证据理论等；尽管这些方法提出的不确定推理常不够严谨，使用上也有许多局限性，但尚能解决一些问题，符合人认识世界的直觉，并可对推理的结果给出令人接受的解释。下面就对主观 Bayes、确定性方法和 D-S 证据理论这三种广泛使用的不确定推理方法分别加以介绍和分析。

17.2.10.1 主观 Bayes 方法

主观 Bayes 方法处理推理过程中不确定性的主要理论基础是传统概率论中的 Bayes 理论。Bayes 理论的应用要求收集大量的样品事件来作统计，以便获得事件发生的概率来表示信念的强度（确定性程度）并作相关计算。然而，在许多情况下，同类事件出现（或可采集）的频率并不高，甚至很低，无法作概率统计。例如，“某地明年要发生地震”，就无法获取其概率。不过，根据观测到的数据，地震专家可以凭经验给出一些主观上的判断，称为主观概率，并据此推断地震发生的可能性。

1．应用 Bayes 理论于不确定推理

设推理规则 $P \Rightarrow Q$ 是不确定的，其不确定性可以用条件概率 p(Q/P)表示；若已知前提 P 成立的概率 p(P)，则可求得 P∧Q 成立的概率（P，Q 联合概率）p(P，Q)＝p(Q／P)·p(P)。 依据 Bayes 理论，有以下条件概率公式：

$$p(Q/P)=[p(P/Q)\cdot p(P)]/p(P) \qquad ①$$

其中，p(P)和 p(Q)分别指示前提 P 和结论 Q 的先验概率（不依赖规则 $P \Rightarrow Q$）；p(P /Q）称为后验概率，指示结论 Q 成立时前提 P 成立的概率。值得注意的是往往后验概率比条件概率更易于获取，所以可不经由统计手段去获得条件概率，而是由公式①计算之。鉴于由征兆推测原因是常用的推理方式，许多医疗诊断、机器故障诊断、矿藏预测系统都使用基于这种方式的不确定推理。

通常，一个征兆可以由多种不同的原因引起，所以估计其先验概率比获取原因的先验概率更困难。为克服此困难，可以对 Bayes 理论作一些改进，使 p(P)不出现。首先计算条件概率 $p(\neg Q / P)$：

$$p(\neg Q / P)=1-p(Q / P)=1-[\ p(P/Q)*p(Q)]/\ p(P)=\ [\ p(P)-p(P,Q)]/\ p(P)$$

$$=\ [p(P,\neg Q)+p(P,Q)-p(P,Q)]/\ p(P)=\ p(P,\neg Q)/\ p(P)=\ p(P,\neg Q)*p(\neg Q)/\ p(P) \qquad ②$$

将公式①，②两边分别相除，得：

p(Q/P)/p(¬Q/P)= p(P/Q)*p(P)/ p(P/¬Q)*p(¬Q)

引入 Q 的先验似然比 O(Q)和条件似然比 O(Q/P)：

$$O(Q)=p(Q)/p(\neg Q)=\frac{p(Q)}{1-p(Q)}$$

$$O(Q/P)=\frac{p(Q/P)}{p(\neg Q/P)}=\frac{p(P/Q)}{p(P/\neg Q))}\cdot O(Q)$$

O(Q)意指命题（这里是结论）成立的先验概率和其不成立的先验概率之比。显然 O(Q)随 p(Q)增大而增大；极端情况下，p(Q) = 0,O(Q) = 0；p(Q) = 1，O(Q) = ∞。O(Q/P)则意指前提 P 成立情况下，Q 和¬Q 条件概率之比。令

$$LS=\frac{p(P/Q)}{p(P/\neg Q)}$$，则有　$O(Q/P)=LS\cdot O(Q)$　　③

LS 称为规则 P⇒Q 成立的充分性因子，用以指示规则强度的似然率，即 P 成立对 Q 成立的影响力；公式③则称为 Bayes 理论的似然公式，指出结论 Q 的条件似然比可以由其先验似然比和规则的充分性因子 LS 来计算。进一步可经由

$$O(Q/P)=\frac{p(Q/P)}{1-p(Q/P)}$$，即　$p(Q/P)=\frac{O(Q/P)}{1+O(Q/P)}$

来计算出 p(Q/P)。类似地，我们可以令

$$LN=\frac{p(\neg P/Q)}{p(\neg P/\neg Q)}$$

并有

$$O(Q/\neg P)=LN\cdot O(Q)$$　　④

LN 称为规则 P ⇒ Q 成立的必要性因子，用于指示 P 不成立时对 Q 成立的影响程度。下面考虑 LS 和 LN 的不同取值范围对规则强度的影响：

$$LS=\begin{cases}1, & \text{则}O(Q/P)=O(Q),\text{意指}P\text{对}Q\text{无影响}\\ >1, & \text{则}O(Q/P)>O(Q),\text{意指}P\text{在一定程度上支持}Q\\ <1, & \text{则}O(Q/P)<O(Q),\text{意指}P\text{在一定程度上不支持}Q\end{cases}$$

$$LN=\begin{cases}1, & \text{则}O(Q/\neg P)=O(Q),\text{意指}\neg P\text{对}Q\text{无影响}\\ >1, & \text{则}O(Q/\neg P)>O(Q),\text{意指}\neg P\text{在一定程度上支持}Q\\ <1, & \text{则}O(Q/\neg P)<O(Q),\text{意指}\neg P\text{在一定程度上不支持}Q\end{cases}$$

值得指出的是，LS（和 LN）的引入大大促进了 Bayes 理论在不确定推理中的应用。众所周知，无论是先验概率还是后验概率都必须基于大量的统计数据，工作量很大，而且在许多情况下不可能获得它们。由于 LS 指示的是规则强度，即前提 P 对结论 Q 的影响程度，使得在缺乏大量统计数据的情况下，有经验的专家仍然能作出近似的估计。虽然这种估计不太精确，但十分有用，因为许多实际应用场合并不需要精确计算，只须给出近似估计即可。我们把基于专家主观估计的 LS（和 LN）值而演算出来的条件概率 p(Q/P)（经由 O(Q/P)）称为主观概率，而这种推算主观概率的方法称为主观 Bayes。

从 LS 和 LN 的定义可以看出，两者并非独立，尽管它们是分别独立提供的。对 LN 的定义作适当变换（其中 p（P/Q）= LS·p（P/¬Q））：

$$LN=\frac{p(\neg P/Q)}{p(\neg P/\neg Q)}=\frac{1-p(P/Q)}{1-p(P/\neg Q)}=\frac{1-LS\cdot p(P/\neg Q)}{1-p(P/\neg Q)}$$

可见，当 LS＞1 时，LN＜1；LS＜1 时，LN＞1；LS = 1 时，LN = 1。当然，领域专家凭经验给出 LS 和 LN 时，常未意识到这种约束关系，所以不确定系统的设计者必须考虑这种约束，并要求专家作相

应修正。

2．**不确定性的传递**

前提（即导致结论的证据）的不确定性可以设想为与另一事件 P′ 有关，即 P′ ⇒ P，则有 P′ ⇒ P ⇒ Q。若 P′ 为确定性真（广义地 P′ 指示 P 所处的环境），则由 P 的不确定性而引起的 Q 为真的概率可以用 p(Q / P′)来表示。变换 p(Q / P′)：

$$\begin{aligned} p(Q/P') &= p(Q,P')/p(P') \\ &= \frac{p(Q,P,P')+p(Q,\neg P,P')}{p(P')} \\ &= \frac{p(Q,P,P')\cdot p(P,P')}{p(P')\cdot p(P,P')} + \frac{p(Q,\neg P,p')\cdot p(\neg P,P')}{p(P')\cdot p(\neg P,P')} \\ &= p(Q/P,P')\cdot p(P/P') + p(Q/\neg P,P')\cdot p(\neg P,P') \end{aligned} \qquad ⑤$$

注意到 p(Q / P，P′)和 P(Q / ¬P，P′) 是 P′ 为真而 P 为确定性真或假条件下 Q 为真的概率；鉴于 P′ 是通过 P 去影响 Q 的，而这里 P 已是确定性真或假，就可忽略 P′ ，从而有

$$p(Q/P') = p(Q/P)\cdot p(P/P') + p(Q/\neg P)\cdot p(\neg P/P') \qquad ⑥$$

由前所述，p(Q/P)和 p(Q / ¬P)可从公式③和④算出；类似地，只要给出 p(P)，就可以求得 p(P / P′)，进而 p(¬P / P′)= 1 - p(P / P′)；由此容易求出 p(Q / P′)，即 P 不确定情况下 Q 为真的概率。

公式⑥称为不确定性传递公式，其有一个有意义的特性，即当 p(P / P′) = p(P)时，p(Q / P) = p(Q)（可从公式⑥推出）。这完全符合人的直觉：既然无关于前提 P 真实程度的任何信息（仅有先验概率 P(P)），也就不可能从规则 P ⇒ Q 获得关于结论 Q 真实程度的任何信息（仅有先验概率 p(Q)）。

不确定性的传递可以有更长的路径，例如　P′ ⇒ P ⇒ Q ⇒ W

则有　p(W / P′) = p(W / Q)•p(Q / P′)十 p(W / ¬Q)•p(¬Q / P′)

回到公式⑥。考虑到先验概率 p(P)和 p(Q)往往只能由领域专家凭主观经验给出，LS 和 LN 更是领域专家给出的近似估计，所以当 p(P / P′) = p(P)时，按公式⑥计算出来的 p(Q / P′)与领域专家给出的 p(Q)常不一致。为避免这种不一致而引起的矛盾，主观 Bayes 方法建议采用分段线性插值的手段来计算 p(Q / P′)的实际值。相应地，计算公式⑥改变为

$$p(Q/P') = \begin{cases} p(Q/\neg P) + \dfrac{p(Q)-p(Q/\neg P)}{p(P)}\cdot p(P/P') \\ \qquad 0 \le p(P/P') < p(P) \\ p(Q) + \dfrac{p(Q/P)-p(Q)}{1-p(P)}\cdot (p(P/P')-p(P)) \\ \qquad p(P) \le p(P/P') \le 1 \end{cases}$$

3．**不确定性的组合**

常常会遇到多个前提（证据）支持同一结论的情况，可表示为：

$P_1' \Rightarrow P_1 \Rightarrow Q$

$P_2' \Rightarrow P_2 \Rightarrow Q$

…　…

$P_n' \Rightarrow P_n \Rightarrow Q$

设想 P_i' 相互独立，从而 P_i 也相互独立。首先考虑只有两个前提的情况，类似于公式⑤的推导(P 以 P_2 取代　以 P_1'，P_2' 取代)，有

$$p(Q/P_1', P_2') = p(Q/P_2, P_1', P_2')\cdot p(P_2/P_1', P_2') + p(Q/\neg P_2, P_1', P_2')\cdot p(\neg P_2/P_1', P_2')$$

显然，P_2 独立于 P_1'，且 P_2 为确定性真或假情况下，可以忽略 P_2' 的影响（P_2' 通过 P_2 影响 Q），故上式可化简为：

$p(Q/P_1', P_2') = p(Q/P_2, P_1') \cdot p(P_2/P_2') + p(Q/\neg P_2, P_1') \cdot p(\neg P_2/P_2')$　⑦

作以下演算

$$p(Q/P_2, P_1') = \frac{p(Q, P_2, P_1')}{p(P_2, P_1')} = \frac{p(P_2/Q, P_1') \cdot p(Q/P_1') \cdot p(P_1')}{p(P_2, P_1')}$$

$$p(\neg Q/P_2, P_1') = \frac{p(\neg Q, P_2, P_1')}{p(P_2, P_1')} = \frac{p(P_2/\neg Q, P_1') \cdot p(\neg Q/P_1') \cdot p(P_1')}{p(P_2, P_1')}$$

此二式两边相除，有

$$O(Q/P_2, P_1') = \frac{p(Q/P_2, P_1')}{p(\neg Q/P_2, P_1')} = \frac{p(P_2/Q, P_1')}{p(P_2/\neg Q, P_1')} \cdot \frac{p(Q/P_1')}{p(\neg Q/P_1')} = \frac{p(P_2/Q)}{p(P_2, \neg Q)} \cdot O(Q/P_1') = LS2 \cdot O(Q/P_1')$$

类似地可得

$$O(Q/\neg P_2, P_1') = \frac{p(\neg P_2/Q)}{p(\neg P_2/\neg Q)} \cdot O(Q/P_1') = LN2 \cdot O(Q/P_1')$$

鉴于可从公式⑥（以 P_1' 取代 P'）计算 $p(Q/P_1')$，再算出 $O(Q/P_1')$；只要给出 LS2 和 LN2，就可算出 $O(Q/P_2, P_1')$ 和 $O(Q/\neg P_2, P_1')$，进而得到 $p(Q/P_2, P_1')$ 和 $p(Q/\neg P_2, P_1')$ 由此，再给出 $p(P_2/P_2')$，即可得到 $p(Q/P_1', P_2')$。公式⑦展示的两个前提相互独立地支持同一结论的不确定性组合方法，可以很容易推广到 n 个前提相互独立地支持同一结论的情况。

4．在推理网络中传递不确定性

不确定推理在基于规则的专家系统中具有重要地位，因为许多实际问题中规则都具有不确定性。这些系统中往往存在许多中间结果，它们既是这些规则的结论又是另一些规则的前提，从而形成由规则构成的推理网络。

为在推理网络中传递不确定性，应首先搜索推理路径，然后沿推理路径从网络末端节点向目标节点传递概率计算。探矿系统 Prospector 可以作为应用推理网络传递不确定性的典型示例。

17.2.10.2 确定性方法

在原始证据具有相互独立性，并能提供精确且一致的主观概率数据情况下，主观 Bayes 方法可令人满意地处理不确定推理。遗憾的是实际问题求解中，这些条件常不能满足；而且当需提供的主观概率数据增多时，处理概率数据的不一致性任务繁重。其实，许多问题并不要求概率计算有较高的精度，如医疗、气象预报、探矿等，只须粗略地估计推出结论的可信程度即可。例如，为了更方便地处理不确定推理，医疗系统 MYCIN 提出了比较简单、直观且易于掌握和使用的确定性方法模型。尽管该方法未建立在严格的理论推导基础上，但对于许多应用领域，仍可以得到比较合理的和令人满意的结果。

1．规则中不确定性的表示方法

MYCIN 提出的确定性方法表示推理规则为如下形式：

$$E \Rightarrow H,\ CF(H,E)$$

规则前提 E 可以是命题的合取和析取组合，结论 H 为单一命题；CF(H, E)为确定性因子，简称 ，用以量度规则的确定性（可信）程度。令

CF(H, E) = MB(H, E) - MD(H, E)

其中 MB(H, E)指示信任量度，MD(H, E)指示不信任量度，分别定义如下：

$$MB(H,E)=\begin{cases}1 & 若p(H)=1\\ \dfrac{\max[p(H/E),p(H)]-p(H)}{\max[0,1]-p(H)} & 其它情况\end{cases}$$

$$MD(H,E)=\begin{cases}1 & 若p(H)=0\\ \dfrac{\min[p(H/E),p(H)]-p(H)}{\min[1,0]-p(H)} & 其它情况\end{cases}$$

从这些定义中可见，对于特定情况下的证据 E，其可以增加或减少结论 H 存在的可能性，但不可兼有。当 p(H/E)＞p(H)时，表示证据 E 支持结论 H，则有 MB>0，MD=0；反之，当 p(H/E)<P(H)时，表示 E 不支持 H，则有 MB=0，MD>0；当 p(H/E)＝p(H)时，表示 E 对 H 无影响，则有 MB＝MD＝0。值得注意的是，可信度 CF(H, E)(即 MB, MD)的值通常并不是经由 p(H/E)和 P(H)来计算的，而是在建立规则库时由领域专家凭经验主观确定的；再有，CF(H, E)指示的是增量[p(H/E)-p(H)]对[1-p(H)](MB>0) 和 p(H) (MD>0 时) 的比值。显然,CF=1 指示 E 确定性地导致 H 为真，即 p(H/E) = 1；CF=-1 指示 E 确定性地导致 H 为假，即 p(H/E)=0，或 p(¬H/E)=1 ；CF = 0 指示 E 对 H 无影响，即对应于 MB = MD = 0。

2．证据中不确定性的表示方法

设证据 E 所在的环境为 E′ ，则可用可信度 CF(E, E′)来表示 E 在 E′ 下的确定性程度，并有：

$$CF(E,E')=MB(E,E')-MD(E,E')$$

若 E′ 下 E 为真，则 CF(E, E′) = 1；E 为假，则 CF(E, E′) =-1；若 E′ 对 E 的真值无影响，则 CF(E, E′)= 0。类似于规则的不确定性，证据的可信度往往可由领域专家凭经验主观确定。

3．不确定性的传递

在证据 E 具有不确定性的情况下，规则 E⇒H 的结论 H 的不确定性可以通过不确定性的传递来计算。相应的推理过程记为 E′ ⇒ E⇒H，而结论 H 在环境 E′ 下的可信度记为 CF(H, E′)。

不确定性的传递算法定义如下：

CF(H, E′)= CF(H,E) • max [0，CF(E, E′)]

=（MB(H,E)- MD(H,E)）• max[0，CF(E, E′)]

4．不确定性的组合

在环境 E′ 下，若两个证据的合取或析取支持结论 H，则可表示为

$$E'\Rightarrow E_1\wedge E_2\Rightarrow H或E'\Rightarrow E_1\vee E_2\Rightarrow H$$

证据的不确定性组合定义为

CF(E_1∧E_2, E′) = min[CF(E_1, E′), CF(E_2, E′)]

CF(E_1∨E_2, E′) = max[CF(E_1, E′), CF(E_2, E′)]

当两条规则支持同一结论 H 时，可表示为

$$E_1\Rightarrow H$$

$$E_2\Rightarrow H$$

假定 E_1和 E_2相互独立，那末规则的不确定性组合可定义为（以＆指示 E_1和 E_2相互独立地支持 H）

$$CF(H,E_1\&E_2)=\frac{MB(H,E_1\&E_2)-MD(H,E_1\&E_2)}{1-\min[MB(H,E_1\&E_2),MD(H,E_1\&E_2)]}$$

$$MB(H,E_1\&E_2)=\begin{cases}0 & 若MD(H,E_1\&E_2)=1\\ MB(H,E_1)+MB(H,E_2)-MB(H,E_1)\cdot MB(H,E_2) & 其它情况\end{cases}$$

$$MD(H,E_1\ \&E_2)=\begin{cases}0 & 若MB(H,E_1\ \&E_2)=1\\ MD(H,E_1)+MD(H,E_2)-MD(H,E_1)\cdot MD(H,E_2) & 其它情况\end{cases}$$

以上定义可以推广应用到多条规则支持同一结论，且规则前提包含多个证据的场合。通常可以分别综合对于结论 H 的信任量度 MB(H)和不信任量度 MD(H)，并记 H 的可信度为 CF(H)，则有：

$$CF(H)=\frac{MB(H)-MD(H)}{1-\min[\ MB(H),MD(H)]}$$

5．方法的应用与不足

MYCIN 的不确定推理构成与或推理树，规则的前提是命题（证据）的合取形式（更一般地，合取和析取的组合），规则之间隐含析取关系。MYCIN 取逆向穷尽推理方式，即从一个假设的结论出发，追索支持结论的证据；若这些证据又是其它规则的结论，则继续逆向追索，直到可由用户直接提供的原始证据为止。当推理树较大时，由可信度的不精确估计（领域专家凭主观经验给出可信度）而产生的积累误差会很大。例如，一个结论由许多规则支持，而经由这些规则到原始证据的路径又较长，则即使每个证据都只有微弱的正面影响，也会因误差的积累导致该结论的可信度达到 1。为抑制微弱证据引起的不良影响，MYCIN 为规则前提指定可信度的门槛值为 0.2，不足者不能激活规则。

为了表示和计算的方便，我们把当前环境下，推理树中某节点 X 的实际可信度简记为 CF(X)，并有：

CF(X) = MB(X) - MD(X)

尽管确定性方法使 MYCIN 和其它一些专家系统能简单有效地实现不确定性推理，但仍存在不少问题。现归纳如下：（1）如何将人表示可信度的术语转变为数字化的 CFs。例如，人的经验规则常涉及"很可能"、"不大可能"等术语，应对应到多大的 CF 值。（2）如何规范化人们对可信度的估计，不同人所作的估计往往相差较大。（3）为防止积累误差，需指定门槛值，但多大合适呢？太小固然不行，但太大也不好，因为可信度的传递需要累计较小的变化。（4）为改进可信度的精确性，需提供从系统的实际执行反馈的信息，并基于反馈信息调整可信度。这实际上是一种机器学习问题，尚未较好地加以解决。正因为这些问题的存在，限制了 MYCIN 提出的确定性方法只能用于对不确定推理的精度要求不高的场合。

17.2.10.3 D-S 证据理论

D-S 证据理论是由丹普斯特（Dempster）提出，并由他的学生莎弗（Shafer）改进的一种不确定推理模型。该理论引入信任函数而非采用概率来量度不确定性，并引用似然函数来处理由不知道而引起的不确定性，从而在实现不确定推理方面显示出很大的灵活性，受到人们的重视。

1．假设中的不确定性描述

为了描述假设的不确定性，D-S 证据理论首先引入关于基本概率分配函数的概念。设论域 U 为所有可能假设（表示为原子命题的结论）的有限集合，且 U 中的元素间是互斥的，则可以在 U 的幂集 2^U 上定义一个基本概率分配函数 m：$2^U \rightarrow [0,1]$，满足

$$m(\Phi)=0,\sum_{A\subseteq U}m(A)=1$$

数值 m(A)称为基本概率，它表示依据当前环境（证据）对假设集 A（U 的子集）的信任程度。

注意，m 是 2^U 上而非 U 上的概率分布，所以基本概率 m(A)不必等于概率 p(A)，而且 $m(A)\neq 1-m(\neg A)$。

下面定义描述证据不确定性的三个函数：信任函数、似然函数和类概率函数。

(1)信任函数 Bel：

Bel：$2^U \rightarrow [0,1]$

对于任意的 $A\subseteq U$，有

Bel(A)= $\sum_{B\subseteq A} m(B)$

Bel(A)表示当前环境下，对假设集 A 的综合信任程度，其为 A 所有子集的基本概率之和。易知 Bel(Φ) = 0，Bel(U) = 1；当 A 为单一元素集时，Bel(A) = m(A)。

在上例中，若再知道 m({兰}) = 0.2, m({绿}) = 0；则

Bel({兰，绿})= m({兰}) + m({绿}) + m({兰，绿})= 0.2 + 0 + 0.3 = 0.5。

(2)似然函数 Pl：

Pl：$2^U \rightarrow [0,1]$

Pl(A)= 1 - Bel $(\neg A) = \sum_{B\cap A\neq\Phi} m(B)$，　$A\subseteq U, B\subseteq U$，

其中，Pl(A)表示对于不否定 A 的信任程度，它是所有与 A 相交的子集的基本概率之和。

似然函数有以下特性：　Pl(A) ≥ Bel(A)　Pl(Φ) = 0　Pl(U)= 1

Pl(A) - Bel(A)表示既不信任 A 也不信任¬A 的程度，即对于 A 是真是假不知道的程度。可以用区间[Bel(A)，Pl(A)] 来综合描述 A 的不确定性。易知存在三个特殊的区间：

[Bel(A)，Pl(A)] = [1, 1]，表示信任 A 为真；

[Bel(A)，Pl(A)] = [0, 0]，表示信任 A 为假；

[Bel(A)，Pl(A)] = [0, 1]，表示对 A 是真是假一无所知。

（3）假设集 A 的类概率函数 f(A)：

$$f(A)=Bel(A)+\frac{|A|}{|U|}\cdot[Pl(A)-Bel(A)]$$

其中，|A|、|U|分别指示 A 和 U 中包含元素的个数。f(A)有如下性质：

f(Φ)= 0，　f(U)＝1

Bel(A) ≤ f(A) ≤ Pl(A)，对于任何 $A\subseteq U$

2．规则中的不确定性描述

定义具有不确定性的规则形如

$E \Rightarrow A$，CF

其中，E 为支持 A 成立的证据集；A={a_1，a_2，…，a_m}，a_i∈U（i＝1，2，…，m），A 为假设集 U 的子集；CF={C_1，C_2，…，C_m}，用于指示前提 E 成立时假设 a_i 成立的可信度。

3．证据中的不确定性描述

证据 E 的不确定性可以用类概率函数 f(E)表示，原始证据的 f(E)应由用户给定，作为中间结果的证据则由下面的不确定性传递算法确定。

4．不确定性的传递

对于上述具有不确定性的规则，定义

$m(\{a_i\}) = f(E)\cdot C_i$　　(i = 1,2,…,m)

或缩简记为

$$m(\{a_1\},\{a_2\},\Lambda,\{a_m\}) = (f(E)\cdot C_1, f(E)\cdot C_2,\Lambda, f(E)\cdot C_m)$$

规定 $m(U) = 1-\sum_{i=1}^{m} m(\{a_i\})$，则对于 U 的所有其它子集 H，均有

m(H)＝0；

所以当 A 为 U 的真子集时，有

$$Bel(A)=\sum_{B\subseteq A}m(B)=\sum_{i=1}^{m}m(\{a_i\})$$

进一步可以计算 Pl(A)和 f(A)。

5．不确定性的组合

当规则的前提（即证据）E 是多个命题的合取或析取时，定义

$f(E_1\wedge E_2\wedge\cdots\wedge E_n)=\min[f(E_1),f(E_2),\cdots,f(E_n)]$

$f(E_1\vee E_2\vee\cdots\vee E_n)=\max[f(E_1),f(E_2),\cdots,f(E_n)]$

当有多条规则支持同一结论 A（$A=\{a_1,a_2,\cdots,a_m\}$）时，即

$$\begin{aligned}&E_1\Rightarrow A,CF_1 \quad (CF_1=\{C_{11},C_{12},\cdots C_{1m}\})\\&E_2\Rightarrow A,CF_2 \quad (CF_2=\{C_{21},C_{22},\cdots C_{2m}\})\\&\quad\vdots\qquad\qquad\vdots\\&E_n\Rightarrow A,CF_n \quad (CF_n=\{C_{n1},C_{n2},\cdots C_{nm}\})\end{aligned}$$

若这些规则的前提相互独立地支持结论 A 的成立，可以先计算

$$m_i(\{a_1\},\{a_2\},\cdots,\{a_m\})=(f(E_i)\cdot C_{i1},f(E_i)\cdot C_{i2},\cdots,f(E_i)\cdot C_{im});$$

其中，i = 1,2,…,n；然后再通过 m_i 的正交和，组合所有规则对结论 A 的支持。组合计算从组合规则 1 和 2 对结论 A 的影响开始，令 $m=m_1\oplus m_2$ 表示 m_1（对应于规则 1）和 m_2（对应于规则 2）的正交和，然后再累加其它规则的影响。

定义
$$m(A)=(m_1\oplus m_2)(A)=\begin{cases}0 & \text{其中}A=\Phi\\ K\cdot\sum_{X\cap Y=\Phi}m_1(X)\cdot m_2(Y) & \text{当}A\neq\Phi,X\subseteq U,Y\subseteq U\end{cases}$$

$$K^{-1}=1-\sum_{X\cap Y=\Phi}m_1(X)\cdot m_2(Y)=\sum_{X\cap Y\neq\Phi}m_1(X)\cdot m_2(Y)$$

一旦累加的正交和 m(A)计算出来后，就可如前述那样计算 Bel(A)，Pl(A)和 f(A)。

17.2.10.4 模糊逻辑和模糊推理

由扎德（Zadeh）于 1983 年提出的模糊逻辑（Fuzzy Logic）是建立在模糊集理论的基础之上的，是一种处理不精确描述的软计算。与不确定推理处理随机事件发生的可能性相对照，模糊逻辑主要面向事物特性和能力的不精确描述。模糊逻辑中的一个核心概念是“语言变量”。例如，当我们将人的年龄作为一个语言变量时，其可有三个以“术语”表示的定性值：轻、中、老。其每个值均可由被称为“隶属度函数”的一个函数来加以定义。尽管当“年龄”作为数值变量时其变量值更简单（如“年龄”等于 25），但其值域有许多值（如 1－100）。所以语言变量是一种“抽象形式”的数据压缩（年龄只有三个定性值）。但这种压缩不同于定性物理中的量（值间隔）概念，因为语言变量的定性值是一种模糊值间隔，相互重叠，不存在用于分割连续值域的界标。使用语言变量的主要方式是模糊图和模糊规则。

1．模糊集合及其运算

一个论域 U 中的元素 x 可以按其属性划分为子集。例如具有属性 a 的元素构成子集 A，表示为 A = {x/a(x)}

传统的集合论中，元素 x 与子集 A 的关系只可能有两种：x∈A（a(x)＝1）或 $x\notin A(a(x)=0)$，视 x 是否有属性 a 而定。所以，a(x）也称为特征函数。然而，真实世界中的许多事物和概念却不能这样简单地描述。例如，上述年龄“轻”这个概念就找不到一个年龄数值作为年轻和中年的分界线。通常，30 岁以下的人认为是年轻的，但 30-40 岁之间的人属年轻或中年就是很模糊的，且因人的观念和场合而异。

为表示类似这样的一些模糊概念，扎德于 1965 年提出模糊集合理论，其基本思想就是把传统集合论中由特征函数决定的绝对隶属关系模糊化，使元素 x 对子集 A 的隶属程度不再局限于取 0 或 1，而是可以取［0，1］上的任何值，以指示元素 X 隶属于子集 A 的模糊程度。

为此，可以在论域 U 上定义一个模糊子集（简称模糊集） $\underset{\sim}{A}$，其对 U 的任意元素 x 均指定一个值 $\mu_A(x)\in[0,1]$，以表示它对 A 的隶属程度，即有

$$\mu_A:\ U\to[0,1]$$
$$\underset{\sim}{A}=\{x/\mu_A(x)\}$$

其中，μ_A称为 $\underset{\sim}{A}$的隶属函数。显然，当 $\mu_A(x)=1$ 时，x 确定性隶属于 $\underset{\sim}{A}$；而 $\mu_A(x)=0$ 时，x 确定性非隶属于 $\underset{\sim}{A}$；x 取其它值时，隶属程度模糊。当 U 是离散元素的有限集时，即 $U=\{x_1, x_2, \cdots, x_n\}$，模糊集 $\underset{\sim}{A}$可表示如下：

$$\underset{\sim}{A}=x_1/\mu_A(x_1)+x_2/\mu_A(x_2)+\Lambda+x_n/\mu_A(x_n)$$

其中，$\mu_A(x_i)=0$ 的项可以省略。总之，一个模糊集 $\underset{\sim}{A}$是以隶属函数 $\mu_A(x)$来描述的，隶属程度的概念构成模糊集理论的基石。

与普通集合一样，对模糊集可以进行各种逻辑运算，主要的运算有并、交、余。设 $\underset{\sim}{A}$和 $\underset{\sim}{B}$均为论域 U 上的模糊集，则对于元素 x， $\underset{\sim}{A}$与 $\underset{\sim}{B}$并、交、余运算定义如下：

$$\mu_{A\cup B}(x)=\max[\mu_A(x),\ \mu_B(x)]$$
$$\mu_{A\cap B}(x)=\min[\mu_A(x),\ \mu_B(x)]$$
$$\mu_{\overline{A}}(x)=1-\mu_A(x)$$

可见，模糊集合的逻辑运算实质上就是隶属函数的组合运算过程

2．模糊逻辑

模糊逻辑的基本思想是将常规数值变量模糊化，使变量成为以定性术语（也称语言值）为值域的语言变量。例如前述的语言变量"年龄"就以三个定性术语作为其值域。当用语言变量来描述对象时，这些定性术语就构成 模糊命题。可以省略被描述的对象，则模糊命题可表示为"＜语言变量＞＜定性值＞"形式。例如张三“年龄轻”就是一个模糊命题，其模糊程度用定性术语“轻”的隶属函数来表示。然后可以对模糊命题作合取、析取、取反等逻辑操作。

每个模糊命题均由相应的一个模糊集作细化描述，所以模糊逻辑操作与模糊集操作是一致的。设 P_1，P_2，…，P_m；为论域 U_1，U_2，…，U_n上的一组模糊命题，相应的隶属函数为 μ_{P_1}，μ_{P_2}，…，μ_{P_m}，当前观察的论域元素分别为 x_{P_1}，x_{P_2}，…，x_{P_m}；若令 $P_\vee$和 $P_\wedge$分别表示这些命题的析取和合取，则 $P_\vee$和 $P_\wedge$的隶属程度为

$$\mu_{P_\vee}=\max[\mu_{P_1}(x_{P_1}),\mu_{P_{21}}(x_{P_2}),\cdots,\mu_{P_m}(x_{P_m})]$$
$$\mu_{P_\wedge}=\min[\mu_{P_1}(x_{P_1}),\mu_{P_{21}}(x_{P_2}),\cdots,\mu_{P_m}(x_{P_m})]$$

令析取和合取隐含着 max 和 min 操作，则此二式可简写为

$$\mu_{P_\vee}=\bigvee_{i=1}^{m}\mu_{P_i}(x_{P_i})$$

$$\mu_{P_n} = \hat{\bigcap_{i=1}^{m}} \mu_{P_i}(x_{P_i})$$

其中，不同的模糊命题可以面向同一论域，即使用相同的语言变量，但取用的定性值不同，隶属函数也不同。对于模糊命题的取反，则有

$$\mu_{\neg p_i}(x_{P_i}) = 1 - \mu_{p_i}(x_{P_i})$$

让模糊命题的析取和合取隐含 max 和 min，在一定程度上反映了客观规律，但这种平等看待各模糊命题的观念有时仍不符合实际。在真实世界中，影响问题求解的因素往往具有不相同的相对重要性。例如一个模糊控制器的输入变量是与标准温度的温差 θ 和温度变化速率 dθ，输出为恒温加热器液体燃料流量的修正量 y；为保持恒温，若控制温差比控制温度变化率更有效，就应加强温差对流量修正的影响。这可以通过引入加权模糊逻辑来解决

3．模糊推理

模糊推理有多种模式，其中最重要的且广泛应用的是基于模糊规则的推理。模糊规则的前提是模糊命题的逻辑组合（经由合取、析取和取反操作），作为推理的条件；结论是表示推理结果的模糊命题。所有模糊命题成立的精确程度（或模糊程度）均以相应语言变量定性值的隶属函数来表示。

模糊规则由应用领域专家凭经验知识来制定，并可在应用系统的调试和运行过程中，逐步修正和完善。模糊规则连同各语言变量的隶属函数一起构成了应用系统的知识库。基于规则的模糊推理实际上是按模糊规则指示的模糊关系 $\underset{\sim}{R}$ 作模糊合成运算的过程。

建立在论域 U_1，U_2，…，U_n 上的一个模糊关系 $\underset{\sim}{R}$ 是笛卡尔积 $U_1 \times U_2 \times \cdots \times U_n$ 上的模糊集合。若这些论域的元素变量分别为 $X_{U_1}, X_{U_2}, \cdots, X_{U_n}$，则 R 的隶属函数记为 $\mu_R(X_{U_1}, X_{U_2}, \cdots, X_{U_n})$。模糊关系 $\underset{\sim}{R}$ 可形式地定义为

$$\mu_R : U_1 \times U_2 \times \cdots \times U_n \to [0,1]$$

$$\underset{\sim}{R} = \{(X_{U_1}, X_{U_2}, \cdots, X_{U_n}) / \mu_R(X_{U_1}, X_{U_2}, \cdots, X_{U_n})\}$$

在模糊推理中，尚未建立一致的理论去指导模糊关系的构造。这意味着存在着多种构造模糊关系的方法，相关的模糊合成运算方法也不同，从而形成了多种风格的模糊推理方法。不过，基于 max-min 原则的算法占居了目前模糊推理方法的主流。尽管这些算法不能说是最优的，但易于实现并能有效地解决实际问题，因此它们已广泛地应用于模糊推理。

（1）直接基于模糊规则的推理

当模糊推理的输人信息是量化的数值时，可以直接基于模糊规则作推理，然后把推理结论综合起来，典型的推理过程可以分为两个阶段，其中第一阶段又分为三个步骤，表述如下：

（a）计算每条模糊规则的结论：①输入量模糊化，即求出输入量相对于语言变量各定性值的隶属度；②计算规则前提部分模糊命题的逻辑组合（合取、析取和取反的组合）；③将规则前提逻辑组合的隶属程度与结论命题的隶属函数作 min 运算，求得结论的模糊程度。

（b）对所有规则结论的模糊程度作 max 运算，得到模糊推理结果。

（2）基于模糊关系的推理

当模糊推理的输人信息是定性术语（以相应的模糊集表示）时，可以基于模糊关系作推理。如前所述，存在多种构造模糊关系的方法，我们这里仅介绍简单直观的 Mamdani 方法。

设模糊规则形如 $P \Rightarrow H$，模糊命题 P 和 H 相应的模糊集 $\underset{\sim}{A}_P$ 和 $\underset{\sim}{A}_H$ 分别建立在论域 UP 和 UH 上（相应的元素变量为 xP，xH）。令 $\underset{\sim}{R}(P; H)$ 指示从 P 推出 H 的模糊关系，则定义

$$\underset{\sim}{R}(P;\ H)=\underset{\sim}{A}_{P}\times\underset{\sim}{A}_{H}=\{(x_P,x_H)/\mu_R(x_P,x_H)\}$$

$$\mu_R(x_P,x_H)=\mu_{A_P}(x_P)\wedge\mu_{A_H}(x_H)$$

当实际的输人信息是模糊命题 P′（相应的模糊集为 $\underset{\sim}{A}_{P'}$）的，则模糊推理的输出 H′（相应的模糊集为 $\underset{\sim}{A}_{H'}$）表示为

$$\underset{\sim}{A}_{H'}=\underset{\sim}{A}_{P'}\cdot\underset{\sim}{R}(P;\ H)$$

$$\mu_{A_{H'}}=\bigvee_{x_P\in U_P}(\mu_{A_{P'}}(x_P)\wedge\mu_R(x_P,x_H))$$

设有 m 条形如 $P_i\Rightarrow H_i$ 的规则，相应于每条规则的模糊关系分别为 $\underset{\sim}{R}_1$，$\underset{\sim}{R}_2$，…，$\underset{\sim}{R}_m$，则综合的模糊关系 $\underset{\sim}{R}$ 定义为　$\underset{\sim}{R}=\underset{\sim}{R}_1\bigcup\underset{\sim}{R}_2\bigcup\cdots\bigcup\underset{\sim}{R}_m=\bigcup_{i=1}^{m}\underset{\sim}{R}_i$　　$\mu_R(x_P,x_H)=\bigvee_{i=1}^{m}\mu_{R_i}(x_P,x_H)$

在实际应用中，规则的前提常表示为若干模糊命题的合取，则

$$\left(\bigwedge_{j=1}^{n}P_{ij}\right)\Rightarrow H_i\qquad\underset{\sim}{R}=\bigcup_{i=1}^{m}\underset{\sim}{R}_i=\bigcup_{i=1}^{m}\underset{\sim}{A}_{P_{i1}}\times\underset{\sim}{A}_{P_{i2}}\times\cdots\times\underset{\sim}{A}_{P_{in}}\times\underset{\sim}{A}_{H_i}\qquad\underset{\sim}{A}_{H'}=(\underset{\sim}{A}_{P_1'}\times\underset{\sim}{A}_{P_2'}\times\cdots\times\underset{\sim}{A}_{P_n'})\cdot\underset{\sim}{R}$$

或者

$$\underset{\sim}{A}_{H'}=\bigcup_{i=1}^{m}(\underset{\sim}{A}_{P_1'}\times\underset{\sim}{A}_{P_2'}\times\cdots\times\underset{\sim}{A}_{P_n'})\cdot R_i$$

（3）典型应用实例--模糊控制

基于模糊逻辑的推理系统已发展为一个重要的学科领域，成功的实用系统也与日俱增。最成功的应用领域是对各种物理和化学特征，如温度、电子流、液流、机械运动等的模糊控制。模糊控制技术已得到广泛采用，数以千计的模糊系统的开发使微波炉、洗衣机。空调机和照相机具有了能自动优化任务的“机器智能”。

与传统的 PID（比例、积分、微分）控制方法相比，模糊控制采用了完全不同的原理和方法，从而具有许多 PID 控制没有的优点，特别适合于难以建立精确数学模型、非线性和大滞后的过程。一个典型的模糊控制器如图 17.2.9 所示。通常模糊控制器的输入（如温差 θ 和温度变化率 dθ）和输出都是精确的数值，这就需要定义相应于输入的语言变量及其定性值，实现输入数据的“模糊化”；并将模糊推理的结果转变为数值，实现输出数据的模糊判决[反模糊化]。

模糊化的关键在于设计语言变量定性值的隶属函数。在实际控制过程中，经常把物理量划分为“正大、正中、正小、零、负小、负中、负大”这 7 个级别，并且以 PB，PM，PS，ZO，NS，NM，NB 加以表示，作为相应语言变量的定性值。若控制精度要求不高，则隶属函数可以采用梯形或三角形。

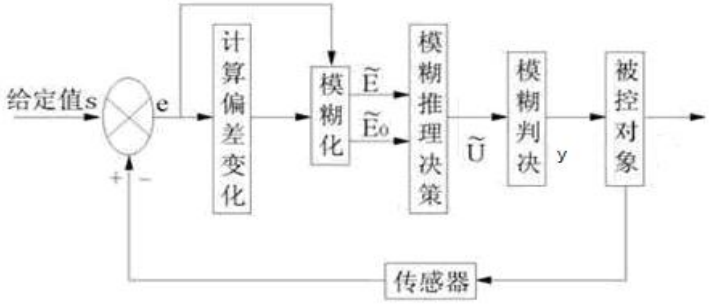

图 17.2.9　模糊控制器示意图

模糊推理建立在表示为模糊规则的知识库上，模糊规则的多少取决于输入和输出物理量的个数以及所需的控制精度。对于常用的二输入、一输出控制过程，若每个输入量划分为 7 个等级，则有 49 条规则就可覆盖全部情况。当然，实际应用中往往只需定义少得多的规则。

反模糊化[模糊判决]有多种方法，其中最简单的是最大隶属度法。即取推理结果（模糊集）中模糊度最大的那个元素。但这种方法精度较差，因为完全排除了其它隶属度较小的元素的影响和作用。较为合理也是最常用的方法是加权平均法，输出量

$$y = \frac{\sum \mu_y(y_i) \cdot y_i}{\sum \mu_y(y_i)}$$

yi（i = 1，2，…，n）为输出量论域中的元素。若论域是连续域，则可用面积重心法。

模糊控制实际上是周期性执行模糊化、模糊推理和反模糊化的过程，在输入和输出量不多的情况下，适合于微机来完成。但周期性执行导致大量重复计算，效率低下。这可以通过引入查表法来改进。首先将输入和输出物理量的值域划分为若干等级（例如 13），并归化到某一标准区域上（如[-6，6]，然后以二维模糊化表的方式定义隶属函数。

模糊规则的定义也可表格化。实际上基于输入模糊化表、输出模糊化表和模糊规则表的计算和推理可以在模糊控制器实际工作之前预先完成，并将控制器的输入和输出数据整理成二维表。由此模糊控制器的工作就简化为从输入量等级查表决定输出量等级的处理了。显然，查表法可以显著提高模糊控制的效率，而且表格和查表程序只占据很少内存，可以制作在 ROM 芯片上，模糊控制家电产品一般都采用这种方法。查表法的缺点在于变更模糊规则和隶属函数不方便，而且当输入量个数增加时，表格的存储容量将指数级增长。

17.2.10.5 应用不确定推理的准则

不确定推理仅是实现软计算的一种方式，并有其适用范围。应用不确定推理的准则可以归纳为以下三点：

（1）尽可能避免使用统计表示，能确定性地解决问题的场合不应使用不确定推理，因为主观概率是不精确的，且在许多场合难以估计。例如，字符识别系统应把字符表示为一组高级特征，作结构化模式识别；而不要表示为黑点集，作统计模式识别。

（2）在必须采用不确定推理时，应将其限制在小范围内（相应于推理中的逻辑步）；而不要在不能反映问题结构的大跨度操作中执行。

（3）切记不确定推理结果的精度决不会超过输入数据的精度，不管采用什么技巧也无济于事，所以应尽量保持输入数据的精确性，否则结论的可信度只是误导。

17.2.11 基于知识的问题求解系统曾经的实现策略及示例【内容过时，仅供参考】

17.2.11.1 曾经的基于知识的问题求解系统的开发工具和环境

尽管基于知识的问题求解系统的设计取决于应用领域特征和问题求解任务的要求，但基于知识的问题求解系统之间仍具有很多共性，特别是执行同类任务或同一应用领域内任务的系统，在知识表示、体系结构等方面有不少共同之处，使研制各种开发知识系统的工具和环境成为可能。从各种已经推广使用的工具和环境的应用情况表明，开发工具和环境的应用可大大减少设计知识系统的重复性工作。换言之，应用工具提供的知识表示和推理控制结构，可大大提高开发效率，缩短开发周期，使知识系统的开发容易进行。

常用的开发工具和环境可以分为三类：外壳(骨架系统)、表示语言和开发工具箱(开发环境)。外壳给知识工程师提供了一个现成的实现知识系统的骨架，只要按骨架规定的表示方式编写专门的

知识，就可以形成应用领域的知识系统。为方便使用，外壳型工具往往提供知识获取的辅助设施和知识编辑器等。曾经的典型的外壳型工具有 EMYCIN 和 EXPERT，它们都是从已经建立的专家系统中抽取出的通用骨架。外壳型工具的缺点在于对任务的特征有严格的要求，若任务特征不能与外壳提供的表示语言和体系结构匹配，就不能应用外壳来开发相应的知识系统。即这种工具仅有较窄的应用范围，但对于合适的任务，使用外壳能很快开发出知识系统。

表示语言类工具可为知识系统开发工程师提供面向知识处理的高级编程语言。工具只提供通用的控制结构，知识工程师可以通过编程语言来实现特别的控制结构(建立在通用控制结构的基础上)，所以能适用于较宽广的应用领域。缺点在于编程语言不能直接描述控制结构，只能隐含于编程语言表示的知识中，增加了知识系统的开发和维护的困难。OPS5 就曾是一个典型的表示语言类工具。

开发工具箱(或称开发环境)可为知识系统的生命周期--开发、运行、维护、评价、进化等各个阶段提供工具，甚至可以提供多种外壳和表示语言，以及综合它们建立复杂知识系统的手段。曾经的典型的开发环境有 ART，KEE，Knowledge-Craft 等。也曾大有取代单一外壳和表示语言工具的趋势。其中，KEE 全称为 Knowledge Engineering Environment，它结合了 Lisp 语言和面向对象程序设计的优点。KEE 的知识表示方式是框架系统(面向对象的表示语言)，允许对象中封装的处理方法设计为基于规则的推理，从而把框架和规则这两种不同的知识表示方式紧密地结合在一起。推理控制允许正向和逆向推理，对象(框架)之间允许以消息收发的方式交互。KEE 也为知识工程师提供图文编辑器，以辅助知识系统的快速建立和维护扩展。此外，它还提供存取关系数据库的接口，使知识系统技术易于和常规软件技术集成。

17.2.11.2 一个产生式系统设计示例【仅供参考】

我们知道，产生式是一种得到广泛应用的知识表示技术，特别适用于复杂程度不同的中小型知识系统的设计。于是，基于产生式表示的知识系统开发工具也应运而生，其中最著名的就有产生式系统语言 OPS5（之后发展为 OPS-83）。OPS5 采用条件-动作型产生式规则，只允许正向推理，规则的右部可以是任何操作函数的序列。下面介绍的是一个命名为 Xps 的模拟 OPS5 实现的实验型产生式系统，它取自高济的课件，仅用来说明开发一个产生式系统的基本过程。

OPS5 属表示语言型专家系统开发工具，其能提供比骨架型工具更为通用的推理控制机制和知识表示语言去适应于较宽范围的应用领域，尤其是专家系统开发者可以通过 OPS5 的表示语言去设计特别的控制要求。

我们知道，产生式系统主要由三个部分组成：规则库、综合数据库和控制系统。规则库是知识系统的知识库，综合数据是存放问题的初始状态描述和推理中间结果的工作区域，而控制系统则按一定的策略控制基于规则的问题求解过程。

1．规则的表示

规则的表示形式以 BNF 定义如下：　<规则> := <规则名> {<匹配模式>}* ＝>{<操作>}+

匹配模式相当于谓词公式，只不过表示为元素的列表形式；列表的首元素指示谓词，其它元素就是谓词公式的参数项，可以是常量和变量；变量以“?”作为前缀，也称为模式变量。操作以表示为元素列表形式的函数来实现，表首元素即函数（操作）名，其余元素为函数的参数项。用元素列表来表示谓词公式和函数是人工智能程序设计语言 LISP 的风格，所以 Xps 可方便地由 LISP 语言编程实现，但用其它语言（例如，C 和 C++）实现也不难。

可以用规则定义函数 Define-Rule 定义一条新规则，并将其置于规则库。例如下面定义了一条

有关饮食问题的规则：

```
(Define-Rule Eat
        (Hungry ?Person) (Edible ?Food) =>
        (Write '(?Person eats the ?Food)))
```

其中，出现于规则条件部分的多个（这里 2 个）匹配模式之间隐含合取关系，所以只有这些匹配模式全部得到满足，即与综合数据库中的事实元素或中间结果匹配，规则才会激活。模式变量是仅在单一规则内部才有效的局部变量，无初始束缚值，只能在对匹配模式进行匹配检查的过程中取得束缚值（通过合一和置换）。列表前面的“'”号指示先将列表中的模式变量用其束缚值置换，然后整个列表作为字符串使用（相当于用双引号括起）。

2．综合数据库的表示

综合数据库的内容是描述问题求解初始状态的事实元素和推理过程的中间结果，都表示为以列表形式描述的谓词公式。可以用存储函数 DB-Store 将它们插进综合数据库。例如，在初始化有关饮食问题的综合数据库时，若执行：

```
(DB-Store '(Hungry Peter))
(DB-Store '(Hungry Paul))
(DB-Store '(Edible Hot-Dog))
(DB-Store '(Edible Turkey-Leg))
(DB-Store '(Edible Muffin))
```

则综合数据库的初始内容就由这 5 个事实元素（命题公式）构成，并且每个元素附加一个时间标签，以指示它们进入综合数据库的先后顺序：

时间标签	事实元素
1	(Hungry Peter)
2	(Hungry Paul)
3	(Edible Hot-Dog)
4	(Edible Turkey-Leg)
5	(Edible Muffin)

时间标签按事实元素进入综合数据库的顺序，从 1 开始，依次加 1。时间标签在下面介绍 Xps 的冲突解法中起关键作用。

3．控制系统

控制机制采用前述的识别-行动循环控制流。在每个识别-行动循环的识别阶段均有可能激活多条规则，且每条激活的规则可有多个激活例，这些规则激活例构成了所谓冲突集。之所以一条规则可以有多个激活例，原因在于该规则的同一条件模式可能会与综合数据库中的不同事实元素相匹配，例如上述有关饮食问题的规则就存在多个满足综合数据库的激活例，并由此建立了以下冲突集：

规则名	激活例序号	变量置换	时间标签表
Eat	1	{Peter/Person, Hot-Dog/Food}	(1 3)
Eat	2	{Peter/Person, Turkey-Leg/Food}	(1 4)
Eat	3	{Peter/Person, Muffin/Food}	(1 5)
Eat	4	{Paul/Person, Hot-Dog/Food}	(2 3)

Eat　　5　　{Paul/Person, Turkey-Leg/Food}　　(2 4)

Eat　　6　　{Paul/Person, Muffin/Food}　　(2 5)

其中，时间标签表记载了与规则条件部分匹配模式匹配的事实元素的时间标签。

Xps 采用的冲突解法是：新近和特殊的规则激活例优先选用，冲突集可以有三种情况：

- 空集--则系统无法继续推理过程，失败结束；
- 单一规则激活例--直接执行该激活例；
- 多个规则激活例--执行冲突解法。

冲突解法分三个步骤，分别由三个筛选器执行：

（1）折射（Refraction）筛选--将已使用过，又再一次激活的规则例删除，不让其进入冲突集。重复激活的规则例是可能存在的，例如某规则例被选中执行时，其右部并未提供操作去删除综合数据库中与规则条件部分匹配模式匹配的事实元素，就会发生这种情况。在规则激活例中记载的时间标签表，使得检查规则例是否重复激活成为可能。上面有关饮食例子中，若从冲突集中选用第 6 个规则激活例执行，则在下一个识别-行动循环的识别阶段，该规则例仍会重复激活；但经由折射筛选，其就不会再进入冲突集。

（2）新近性（Recency）筛选--筛选的原则是能与最新近进入综合数据库的事实元素相匹配的规则激活例优先选用。在上述有关饮食问题的冲突集中，显然第 6 个规则激活例最新近，就选用它。由于规则条件部分往往有多个匹配模式，所以必须综合评价它们的新近性。鉴于规则激活例已记载了时间标签表，很容易基于该表加以评价。该方法如下：首先将各规则激活例的时间标签表按数字从大到小排列其包含标签的顺序（并删除重复的标签），然后再依次比较经排序后的时间标签表的相应元素，就可鉴别出新近性的不同。例如，有以下各时间标签表：

（1 10 3）（3 10 1）（9 1 3）（8 6 9 7）（10 3）（3 1 2 9）（3 1 10 1）

则先对各时间标签表进行排序得：

（10 3 1 ）（10 3 1）（9 3 1）（9 8 7 6 ）（10 3）（9 3 2 1）（10 3 1）

其中最后一个时间标签标有二个均为 1 的时间标签，意味着标签为 1 的事实元素与规则条件部分的 2 个不同的匹配模式匹配，所以不应作为 2 个标签，而将它们合而为一。按新近性原则，相应于时间标签表

（1 10 3）（3 10 1）（3 1 10 1）

的规则激活例新近性最好。用新近性原则筛选规则激活例，在一定程度上体现了深度优先的推理（搜索）控制策略。

（3）特殊性（Specificity）筛选--特殊性意指规则的条件部分具有更多的匹配模式。显然，特殊性高的规则难以激活，所以一旦激活，并通过了新近性筛选，就应优先选用这种规则的激活例。于是对于上例新近性筛选留下的三个规则激活例，按特殊性原则，就应选用对应于时间标签表（3 1 10 1）的那个。

若经由上述三个步骤的筛选后仍留下多于一个的规则激活例，则从中随机选用一个。

4．Xps 的实现

实现 Xps 的程序设计分三个部分进行：规则库管理、综合数据库管理和推理引擎。

（1）规则库管理。规则库设计为一个散列表，用前述函数 Define-Rule 定义产生式规则，并在对其作简单的句法检查后，以规则名作为索引插入散列表。为提高使用效率，规则转变为内部形式的

数据结构存放，包括5个数据场：规则名、匹配模式列表、模式变量表、操作函数列表和时间标签表集合。模式变量表用于集中存放规则中出现的模式变量及其在匹配检查的过程中得到的束缚值，以利取用。时间标签表集合存放已被执行（右部执行）过的该规则激活例的时间标签表，以备检查。

（2）综合数据库管理。综合数据库设计为树状层次索引网。由于事实元素表示为列表形式的谓词公式，可按列表中的元素次序逐层建立事实元素的索引，并将事实元素置于索引路径的末端。前述有关饮食问题的综合数据库的初始内容就可以这种方式存放。面向综合数据库的管理操作包括PS-Store、PS-Erase和PS-Fetch，分别实现事实元素的插入、删除和取用。

（3）推理引擎。推理引擎也称为解释器，其主要工作就是对规则库中的规则进行解释性执行。基本的控制流是识别-行动循环，每个循环分三个步骤执行：建立冲突集，解决冲突和执行选中的规则例（执行该规则例的右部）。

1）建立冲突集。推理引擎遍历规则库，按排列顺序依次对每条规则条件部分的每个匹配模式，应用管理操作PS-Fetch将综合数据库中所有匹配的事实元素取到（此时模式变量都默认为与事实元素中相对应的参数项匹配），并进一步作合一检查。每当一个规则条件部分的所有匹配模式都找到匹配的事实元素时，就建立该规则的一个激活例，记载模式变量束缚值和时间标签表。随即检查该标签表是否出现于该规则的标签表集合中，若出现，则该激活例已使用过，不再进入冲突集，否则加进冲突集。如此，在遍历规则库后，就建立起包含对应于若干个规则的若干激活例的冲突集。

2）解决冲突。由于“折射”筛选步已在建立冲突集的过程中完成，解决冲突实际上就是进行后二步：新近性筛选和特殊性筛选，若筛选后冲突集中还剩余多个规则激活例，就随机取一个。

3）执行选用的规则例。主要做二件工作：首先把该规则激活例的时间标签表加进相应规则内部结构的时间标签表集合；然后将模式变量的束缚值取出，作为参数调用规则右部的操作函数加以执行。

4)特别处理。为增加产生式规则的表示功能，Xps允许在规则的条件部分应用特殊谓词Assign，连词AND和NOT，关系表达式（以前缀方式表示）和任何真值函数（以$符号作为函数名前缀）。

应用Assign谓词的表达式形如：（Assign <模式变量> <匹配模式>）

用于提高规则表示的便易性。其用法通过下面例子加以说明：

```
(Define-Rule Fill-Big-Box
    (Assign ?A (On ?X Table))
    (Color ?X Green)
    =>
    (PS-Store '(In ?X Big-Box))
    (Write '(?X is now in the big box))
    (PS-Erase ?A))
```

这里谓词Assign,仅指示将与匹配模式匹配的事实元素作为模式变量?A的束缚值。换言之，应用Assign谓词的表达式并不直接用于作为匹配模式，模式变量（该例子中的?A）后的匹配模式才是需作匹配检查的。

连词AND的应用使多个匹配模式联合作为单一的匹配模式。例如某规则条件部分形如：

```
(P ?X ?Y)(AND (Q ?X ?Y)(W ?Y ?Z))
```

则相当于该规则只有2个匹配模式，规则激活例的时间标签表也只包含2个时标。连词AND辖域内

的匹配模式仍分别作匹配检查，只是仅将匹配的事实元素中最大的时间标签作为整个 AND 匹配模式的时间标签。

连词 NOT 的应用引入了否定的匹配模式，例如：

（NOT (P ?X ?Y)）

只有（P ?X ?Y）不能满足（即综合数据库中不存在与其匹配的事实元素）的情况下，NOT 匹配模式才满足。显然相应于满足的 NOT 匹配模式，不可能取得时间标签；但为了表示因 NOT 匹配模式的引入增加了规则的特殊性，可产生一个以数字"0"指示的空时间标签。例如时间标签（9 0 3）就意指条件部分第 2 个匹配模式是 NOT 匹配模式。

关系表达式（使用关系符>, >=, =, <=, <）和真值函数不需要在综合数据库中进行匹配检查，而是依据关系符的语义或真值函数的执行来确定真值（T 或 F）。因此，同样不可能取得时间标签，可相应地引入空时间标签。

5．系统性能的改进

从 Xps 推理引擎的设计和关于家族树的例子可以看出，Xps 的一个严重缺点就是重复地产生冲突集。即每个识别-行动循环都重新生成一个冲突集，但系统只从中选一个规则激活例执行，其余规则激活例全部抛弃。实用上，相邻二个循环之间综合数据库的内容往往变化很小，造成许多同样的匹配检查工作重复地进行，浪费时间，降低问题求解效率。

作为改进，Xps 应设计成不是每个循环重新生成一个冲突集，而是始终保持一个全局冲突集。一旦初始冲突集生成，在以后的识别-行动循环中只需依据综合数据库的增删变化（因规则激活例的执行而变化），加新的规则激活例到冲突集，删去条件部分变得不满足的规则激活例。为此，需对 Xps 作以下改进：

（1）将规则条件部分的匹配模式置于另一树状层次索引网，并记载匹配模式在规则条件部分中的排列次序和模式变量束缚值。每当执行选中的规则激活例而导致综合数据库增、删事实元素时，就到该索引网寻找与这些事实元素匹配的匹配模式，进而包含这些匹配模式的规则例就有可能加入冲突集或从冲突集删除。换言之，我们不再用规则去匹配事实元素，而是反过来用新增、删的事实元素去匹配规则。

（2）记载规则的部分满足状态。规则条件部分仅部分匹配模式与事实元素匹配，称为部分满足。由于记载了部分满足状态，以后一旦其余匹配模式得到满足，规则就激活，不需重复检查已满足的匹配模式。

（3）处理否定的匹配模式（带连词 NOT）。由于这种匹配模式的引入，使维持全局冲突集变得困难起来。因为此时，增加事实元素也会引起已有规则激活例不再激活，而删除事实元素又会激活新的规则例。数据从属技术可用于有效地处理这种困难。

所有这些优化技术已实现于 OPS5，称为 Rete 算法，有兴趣的读者可查阅 OPS5 的相关文献。可以说 Xps 的主要功能与 OPS5 相当，推理控制机制也相同；主要区别在于知识表示形式和内部存储方式。

上述 Xps 较好地模仿了 OPS5 的实现，主要差别在以下三个方面：

1）对象表示。Xps 用一组相互独立的事实元素来描述一个对象，而 OPS5 则用对象子句集中描述一个对象，形如：（<对象>｛ ^ <属性> <值> }+）其中符号"^"指示属性名。例如 Xps 用一组事实元素来描述某个人 PENELOPE：

（Parents Penelope Jessica Jeremy）

(Age Penelope 20)

(Sex Penelope Female)

而在OPS5中则紧凑地表示为：

(Person ^Name Penelope ^Age 20 ^Sex Female ^Parents Jessica Jeremy)

2）对象的存储形式。Xps以树状层次索引网存储事实元素,OPS5则为每类对象（例如上述Person）定义一个类（Class）结构，使类的每个实例（即对象）具有固定数量的属性和固定的属性名。OPS5以LITERAL1ZE格式定义类，例如类Person定义为：（LITERALIZE Person Name Age Sex Parents）

类定义允许最后一个属性取多值，这种属性称为向量属性，上述中Penelope的父母亲属性就是由二个值（Jessica和Jeremy）组成的向量。正由于这种类定义，类实例（对象）在内存中只需存放遵从类定义的值向量，不存放属性名。OPS5用散列（Hash）表索引值向量，以提高检索效率`。

3）规则条件。规则条件部分匹配模式中的模式变量常会受到一些值束缚限制，例如要求模式变量?X非空。Xps以插入规则条件部分的（NOT (EQUAL ?X NIL)）来表示；OPS5中规则前提部分的模式则以带变量的类实例（对象）来实现更为方便和紧凑的表示。就以前述家族树最后一条规则的条件部分为例，OPS5将其表示为：（Request ^Type Ancestors ^Target {<Name> <> NIL}）

-(Request ^Type Ancestors ^Target {<> <Name> <> NIL})

其中，Type和Target均是属性名，符号"-"的作用相当于逻辑符"NOT"，花括弧指示模式变量及其值束缚限制。第一个匹配模式中的模式变量<Name>（变量名用尖括号括起，而不是用"？"作为前缀），由于其初次出现，能与相应事实元素中对应属性的任意取值匹配，并以该值作为变量的束缚值。变量<Name>后面的"<>"指示对于该变量束缚值的附加约束：不等于NIL（即非空）。第二个条件模式中的模式变量<Name>，由于是第二次重复出现，已经有束缚值。该变量前、后的"<>"分别指示相应事实元素中对应属性的值不能与变量束缚值相同，也不能为空。 稍加分析，可以得知OPS5关于规则条件部分的表示完全等价于Xps，但更为简洁。前者只须二个匹配模式，后者则需五个匹配模式。

OPS5的一个不同于骨架型工具的重要特点是允许专家系统开发者定制特别的控制要求。定制建立在二个重要概念的基础上：目标模式和控制元素。产生式规则条件部分的第一个匹配模式称为目标模式，而与其匹配的综合数据库中的事实元素称为控制元素。显然，只有与控制元素匹配的规则才可能激活，这意味着同目标模式的规则构成一个规则组，可用来服务于解决某个子问题。所以，通过设计目标模式和相应的控制元素，就可定制适当的控制策略，进而让子问题（子目标）按预先确定的次序逐个解决。一个附加的收益是提高了问题求解的效率，因为通过设置适当的索引，可以限制规则的匹配检查工作到新插入综合数据库的控制元素所对应的一组规则。

OPS5提供的这种推理控制的定制能力既是优点也是缺点。优点体现在能够定制控制要求以适应于问题特征和更有效地求解问题，也有利于知识库维护；缺点体现在要求专家系统开发者具有一定的技术水平，无经验的开发者会感觉到难以使用。作为对照，骨架型工具完全固定了推理控制流程（如EMYCIN的逆向推理链和深度优先的控制策略），尽管不灵活，但若适合于解决手头问题，则使用起来十分便利，开发者只需用工具提供的表示语言表示知识，就可快速建立起专家系统。

17.2.11.3 专家系统实例—MYCIN简介【简单示例，仅供参考】

MYCIN是一个通过提供咨询服务来帮助普通内科医生诊治细菌感染性疾病的专家系统，其于

1972 年开始研制，74 年基本完成，并投入实际应用。MYCIN 的取名来自多种治疗药物的公共后缀，如 clindamycin、erythromycin、kanamycin 等。如果说能推测有机化合物分子结构的 DENDRAL 是世界上第一个有重要实用价值的专家系统，那末 MYCIN 则是最有影响力的专家系统。围绕着 MYCIN 的各种研究工作一直沿续了 10 年，对于推动知识工程以及专家系统学科的建立和发展具有重要影响。可以说，早期的专家系统，尤其是医疗诊断和咨询型专家系统，许多都参照了 MYCIN 系统的技术，如知识表示、不确定推理、推理解释、知识获取等。

MYCIN 也设计为典型的产生式系统，由规则库、综合数据库和控制系统三个部分组成；只是基于规则的推理采用逆向方式，即从问题求解的目标出发，搜寻原始证据对于目标成立的支持，并传递和计算推理的不确定性。从 KB 系统的组成来看，规则库就是 MYCIN 的知识库，综合数据库和控制系统联合形成推理机。其中，综合数据库用以保存问题求解的原始证据（初始状态）和中间结果。由于当时尚未出现视窗技术，用户界面只提供基于文本(text)的问答过程和结果显示。

MYCIN 系统采用 INTERLISP(人工智能程序设计语言 LISP 的一种版本)编程，运行于 DEC PDP-10 的操作系统 TENEX 下，MYCIN 系统的执行代码有 50KB，规则库占据 16KB，系统的咨询服务（包括提供解释）平均每次 20 分钟。

下面首先介绍知识库的结构，然后讨论推理机的设计，再阐述支持 MYCIN 应用的各种设施，最后介绍通过抽取 MYCIN 骨架而产生的专家系统工具 EMYCIN。

1．知识库的构造

MYCIN 的知识库以前提-动作型产生式规则来表示诊断和治疗细菌感染性疾病的专家级医学知识，以实现专家级诊断和治疗能力。规则的表示结构以 BNF 范式描述如下：

〈规则〉 := RULE 〈规则号〉
　　PREMISE ($AND {〈条件〉}+)
　　ACTION {〈动作〉}+
〈条件〉 := 〈简单条件〉 | ($OR {〈简单条件〉}+)

其中，简单条件以 MYCIN 提供的 4 类简单函数来表示。最常用的函数是 SAME，其 BNF 范式描述如下：

（SAME 〈对象〉〈属性〉〈值〉）

在推理机对规则进行解释执行时,SAME 函数指示推理机请求用户证实该对象是否有该属性值(当属性值是原始观测数据时），或经由逆向推理证实该属性值。这里证实的程度以所谓的可信度（CF-Certainty Factor）来指示。CF 的取值范围是[-1，+1]，-1 表示假，+1 表示真，0 指示无法确定真假的程度，其它值或多或少指示真假的程度。CF 可以由用户在回答关于证实的请求时直接给出，或由逆向推理产生。

规则中的动作也以简单函数表示，最常用的是 CONCLUDE，其将对象的属性值作为推理结论加进综合数据库，并记载推理结论的实际可信度。CONCLUDE 函数的 BNF 范式描述如下:

(CONCLUDE 〈对象〉〈属性〉〈值〉 TALLY 〈结论 CF〉)

其中，TALLY 用于存放规则前提的 CF。结论 CF 意指规则本身的可靠（可相信）程度，即在规则前提 CF 为 1（真）的情况下，结论为真的可能程度。实际上 MYCIN 知识库收集的大多数规则均是启发式关联知识，取自医疗专家多年积累的经验，不保证完全正确，所以引入结论可信度是十分必要的。结论 CF 的取值范围也是[-1,+1],其与规则前提在推理中使用时的实际 CF 相乘，即可得到规则结论

的实际 CF。例如，TALLY 当前值为 0.8，而结论 CF 为 0.4，则结论的实际 CF 为 0.32。

MYCIN 系统建立的初期就以上述格式表示和收集了 200 多条规则于知识库，其中 047 号规则表示如下：

```
RULE 047
    PREMISE ($AND (SAME CNTXT SITE BLOOD)
                  (NOTDEFINITE CNTXT IDENT)
                  (SAME CNTXT STAIN GRAMNEG)
                  (SAME CNTXT MORPH ROD)
                  (SAME CNTXT BURN T))
    ACTION (CONCLUDE CNTXT IDENT PSEUDOMONAS TALLY 0.4)
```

该规则的英语形式（已翻译为汉语）如下：

规则 047

如果：1）培养物取自血液，且 2）病原体的身份未鉴别，且 3）病原体的染色是革兰氏阴性，且 4）病原体的形态为杆状，且 5）病人被烧伤；

那么：该病原体的身份应鉴别为假单胞细菌，且可信度为 0.4。

其中，CNTXT 意指推理过程中需考察的相应对象，作为综合数据库的主要内容，MYCIN 系统称其为上下文，并区分为 10 类。上述规则中已涉及到三类：病人(PERSON)，从病人身上提取的培养物（CURCULS），从培养物中分离出的病原体（CURORGS）。另外还有给病人使用的抗生药物和治疗手术等。每一类上下文对象都有其特有的属性（也称临床参数），属性又可按其取值特性区分为单值型（只可取单一值）、多值型、可问型（可通过向用户询问来取值）、可导型（可基于规则推导出值）等。对象、属性和值构成所谓的关联三元组，由于属性名隶属于特别类型的上下文对象，所以规则 047 前提中的 CNTXT 隐含地指示了三种不同类型的上下文对象。

2．推理机的设计

MYCIN 系统的推理机分二个阶段四个步骤诊断和治疗细菌感染性疾病。整个推理过程通过称为目标规则的 092 号规则来启动：

规则 092

如果：1）存在一种病原体需要治疗，且 2）可能存在其它需要治疗的病原体，尽管它们尚未从目前的培养物中分离出来；

那么：1）依据病原体对药物的敏感情况，制定能有效抑制这些病原体的治疗方案（可以有多个），且 2）从中制定最佳的综合治疗方案；

否则：病人不必治疗。

该规则反映了医生诊断和治疗疾病的决策过程，即首先确定病人有无治疗细菌感染的需要，并进一步确定引起感染的细菌；然后制定若干可能的治疗方案，并从中制定最佳的综合治疗方案。前二个步骤对应于诊断阶段，并由该规则的前提部分表示；后二个步骤对应于治疗阶段，并由该规则的结论部分指示。显然，在前提不满足的情况下，病人不必治疗。

（1）诊断的推理控制

诊断的推理控制采用逆向推理和深度优先的搜索策略。MYCIN 的咨询模块由医生启动后，首先在综合数据库（MYCIN 称为动态数据库）中建立上下文对象：病人-1（patient-1），作为一棵上下

文树的根节点，并依据病人特有的属性，向医生询问病人的姓名、年龄和性别，然后以建立病人的治疗方案(REGIMEN)为目标，激活上述规则 092。由于该规则前提包含的二个条件不能通过询问用户来证实，就需寻找能推出这些条件（使这些条件满足）的规则（这些规则以这二个条件之一作为结论），例如规则 090 和 149。这些规则的前提包含的条件又可作为子目标，应寻求其它规则来支持。如此类推，形成规则链，直到链末端规则的前提包含的条件都能直接由原始证据（医生提供的观测结果）证实为止。

这样的推理过程导致推理树（或称目标树）的建立。由于规则 092 的前提涉及到病原体的属性 TREATFOR（需要治疗）和培养物的属性 COVERFOR(有病原体隐藏)，就需在上下文树中增加新的节点，以记载培养物-1 和病原体-1 的属性值。随着推理过程的进展，推理树将逐渐增大，上下文树也逐渐形成。由于导出相同结论的规则（如 090 和 149）相互独立地支持结论的成立（有"或"关系），而规则前提包含的条件又有"与"关系，所以推理树成为与或树。

MYCIN 系统通过两个相互调用的程序 MONITOR 和 FINDOUT 去推进整个推理（咨询）过程。前者分析相关的规则能否激活，后者则搜索规则激活所需的数据（属性值及其 CF）。正是由于 MONITOR 和 FINDOUT 的相互嵌套调用，导致了深度优先的穷尽搜索过程。为消除因反复对知识库进行穷尽搜索（对于推理树中每个不能通过询问用户来证实的目标，都要搜索一遍知识库)而导致的低效性，MYCIN 将规则按上下文对象分类，使得每次对于一个目标作推理时，只需考虑该目标涉及的那个上下文对象相关的规则，从而大幅度提高了推理的效率。

（2）不确定推理

如前所述，原始证据和推理的结论都可以用可信度 CF 来指示其不确定性，所以如何处理不确定性是 MYCIN 推理机需提供的重要功能。鉴于推理过程生成了与或推理树，MYCIN 的不确定推理既要处理 CF 沿推理链的传递，又要处理 CF 的与或组合（详细阐述参见 7.2.2 节）。

由于推理过程采用深度优先的策略，推理树左部推理链将优先建立。一旦推理链沿伸到原始证据，就可反向计算和传递 CF。MYCIN 系统规定 0.2 是规则前提得以满足的 CF 门槛值，前提 CF 低于 0.2 的规则不能激活，并从推理树中删除，从而使得推理树的规模可以限制在较小的范围内。

（3）治疗选择机制

治疗选择机制处理目标规则 092 的动作部分，先建立若干可能的治疗方案，然后综合制定最佳治疗方案。所谓治疗方案，就是依据推断出的可能病菌（病原体）选用适当的治疗药物。知识库中已包含一组治疗规则，每条规则为一种病菌制定一个药物治疗方案。例如，下面是一条治疗假单胞细菌的规则：

如果：病原体鉴别为假单胞细菌

那么：推荐以下药物作为可选的治疗方法：

1）colistin (0.98)

2) polymyxin (0.96)

3) gentamicin (0.96)

4) carbenicillin (0.96)

5) sulfisoxazole (0.64)

其中，每种药物后的数字指示细菌对该药物的敏感性。

由于导致病人感染疾病的可能病菌及其可信度 CF 已记载于该病人的上下文树，能够由系统依据

这种规则自动生成针对每种可能病菌的治疗方案。

为综合制定最佳治疗方案，MYCIN 遵循以下药物选配准则：1）细菌对药物的敏感性，2）药物是否已给病人配用过，3）药物的相对功效，例如药物是杀菌性的还是抑菌性的，是否会引起病人的过敏反应等。

MYCIN 系统还制定了若干面向药物选配的启发式，例如：

• 若物药-1 是治疗细菌-1 的最优药物，又是治疗细菌-2 的次优药物，则将物药-1 提升为治疗细菌-2 的最优药物。• 某些药不能单独使用。• 不要从同类药物中选配多于一种的药物。

上述药物选配准则和启发式未表示成规则形式存放于知识库，而是直接编程于治疗选择机制中。

一旦综合制定的最佳治疗方案形成，就要依据病人的特别信息（如药物敏感史、年龄等）作禁忌检查。MYCIN 提供了一组规则用于自动检查治疗方案是否违反禁忌，若违反，则将违反禁忌的药物除掉，重新开始治疗方案的综合制定，直到产生不违反禁忌的治疗方案为止。当然，医生对药物的使用有最后决定权，若发现用药不合理，可以要求除掉这些不合理的药物，并重新启动治疗方案的综合制定。

3 系统服务设施

MYCIN 系统提供了多种服务设施去支持系统的应用和维护，包括推理解释、知识库维护和用知识库支持教学等，这些研究丰富了 KB 系统技术，并具有示范作用。

（1）推理解释

鉴于 MYCIN 系统针对医生的每次咨询都建立相应于病人的与或推理树和上下文树，所以系统在推理过程中或推理结束后可以回答医生（用户）对推理过程和推理结果的各种询问。询问分为三种：

1）WHY--主要用于推理过程中系统请求医生提供观测数据时。让医生了解提供所需观测数据的用处和重要性，有时是至关重要的，因为只有医生了解到这些，才会提供更为准确的数据。这可以通过向医生显示相关的规则来实现。例如系统可能会问医生?quot;观察到这种细菌有足够数量吗？″；医生可以反问系统 WHY(通过菜单、按钮)，即相当于问：″该细菌的数量对于诊断很重要么？″。系统可以把应用该观察数据作为激活条件的规则（可能不止一条）显示给医生看，并通过规则的结论″确定病人疾病的相应感染点″，让医生明白索取该观测数据的作用。

2）HOW--主要用于推理结束后，回答医生对推理结果提出的疑问，这可以通过与或推理树中相应于该结果的推理链来给出解释。

3）WHYNOT--医生询问某条规则为何未被使用，系统的解释通常是该规则的前提不能满足。

显然，这些解释设施属于 4.1.1 节中介绍的规则追踪范畴。为了使系统提供的解释适合于不同专业的医生（甚至病人），MYCIN 的研制者于 1982 年提出了能依据用户知识水平加以裁剪的解释；并设置了二个参数：复杂性和重要性，来量化知识单元（规则和对象属性）的可解释性。① 复杂性--用以指示为理解-知识单元，需要用户自身具备的知识水平。② 重要性--为让用户理解给出的解释，该知识单元不可缺少的程度。

与剪裁解释相关的另一工作就是测量用户的实际知识水平和对于解释的深化（细化）要求。通常，用户的实际知识水平可以由用户自己输进系统（系统按预先确定的等级让用户认定），而对于解释的深化要求则由系统依据用户的知识水平自动算出。

MYCIN 的推理解释建立在推理链上，而推理链已记载于推理树中。为解释诊断结果或某一中间结果，首先从推理树中找出相应的推理链，然后对链上的所有知识单元，即规则和出现于这些规则

前提及动作中的对象属性，按用户的知识水平加以裁剪。凡是复杂度落在用户知识水平和解释深化要求之外的知识单元（包括复杂度过高的或过低的），都应修剪掉；只是对于一些重要的知识单元（通过动态设置的门槛值来衡量），可以不计其复杂性而留下。MYCIN 系统已为重要的且复杂性高的推理设置了封装的细化解释（超出规则本身的文字描述），以解释规则的前提和结论间的因果关联细节。所以，一旦超出用户知识水平的但又重要的知识单元保留在推理解释中。相应规则的封装解释可以作为补充说明，帮助用户理解系统给出的解释。例如，考虑以下推理链：

r1　　r2　r3　　r4　r5

A -->B -->C -->D -->E -->F

其中大写字母指示相应于对象属性的知识单元，并分别包含于规则（r1,r2,…,r5）的前提和结论。假设 A、D 和 F 的复杂度符合用户的知识水平和解释细化要求，且在对该推理链的解释中，C 的重要性超过了动态设置的门槛值，则推理链将被系统重新整理为 A->C->D->F。 既然规则 r3（C->D）的复杂度超出了用户可理解的水平，系统就在按新推理链提供解释的同时，将为 r3 封装的细化解释也显示给用户，作为一种补充说明。

（2）知识库维护

MYCIN 系统知识库中包含的推理规则，尽管形式上相互独立，但语义上却相互关联，并由此形成推理树。正是这种语义上的关联，使知识库的维护面临困难，尤其是在知识库包含众多规则的情况下。以增加规则到知识库为例，通常会出现以下三类问题：

1）包含（Subsumption）问题--在知识库包含规则 r: A ∧ B => D 的情况下，增加新规则 r′：A ∧ B ∧ C => D，就会产生包含问题：规则 r′ 的前提包含了规则 r 的前提。显然，这二条规则不应同时存在于]知识库（会引起冗余）。究竟保存哪条规则于知识库取决于是否存在这样的情况：A ∧ B ∧ ¬C => D。

2）单一规则的不一致--在知识库包含规则 A => B 的情况下，增加新规则 A => ¬B，就会产生这种不一致问题。显然，让系统自动发现该问题是容易的。

3）多规则的不一致--在知识库包含规则 A => B，B => C，C => D 的情况下，增加新规则 A => ¬D 就会产生这种不一致问题。由于涉及到一条推理链，要查出这种不一致问题往往既困难又耗时。

MYCIN 系统的推理解释机制为知识库的维护提供了有力的支持。在对知识库作规则的增、删和更新后，知识库维护者和医疗专家可以利用这种解释机制追踪推理链，以及时发现可能的问题。作为与 MYCIN 系统配套的知识获取工具，TEIRESIAS 提供了一个高性能编辑器，去帮助专家小组更新 MYCIN 的知识库。TEIRESIAS 能自动发现前述的包含和单一规则不一致问题，并为专家小组发现多规则不一致问题提供帮助。

（3）教学

MYCIN 的知识库包含了医学专家提供的丰富经验知识，可以作为医疗教学的知识来源。一个称为 GUIDON 的系统实现了基于 MYCIN 知识库的医疗教学。GUIDON 的教学工作从展示一个医疗案例给学生开始，学生在试探性地诊断该案例的过程中，可以请求提供附加的信息，而 GUIDON 则通过追踪学生请求的信息去推测学生使用的问题求解方式。一旦发现学生采用的方式不合适，GUIDON 会中断学生的诊断过程，重定位学生的注意力，或从 MYCIN 知识库中选取适当规则加以示教。

4 开发工具 EMYCIN

EMYCIN 曾是从 MYCIN 系统抽取出的与应用领域无关的一个骨架型专家系统开发工具。由于去除了与细菌感染性疾病的诊断和治疗相关的特别处理机制，可用于开发任何旨在提供咨询服务的专家系统，尤其适合解决故障诊断问题。

EMYCIN 继承了 MYCIN 的主要特点，如下：• 采用逆向链深度优先的控制策略；• 使用产生式规则表示领域知识；• 允许事实和规则具有不确定性（以可信度指示）。

EMYCIN 定义的规则可以用 BNF 范式描述如下：

〈规则〉:=（IF〈前提〉THEN〈动作〉[ELSE〈动作〉]）

〈前提〉:=〈条件〉的与或组合

〈条件〉:=〈关联三元组〉

〈动作〉:=〈关联三元组〉

〈关联三元组〉:=（〈属性〉〈对象〉〈值〉）

EMYCIN 支持逆向推理：顶层目标是一个关联三元组，由 EMYCIN 的最终用户（即专家系统的用户）提出 ；推理机寻找动作与其匹配的规则，而这些规则前提部分的每个条件又可作为子目标。推理控制遵从深度优先的策略去推进解答的搜索过程，直到每个底层子目标都与事实（用户输入的初始或中间数据）匹配为止。可以有多条规则匹配同一子目标，使搜索有多个分支；某分支推理（搜索）失败时，回溯，向另一分支推理。

17.3 认知模拟的生理模式--神经网络方法

利用计算机（智能机器）来模拟人类的认知心理活动，是智能模拟和工程化的核心内容。20 世纪 50 年代以来，基于符号主义和思维模拟的人工智能曾迅速兴起，成为了智能模拟和工程化的主流。Simon 曾将心理学研究分为 3 个水平：第一级是对复杂行为的研究，如问题解决和概念形成等；第二级是对简单信息加工的过程的研究，如感知觉等；第三级是生理水平上的研究，如神经结构和神经过程。Simon 和 Newell 从对人类问题解决的实验研究和计算机模拟中得到启发，提出了信息加工的观点；在 Simon 等人看来，认知心理研究的目的，就是要说明和解释人在完成认知活动时是如何进行信息加工的。而信息加工心理学更认为，智能的本质就是对信息的加工，即对符号的操作，人类的认知过程就是信息加工的工程，也就是符号操作的过程。由此进一步发展为的物理符号系统假设（physical symbol system hypothesis），更将人类所具有的观念、概念、能力以及脑内的加工过程都看作是物理符号事件，由此，我们也就可以把人的心理事件放在物理事件的理论体系中加以探讨，人类思维中的各种抽象的概念和符号也可以像物理对象一样加以复制、转换和相互连接等。

怀特海德在《符号主义》一书中曾写到：“人的心智是以符号的形式进行活动的，即经验的某些成分发出意识、信念和情感，而这些活动又与经验中的其他成分相关联。这里，前一种成分的集合就是‘符号’，后一种成分的集合构成了符号的‘意义’。这种将符号和其意义相连接的过程就是‘符号的指定’”。怀特海德的这一观点无疑得到了符号主义学派的认同。符号主义学派将其阐述为“**符号化的思维和符号化的行为是人类生活中最富于代表性的特征**”。于是，智能模拟和工程化的研究也致力于研究符号在计算机中是如何被表示和被加工的。

在物理符号系统假设中，“符号”具有广义的理解，指的是人们能够加以区分的任何一种模式。

如果一个模式能够表示外界客体及其关系，它就是一个符号。一个物理符号系统就是一组符号，它的功能就是操作符号。一个完整的物理符号系统应该具有对符号进行输入(input)、输出 (output)、存储(store)、复制(copy)、创建符号结构(build symbol structure)和条件性迁移(conditional transfer)的基本功能。物理符号系统假设无疑在人类心理操作和计算机信息处理之间建构了一座桥梁。由于二者的加工原则是一致的，人类复杂的心理结构和过程，就可以用内部机理基本一致的另一种智能形式来表现；通过类比和形式化操作，即可在计算机上建立起人类心理活动的模拟模型。

早期的人工智能研究主要着重于较简单的问题处理、弈棋和难度不大的定理证明等工作，目的多在于探讨人的问题解决策略。20 世纪 70 年代开始，人工智能由理论探索开始转向实际应用，发展了以专家系统为中心的知识工程。在专家系统中包括了大量的专门领域的知识、环境知识和一般常识，知识的表示和运用当时即成为问题的关键。尽管 20 世纪 80 年代，已经开发出了专家系统的商品化成果，如医学专家系统、地矿勘探系统等；人工智能已能用计算机模拟人脑的某些功能，并用于探索人类智能的某些规律，人工智能成果已可用于解决工业、商业、医学等领域的一些决策问题，用人工智能装备的“深蓝”已有战胜象棋大师的记录；在智能模拟方面人工智能取得了不少重大进展；但是，它也遇到了难以克服的许多困难，如图象识别、自然语言理解等对于人脑而言轻而易举的任务，对人工智能计算机却是难题。这使人们开始思考，自上而下、串行加工的信息处理方式，是否就是人脑的信息加工模式？人脑信息处理的灵活可靠，应该用什么形式再现出来？20 世纪 80 年代，在神经生物学的进展和神经网络研究的基础上，以并行分布处理(PDP)为代表的联接主义的神经网络的研究再度兴起，算法上的突破使多层前馈网络在模式识别问题的应用中取得成效，人工神经网络(ANN)又一次成为智能模拟的潮流。如今，综合了多项智能技术的智能计算技术，以及综合了符号智能、计算智能和现场认知智能的智能技术等正成为智能模拟和工程化重要的发展方向。

实际上，人工神经网络的出现是先于人工智能的。早在 1943 年，麦克洛奇（W.S.McCulloch）和皮兹（W.Pitts）就合写了一篇关于神经元如何工作的文章，并用电路构成了简单的人工神经网络模型，即被称为**M-P 模型**的神经元模型。Hebb 于 1949 年提出了突触联结强度的改变是神经记忆的基础的设想，并明确指出了突触可塑性条件，凡在突前纤维与相连的突后细胞同时兴奋时，突触的联结加强，这就是 **Hebb 学习律**。在神经元模型的基础上，人们试图用这些单元组成网络以表现脑的功能。20 世纪 50 年代末，罗圣勃拉特（F. Rosenblatt）提出了著名的视知觉模型—**感知机**(perceptron)。它具有与脑结构相似的结构。简单的感知机共分三层，信息由感知层输入，经过联合层到决策层，由决策层输出，每一层都是由一组神经元组成。感知层与联合层的联结是随机的，联合层与输入层之间的联系是可塑的，它可以通过学习来改变。这样的感知机以神经元模型为单元，在某些方面考虑了神经系统的可能的联结方式，如多层的结构，各层神经元有不同的作用，以及神经元之间联结的可塑性和随机性等。感知机有明确的功能，它能通过“示教”学会正确分类事物。在开始时，因为感知层与联合层之间的联系是随机的，感知机的判断可能不正确，根据判断是否正确，可以训练感知机改变单元层间的联结权重，最终学会正确的分类。感知机通过学习学会分类事物的能力曾引起广泛的关注，并促进了模式识别研究的发展。然而，由于以感知机为代表的早期神经网络缺乏先进的理论和实现技术，信息处理能力低下，甚至连 XOR 这样的简单非线性分类问题也解决不了；加之知识工程的兴起，使得从宏观功能的角度模拟人脑思维行为的研究欣欣向荣，降低了人们从模拟人脑生理结构来研究思维行为的热情。明斯基（Minsky）等人的批评，更直接导致了神经网络研究进入萧条时期，人工神经网络的研究曾陷于停滞状态。直到 1982 年，霍普菲尔特（John

Hopfield）提出具有联想记忆能力的 Hopfield 神经网络模型，根据其对神经网络的数学分析和深入理解，Hopfield 揭示了每种模型的工作原理和实用性；与此同时，人工智能的研究遇到了阻碍，于是，人们再一次把目光投向人工神经网络，才使神经网络的研究再一次兴盛起来。

Aecht Nielsen 为人工神经网络所下的定义是：**“神经网络是由多个非常简单的处理单元按某种方式相互连接而形成的计算系统，该系统是依靠其状态对外部输入信息的动态响应来处理信息的。”**在人工神经网络中，没有确定的存储器，而是由许多互连的简单处理单元组成。其中每个处理单元的功能是求其输入信号的加权和，当该和超过某一阈限时，输出呈现兴奋状态；低于某一阈限是，输出呈现抑制状态。它并不执行指令序列，结果也不存在特定的存储单元中，当它达到某种平衡状态时，网络的整个状态就是所求结果。

从结构的角度看，（人工）神经网络是由许多具有适应性的简单单元[神经元],按照一定的拓扑结构广泛并行互连而成的一种具有并行计算能力的网络系统。它能够模拟生物神经系统对真实世界事物所作出的交互反应,具有较强的非线性拟合能力和多输入/多输出的并行处理能力。（人工）神经网络是人们试图通过模拟人脑神经网络来处理信息的一种尝试，是希望从另一个角度来构建具有人脑信息处理能力的系统。由于对（人工）神经网络的研究和对人的神经系统的研究密不可分，所以人们也把人工神经网络直接称为神经网络。

在感知机的研究中，人们就已经发现，多层前馈神经网络具有较强的信息处理能力，如模式的移动和旋转的不变性。事实上，多层前馈联结是神经系统内信息传递与处理的基本形式。多层前馈网络的第一层为输入层，最后一层为输出层，中间可以有多层，称为隐藏层。每层由一组神经元所组成，各层之间的联结权重均认为是可塑的。20 世纪 80 年代中期，已经提出了不少算法使这种网络可通过学习获得所要求的信息处理能力。其中，鲁梅尔哈特（D. Rumelhart）等人提出的面向多层前馈神经网络的 **BP（Back-Propagation，反向传播）学习算法**最为著名。BP 算法实际上是最优化的一种梯度算法，故有可能会落入局部最小点。但在实际应用中，许多问题都能得到期望的结果。现在已经证明，多层前馈神经网络可以通过学习完成任意非线性变换，是一个**“万能映射（变换）器”**。这说明，我们可能利用这种网络来完成任何信息处理任务。但是，目前的算法都要求根据各层的的许多突触的状况来修改每一个突触的权重，训练需反复进行。一个有限的网络是否可以在有限的时间内完成一个“任意的”学习任务，还是一个须进一步研究的问题。

人的记忆具有联想性质，常常可从一事物联想到另一事物。Laggnst-Higgins 在光学全息存储的启发下，曾提出了实现全息存储的网络模型—**联想记忆网络模型**（associative memory）。这一模型的结构相当简单，由输入输出两组神经元组成，一组神经元与另一组内所有神经元的突触联系，所有的突触都是可塑的。Hopfield 曾把联想记忆模型推向了新阶段。Hopfield 在其研究中引入了能量函数，用非线性动力学研究神经网络的状态变换过程，并指出信息是存储在网络中神经元之间的连接上。他认为，稳定系统的能量函数有趋向最小的性质，稳定平衡点即对应着某种正确的“记忆”。由于在一定的外界条件下网络具有趋于能量最小的特性，因而可以实现特定的联想记忆和优化功能。联想记忆网络有人脑记忆的某些特征，如并行分布存储和容错补全能力，但它仅是记忆系统的局部网络。Hopfield 曾成功地用神经网络解决了具有 NP 复杂度的旅行商计算难题，曾激发了人们对神经网络的研究热情，也推动了波尔兹曼机（一种具有自学习能力的神经网络）和 BP 学习算法等的研究。不过，人脑的记忆存取可以连续进行，不会停止于某一稳定状态，这也是需要继续研究的。

不同于经典人工智能的基于问题解决的符号信息加工，人工神经网络主要基于模仿生物大脑的

结构和功能，来构成一种信息处理系统。尽管有人认为，以并行分布处理为特点的（人工）神经网络，虽然有许多类似人脑信息加工的功能和结构，但只适合低层次的脑内信息处理过程描述。而脑内的高级功能，常常显示为串行处理，因而，基于符号处理的人工智能才是脑内高级功能的合适描述。实际上，脑内所有的信息处理过程，无论低层次还是高层次，都是在脑的神经网络中进行的。神经网络技术的深入发展，有可能建立（类人）神经系统的近似描述。对神经网络的研究，需要集中在三个方面：（1）对认知科学的研究，如对可模拟的知觉、记忆、语言、思维等的研究；（2）利用物理学的方法进行（网络）单元间相互作用理论的研究；（3）数学领域的非线性动力学系统的研究。我们希望能通过多学科的结合，建立起更接近人类认知的神经网络模型；同时，找到人脑信息处理灵活性和创造性的条件和规律，建立起基于人类认知模型的人工神经网络模型，从而开辟出认知模拟的工程途径。

为了模拟人脑信息处理的机能，目前的人工神经网络已显示出了人脑神经系统所具有的很多基本特征，如：① 信息可分布存储，且容量大，容错性较好；② 可大规模并行处理收集到的网络信息；③ 自学习，自组织，自适应性强；有很强的学习功能、联想功能和容错功能；④ 其行为是大量神经元的集体行为，而不是各个（信息处理）单元性能的简单相加；常表现出一般复杂非线性动态系统的特征；⑤ 可以处理一些环境十分复杂，知识背景不清楚和推理规则不明确的问题。目前，人工神经网络已被广泛而深入地进行研究，并开始进入实际应用。其应用的领域包括：系统辨识、模式识别（如语音识别）、智能控制（如家电和工程控制）和机器学习等。

17.3.1 人工神经元和神经网络模型简介

人工神经网络是从人脑的生理结构来模拟人的智能行为，认为智能行为存在于大量神经元的相互连接之中。人类大脑包含大约 10^{11}-10^{12} 个神经元，每个神经元都包括细胞体、树突和轴突三部分。树突接受来自其它神经元的传递信号，轴突通过其末梢将信号传送给其它神经元。两个神经元之间传递信息的接合部称为突触，它通常是一个神经元轴突末梢和另一个神经元树突的接触处。神经元细胞体相当于一个初等处理器，对来自其它神经元的信息综合求和。当其总和超过某一阈值时，细胞体进入兴奋状态，产生一电脉冲（称为激活），并将该脉冲信息沿轴突传播到周围的神经元。为了模拟人脑神经元的这些信息处理功能，在人工神经网络中，神经元通常被简化为一个多输入单输出的非线性阈值器件。其基本模型如图 17.3.1 所示。用数学方式表达则为：

设 $\mathbf{X}=\{x_1, x_2, \cdots, x_n\}$ 为某一神经元 u_i 的 n 个输入，$\mathbf{W}=\{w_{ji}\}$ 表示神经元 u_j 与神经元 u_i 间突触的连接强度，也称为连接权；令 a_i 表示神经元 u_i 的输入加权和，用于指示神经元的活跃程度；0_i 指示其输出；则有：

$$a_i = \sum_{j=1}^{n} w_{ji} x_j \qquad (17.3.1)$$

$$o_i = f(a_i) \qquad (17.3.2)$$

其中，$f(a_i)$ 称为作用函数或传递函数，表示神经元输入和输出间的关系。连接权 w_{ji} 通常取值于区间[-1,+1]。

（1）当 $w_{ji}>0$，称为正连接，表示神经元 u_j 对神经元 u_i 有激活作用；

（2）当 $w_{ji}<0$，称为负连接，表示神经元 u_j 对神经元 u_i 有抑制作用。

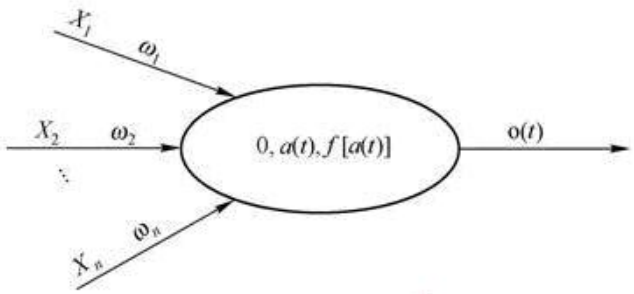

图 17.3.1　神经元模型示意图

常用的作用函数可归结为三种形式：阈值型、S 型和伪线性型。具有阈值型作用函数的神经元是最简单的，其输出状态只有两个可取值：0 和 1，分别指示神经元处于激活和抑制状态。早期的神经网络模型 M-P 就使用了这种阈值型作用函数，神经元的输出以数学表达式表示为

$$o_i = f(a_i) = \begin{cases} 1 & a_i > 0 \\ 0 & a_i \le 0 \end{cases} \tag{17.3.3}$$

采用 S 型作用函数的神经元具有连续的输出取值函数，常见的 S 型作用函数以指数、对数或双曲线表示神经元的非线性输入-输出特性，例如

$$o_f = f(a_i) = \frac{1}{1+e^{-a_i}} \tag{17.3.4}$$

采用伪线性型作用函数的神经元在一定的区间内满足线性输入-输出特性，输出可表达为（c，A 均为常量）

$$o_i = \begin{cases} 0 & i \le 0 \\ ca_i & 0 < a_i \le A \\ 1 & A < a_i \end{cases} \tag{17.3.5}$$

多个神经元以一定方式连接在一起，即构成神经网络。现代人工神经网络从模拟人脑的感知行为出发，希望基于大量神经元间的连接来实现感知信息的大规模并行、分布式存储和处理，并提供自组织、自适应和自学习能力，特别是希望能用它来处理涉及诸多因素和条件的、不精确和模糊的信息处理问题。为此，所构造的神经网络往往有比较复杂的联结结构，最常见的是三层网络结构。在典型的层次网络，如 BP 网络中，相邻层次的神经元间形成全互连连接，而同层次中的神经元之间没有连接。用作联想存储器的 Hopfield 模型则只有单一层次，且各神经元之间是全互连的。由于单个神经元的功能极其有限，神经网络的特性和能力主要取决于神经网络的互连结构。

人工神经网络的一个重要特征是其自适应学习能力。神经网络的最基本学习规则是 Hebb 规则，所有的学习规则都可视为 Hebb 规则的变种。该规则的基本思想是：如果一个单元 u_i 从另一单元 u_j 接收输入，且这两个单元都处于兴奋（激活）状态，则从 u_j 到 u_i 的连接权 w_{ji} 应加强。该规则可表示为：

$$\Delta w_{ji} = g(a_i(t), t_i(t)) \cdot h(o_j(t), w_{ji}) \tag{17.3.6}$$

其中，$a_i(t)$ 和 $t_i(t)$ 是神经元 u_i 在 t 时刻的实际和基准活跃值，$o_j(t)$ 是 t 时刻神经元 u_j 的输出值。连接权 w_{ji} 的修正量由关于 $a_i(t)$ 和 $t_i(t)$ 的函数 g 以及关于 $o_j(t)$ 和 w_{ji} 的函数 h 的乘积确定。学习规则的最简单形式是没有基准活跃值，也不计时间特性，且 g 和 h 都是关于第一参数的正比例函数，则学习规则简化为

$$\Delta w_{ji} = \eta x_i o_j \tag{17.3.7}$$

其中 η 是影响学习速率的比例常数。更通用的形式为

$$\Delta w_{ji} = \eta \cdot (t_i - a_i) \cdot o_j$$

令 $\delta_i = t_i - a_i$，则

$$\Delta w_{ji} = \eta \delta_i o_j \tag{17.3.8}$$

称为 Delta 学习规则，被应用于早期的以罗圣勃拉特提出的感知机为代表的依赖于阈值型作用函数的神经网络。现行的多层（主要是三层）神经网络在输入层和输出层之间加入一个隐含层，以求通过抽取输入空间的特征来划分输入空间，实现出强大的对于输入信息的模式分类（识别）能力。模式分类可视为从输入到输出的映射，理论上，只要在隐含层设置足够多的神经元，就可实现任意的映射。因此，神经网络也被认为是“万能映射转换器”。然而，隐含层的加入也增加了学习算法的难度和复杂度，推动了多种优化学习算法的研究。神经网络是一个高度非线性动力学系统。

17.3.2 面向映射变换的 BP 神经网络

BP 神经网络是面向映射变换的神经网络中应用最广泛的一种，其结构如图 17.3.2 所示。

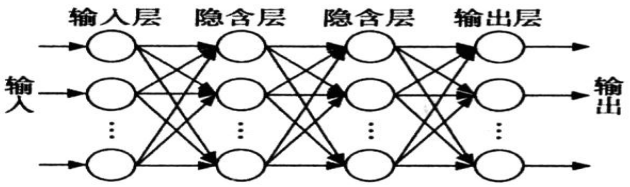

图 17.3.2　BP 神经网络结构示意图

典型的 BP 网络有三个层次：输入层、隐含层和输出层，相邻层次神经元间采用全互连形式，同层神经元间则不相连。通常，设 **X**={ $x_1, x_2, \cdots, x_m$} 为系统 m 个输入信息；{ $v_1, v_2, \cdots, v_k$} 为系统 k 个隐节点；Y={ $y_1, y_2, \cdots, y_n$} 为系统输出信息。连接输入层与隐含层的权值记为 **W**={w_{ik}}，连接隐含层与输出层的权重记为 V={v_{kj}}。

对于输入层神经元，其输出与输入相同。输入层神经元与中间隐含层神经元的操作特性表示为：

$$a_k = \Sigma w_{ik} \bullet o_i \quad o_k = f(a_k) \quad i=1,2,\cdots,m;\ k=1,2,\cdots,k; \tag{17.3.8}$$

作用函数 f 设计为非线性的输入-输出关系，一般选用下面形式的 S 型函数：$f(a_k)=1/(1+e^{-ak})$.

中间隐含层和输出层的神经元的操作特性表示为:

$$b_j = \Sigma v_{kj} \bullet o_k \quad o_j = g(b_j) \quad k=1,2,\cdots,k; j=1,2,\cdots,n. \tag{17.3.9}$$

鲁梅尔哈特等人为 BP 网设计了依据反向传播的误差来调整神经元连接权的学习算法，有效地解决了多层神经网络的学习问题。该算法的基本思路是：当给网络提供一个输入模式时，该模式由输入层传到隐含层，经隐含层神经元作用函数处理后传送到输出层，再经由输出层神经元作用函数处理后产生一个输出模式。如果输出模式与期望的输出模式有误差，就从输出层反向将误差逐层传送到输入层，把误差“分摊”给各神经元并修改连接权，使网络实现从输入模式到输出模式的正确映射。对于一组训练模式，可以逐个用训练模式作为输入，反复进行误差检测和反向传播过程，直到不出现误差为止。这时，BP 网就完成了学习阶段，具备所需的模式分类（识别）能力。

BP 网学习算法是典型的有导师指导的学习，是 Delta 学习算法的拓广。设 BP 网神经元的作用函数 S 型，训练集包括 m 个样本。每个样本包括一个训练模式（作为 BP 网的输入）和一个期望模式（作为相应于训练模式所期望的 BP 网络标准输出）。记第 p 个训练模式输入后神经元 ui 的活跃值为 a_i^p，输出为 o_i^p；神经元 u_i 的输入连接有 n 个神经元，每个神经元 uj 的输出为 o_j^p（j＝1，2，…，n），则有

$$a_i^p = \sum_{j=1}^{n} w_{ji} \cdot o_j^p \tag{17.3.10}$$

$$o_i^p = f(a_i^p) = 1/(1 + e^{-a_i^p}) \tag{17.3.11}$$

神经网络中各神经元的连接权初始值可在一定的约束条件下随意设置，致使对于每个输入模式 P，网络的输出与期望模式有一定误差。定义 BP 网某个层输出误差为

$$E = \sum_{p=1}^{m} E_p \tag{17.3.12}$$

$$E_p = \frac{1}{2}\sum_{i=1}^{n} (t_i^p - o_i^p)^2 \tag{17.3.13}$$

其中，E 为训练集所有训练模式产生的输出误差之和，t_j^p 为神经元 u_i 的输出期望（标准输出）。Delta 学习规则的实质是利用梯度下降法，使神经元 u_i 连接权沿误差函数 E_p 的负梯度方向改变。记相应于训练模式 p 而作的权修正量为 $\Delta_p w_{ji}$，则定义

$$\Delta_p w_{ji} \propto -\frac{\partial E_p}{\partial w_{ji}} \tag{17.3.14}$$

由于

$$\frac{\partial E_p}{\partial w_{ji}} = \frac{\partial E_p}{\partial a_i^p} \cdot \frac{\partial a_i^p}{\partial w_{ji}}$$

而从公式(17.3.10)可得

$$\frac{\partial a_i^p}{\partial w_{ji}} = o_j^p$$

再令　　$\delta_i^p = -\frac{\partial E_p}{\partial a_i^p}$

用其表示误差修正量，则有

$$\Delta_p w_{ji} = \eta \delta_i^p o_j^p \tag{17.3.15}$$

这就是前述的 Delta 学习规则，其中 η 为比例常数。接下来的处理是求得与网络中各神经元 u_i 相应的 δ_i^p，这可以通过将神经网络输出层发现的误差反向传播到输入层来计算。对于输出层神经元，则有

$$\begin{aligned}\delta_i^p &= -\frac{\partial E_p}{\partial a_i^p} = -\frac{\partial E_p}{\partial o_i^p} \cdot \frac{\partial o_i^p}{\partial a_i^p} \\ &= -[-(t_i^p - o_i^p)] \cdot f'(a_i^p) = f'(a_i^p) \cdot (t_i^p - o_i^p)\end{aligned} \tag{17.3.16}$$

式中，$(t_j^p - o_j^p)$ 反映输出层神经元 u_i 的输出误差，$f'(a_j^p)$ 则作为对于输出误差的比例因子，按神经元 u_i 的活跃值动态影响误差修正量。

对于隐含层的神经元 u_i，无法直接获取输出误差，可以令

$$\frac{\partial E_p}{\partial o_i^p} = \sum_{k=1}^{n} \frac{\partial E_p}{\partial a_k^p} \cdot \frac{\partial a_k^p}{\partial o_i^p} = -\sum_{k=1}^{n} \delta_k^p w_{ik} \tag{17.3.17}$$

则有

$$\delta_i^p = f'(a_i^p) \cdot \sum_{k=1}^{n} \delta_k^p w_{ik} \qquad (17.3.18)$$

由于隐含层神经元 u_i 的误差修正量是通过对所有与 u_i 相连的输出层神经元 u_k 的误差修正量 δ_k^p 求加权和（权为 w_{ik} 的）来获得的，所以称为误差反向传播。如此，公式(17.3.15)、(17.3.16)、(17.3.17)联合构成拓广的 Delta 学习算法，称为 BP 学习算法。

从输入训练模式、计算输出误差，到经过误差反向传播来调整输出层和隐含层神经元的连接权，完成一个学习周期。连接权的调整是个迭代过程，为使实际输出模式达到期望的输出模式，往往需经由多个学习周期的迭代。在这个拓广的 Delta 学习算法中，η仍表示学习速率。η较大时，权值修正量就较大，学习速率比较快，但有时可能产生振荡，使误差修正量 δ_i^p 不能平稳变小。而当η较小时，权值修正量就较小，学习速度减慢，但 δ_i^p 能平稳变小。在实际应用中，可引入适当的附加项，使学习过程快速平稳地收敛。

由于引人隐含层，BP 网能完成从输入模式到输出模式的复杂映射，大大提高了神经网络的分类能力。

BP 网具有很强的模式分类能力，这种能力是将隐含层单元作为分类特征抽取器而获取的，反向传播学习算法为此提供了保证，然而学习算法存在着收敛速度较慢、学习过程易于陷入局部极小（假饱和）状态、隐含层神经元数量的确定缺乏理论指导等问题，需要通过深入的研究来改进。

17.3.3 面向联想记忆的 Hopfield 神经网络简介

根据神经网络运行过程中的信息流向，人们一般将神经网络分为前馈式和反馈式两种基本类型。前馈网络的输出仅由当前输入和权矩阵决定，而与网络先前的输出状态无关。反馈网络则不同。由于人们希望神经网络除了具有映射和模式分类能力之外，还希望其具备联想记忆能力。因而，研究面向联想记忆的反馈网络也就成为了神经网络研究的一个重要方面。Hopfield 曾提出一种单层反馈神经网络，后来人们将这种反馈网络称作 Hopfield 网络。这类全互连反馈型神经网络是提供联想记忆功能的典型，其通过引入网络的能量函数和稳定性概率，可产生出非线性动力学系统所表现出的丰富的动态特性。

一个典型的 Hopfield 神经网络通常如图 17.3.2 所示。

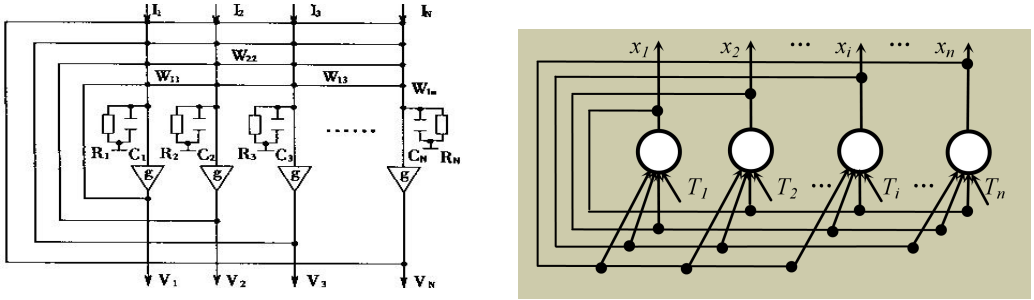

图 17.3.3　一个典型的 Hopfield 神经网络

1. 二值 Hopfield 网络

Hopfield 网络分为离散型和连续型两种网络模型，分别记作 DHNN (Discrete Hopfield Neural Network) 和 CHNN (Continues Hopfield Neural Network)。

二值 Hopfield 网络只有一个神经元层次，神经元间全互连，且具有对称连接，即 $w_{ji}=w_{ij}$，每个神经元 u_j 只有两个状态，以输出值 x_j 指示：0 或 1。神经元作用函数的计算规则如下：

$$x_j = f(net_j)$$

$$net_j = \sum_{i=1}^{m}(w_{ij}\cdot x_i) - q_j)$$

$$x_j = sgn(net_j) = \begin{cases} 1, net_j \geq 0 \\ 0, net_j < 0 \end{cases}$$

（j=1, 2, …, n ）　　(17.3.19)

其中 T_j 是神经元 u_j 激活（x_j=1）的阈值。显然，当网络包含 n 个神经元时有 2^n 个可能状态。在任意给定的时刻，网络状态表示为 X=(x_1，x_2，…，x_n)，其是只包含 0 和 1 的向量，或由"＋"和"-"构成的符号向量（"＋"对应于 1，"-"对应于 0）。

研究表明，Hopfield 网络具有非线性动力学系统的两个重要特征：（1） 有若干稳定状态，网络总可以从某一初始状态变动到这些稳定状态之一。（2） 网络的稳定状态可以通过设置神经元间的适当连接权来确定。

稳定状态是实现联想记忆的基础，每个稳定状态对应于一个记忆样本。神经网络变动到稳定状态的基本工作方式有两种：·串行异步方式--任意时刻随机地或确定性地选中网络中的一个神经元进行状态更新，其余神经元的状态保持不变。·并行同步方式--任意时刻可以有多个神经元的状态同时更新。

可以证明，对于具有对称连接、无自身反馈的二值 Hopfield 神经网络，若网络状态按串行异步方式更新，则必定收敛于某一稳定状态。引人网络状态的能量定义有助于研究网络的状态变迁。神经元 u_i 的状态能量定义为

$$E_i = -a_i x_i = -(\sum_{j\neq i} w_{ji} x_j - q_i) x_i \qquad (17.3.20)$$

依据二值 Hopfield 网络的作用函数，神经元 u_i 总是向能量减小的状态变迁。变迁只有两种情况：a_i＞0，x_i=0，则 x_i 转变为 1，能量从 0 下降为负值；a_i＜0，x_i=1，则 x_i 转变为 0，能量从正值下降为 0。整个神经网络的状态能量则定义为所有神经元状态能量之和：

$$E = -\frac{1}{2}\sum_i\sum_{j\neq i} w_{ji} x_j x_i + \sum_i q_i x_i \qquad (17.3.21)$$

其中常量 1 / 2 的引入是因为神经元能量求和时，能量定义的第一项被重复计算两次。当所有神经元都不能改变其状态时，神经网络到达稳定状态，其对应于神经网络状态能量的极小点。所以只要把需记忆的样本信息存储（记忆）于稳定状态，当神经网络接受某一输入模式（作为初始状态）时，就会将状态向能量减小方向变迁，直到某一稳定状态，从而导致对已记忆样本的联想。

神经网络稳定状态（即能量极小点）的配置由连接权（和阈值）决定，有以下两种方式：（1）根据联想记忆需求（要记忆的样本状态）直接计算神经元间的连接权（和阈值），以静态确定状态。（2）通过适当的学习算法，自动调整连接权（和阈值），以动态产生期望的稳定状态。

若网络状态按串行异步方式更新，且随机地选择神经元进行状态变迁，则变迁倾向于能量最接近的稳定状态，但有时这会引起错误的联想记忆。这种联想记忆的错误在需记忆的样本超出神经网络有效记忆容量时会迅速加剧。霍普菲尔特通过计算机仿真实验确定，对于具有 n 个神经元的二值 Hopfield 网络，记忆容量约为 0.15n 个样本。可以通过调整神经网络结构（增加神经元，改变连接权和阈值等）或采用梯度最快下降（转变到能量下降得最多的下一状态）等启发式知识来消除记忆错误，但仍然不能解决所谓的能量局部极小问题。这个问题导致神经网络有可能变迁到系统的能量局部极小状态，而不能到达全局最小状态。

2．连续值 Hopfield 网络

连续值 Hopfield 网络的拓扑结构与二值 Hopfield 网络相同，但每个神经元 u_i 的状态取值（输出值）是随时间连续变化的。神经元作用函数取 S 型：

$$x_i = f(a_i) = \frac{1}{1+e^{-a_i}} \tag{17.3.22}$$

为使神经元活跃值随时间连续变化，在实际实现时常采用具有饱和非线性的反馈放大器模拟神经元的活跃规律。其拓扑结构可如图 17.3.3 所示。此时，每个神经元 u_i 由一个反馈放大器模拟，包括输入电阻 r_i 和输入电容 C_i，以刻画 u_i 活跃值的时间常数。将神经元 u_i 的输入端视为树突，神经元的输出端视为轴突。两者的突触连接由电导 G_{ji}（其为连接电阻 R_{ji} 的倒数）定义。为区分突触连接的兴奋性和抑制性，每个神经元（反馈放大器）有两个输出：同相（V_i）和反相（$-V_i$），连接电阻 R_{ji} 与其中之一接通。此外，每个神经元均有一个用于设置其激活电平的外界输人偏置电流 I_i。此时，神经元 u_i 活跃值的状态变化规律可表示为：

$$C_i \frac{da_i}{dt} = \sum_{j=1}^{n} G_{ji} V_j - \frac{a_i}{R_i} + I_i \tag{17.3.23}$$

$$V_i = x_i = f(a_i) \tag{17.3.24}$$

其中 R_i 为 u_i 输入电阻 r_i 与各连接电阻 R_{ji} 的并联电阻，

$$\frac{1}{R_i} = \frac{1}{r_i} + \sum_{\substack{j=1 \\ j \neq i}}^{n} \frac{1}{R_{ji}} \tag{17.3.25}$$

从上述对于连续值 Hopfield 网络的描述，可以看出，单个神经元的特性与实际生物神经元相比是十分简单的。但作为模型，生物神经元的基本特性得到体现：神经元输入到输出的响应具有平滑的 S 型曲线，神经元互连的时空效应，突触连接的兴奋性和抑制性。

连续值 Hopfielid 网络的能量可定义为

$$E = -\frac{1}{2}\sum_i \sum_{j \neq i} G_{ij} V_j V_i - \sum_i V_i I_i + \sum_i \frac{1}{R_i} \int_0^{V_i} f^{-1}(V) dV \tag{17.3.26}$$

其中，$f^{-1}(V)$是 $V=f(a_i)$ 的反函数。可以证明，只要 f^{-1}为单调递增连续函数，则能量函数 E 单调递减。所以，从任意初始状态出发，连续值 Hopfield 网络均能朝能量单调减小的方向变迁，并最终到达（或渐近）能量全局最小状态。

连续值 Hopfield 网络为解决优化问题开辟了新途径，其基本思想是把优化问题映射为连续值 Hopfield 网络，通过网络状态的动态演变自动趋向稳态，从而实现优化解答的自动搜索。霍普菲尔特于 1984 年就应用这种方法成功地解决了具有 NP 复杂度的旅行商问题，引起世人的震惊。

17.3.4 随机神经网络--波尔兹曼（Bollzmann）机

许多应用需要将网络能量全局最小的稳定状态作为问题解答，如优化问题、约束满足问题求解等。欣顿（Hinton）等人通过引进随机扰动机制和模拟退火算法，提出遵从波尔兹曼效应的神经网络模型-波尔兹曼机，较好地解决了摆脱能量局部极小状态，进人能量全局最小状态的问题。

波尔兹曼效应指示，气体分子的随机运动所具有的能量与温度直接相关，温度越高，气体分子的运动越快。这种效应可以比拟为神经网络中状态变迁的波动，使变迁不仅转向能量减少的状态，也可以一定的概率转向能量增加的状态。温度高时，状态变迁波动大，有利于神经网络摆脱能量局部极小状态；而温度降低时，有利于稳定在能量全局最小状态。

波尔兹曼机在结构上与前述典型的 Hopfield 神经网络相同，主要的不同在于当神经元 u_i 被选定作状态变迁时，其作用函数 $f(a_i)$是一个随机函数。依据该函数，u_i 下一状态为 1（即 $x_i=1$）的概率为

$$p_i(1) = \frac{1}{1 + e^{-a_i/T}} \tag{17.3.27}$$

式中，T 表示网络的“温度”参数，常取[0，1]区间上的某个值。$p_i(1)$ 也称为激活概率函数（x_i=1 表示神经元激活），是 S 型函数。u_i 下一状态为 0（即 xi＝0）的概率则为

$$p_i(0) = 1 - p_i(1) \tag{17.3.28}$$

显然，激活概率函数的应用使 u_i 能以一定的概率转变到能量增加的状态（而不像二值 Hopfield 网络那样，只能转变到能量减小的状态），使神经网络能有机会跳出能量局部极小状态。在这个意义上，激活概率函数成为状态变迁波动的动力。对于相同的 a_i，T 增大，神经元向能量增大的状态变迁的机会增大（即 a_i＞0 时状态变为 0，而 a_i＜0 时状态变为 1）。相反，温度变低，这种变迁机会减小。当温度趋于 0 时，能量增加的变迁不可能，波尔兹曼机退化为 Hopfield 网络。

采用模拟退火技术是通过上述随机神经网络解决能量局部极小问题的有效方法。为产生金属的低能态晶格，可以先将金属加温到熔化，使金属中的粒子处于高能态，可自由运动，然后逐渐降温，使金属结晶。只要在接近凝固点时温度下降足够慢，粒子就会摆脱局部应力的束缚，形成具有最低能态的晶体。类似地，模拟金属的退火过程，只要神经网络的初始“温度”选得足够高（以促进摆脱能量局部极小状态），温度下降速度足够慢，就可使神经网络到达能量全局最小状态。

模拟退火算法的基本步骤如下：（1）设置 $X=S_i$（当前状态 X 为某一任意的初始状态 S_i），$T=T_0$（初始温度）；（2）随机选择某个邻近状态 S_j（一般指只需变动单一神经元状态后产生的神经网络下一状态）；（3）若该邻近状态为一能量增加状态，则在［0，1］区间上产生一个均匀分布的随机数 β，若 exp［△E／T］≤β （△E 表示增加的能量），转（2）；（4）令 $X=S_j$，检查神经网络是否到达热平衡态；（5）若不满足结束条件，则令 T=λT（λ取（0，1）区间上某个值），转（2）；（6）算法结束。

算法中要检查的热平衡态是一种概率意义下的稳态。此时，网络状态仍可变化，但状态出现的概率分布不再变化。足够高的温度，将使热平衡态出现在能量全局最小状态的某个邻近范围内。而温度的降低则使这个范围缩小。只要温度下降足够慢，可以确保热平衡态不落入以某个能量局部极小状态为中心的"陷阱"内。通常算法第（5）步中的 λ 取值于 0.2-0.99 之间，并需根据应用问题特征确定。结束条件可以通过检查神经网是否已达能量全局最小状态（其热平衡态范围足够小）来确定，具体检查策略尚无一般化规则。

17.3.5 神经网络的工程实现与工程应用

神经网络的工程实现技术可归入两条途径：专用硬件和软件模拟。前者采用硬件系统中处理单元及连接与神经网络中神经元及连接一一对应的方式，即每个神经元和每个突触连接均有与之相应的物理器件，可以是数字式的或模拟式的。专用硬件是为特别的应用需求专门设计的，以求获得较高的工作效率；但缺乏通用性、灵活性和可编程性。软件模拟是在通用计算机上经由软件编程来模拟实现神经网络的拓扑结构和运行，具有通用性强、灵活性好的优点，适用于理论研究、应用研究和实时性要求不高的实用系统。

在通用计算机上模拟实现神经网络曾经是普遍采用的也是最经济的方式。曾经实际应用于计算机上的开发工具和环境也有许多种，为神经网络的研究和应用提供了很大方便。但由于通用计算机存储容量和速度的限制，导致速度慢，不利于体现神经网络的并行分布处理特点，因而出现了带加速板的计算机。**加速板**是一种协处理器，作为通用机的插件，可提高神经网络的规模和速度。当然，也可以根据神经网络的结构特点和信息处理方式设计通用并行处理机阵列。以前研究得较多的是 Systolic 阵列机（也称脉动阵列机）。在最佳情况下，具有 n 个处理单元的 Systolic 陈列机，其处理效率是单个处理单元的 n 倍。如今基于大模型的智能系统，其算力是重要基础，而其算力中心所采用的目前主要是英伟达的芯片。

采用专用硬件实现的神经计算系统有**数字神经计算机。**它可采用现已成熟的 VLSI 等技术来制造。

这种专用硬件的主要问题在于神经网络中存在广泛的互连，连线占用大量芯片面积，且连线间的寄生电容给电子布线造成困难。新型的三维 VLSI 制造技术可以作为克服此困难的方法。光学技术是实现光学神经计算机的一个比较理想的选择，具有广阔的发展和应用前景。光学处理元件通过光束互联，无须导线，也摆脱了 VLSI 芯片的布线困难，具有互连性高、通信带宽大、速度高的优点。光学处理元件的弱点是缺乏执行复杂逻辑运算的能力，而这种能力恰好对神经计算并不重要。当然，光学计算技术目前尚不太成熟，但其应用潜力已引起人们关注。采用生物工程技术将有机分子或人工蛋白质制作为计算机元件--生物器件，是硬件实现神经网络的另一发展方向。由于生物器件的尺寸是分子量级的，故生物[分子]神经计算机将具有体积很小、集成度极高和运行速度十分快的优点。

神经网络的软件模拟需通过编程来实现。尽管从理论上讲，神经网络不需要编程，因为其是根据外界环境自适应调节网络参数来完成信息处理的。然而模拟实现需作从虚拟处理单元（即神经元）到物理处理单元的映射，这正是要由软件编程来解决的。软件模拟实现不仅要求开发人员有熟练的编程技术，也应对神经网络的拓扑结构和学习算法有较深刻的理解。为此，提供神经网络描述语言和开发环境，将大大减轻开发人员的编程负担，加速理论研究和应用开发过程。

神经网络描述语言通常分三个部分来描述为解决应用问题所需的神经网络：拓扑结构、运作规则和运作环境。拓扑结构包括神经网络的分层（分块）、神经元间的互连方式、连接权和阈值、作用函数和学习规则的定义。运作规则指定神经网络在学习和工作状态下神经元操作的时序和策略。运作环境定义神经网络的外部环境和信息交互方式，使神经网络能有效地从外部环境接收数据（如需识别的图象信息），并将处理结果传送到外部环境。目前专用于神经网络的通用描述语言尚不多。得到广泛应用的曾有 AXON 和 ANSPEC，用这两种语言描述的神经网络可以编译执行。

开发环境为神经网络的应用研究人员提供工作平台，支持应用的快速开发。例如 CaseNet 是一个面向 PC 机的神经网络软件开发环境，它提供图形用户接口，使用户可直接在图形屏幕上绘制网络结构，并由菜单输入网络特性。如此生成的神经网络定义被编译为优化的机器代码，以实现高效的网络运作。CaseNet 开发环境包括五个主要工具：图形网络编辑器、网络语法分析器、网络分析器、网络代码生成器和网络编译器。

在神经网络实现技术的支持下，神经网络的实用化已出现于许多商业产品，并加速向工业系统推广。线性神经网络由于具有结构简单和易于训练的优点，已广泛应用于模式分类只需线性处理的工商业领域。例如，远程通信中，电话通道 Modems 用的均衡器就可使用神经网络来实现；神经网络也被用于在空调和其它自动系统中主动控制振动和噪音，在高能加速器中控制粒子束等。

多层次非线性神经网络技术的兴起及算力芯片价格的大幅度下降，推动了神经网络技术向工商业的广泛渗透。比如，非线性模式分类技术已成功地应用于检查信用卡使用中的欺诈行为、印刷体和手写体的识别、制造业的质量控制、石油勘探和医疗卫生等。联想记忆技术则可用于辅助决策支持，如评价商业贷款风险（结合常规的模式分析技术）、市场分析和预测（如优化市场策略、鉴别信誉不良的客户等），预测航班客流量和分配座位等。

控制和优化也是适合于神经网络技术应用的广阔领域。1990 年代后，神经网络与模糊逻辑技术结合，已应用于诸多家用电器产品，如空调、取暖器、炊具、微波炉、冰箱、洗衣机、照相机等。工业上的应用也很多，如已实用的有电弧炉电极状态控制、半导体制作过程控制、化学过程控制、石油提炼过程控制（温、压、流速等）、钢生产中的连续铸造控制等。有潜力的非线性神经网络应用也有导弹引导和起爆控制、战机飞行和战役模式指导、光学望远镜聚焦、生物医学分析等。随着大规模神经网络实现技术的开拓，工业应用的前景将更趋明朗。

神经网络或神经计算技术最突出的应用应该是它在大模型方面的应用。如今，几乎所有的大模型都是基于神经网络技术来实现的。对此，我们将在 17.3.7 节做进一步介绍。

17.3.6 关于神经网络的一些理论研究简介

在智能模拟和工程化领域，神经网络无疑是一个十分值得重视的发展方向。多年来，有关神经

计算的研究一直受到了人们极大的重视，并取得了大量成果，但也暴露出了很多以往研究中存在的不足，有待进一步去深入研究。下面，仅介绍其部分研究成果。

1．关于神经网络的模拟模型

作为对人类认知模拟的一类生理模式，神经网络的研究，其基础应是各类神经模型的构建。尽管经过多年的研究，人们已经提出了功能和结构各具特色的众多的神经网络模型，但是，这些模型无疑都是对生物神经系统的某种简化。脑科学的众多研究，曾给神经网络的研究以多方面的启示；但脑科学的成果，或如大脑皮层的功能定位般过于宏观，或如许多神经生理学的实验般过于微观，尚未能使我们对大脑微观结构与思维宏观过程之间的纽带有一个清楚的了解。尽管认知的生理模拟并不一定是人脑真实的思维过程，但对真实的思维的了解无疑是有赖于认知神经科学的研究的，也只有在此基础上也才有可能获得理想的认知模型。人类希望从突触、神经元等微观水平到全脑、行为等宏观水平上理解在感知客体、形成表象、使用语言、记忆特征、实施推理时的信息加工过程及其机制，而这无疑是十分艰巨的任务，需要认知神经科学的更丰富的结论。而生理模型的建立，更需要认知神经科学和智能模拟领域的专家通力合作。比如，我们知道，在人脑内，神经元的兴奋是人脑信息的主要载荷者；各种不同的神经元兴奋模式，反映着脑内信息传递和加工的不同内容；而各种抑制作用，也使得外部传入的信号受到精细的加工。脑内的这些不同的兴奋性和抑制性调控在脑功能实现中的作用，就是很值得探索的。而这种探索还只是初步的。比如，（1）**侧抑制网络**。侧抑制是神经系统信息处理的基本形式。Hartline 及其同事曾从鲎的视网膜神经网络入手，深入研究了这种抑制方式的生物学原理和数学模型。这种侧向抑制在生物传递通路中是普遍存在的现象。侧抑制网络在信息处理中可有如下功能：突出边框、增强图形反差；空间频率的高通滤波器，抑制低频成分和噪声；对由光学系统的缺陷引起的成像模糊有补偿作用等。（2）**反馈抑制网络**。反馈抑制在中枢神经系统中广泛存在。一般来说，兴奋性神经元受到刺激兴奋后，除由轴突传出兴奋信息到其他神经元去之外，还有轴突侧支使兴奋性中间神经元兴奋。中间神经元的兴奋又反过来输入到兴奋性神经元，使其兴奋受到抑制。通常多个兴奋性神经元的轴突末梢会聚于一个中间抑制性神经元，一个抑制性神经元的输出又发散到多个兴奋神经元，呈相互交叉。反馈抑制网络在信息加工的中的优点有：可保证在一个区域内神经元的总兴奋水平较低，并大致恒定，这样可减少能量消耗，且形成稀疏编码，稀疏编码可增加系统的存储容量；可使兴奋模式分化，使不同输入的兴奋模式更接近与正交，有利于存储和加工；可提供对某一脑区的总兴奋水平进行控制的条件。另外，通过远距控制输入到抑制性中间神经元群，可以改变脑区内总的兴奋水平。注意和选择系统可能通过这些抑制神经元实现所需的控制，等等。如何据此构建合适的网络模型，是值得我们去进一步研究的。

2．关于神经网络学习理论的一些深入研究[1738]

目前，尽管神经计算已经在很多领域得到了成功的应用，但是，由于缺少一个统一的理论框架，神经网络的构建和学习训练，经验性成分相当高。这使得研究者们难以对各种神经计算模型的性能及其适用范围进行理论分析，仅能用不十分可靠的实验性比较来评价其优劣。另一方面，在利用神经计算解决问题时，也只能采取具体问题具体分析的方式，通过大量费力耗时的实验摸索，确定出合适的神经网络模型、算法以及参数设置。这些缺陷已经对神经计算的进一步发展造成了极大的阻碍。如果能提供一套比较完备的理论方法，将可望解决上述问题。

已有很多研究者进行了这方面的研究，力图在一个统一的框架下来考虑网络的学习与泛化的问题。比如，PAC（Probably Approximately Correct）学习模型就曾是这样一个框架。作为 PAC 学习的

核心以及学习系统学习能力的度量，VC 维（Vapnik-Chervonenkis dimension）在确定神经网络的容量（capacity）、泛化能力（generalization）、训练集规模等的关系上有着重要作用。如果可以计算出神经网络的 VC 维，则我们或可以估计出要训练该网络所需的训练集规模；反之，在给定一个训练集以及最大近似误差时，我们也可以确定所需要的网络结构。联系到 Hornik 等人所证明的结论，即**“仅有一个隐层的网络就可以以任意精度去逼近任何函数，但确定该网络的结构却是一个 NP 难问题”**，显然，神经网络 VC 维计算的研究对神经网络的发展将会产生极大的促进作用。

学习系统的容量对其泛化能力有重要影响。低容量学习系统只需要较小的训练集，高容量学习系统则需要较大的训练集，但其所获的解将优于前者。对给定训练集来说，高容量学习系统的训练集误差和测试集误差之间的差别将大于低容量学习系统。Vapnik 曾指出，对学习系统来说，训练集误差与测试集误差之间的差别是训练集规模的函数，该函数可以由学习系统的 VC 维表征。换言之，VC 维表征了学习系统的容量。

Anthony 将 VC 维定义为：设 F 为一个从 n 维向量集 X 到{0, 1}的函数族，则 F 的 VC 维为 X 的子集 E 的最大元素数，其中 E 满足：对于任意 $S \subseteq E$，总存在函数 $f_s \in F$，使得当 $x \in S$ 时 $f_s(x)=1$，$x \notin S$ 但 $x \in E$ 时 $f_s(x)=0$。

VC 维可作为函数族 F 复杂度的度量，它是一个自然数，其值有可能为无穷大，它表示无论以何种组合方式出现均可被函数族 F 正确划分为两类的向量个数的最大值。对于实函数族，可定义相应的指示函数族，该指示函数族的 VC 维即为原实函数族的 VC 维。

为便于讨论，这里只针对典型的二元模式识别问题进行分析。设给定训练集为{(x_1, y_1)，(x_2, y_2)，…，(x_l, y_l)}，其中 $x_i \in \mathrm{R}^n$，$y \in \{0, 1\}$。显然，x_i 是一个 n 维输入向量，y 为二值期望输出。再假设训练样本与测试样本均满足样本空间的实际概率分布 $P(x, y)$。

对基于统计的学习方法来说，学习系统可以由一族二值函数$\{f(x, \alpha), \alpha \in \Lambda\}$表征，其中参数$\alpha$可以唯一确定函数 $f(\cdot)$，Λ为α所有可能的取值集合。因此，$\{f(x, \alpha), \alpha \in \Lambda\}$的 VC 维也表征了该学习系统的复杂度，即学习系统的最大学习能力，我们称其为该学习系统的 VC 维。学习的目的就是通过选择一个参数α^*，使得学习系统的输出 $f(x, \alpha^*)$与期望输出 y 之间的误差概率最小化，即出错率最小化。出错率也称为期望风险（Expected Risk），其值

$$R(\alpha)=\int \frac{1}{2}\left|y-f(\mathbf{x},\alpha)\right| dP(\mathbf{x},y) \qquad (17.3.29)$$

其中 $P(x, y)$为样本空间的实际概率分布。由于 $P(x, y)$通常是未知的，因此无法直接计算 $R(\alpha)$。但是，对给定的训练集，其经验风险（Empirical Risk）$R_{emp}(\alpha)$却是确定的，即

$$R_{emp}(\alpha)=\frac{1}{2l}\sum_{i=1}^{l}\left|y_i-f(\mathbf{x}_i,\alpha)\right| \qquad (17.3.30)$$

其中(x_i, y_i)为训练样本，l 为训练集中样本数，即训练集规模。由数理统计中的大数定理可知，随着训练集规模的扩大，$R_{emp}(\alpha)$将逐渐收敛于 $R(\alpha)$。

基于统计的学习方法大多建立在经验风险最小化原则基础上，其思想就是利用经验风险 $R_{emp}(\alpha)$代替期望风险 $R(\alpha)$，用使 $R_{emp}(\alpha)$最小的 $f(x, \alpha_l)$来近似使 $R(\alpha)$最小的 $f(x, \alpha_0)$。这类方法有一个基本的假设，即如果 $R_{emp}(\alpha)$收敛于 $R(\alpha)$，则 $R_{emp}(\alpha)$的最小值收敛于 $R(\alpha)$的最小值。Vapnik 与 Chervonenkis 证明，该假设成立的充要条件是函数族$\{f(x, \alpha), \alpha \in \Lambda\}$的 VC 维为有限值。

Vapnik 还证明，期望风险 $R(\alpha)$满足一个上界，即任取η满足 $0 \leqslant \eta < 1$，下列边界以概率 $1-\eta$成立：

$$R(\alpha) \le R_{emp}(\alpha) + \sqrt{\frac{h(\ln(2l/h)+1) - \ln(\eta/4)}{l}} \qquad (17.3.31)$$

其中 h 为函数族$\{f(x, \alpha), \alpha \in \Lambda\}$的 VC 维，$l$ 为训练集规模。式(17.3.31)右侧第二项通常称为 VC 置信度（VC Confidence）。由式(17.3.31)可以看出，在学习系统 VC 维与训练集规模的比值很大时，即使经验风险 $R_{emp}(\alpha)$较小，也无法保证期望风险 $R(\alpha)$较小，即无法保证学习系统具有较好的泛化能力。因此，要获得一个泛化性能较好的学习系统，就需要在学习系统的 VC 维与训练集规模之间达成一定的均衡。

由于神经网络也是一种基于统计的学习方法，因此其 VC 维也满足关于一般学习系统的讨论。从功能上来说，每一个权值、阈值等参数都已被确定的神经网络就相当于一个函数。不妨假设网络有 n 个输入神经元，m 个输出神经元，则该网络对应于一个函数 $f: R^n \to R^m$。如果我们用α来表示所有权值、阈值等可调参数的集合，Λ表示该集合可能的取值集合，则神经网络的学习过程可被视为通过选择α^*来确定函数$f(x, \alpha^*)$。这样，如果求出函数族$f(\cdot)$的 VC 维，我们就得到了该神经网络的 VC 维。由于神经网络的 VC 维取决于网络中可调参数的数目，而后者又是由网络的拓扑结构所确定的，因此，网络的 VC 维与拓扑结构之间有必然的联系。在给定训练集的情况下，如果我们能求出合适的 VC 维，则可以帮助确定网络的结构；反之，在给定网络结构的情况下，如果我们能求出其 VC 维，则可以确定合适的训练集规模。显然，这对寻找 Hornik 指出的最优解有重要的启发作用。

对 VC 维的研究是有成效的。比如，Cover 最早进行了神经网络 VC 维的计算工作，在此之后，Vapnik 在相关的统计学方面做了大量的工作，他们的成果与 Blumer 等人在计算学习理论方面的工作一起，被 Valiant 引入 PAC 学习模型中。从此，神经网络 VC 维的计算受到了极大的重视。1997 年，Vidyasagar 通过构造性方法证明，神经网络的 VC 维并不一定是有限的。因此，对神经网络 VC 维的讨论必须针对一定的网络结构，这样才能保证 VC 维为有限值。目前，这方面的研究成果主要集中在以阈值函数（threshold function）、分段多项式函数（piecewise polynomial function）和 *Sigmoid* 函数为响应函数的神经网络上。而对其在将来的研究，文献[1738]认为，

(1) 对于阈值前馈网络和分段多项式前馈网络，其 VC 维已经得到了充分的研究，特别是前者，已经得到了对实际应用具有指导意义的 VC 维具体数值 $w\log w$。但是，由于这两类网络的学习能力非常有限，实际使用得最多的是以 *Sigmoid* 函数或 *Gaussian* 函数等连续函数为响应函数的前馈网络。目前，关于此类网络的 VC 维研究还有待深入。虽然 *Sigmoid* 前馈网络已得到了一些研究结论，但其 VC 维上界 $O(w^2k^2)$与下界$\Omega(w^2)$之间的差距太大，对实际应用缺乏指导意义。如何尽可能缩小该差距，将是一个重要研究课题。

(2) 目前，对神经网络 VC 维的研究主要侧重于确定其上、下界。由于求出的上、下界之间往往有较大的差距，使得该领域的研究成果难以直接反映到神经网络结构设计、训练集样本选择中。如果能找到一种方法，可以确定网络 VC 维的具体值，将有重要的实际意义。在这方面的研究中，Carter 和 Oxley 的方法也许是一个值得深入的研究方向。

(3) 神经网络技术发展到现在，其模型、算法种类已非常多，它们在解决不同的问题时往往具有不同的能力。例如，循环神经网络在处理时序问题时其效果远优于前馈网络。但是，目前 VC 维的研究主要集中在前馈网络上，对其他类型的网络还研究得很少。这方面的工作如能得到加强，将有助于改善各种类型的神经网络在实际应用中的性能。

(4) 近年，神经网络集成（neural network ensemble）已成为神经网络界研究的热点。由于集成的目的是通过充分利用训练样本来改善网络的泛化能力，因此，集成中网络的类型、结构与训练集规模有密切的关系。集成中虽然包含多个网络，但从更高的抽象级来看，整个集成也可以看作一个学习系统。因此，神经网络集成也应该具有 VC 维，如能找到它的计算方法，将对集成技术的发展起到重要的促进作用。

(5) 由于 VC 维考察的是学习系统在最坏情况下的样本复杂度，因此其结论通常是比较“悲观”的，与实际应用时的需要有较大的偏差。此外，在进行 VC 维分析时，通常假定训练样本是“一致可学习”的，即训练样本均匀地分布在样本空间中。但在实际应用中，该假设往往很难满足。由此可知，VC 维本身仍存在一些缺陷。如能改进 VC 维分析方法，或提出更好的方法，将对神经网络乃至整个机器学习技术的发展起到深远的影响。在这方面，Haussler 等人和 Takahashi 等人已进行了一些研究，并取得了初步的成果。

3．关于神经网络集成的一些研究[1738]

由于缺乏严密理论体系的指导，神经计算技术的应用效果常常取决于使用者的经验。虽然 Hornik 等人证明，只需一个具有单隐层的前馈网络就可以逼近任意复杂度的函数，但如何找到合适的网络配置却是一个 NP 问题。在实际应用中，由于缺乏问题的先验知识，往往很难找到理想的网络结构，这就影响了网络泛化能力的提高。如果能建立一套方法，避开网络配置的问题，从另一个角度寻求提高学习系统泛化能力的途径，具有重要的现实意义。

1990 年，Hansen 和 Salamon 曾提出了一种方法，即神经网络集成（neural network ensemble），为上述问题的解决提供了一个简易可行的方案。使用这种方法，可以简单地通过训练多个神经网络并将其结果进行合成，显著地提高学习系统的泛化能力。由于其易于使用且效果明显，因此，对神经网络集成的研究不仅会促进神经计算乃至所有统计学习方法的理论研究，还会极大地促进神经计算技术进入工程应用的进程。

Kearns 和 Valiant 指出，在 PAC 学习模型中，若存在一个多项式级学习算法来识别一组概念，并且识别正确率很高，那么这组概念是强可学习的；而如果学习算法识别一组概念的正确率仅比随机猜测略好，那么这组概念是弱可学习的。Kearns 和 Valiant 又提出了弱学习算法与强学习算法的等价性问题，即是否可以将弱学习算法提升成强学习算法。如果两者等价，那么在学习概念时，我们只要找到一个比随机猜测略好的弱学习算法，就可以将其提升为强学习算法，而不必直接去找通常情况下很难获得的强学习算法。

上述等价性问题可视为神经网络集成思想的出发点。Sollich 和 Krogh 曾将神经网络集成定义为：“神经网络集成是用有限个神经网络对同一个问题进行学习，集成在某输入示例下的输出由构成集成的各神经网络在该示例下的输出共同决定”。但是，也有一些研究者认为，神经网络集成指的是多个独立训练的神经网络进行学习并共同决定最终输出结果，并不要求集成中的网络对同一个（子）问题进行学习。

在神经网络集成的研究中，始终存在着两方面的内容。一方面，研究者们试图设计出更有效的神经网络集成实现方法，以直接用于解决问题。另一方面，研究者们试图对神经网络集成进行理论分析，以探明这种简单的方法为何有效、在何种情况下有效，从而为实现方法的设计提供指导。

对神经网络集成实现方法的研究主要集中在两个方面，即：怎样将多个神经网络的输出结论进行结合，以及如何生成集成中的各网络个体。在结论生成方法方面，当神经网络集成用于分类器时，

集成的输出通常由各网络的输出投票产生；采用的方法包括绝对多数投票法（某分类成为最终结果当且仅当有超过半数的神经网络输出结果为该分类）或相对多数投票法（某分类成为最终结果当且仅当输出结果为该分类的神经网络的数目最多）。理论分析和大量试验表明，后者优于前者。因此，在对分类器进行集成时，目前大多采用相对多数投票法。当神经网络集成用于回归估计时，集成的输出通常由各网络的输出通过简单平均或加权平均产生。Perrone 等人认为，采用加权平均可以得到比简单平均更好的泛化能力。但是，也有一些研究者认为，对权值进行优化将会导致过配（over-fitting），从而使得集成的泛化能力降低，因此，他们提倡使用简单平均。此外还存在多种结合方式。例如，有些研究者利用神经网络这样的学习系统，通过学习来对多个预测进行结合；有些研究者通过对一组子网进行进化，使各子网都可以较好地处理一个输入子空间，从而一步步地进行结合。

在生成集成中个体网络方面，最重要的技术有 Boosting 和 Bagging。Boosting 最早由 Schapire 提出，Freund 对其进行了改进。通过这种方法可以产生一系列神经网络，各网络的训练集决定于在其之前产生的网络的表现，被已有网络错误判断的示例将以较大的概率出现在新网络的训练集中。这样，新网络将能够很好地处理对已有网络来说很困难的示例。另一方面，虽然 Boosting 方法能够增强神经网络集成的泛化能力，但是同时也有可能使集成过分偏向于某几个特别困难的示例。因此，该方法不太稳定，有时能起到很好的作用，有时却没有效果。值得注意的是，Schapire 和 Freund 的算法在解决实际问题时有一个重大缺陷，即它们都要求事先知道弱学习算法学习正确率的下限，这在实际问题中很难做到。后来，Freund 和 Schapire 提出了 AdaBoost（Adaptive Boost）算法，该算法的效率与 Freund 算法很接近，却可以非常容易地应用到实际问题中。

Bagging 方法中，各神经网络的训练集由从原始训练集中随机选取若干示例组成，训练集的规模通常与原始训练集相当，训练例允许重复选取。这样，原始训练集中某些示例可能在新的训练集中出现多次，而另外一些示例则可能一次也不出现。Bagging 方法通过重新选取训练集增加了神经网络集成的差异度，从而提高了泛化能力。Breiman 指出，稳定性是 Bagging 能否发挥作用的关键因素，Bagging 能提高不稳定学习算法的预测精度，而对稳定的学习算法效果不明显，有时甚至使预测精度降低。学习算法的稳定性是指如果训练集有较小的变化，学习结果不会发生较大变化，例如，k 最近邻方法是稳定的，而判定树、神经网络等方法是不稳定的。Bagging 与 Boosting 的区别在于：Bagging 的训练集的选择是随机的，各轮训练集之间相互独立，而 Boosting 的训练集的选择不是独立的，各轮训练集的选择与前面各轮的学习结果有关；Bagging 的各个预测函数没有权重，而 Boosting 是有权重的；Bagging 的各个预测函数可以并行生成，而 Boosting 的各个预测函数只能顺序生成。对于象神经网络这样极为耗时的学习方法，Bagging 可通过并行训练节省大量时间开销。此外还存在多种个体生成方法。例如，有些研究者利用遗传算法来产生神经网络集成中的个体；有些研究者使用不同的目标函数、隐层神经元数、权空间初始点等来训练不同的网络，从而获得神经网络集成的个体。

对神经网络集成的理论分析与对其实现方法的研究类似，也分为两个方面，即对结论生成方法的分析，以及对网络个体生成方法的分析。在实际应用中，由于各个独立的神经网络并不能保证错误不相关，因此，神经网络集成的效果与理想值相比有一定的差距，但其提高泛化能力的作用仍相当明显。

在结论生成方法分析方面，Hansen 和 Salamon 证明，对神经网络分类器来说，采用集成方法能够有效提高系统的泛化能力。假设集成由 N 个独立的神经网络分类器构成，采用绝对多数投票法，

再假设每个网络以 $1-p$ 的概率给出正确的分类结果，并且网络之间错误不相关，则该神经网络集成发生错误的概率 p_{err} 为：

$$p_{err} = \sum_{k>N/2}^{N} \binom{N}{k} p^k (1-p)^{N-k} \qquad (17.3.32)$$

在 $p<1/2$ 时，p_{err} 随 N 的增大而单调递减。因此，如果每个神经网络的预测精度都高于 50%，并且各网络之间错误不相关，则神经网络集成中的网络数目越多，集成的精度就越高。当 N 趋向于无穷时，集成的错误率趋向于 0。在采用相对多数投票法时，神经网络集成的错误率比式(17.3.32)复杂得多，但是 Hansen 和 Salamon 的分析表明，采用相对多数投票法在多数情况下能够得到比绝对多数投票法更好的结果。在实际应用中，由于各个独立的神经网络并不能保证错误不相关，因此，神经网络集成的效果与理想值相比有一定的差距，但其提高泛化能力的作用仍相当明显。

Perrone 和 Cooper 证明，在将神经网络集成用于回归估计时，如果采用简单平均，且各网络的误差是期望为 0 且互相独立的随机变量，则集成的泛化误差为各网络泛化误差平均值的 1/N，其中 N 为集成中网络的数目；如果采用加权平均，通过适当选取各网络的权值，能够得到比采用简单平均法更好的泛化能力。

常用的一些神经网络模型在学习过程中容易陷入局部极小，这通常被认为是神经计算的主要缺点之一。然而，Perrone 和 Cooper 却认为，这一特性对神经网络集成泛化能力的提高起到了重要作用。这是因为，如果各神经网络互不相关，则它们在学习中很可能会陷入不同的局部极小，这样神经网络集成的差异度（Variance）就会很大，从而减小了泛化误差。换句话说，各局部极小的负作用相互抵消了。

Krogh 和 Vedelsby 给出了神经网络集成泛化误差计算公式。假设学习任务是利用 N 个神经网络组成的集成对 $f: R^n \to R$ 进行近似，集成采用加权平均，各网络分别被赋以权值 w_α，并满足 $w_\alpha > 0$ 和 $\sum_\alpha w_\alpha = 1$；再假设训练集按分布 $p(x)$随机抽取，网络 α 对输入 X 的输出为 $V^\alpha(X)$，则神经网络集成的输出为：

$$\overline{V}(X) = \sum_\alpha w_\alpha V^\alpha(X) \qquad (17.3.33)$$

神经网络 α 的泛化误差 E^α和神经网络集成的泛化误差 E 分别为：

$$E^\alpha = \int dx p(x)(f(x)-V^\alpha(x))^2 \qquad (17.3.34)$$

$$E = \int dx p(x)(f(x)-\overline{V}(x))^2 \qquad (17.3.35)$$

各网络泛化误差的加权平均为：$\overline{E} = \sum_\alpha w_\alpha E^\alpha$　　(17.3.36)

神经网络 α 的差异度 A^α和神经网络集成的差异度 $\overline{A}$ 分别为：

$$A^\alpha = \int dx p(x)(V(x)-\overline{V}(x))^2 \qquad (17.3.37)$$

$$\overline{A} = \sum_{\alpha} w_{\alpha} A^{\alpha} \tag{17.3.38}$$

则神经网络集成的泛化误差为：$E = \overline{E} - \overline{A}$。

式中的 $\overline{A}$ 度量了神经网络集成中各网络的相关程度。若集成是高度偏向（biased）的，即对于相同的输入，集成中所有网络都给出相同或相近的输出，此时集成的差异度接近于 0，其泛化误差接近于各网络泛化误差的加权平均。反之，若集成中各网络是相互独立的，则集成的差异度较大，其泛化误差将远小于各网络泛化误差的加权平均。因此，要增强神经网络集成的泛化能力，就应该尽可能地使集成中各网络的误差互不相关。

在个体生成方法分析方面，Freund 和 Schapire 以 AdaBoost 为代表，对 Boosting 进行了分析，并证明该方法产生的最终预测函数 H 的训练误差满足

$$H = \prod_{t} [2\sqrt{\varepsilon_t (1-\varepsilon_t)}] = \prod_{t} \sqrt{1-4\gamma_t^2} \le \exp(-2\sum_{t} \gamma_t^2) \tag{17.3.39}$$

其中 ε_t 为预测函数 h_t 的训练误差，$\gamma_t = 1/2 - \varepsilon_t$。

从上式可以看出，只要学习算法略好于随机猜测，训练误差将随 t 以指数级下降。在此基础上，Freund 和 Schapire 用 VC 维对 Boosting 的泛化误差进行了分析。设训练例为 m 个，学习算法的 VC 维为 d，训练轮数为 T，则其泛化误差上限为

$$\hat{\Pr}(H(x) \ne y) + \hat{O}(\sqrt{\frac{Td}{m}}) \tag{17.3.40}$$

其中 $\hat{\Pr}(\cdot)$ 表示对训练集的经验概率。

上式表明，若训练轮数过多，Boosting 将发生过配。但大量试验表明，Boosting 即使训练几千轮后仍不会发生过配现象，而且其泛化误差在训练误差已降到零后仍会继续降低。为解释这一现象，Schapire 等人从边际（margin）的角度对泛化误差进行了分析。边际 margin(x, y)定义为：

$$m\arg in(x,y) = y\sum_{i=1}^{n} \alpha_i h_i(x) \tag{17.3.41}$$

正边际表示正确预测，负边际表示错误预测，较大的边际可信度较高，较小的边际可信度较低。

Schapire 等人认为，在训练误差降为零后，Boosting 仍会改善边际，即继续寻找边际更大的划分超平面，这就使得分类可靠性得到提高，从而使泛化误差得以继续降低。进一步，Schapire 等人还具体地给出了泛化误差的上限：

$$\hat{\Pr}[m\arg in(x,y) \le \theta] + \hat{O}(\sqrt{\frac{d}{m\theta^2}}) \tag{17.3.41}$$

从上式可以看出，Boosting 的泛化误差上限与训练轮数无关，Schapire 的一些实验也证实了这一点。然而，Grove 和 Schurmans 指出，Schapire 等人的边际假说并不能真正解释 Boosting 成功的原因。为证明这一点，他在 AdaBoost 的基础上设计了 LPBoost 算法，通过线性规划来调整各预测函数的权重，从而增大最小边际。Grove 指出，如果边际假说成立，那么 LPBoost 算法产生的学习系

统泛化误差应比较小，然而实验表明，该学习系统的泛化误差并不小，也就是说，边际的增大并不必然导致泛化误差的减小，有时甚至造成泛化误差增大。因此，关于 Boosting 为什么有效，目前仍然没有一个被广泛接受的理论解释。

Breiman 对 Bagging 进行了理论分析。他指出，分类问题可达到的最高正确率以及利用 Bagging 可达到的正确率分别为：

$$r^* = \int \max_j P(j \mid x) P_X(x) \tag{17.3.42}$$

$$r_A = \int_{x \in C} \max_j P(j \mid x) P_X(dx) + \int_{x \in C'} [\sum_j I(\phi_A(x) = j) P(j \mid x)] P_X(x) \tag{17.3.43}$$

其中 C 表示序正确（order correct）的输入集，C'为 C 的补集，$I(\cdot)$为指示函数（Indicator Function）。显然，Bagging 可使序正确集的分类正确率达到最优，单独的预测函数则无法做到这一点。

对回归问题，Breiman 推出

$$|E_L \varphi(x, L)|^2 \le E_L \varphi^2(x, L) \tag{17.3.44}$$

不等号左边为 Bagging 的误差平方，右边为各预测函数误差平方的期望。显然，预测函数越不稳定，即式(17.3.44)右边和左边的差越大，Bagging 的效果越明显。除此之外，Breiman 还从偏向（bias）和差异（variance）的角度对泛化误差进行了分析。他指出，不稳定预测函数的偏向较小、差异较大，Bagging 正是通过减小差异来减小泛化误差的。在此之后，Wolpert 和 Macready 具体地给出了泛化误差、偏向和差异之间的关系：

$$E(C \mid f, m, q) = \sum_d P(d \mid f, m)(h_d(q) - f^*(q))^2 = (h^*(q) - f^*(q))^2 + \sum_d P(d \mid f, m)(h_d(d) - h^*(q))^2 \tag{17.3.45}$$

式中，左边为泛化误差，右边第一项为偏差的平方，第二项为差异。Bagging 就是对 $h^*(q)$进行模拟，使得在偏差相同的情况下差异尽量趋向于零。

值得注意的是，虽然利用偏向和差异来解释 Bagging 获得了一定的成功，但 Freund 和 Schapire 通过一系列实验指出，偏向和差异并不能很好地解释 Boosting。

文献[1738]认为，目前，在神经网络集成的研究中仍然存在着很多有待解决的问题：(1) 关于神经网络集成的研究目前基本上是针对分类和回归估计这两种情况分别进行的，这就导致了多种理论分析以及随之而来的多种不同解释的产生。如果能为神经网络集成建立一个统一的理论框架，不仅可以为集成技术的理论研究提供方便，还有利于促进其应用层面的发展。(2)关于 Boosting 为什么有效，虽然已有很多研究者进行了研究，但目前仍然没有一个可以被广泛接受的理论解释。如果能够成功地解释这种方法背后隐藏的东西，不仅会促进统计学习方法的发展，还会对整个机器学习技术的进步发挥积极作用。(3) 现有研究成果表明，当神经网络集成中的个体网络差异较大时，集成的效果较好，但如何获得差异较大的个体网络，以及如何评价多个网络之间的差异度，目前仍没有较好的方法。如果能找到这样的方法，将极大地促进神经网络集成技术在应用领域的发展。(4) 在使用神经网络集成，尤其是 Boosting 类方法时，训练样本的有限性是一个很大的问题，Bagging 等算法正是通过缓解该问题而获得了成功。如何尽可能地充分利用训练数据，也是一个很值得研究的重要课题。(5) 神经网络的一大缺陷是其“黑箱性”，即网络学到的知识难以被人理解，而神经网络集成则加深了这一缺陷。目前，从神经网络中抽取规则的研究已成为研究热点，如果能从神经网络集成中抽取规则，则可以在一定程度上缓解集成的不可理解性。

4．关于模糊神经网络的一些研究

长期以来，神经网络与模糊技术的结合一直是神经计算中的一个研究热点。该领域的研究不仅包括模糊逻辑和神经网络的结合研究，还包括对模糊神经网络的理论和应用研究等。

我们一直看好神经网络技术与模糊技术的结合。认为，在智能化信息处理领域，特别是不确定性信息处理领域，神经网络技术和模糊逻辑技术的完美结合，是非常有发展潜力的方向。神经网络以模拟人脑结构的方式来模拟人的思维功能，具有较强的自适应学习能力、大规模并行处理能力和联想功能，擅长处理感知信息；它人工干预少，模拟精度高，对专家知识的依赖也较少；但缺点是不能处理和描述模糊信息，不能很好利用已有的经验知识；特别是学习及问题的求解具有黑箱特性，其工作不具有可解释性；同时它对样本的要求较高，只能应用于有足够大且健全的训练集可以获取的场合。而且，由于神经网络的工作不能为人直观理解，当神经网络给出的解答不符合期望时，难以解释和调试（纠正）。基于模糊逻辑的推理系统能直接应用学科领域的专门知识，并以模糊规则浓缩地表示；具有推理过程容易理解、专家知识利用较好、对样本的要求较低等优点；但它同时也存在人工干预多、推理速度慢、很难实现自适应学习等不足；如何自动生成和调整隶属度函数和模糊规则，也是一个棘手的问题。可见，神经网络技术和模糊逻辑技术具有较强的互补性。如果将二者有机地结合起来，可以起到互补的效果。

对于模糊神经网络，人们已有深入研究，我们也曾作过深入探讨。但是，对模糊神经网络的理论研究，仍有继续深入的必要。比如，如何根据不同问题类型的要求构建不同类型的模糊神经网络，我们还缺乏必要的理论指导。在知识的导入和知识的导出方面，我们还缺乏完美的进化机制。

6．关于神经网络的其他方面的一些研究

经过半个多世纪的研究，神经计算目前已成为一门日趋成熟、应用面日趋广泛的学科，其研究是卓有成效的。上面，我们介绍了神经网络在模型研究、学习理论、网络集成以及与模糊技术相结合等方面的一些相关研究，并提出了一些有待进一步研究的问题。其实，除了上述内容之外，神经计算中还有很多值得深入研究的重要领域，例如：

• 与符号学习相结合的混合学习方法的研究。通过符号主义与联接主义的结合，可以在一定程度上模拟不同层次思维方式的协作，并能在不同学习机制之间取长补短。这已被认为是当前机器学习的一大研究方向。

• 循环神经网络（Recurrent Neural Networks）的研究。由于善于处理与上下文有关的内容信息，循环神经网络已经在自然语言处理、语音识别、孤立词识别等领域得到了成功的应用。可以预料，随着多媒体技术的发展，循环神经网络的应用面将会日趋广泛，其重要性也会越来越明显。

• 支持向量机（Support Vector Machine）的研究。支持向量机是 Vapnik 等人提出的一类新型机器学习方法。由于其出色的学习性能，该技术已成为机器学习界的研究热点，并在很多领域都得到了成功的应用，如人脸检测、手写体数字识别、文本自动分类等。

• 容错神经网络的研究。一些研究者指出，神经网络在本质上并不具有容错能力，其容错性需要通过适当的机制增强。目前已有一些研究者进行了这方面的研究，可以预料，随着神经网络硬件实现技术的发展，其容错性的改善将日趋重要。

• 神经网络与遗传算法、人工生命的结合研究。进化神经网络已在相当长时期内受到了研究者的关注，在人工生命出现之后，神经网络与遗传算法的结合被认为是再现智能行为的一个很有希望的途径。Terzopoulos 甚至指出，随着计算机图形学技术的发展，利用人工生命技术产生复杂图形学模型将是一个重要研究方向，而神经网络将在其中扮演重要的角色。

我们认为，就像人类大脑的智力活动应该从进化的角度、从生理和心理的角度来研究才能得到深入理解一样，智能模拟也应是一个多种模拟方法综合、逐步完善的过程。人的思维并不是纯粹的逻辑结构，而是形象思维与逻辑思维、直觉（感情、顿悟、创新）和逻辑并重的过程，且深受经验的影响。领域专家既有高深的理论修养，也有丰富的经验知识。因此，在彻底弄清脑的结构和机理

之前，充分重视思维模拟和直觉感知模拟的作用，对于推动智能模拟和工程化有极为重要的意义。过分地强调“逻辑”而忽略经验和形象思维的作用，是不对的；只满足于过分简化的“神经网络”而不思进取，也是不对的。智能模拟需要整合认知模拟和思维模拟、人工智能和神经网络等多方面的力量来进行探索，如此，我们才有可能获得丰富而有意义的成果。

17.3.7 基于神经网络技术的 AI 大模型及其核心技术

17.3.7.1　基于神经网络技术的 AI 大模型

AI 大模型（AI Large Model）是指一类具有大规模参数和复杂计算结构的实用智能系统。AI 大模型作为一个已被验证可行的方向，能够处理海量数据、完成各种复杂的任务，如自然语言处理、计算机视觉、语音识别、多模态分析与融合、疑难问题解答，等等。其“大”主要体现在训练数据集广泛，模型参数和层数大，计算量大。AI 大模型也被称为 AI 基础模型（ AI Foundation Model），其价值主要体现在其智能应用的通用性方面，并且它还具有较好的泛化能力和功能增强能力。

AI 大模型通常是基于**深度神经网络**构建而成的，它本质上就是一个通过海量数据训练而成的包含各类人类知识的深度神经网络模型。它可拥有数十亿甚至数千亿个参数。AI 大模型开发的目的主要是为了提高模型的（基于知识的）表达能力和预测性能，能够处理更加复杂的任务和数据。它可通过训练海量数据来学习复杂的模式和特征，由于其巨大的数据和参数规模，（隐约）实现了智能的涌现，从而可展现出类似人类的智能。而当模型的训练数据和参数不断扩大，直到达到一定的临界规模后，它可表现出一些未能预测的、更复杂的能力和特性。AI 大模型能够从原始训练数据中自动学习并发现新的、更高层次的特征和模式，这种能力也被称为“涌现能力”。而具备涌现能力的机器模型就被认为是独立意义上的大模型。AI 大模型还具有强大的泛化能力，可以对未见过的数据做出准确的预测。

与此相对应地，AI 小模型通常是指那些参数较少、层数较浅的模型。它们通常具有轻量级、高效率、易于部署等优点，常适用于数据量较小、计算资源有限的场景，例如移动端应用、嵌入式设备、物联网等。相比小模型，AI 大模型除了参数较多、层数较深外，还具有更强的表达能力和更高的准确度，但也需要更多的计算资源和时间来训练和推理，因而更适用于数据量较大、计算资源充足的场景，例如云端计算、高性能计算、通用人工智能等。AI 大模型相较于由传统特定领域训练出来的智能系统模型，有更广泛的应用场景。

AI 大语言模型（AI Large Language Model）是 AI 大模型目前最成功的一类，是具有大规模参数和计算能力的自然语言处理类模型。这些模型是通过大量的数据（语料）和参数训练而成的语言处理类智能系统，可依用户要求生成与人类自然类似的文本或回答自然语言类的问题。AI 大型语言模型在自然语言处理、文本生成和智能对话等领域已有广泛应用。其中，GPT（Generative Pre-trained Transformer）类的大模型都是基于 Transformer 架构的语言类（并正扩展至多模态类的）智能系统模型。GPT 类模型已经可以生成自然语言文本并处理各种自然语言处理任务，如文本生成、翻译、摘要等。它通常在单向生成的情况下使用，即根据给定的文本生成连贯的输出。而 ChatGPT 则更专注于交互式对话；它经过特定的训练，可更好地处理多轮对话和上下文理解；可提供流畅、连贯和有趣的对话体验，响应用户的输入并生成合适的回复。

若追溯 AI 大模型的起源与发展，我们可以清晰地看到其深受**深度学习**技术发展历程的影响。深度学习，作为一种模拟人类大脑工作原理的机器学习技术，通过构建多层神经网络来实现对复杂数据的深入学习与理解。在过去的数十年间，深度学习技术经历了多次重要突破与创新，包括**多层感知机（MLP）、卷积神经网络（CNN）、循环神经网络（RNN）、深度残差网络（ResNet）以及 Transformer 模型等**。随着数据量的激增与计算能力的提升，研究人员开始致力于构建更大规模、更加复杂的神经网络模型以进一步提升模型的表征能力与泛化能力。这些大型模型，如 BERT、GPT、DeepSeek 等的出现，已标志着 AI 大模型的时代的来临。

如今，AI 大模型的核心能力，一是其**自然语言处理能力。**包括文本分类、情感分析、机器翻译、问答系统、文本生成等任务；它能够理解和生成人类语言，实现与人类的书面和口语交流。二是其**多模态（文字、图像、视频或音频）信息的识别（辨识）与合成（融合）能力。**它可以对图像中的对象、场景、特征等进行识别和分类，例如识别照片中的人物、物体、地点等；可将语音转换为文本，或者将文本转换为语音，实现语音交互功能。

AI 大模型在实际应用中已展现出诸多优势。一是具有强大的**表征能力**，能够深入理解复杂数据模式与特征，从而在各类任务中表现出色。二是具有较强的**泛化能力**，通过在大规模数据集上进行预训练，它能够学习到**通用的特征表示**，进而适应不同领域与任务的需求。三是具有**多模态融合能力**，它支持多种类型数据的处理与融合（如文本、图像、语音等），为更丰富的应用场景提供了可能。四是具有**自动化特征提取能力**，它能够自动学习数据特征表示，减少了人工设计特征的工作量，提高了模型效率与准确性。五是具有**持续迭代与优化能力**，它具备可迭代性特点，能够不断通过大规模数据进行迭代与优化，进而提升模型性能与精度。

当然，AI 大模型在实际应用中也存在一定的局限性。一是**计算与存储资源需求大**：AI 大模型的训练与推理过程对计算资源与存储空间提出了较高要求，增加了硬件成本与部署难度。二是**可解释性还不足**：由于 AI 大模型的复杂性较高，其内部结构与决策过程往往难以被直观理解与解释，这在一定程度上限制了其在某些领域的应用范围。三是**存在数据隐私与安全风险**：AI 大模型的训练依赖于大量数据资源，这可能导致数据隐私泄露与安全风险增加的问题出现。四是存在**过拟合与泛化能力不足**的情况：特别在小样本或少样本场景下，AI 大模型可能面临拟合问题且泛化能力不足的情况，需要有针对性地进行调优与改进工作。五是**具有环境依赖性**：AI 大模型的性能可能受到环境、数据分布及任务特性等多种因素的影响，需要在不同环境下进行适应性调整与优化工作以确保其稳定运行与高效表现。

AI 大模型的兴起与发展不仅推动了人工智能领域的整体进步，还促进了自然语言处理、计算机视觉、强化学习等多个子领域的快速发展。当然，目前的 AI 大模型在展现出巨大潜力的同时也面临着诸如训练成本高昂、参数规模庞大导致的计算难度增加以及泛化能力有限等问题，这些都是需要我们进行深入的研究与优化的。

如今，AI 大模型已可广泛应用于自然语言处理、计算机视觉、语音识别、推荐系统、自动驾驶等多个领域，可为各行业提供智能化的解决方案。未来，我们有望能出现更加高效、可解释性更强、更易于部署和维护的 AI 大模型。随着边缘计算、云计算等技术的发展，AI 大模型的应用场景将更加广泛。AI 大模型正以其庞大的参数规模、强大的数据处理能力和广泛的应用前景，成为推动人工智能发展的核心力量。随着技术的不断进步和应用的不断深化，AI 大模型也将在未来发挥更加重要的作用。我们有理由相信，在不久的将来，AI 大模型将为我们带来更加智能、高效和便捷的生活方式。

17.3.7.2 AI 大模型的核心技术

作为人工智能领域的一个重要分支，AI 大模型的核心技术涵盖了其算法以及其模型架构设计等的多个方面；它们共同构成了 AI 大模型技术的核心，使得其能够在大规模数据处理中展现出独特的优势。

1. AI 大模型的模型架构及运算技术

AI 大模型的模型架构及运算是 AI 大模型的基础。它涉及到神经网络的结构设计与算法，包括网络层数、每层神经元类型及数量、激活函数、训练及调优策略等。这些架构和算法直接影响着模型的性能和复杂度。在模型架构和算法设计中，需要权衡模型的表达能力和计算复杂度。增加网络层数和神经元数量可以提高模型的表达能力，但也会增加计算成本和过拟合的风险。选择合适的激活函数和 Dropout 策略可以帮助提高模型的泛化能力。

目前，AI 大模型的核心算法主要是基于深度学习。深度学习算法模拟了人脑神经网络的工作原理，通过多层次的神经元和连接权重来实现对数据的处理和分析。这些算法包括**卷积神经网络（CNN）、循环神经网络（RNN）以及 Transformer 模型等。**

卷积神经网络（CNN）在处理图像数据时具有显著优势。它可通过卷积操作提取图像中的局部特征，并通过池化操作降低数据的维度，从而实现对高维数据的有效处理。然而，CNN 在处理序列数据时存在一定的局限性，因为它无法充分利用序列数据中的时序信息。

循环神经网络（RNN）则专门用于处理序列数据。它通过循环连接保留历史信息，并利用这些信息来影响当前时刻的输出。这使得 RNN 在处理诸如自然语言处理、语音识别等任务时具有优势。然而，RNN 也存在着梯度消失和梯度爆炸等问题，这在一定程度上限制了其在大规模数据处理中的应用。

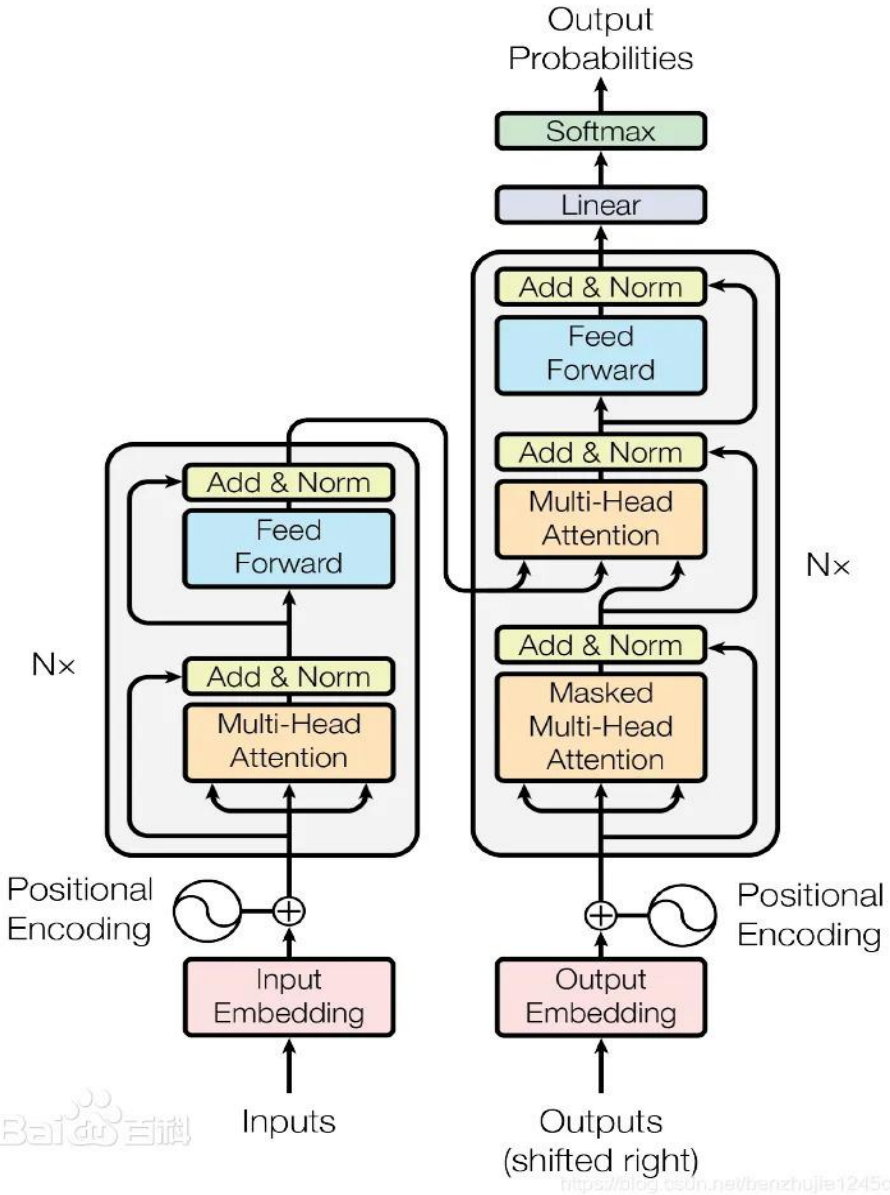

图 17.3.4　Transformer 模型架构

如今，Transformer 模型已是许多 AI 大模型的核心架构，也可以说是 AI 大模型的坚实基石。

Transformer 模型结合了 CNN 和 RNN 的优点，通过自注意力机制实现了对序列数据的全局建模。Transformer 在处理长序列数据时表现出色，且具有较强的并行计算能力。这使得 Transformer 在诸如自然语言处理、机器翻译等领域效果显著。

Transformer 模型无疑开启了深度学习领域的新纪元。在 Transformer 提出之前，自然语言处理领域的主流模型是循环神经网络 RNN，使用递归和卷积神经网络进行语言序列转换，曾是处理序列数据的核心手段。但在面对长序列时，RNN 及其变体常常陷入梯度消失和模型退化的困境。为了解决这一技术瓶颈，Transformer 模型应运而生。Transformer **基于注意力机制(attention)，摒弃了循环递归和卷积**。递归模型通常沿输入和输出序列的符号位置进行计算，来预测后面的值；但这种固有的顺序性质阻碍了训练样例内的并行化，因为内存约束限制了样例之间的批处理。而注意力机制允许对依赖项进行建模，而无需考虑它们在输入或输出序列中的距离。Transformer 避开了递归网络的模型体系结构，并且完全依赖于注意力机制来绘制输入和输出之间的全局依存关系。

Transformer 模型由**编码器和解码器**组成，各部分由若干相同构造的“层”堆叠而成。这些层巧妙地将自注意力子层与线性前馈神经网络子层结合；自注意力子层运用点积注意力机制为输入序列的每个位置构建独特表示；线性前馈神经网络子层则汲取自注意力层的智慧产出富含信息的输出表示。同时，编码器和解码器还各有一个位置编码层，用于捕捉输入序列中的位置信息。

Transformer 模型解决了梯度消失和模型退化问题，凭借自注意力机制能有效捕捉序列中的长期依赖关系。其并行计算能力卓越，在 GPU 上可快速进行训练和推断，在机器翻译、文本分类、语音识别等多项任务中表现出色。不过，Transformer 模型对计算资源需求庞大，训练和推断过程需要大量计算资源支持，同时在处理超长序列时仍面临挑战。

Transformer 架构的引入，为 AI 大模型的发展铺设了坚实的基石。组成 Transformer 架构的编码器和解码器，前者负责将输入序列转化为抽象表达，后者则根据编码器的输出及上下文信息生成目标序列。而作为 Transformer 架构的核心组件，自注意力机制赋予模型在输入序列的所有位置上进行注意力计算的能力，从而实现了对序列内部信息的全局性建模。这一机制能够有效捕捉序列中不同位置间的依赖关系，特别是长距离依赖，进而增强了模型对复杂序列数据的理解和处理能力。

AI 大模型普遍采用了**预训练与微调相结合的训练策略**。在预训练阶段，模型借助大规模无标注数据进行自监督学习或有监督学习，以习得通用的特征表示。随后，在微调阶段，模型针对特定任务的有标注数据进行调整，以适应任务的特定需求。这一策略显著提升了模型的泛化能力和适应性。而作为 Transformer 架构的一种变体，**多头注意力机制**允许模型在多个子空间中并行学习不同的特征表示；通过分散注意力至多个头部，模型能够同时捕获多种语义层次的信息，进而增强了模型的表达能力和学习效率。

基于 Transformer 架构的 GPT、BERT 等 AI 大模型在自然语言处理任务中已取得了突破性成果。这些模型通过大数据预训练学习通用特征，在零样本和少样本情况下具有泛化性，为 AI 大模型的发展奠定了坚实基础。GPT 是生成式预训练（转换）模型（Generative Pre-training Transformer）的集成，具体包括生成（G）、预训练（P）和（转换）算法模型（T）。GPT 技术的基本原理是通过预训练语言模型来提升其在各种 NLP 任务中的表现。它利用大规模语料库进行预训练，学习语言的内在规律和特征，然后通过微调（fine-tuning）来适应具体的任务，如文本生成、问答系统等。GPT 模型通过不断学习和优化，能够生成更加自然和准确的文本输出。

GPT 技术的核心在于自然语言处理技术（NLP），其核心任务是**自然语言理解和生成**。GPT 技术已在多个领域得到了广泛应用，包括语音识别、机器翻译、语言生成等。此外，GPT 还在不断进化，例如 GPT-4 已引入了多模态架构，能够处理图像和文本的混合输入，进一步扩展了其应用场景。

最初的生成式预训练模型 GPT-1 采用了 12 层 Transformer 的结构作为解码器，每个 Transformer 层是一个多头的自注意力机制，然后通过全连接得到输出的概率分布。其模型训练有两

个阶段，第一阶段是无监督预训练，基于海量的文本集通过 Transformer 学习一个大容量的语言模型，第二阶段基于标注数据进行参数微调；得到的一般任务不可知模型（或称为通用模型）可优于经过判别训练的模型。GPT-1 的实践，首次证明了通用模型训练具有很大的价值潜力。之前，用于学习特定任务的标注数据难以获得，导致模型效果不能持续提升，而通过 Transformer 无监督训练+少量标注数据的 Finetune 就取得了更优的效果；同时也证明了增加 Transformer 中间层的有效，在其从 2 层到 12 层的数量增加中，平均每增加 1 层能够提升 9%的准确性；加上 Transformer 本身具备并行能力，这在 GPU 上无疑潜力巨大。GPT-1 发现，在第二步的 Finetune 中添加语言建模作为辅助学习目标，能够提高监督模型的泛化能力，并加速收敛。说明在更海量的数据集时，模型会更收益于辅助学习目标。但 GPT-1 在生成长文本时，仍然会出现信息遗忘和重复等问题，和特定领域的模型对比还有很多不足。于是 GPT-2 重点考虑了更大的模型、更广的数据集及具有更好的泛化能力。GPT-1 是 12 层的 transformer；GPT-2 则是 48 层，共有 15 亿个参数的 transformer，训练集叫 WebText，是从 4500 万个链接提取文本去重后，得到 800 万文档共 40GB 文本。GPT-2 认为，现有系统用单个任务来训练的单个领域数据集，是缺乏模型泛化能力的主要原因，因此在更广的数据集上，GPT-2 采用了多任务（multitask)的方式，每一个任务都要保证其损失函数能收敛，不同的任务共享主体 transformer 参数。最终训练出来的模型在不需要任何参数和模型改动下，在 zero-shot（零样本）任务中，在 8 个数据集中有 7 个表现为业界最优，这个泛化能力可以说已经很强大了，并且在机器翻译场景取得亮眼结果。GPT-3 则采用更大参数和更大数据集。GPT-3 之前的模型要在特定领域有更好表现，依然需要上千条标注样本数据来进行 **finetune**，很大程度影响了模型的通用性；而人类能够根据前面一句话知道语境（in-context)，从而正确回答问题。于是，GPT-3 就通过调大参数(**1750 亿**）来测试 in-context 学习能力，来看看模型参数和样本集对正确性的影响；结果表明，越大的参数对样本具有更强的泛化能力。

ChatGPT 的成功是一个转折点。ChatGPT 是基于 InstructGPT 的。GPT-3 模型的参数为 175B，而 InstructGPT 模型的参数仅为其 130 分之一；但 InstructGPT 模型的输出却优于 GPT-3 模型的输出。

基于人类反馈的强化学习（RLHF）是 ChatGPT 区别于其他生成类模型的最主要特点，该法可帮助模型尽量减少有害的、不真实的及有偏见的输出，提升自然沟通效果。同时，为了更好地支持多轮对话，ChatGPT 引入了一种基于堆栈的上下文管理的机制，帮助 ChatGPT 跟踪和管理多轮对话中的上下文信息，从而在多轮对话中生成连贯自然的回复。

从数学或机器学习的角度来看，现有**语言模型大都是对词语序列的概率相关性分布的建模**，即利用已经说过的语句作为输入条件，预测下一个时刻不同语句甚至语言集合出现的概率分布。GPT 生成式预训练模型也是根据语料概率来自动生成回答的每一个字，ChatGPT 就是在此基础上通过使用基于人类反馈的强化学习（RLHF）来干预**增强学习**以取得更好效果的。

Transformer 的核心结构是其编解码组件结构。Transformer 模型架构本质上就是一个 **Encoder-Decoder** 架构，即由编码组件和解码组件构成的架构。编码组件和解码组件可以有很多层，比如 GPT-1 是 12 层，然后到 GPT-3 是 96 层。每个**编码器**由两个子层组成：Self-Attention 层（自注意力层）和 Position-wise Feed Forward Network（前馈网络，缩写为 FFN），每个编码器的结构都是相同的，但是它们使用不同的权重参数。编码器的输入会先流入 Self-Attention 层。**Transformer 的解码器**也有编码器中这两层，但是它们之间还有一个编解码注意力层（即 Encoder-Decoder Attention），其用来帮助解码器关注输入句子中需要关注的相关部分。

Transformer 的编码器对文本的处理和通常的 NLP 任务一样，首先使用词嵌入算法（Embedding）将每个词转换为一个词向量(vector)。嵌入仅发生在最底层的编码器中，其他编码器接收的是上一个编码器的输出。对输入序列完成嵌入操作后，每个词都会流经编码器内的两层，然后逐个编码器向上传递。

Transformer 的自注意机制（**Self-Attention**）突破了文本关注距离的限制，因此非常关键。比如，在文本中，我们经常看到这样的句子：“The animal didn't cross the street because it was too tired”。这个句子中的"it"代表什么意思，是 animal，还是 street 还是其他？这个对人来说很容易，但对模型来说不简单。self-Attention 就是用来解决这个问题，它可让 it 指向 animal。

在 self-attention 中，每个单词有 3 个不同的向量，它们分别是 Query 向量 Q，Key 向量 K 和 Value 向量 V。它们是通过 3 个不同的权值矩阵由嵌入向量 X 乘以三个不同的权值矩阵 W^Q ， W^K ，W^V 得到；其中三个矩阵的尺寸也是相同的。Query，Key，Value 的概念取自于信息检索系统，举个简单的搜索的例子来说。当你在某电商平台搜索某件商品（年轻女士冬季穿的红色薄款羽绒服）时，你在搜索引擎上输入的内容便是 Query，然后搜索引擎根据 Query 为你匹配 Key（例如商品的种类，颜色，描述等），然后根据 Query 和 Key 的相似度得到匹配的内容（Value)。self-attention 中的 Q，K，V 也是起着类似的作用，在矩阵计算中，点积是计算两个矩阵相似度的方法之一，因此，模型使用了 QK^T 进行相似度的计算。接着便是根据相似度进行输出的匹配，可以使用加权匹配的方式，而权值就是 query 与 key 的相似度。

Transformer 还引入**多注意头机制**（Multi-headed attention）来增强自注意能力，其一是扩展了关注的位置，使之同时关注多个不同位置，其二是它为注意力层提供了多个“表示子空间”，用几个注意头，那就可以有几组不同的 Q/K/V 矩阵，每个输入的词向量都会被投影到多少个表示子空间中进行计算。因此多注意头本质上是用更多个角度进行注意力计算再统一起来，能够增强对句子上下文的完整理解。

残差连接与层归一化也是提升深度神经网络性能的关键技术。残差连接确保了信息在不同层次间的有效传递，有助于缓解梯度消失和梯度爆炸问题；而层归一化则加速了模型的训练收敛过程，并提升了模型的稳定性和泛化能力。

2. **AI 大模型的优化训练技术**

AI 大模型的训练包括预训练与微调（Pre-training and Fine-tuning）。预训练阶段是利用大规模无标注数据进行模型初始化，而后在特定任务的有标注数据上进行微调，以提升模型在特定任务上的表现。

预训练技术是 AI 大模型的核心技术，它严格来说更像是一种训练范式。GPT 等模型通过海量大数据预训练，让模型学习数据通用特征，为特定任务奠定强大基础。预训练技术的核心原理是通过大规模数据预训练提取丰富的语言知识和语义信息。

在预训练阶段，模型利用自注意力机制捕捉文本上下文信息，以自监督方式学习语言规律和结构。随后，在微调阶段，根据具体任务需求，在特定数据集上有监督训练，优化任务性能。预训练技术能显著提升模型性能，通过学习更多语言知识和语义信息，提高准确率、泛化能力和鲁棒性。同时，它还能加速训练过程，提供准确初始权重，加快收敛速度，节省时间和计算资源。预训练技术的应用范围广泛，不仅限于自然语言处理领域，还广泛应用于计算机视觉等领域。通过预训练技术，AI 大模型能够自适应不同的工作和环境，展现出强大的通用性。

AI 大模型是如何提升训练效果的？这里，我们仅以 ChatGPT 为例说明。ChatGPT 所采用的是大语言模型 (LLM) 生成领域的训练范式：RLHF ，即基于来自人类反馈的强化学习来优化语言模型。

关于 RLHF 训练有个 TAMER 框架值得参考。RLHF 是一项涉及多个模型和不同训练阶段的复杂概念，这里我们可以按三个步骤分解：**预训练一个语言模型 (LM) ；聚合问答数据并训练一个奖励模型 (Reward Model，RM) ；用强化学习 (RL) 方式微调 LM**。这里，RLHF 是用生成文本的人工反馈作为性能衡量标准，或更进一步用该反馈作为奖励来优化模型，使得在一般文本数据语料库上训练的语言模型能和复杂的人类价值观对齐。

首先，我们使用经典的预训练目标训练一个语言模型。对这一步的模型，OpenAI 在其第一个

流行的 RLHF 模型 InstructGPT 中使用了较小版本的 GPT-3。然后进行以下步骤：

第一步：训练监督策略语言模型 由于 GPT-3 本身无法识别人类指令蕴含的不同意图，也很难判断生成内容是否高质量，为了解决这一问题，训练过程是从数据集中随机抽取问题，由标注人员给出高质量答案，相当于提供了一系列人工编写的 prompts 和对应的答案数据集，然后用这些人工标注好的数据集微调 GPT3.5 模型，来获得 SFT 模型(Supervised Fine-Tune)。

第二步：训练奖励模型 训练方法是：根据第一阶段的模型，随机抽取问题，给出多个不同的回答，人工选出最优答案进行标注，有点类似教学辅导。将高质量答案的奖励值进入下一轮强化学习 RL，训练一个奖励模型来预测人类偏好的输出。

RM 的训练是 RLHF 区别于旧范式的开端。这一模型接收一系列文本并返回一个标量奖励，数值上对应人的偏好。我们可以用端到端的方式用 LM 建模，或者用模块化的系统建模 (比如对输出进行排名，再将排名转换为奖励) 。这一奖励数值将对后续无缝接入现有的强化学习 RL 算法至关重要。

关于模型选择方面，RM 可以是另一个经过微调的 LM，也可以是根据偏好数据从头开始训练的 LM。微调 LM 被认为对样本数据的利用率更高，但对于哪种 RM 更好尚无定论。

第三步：近端策略优化 (Proximal Policy Optimization，PPO) 使用 PPO 优化奖励模型的策略。使用奖励模型的输出作为标量奖励，并使用 PPO 算法对监督策略进行微调，以优化该奖励。其训练方法是： PPO 的核心目的是将在线的人工学习转为离线学习，机器自己给自己打分。利用第二阶段训练好的奖励模型，在数据集中随机抽取问题，使用 PPO 模型生成多个回答，并用上一阶段训练好的 RM 模型分别给出质量分数。把回报分数按排序依次传递，产生策略梯度，通过强化学习的方式更新 PPO 模型参数。

最后，步骤二和步骤三可以循环迭代，可以不断完善模型。

总体来说，ChatGPT 是在人工标注的 prompts 和回答里训练出 SFT 监督策略模型，再通过随机问题由模型给出多个答案，然后人工排序，生成奖励模型，再通过 PPO 强化训练增强奖励效果。最终 ChatGPT 能够更好理解指令的意图，并且按指令完成符合训练者价值观的输出。

基于人类反馈的强化学习（RLHF）可有效提升模型的性能与可靠性。基于人类反馈的强化学习（RLHF）是一种结合强化学习与人类反馈的调优方法，旨在提升 AI 大模型在特定任务上的性能与可靠性。RLHF 的原理是将强化学习与人类反馈相结合，以人类判断作奖励信号引导模型行为。在 RLHF 的训练过程中，首先选择具备通用能力的预训练模型，然后通过模仿人类标注对话示例进行监督微调。接着，训练奖励模型，根据人类标注学习评估模型行为。最后，以奖励模型为奖励函数进行训练优化。通过这一系列精心设计的步骤，RLHF 能够让 AI 大模型逐步学会如何根据人类的反馈精准调整其行为，从而使其输出更加贴近人类的期望与标准。

RLHF 在大模型技术中发挥着举足轻重的作用。它不仅能够提高模型的性能和可靠性，还能促进模型道德与人类价值观的对齐。通过强化学习与人类反馈的完美结合，RLHF 使得 AI 大模型能够更好地理解和适应特定任务的需求，同时有效减少因环境噪声或数据偏差导致的错误决策。

当然，可使 AI 大模型得到有效训练的技术还有很多。大型模型的训练过程复杂且资源消耗巨大，随着硬件技术的进步和数据的积累，分布式训练、混合精度训练以及增量训练等技术也被广泛应用于 AI 大模型的学习过程之中。这些技术可以提高模型的训练速度和稳定性，使得大规模数据处理变得更加高效和可靠。其中，**分布式训练（Distributed Training）**可通过将训练任务分散至多个计算节点，利用节点间的协同工作加速训练进程，并有效应对大规模数据与模型的计算与存储挑战。**混合精度训练（Mixed Precision Training）**可通过在不同计算阶段采用不同数值精度，如低精度用于参数与梯度计算，高精度用于梯度更新，从而在减少内存占用与计算量的同时，保持训练速度与精度。**数据并行与模型并行（Data Parallelism vs Model Parallelism）训练**，数据并行侧重于并行处理

不同数据批次，而模型并行则聚焦于模型各部分的并行训练。两者可灵活结合，以应对超大规模模型与数据的训练需求。**异步训练（Asynchronous Training）可**在分布式环境中，允许计算节点异步进行训练，无需等待全局同步，从而提高训练效率。**动态学习率调整（Dynamic Learning Rate Adjustment）训练**可根据训练过程中模型的性能变化，动态调整学习率，以优化模型的收敛速度与泛化能力；等等。

在大模型的训练过程中，**分布式计算和并行计算**等技术无疑发挥着重要作用。对于超大规模的模型学习，传统的单机训练方式往往无法满足需求。通过分布式计算等技术，可以将模型拆分成多个部分，并在不同的计算节点上并行训练。这不仅可以加快模型的收敛速度，还可以提高系统的可扩展性和容错性。当然，分布式训练等也面临着一些挑战，如通信开销、数据一致性等问题。为了克服这些挑战，还需要设计高效的通信协议和同步机制，以确保不同计算节点之间的数据一致性和训练进度的一致性。

3. AI 大模型的优化策略与压缩技术

AI 大模型作为人工智能领域的前沿技术，其理论基础和实践应用均受到广泛关注。在训练和优化大规模时，需要我们综合考虑多种（优化）策略和技术，以应对计算资源、模型复杂度、训练效率等多方面的挑战。

首先，**计算资源的灵活动态分配**应是训练大模型时的一个关键。在大模型学习中，通常需要处理海量的数据，并进行复杂的模型训练和推理。这就要求计算资源具备高度的弹性和灵活性，以便根据任务需求动态调整计算资源的规模和配置。如今的**云计算平台**可提供这样的能力，它可以根据模型训练的需求，动态地分配和释放计算资源，从而提高资源的利用率和训练的效率。

其次，**参数调优**也是进一步提高模型性能的关键步骤。学习率的选择、权重的初始化以及正则化技术的应用等，都会对模型的收敛速度和性能有着显著影响。学习率的合适选择可以使模型更快地收敛到最优解，而权重的初始化则直接关系到模型训练的稳定性和效果。而正则化方法不仅可以减少过拟合的风险，还可以帮助模型更好地泛化到未见过的数据。

正则化方法还可以消除特征之间的量纲差异。由于不同的特征可能具有不同的数值范围和单位，这可能导致某些特征在模型训练中的权重过大或过小。通过正则化，可以将所有特征缩放到相似的范围内，使模型能够更平衡地考虑每个特征的影响。各种优化算法，如随机梯度下降（SGD）、自适应学习率优化器、动量法等，旨在提高收敛速度和稳定性；而正则化技术，如 L1 正则化、L2 正则化、Dropout 等，则可用于减少模型的过拟合风险。正式凭借上述基本原理和核心技术，AI 大模型才得以在大规模数据集上进行高效训练，并在众多任务和领域中展现出卓越的性能。

模型压缩技术也是优化大模型的一种重要手段，是 AI 大模型走向实际应用的关键一环。AI 大模型能够快速发展，主要是模型具备很好的并行扩展性；但随着数据量和计算量的增加，工程布局和调优就成为主要挑战。为提升模型效率、降低资源消耗及加速推理过程，大模型的优化与压缩技术就显得至关重要。

模型压缩技术的核心技术包括**模型剪枝（Model Pruning）、量化（Quantization）和知识蒸馏（Knowledge Distillation）**等。**模型剪枝**通过移除模型中的冗余参数与连接，有效减小模型规模与计算量，同时尽可能保持模型性能。权重裁剪则去除不重要权重减小模型大小。**量化**是将模型参数与激活值从高精度浮点数转换为低精度或定点数表示，显著降低模型存储需求与计算复杂度，提升模型在硬件上的运行效率。**知识蒸馏或模型蒸馏（Model Distillation）**技术可将大型复杂模型的知识转移至小型模型，实现模型的有效压缩，同时保持较高的模型性能；实现知识的有效压缩与传承，显著减少模型的存储与计算负担。

模型压缩技术可优化部署与提高效率。通过知识蒸馏、剪枝、量化等方法，可以有效地减小模型的规模和复杂度，从而降低存储需求和计算复杂度。这不仅提高了模型的效率和实时性，还有助

于模型在移动端等资源受限的设备上部署和应用。模型压缩技术通过去除冗余、降低精度和知识迁移等手段，可实现模型的优化。在实际应用中，模型压缩技术能够降低存储和计算需求，使模型更易部署在资源受限设备上，提高推理速度。训练小模型模拟大模型性能，可让学生模型在保持性能的同时拥有更小尺寸。模型压缩技术的应用不仅限于提高部署效率，还能保持较高性能水平，满足设备限制和提高部署速度的需求。同时，它还能降低部署难度和成本，广泛应用于实际场景，为 AI 大模型的普及和应用提供了有力支持。

在 AI 大模型的研究与应用实践中，技术方法的选择与运用占据着核心地位。训练、优化及压缩技术等关键技术方法，可应对复杂多变的模型训练与应用挑战。基于人类反馈的强化学习（RLHF）和模型压缩技术，不仅提升了 AI 大模型的性能和可靠性，还推动了 AI 技术在自然语言处理、计算机视觉、自主驾驶等多个领域的广泛应用，它们共同构成了 AI 大模型发展的坚实基石。**AI 大模型的有效训练和优化需要综合运用多种策略和技术。**从计算资源的分配到参数的调优，从正则化方法的应用到模型压缩的实现，再到分布式计算和并行计算技术的利用，每一个环节都对模型的性能和效率产生着重要影响。**分布式训练、模型压缩、知识库集成、多模态融合、人类反馈强化学习、AI Agent 等，都是解决落地瓶颈的关键**。未来，随着技术的不断进步和应用场景的不断拓展，我们期待看到更多创新和突破在 AI 大模型学习领域涌现。

17.3.7.3　AI 大模型的典型应用领域及未来技术发展

AI 大模型作为人工智能领域的重要分支，其理论基础和应用实践均取得了显著的进展。特别是在自然语言处理、多模态融合等具体领域，AI 大模型均发挥着越来越重要的作用。其应用场景已广泛覆盖自然语言处理、机器人视觉、医疗健康等多个关键领域。通过深入剖析这些应用场景，我们能够更加清晰地认识到 AI 大模型在解决现实复杂问题中的核心作用与深远潜在意义。

首先是自然语言处理（NLP）领域，AI 大模型已展现出强大的语言理解和生成能力，被广泛应用于机器翻译、情感分析、对话系统等。借助如 BERT、GPT 等 AI 大模型，已实现了情感分析、命名实体识别、文本分类等语言理解任务的高效执行。利用 AI 大模型，可成功生成包括文章、对话系统在内的多种文本内容。通过采用 Transformer 等先进架构，实现了多语言翻译任务的高精度完成。GPT 模型可以将一种语言的文本自动转换为另一种语言的文本，实现跨语言交流，为国际交流与合作提供了便利。

其次是**多模态融合及计算机视觉（Computer Vision）领域**，AI 大模型已实现了对图像数据的高效处理与理解，在图像分类、目标检测等任务中取得了优异的表现。借助生成对抗网络（GAN）和变分自编码器（VAE）等模型，AI 大模型已实现了图像超分辨率、风格迁移等高级图像生成功能。

在**自动驾驶与智能交通领域**，AI 大模型在处理传感器数据、环境感知、路径规划及行为预测等方面可发挥关键作用，推动了自动驾驶技术的快速发展。在智能交通管理方面，模型可实现交通流预测、拥堵管理等智能交通管理任务的高效执行。

在**医疗与生物信息学领域**，AI 大模型在医学影像数据的分析中展现出强大能力，为疾病诊断、病灶检测等提供了有力支持。在药物设计与发现方面，人们已利用 AI 大模型进行药物筛选与分子对接，加速了药物研发与发现的进程。

在**金融与风控领域**，AI 大模型可实现对客户数据的深入分析，为信用评分与风险管理提供了科学依据。借助逻辑回归等模型，可有效提升交易数据的分析能力，为欺诈检测与风险预警提供有力保障。

在**教育与辅助学习领域**，利用 AI 大模型对学生数据进行深度挖掘，可实现个性化教育方案与学习路径的精准规划。通过聊天机器人等智能工具，可实现学习过程的实时监控与高效智能辅导。

总之，随着 AI 大模型技术的迅猛发展，其在多个领域的应用都已取得了显著的成果。与此同时，AI 大模型技术的发展也存在一系列问题，需要我们去深入的思考和探讨。

一是计算资源的瓶颈问题。AI 大模型学习通常需要大量的计算资源（算力）进行训练和推理，这使得许多研究机构和企业面临计算资源不足的挑战。为了解决这个问题，可以采用分布式计算、云计算等技术，将模型训练过程分散到多个计算节点上，从而提高计算效率；同时，通过模型压缩、剪枝等技术，也可以降低模型复杂度，减少计算资源需求。

二是数据质量与多样性问题。数据是 AI 大模型学习的核心，但现实中往往存在数据质量不高、多样性不足等问题。为了解决这些问题，我们需要加强数据预处理和清洗工作，去除噪声和异常值；同时，可通过数据增强、迁移学习等技术，提高模型的泛化能力和鲁棒性。

三是算法偏见问题。AI 大模型在训练过程中可能会受到训练数据的影响，从而产生偏见。这些偏见可能导致模型在处理某些任务时表现出不公平或歧视性的行为，给社会带来不良影响。为了解决算法偏见问题，我们需要从多个方面入手。首先，我们需要保证训练数据的多样性和代表性，避免数据中的偏见对模型产生影响。其次，我们可以采用一些技术手段来检测和纠正模型中的偏见，如引入公平性约束、采用对抗性训练等方法。此外，我们还需要加强对算法模型的监管和评估，确保其在使用过程中符合公平、公正的原则。现实说明，如果 AI 大模型有价值观的话，其所体现的，更多的是标注者的价值观念而不是更广泛的人的价值观。

三是模型安全与隐私问题。随着 AI 大模型在各领域的应用越来越广泛，模型安全和隐私保护问题也日益凸显。由于模型通常具有复杂的结构和大量的参数，因此可能存在潜在的漏洞和攻击点。恶意攻击者可能会利用这些漏洞对模型进行攻击，导致模型失效或产生错误的输出。为了提高模型的安全性，我们可以采取一系列的安全防护措施。例如，加强对模型的测试和验证，确保其在各种场景下都能保持稳定的性能；同时，采用一些防御性编程技术，如输入验证、异常处理等，防止恶意输入对模型造成损害。此外，我们还需要加强对模型的安全审计和漏洞检测，及时发现并修复潜在的安全问题。

由于 AI 大模型通常需要大量的数据进行训练和优化，这些数据往往包含用户的个人信息和敏感数据。因此，数据隐私保护也成为了 AI 大模型学习中不可忽视的问题。一方面，我们需要确保在数据采集、存储和使用过程中，用户的隐私得到充分的保护；另一方面，我们还需要防止数据泄露和滥用，避免给用户带来不必要的损失和风险。为了保障用户隐私和数据安全，一方面，我们需要加强模型的安全审计和漏洞检测工作，及时发现并修复其潜在的安全问题；另一方面，我们可以采取一系列的技术手段和政策措施。例如，通过数据加密、匿名化处理等技术手段，确保数据在传输和存储过程中的安全性；同时，建立健全的数据管理制度和隐私政策，明确数据的收集、使用和共享规则，确保用户的知情权和选择权。确保 AI 大模型学习技术的发展能够在保障人类社会福祉的前提下稳步前进，保护用户数据不被滥用和泄露。

随着技术的不断进步和应用领域的拓展，AI 大模型已成为人工智能领域的重要研究方向。未来，AI 大模型将继续迎来新的发展趋势，但同时也面临着诸多挑战。随着新技术、新方法的出现，展望 AI 大模型的未来发展，我们可以看到 AI 大模型的发展存在如下一些明显的发展趋势：

（1）跨模态学习与融合：未来的 AI 大模型将更加注重跨模态信息的融合与利用。通过将文本、图像、语音等不同模态的数据进行联合学习，模型将能够更全面地理解世界，并在多个领域实现更高级别的智能应用。

（2）自动化与智能化模型设计：随着深度学习技术的不断发展，自动化和智能化的模型设计将成为可能。通过引入神经架构搜索（NAS）、自动超参数调整等技术，AI 大模型的设计将更加高效和精准，从而减少人工干预和试错成本。

（3）可解释性与鲁棒性提升：为了提高模型的可靠性和安全性，未来的 AI 大模型学习将更加注重模型的可解释性和鲁棒性。通过引入注意力机制、可视化技术等手段，我们可以更好地理解模型的决策过程；同时，通过对抗性训练、防御性编程等技术，可以提高模型对噪声、攻击等干扰的

抵抗力。

而对 AI 大模型**研究，可有如下一些值得注意的方向：**

（1）新型网络结构与优化算法研究。探索更加高效、稳定的网络结构和优化算法，以提高 AI 大模型学习的性能和效率。

（2）多任务学习与迁移学习研究。研究如何将多任务学习和迁移学习技术应用于 AI 大模型学习中，以提高模型的泛化能力和适应性。

（3）知识与数据融合研究。探索如何将知识推理与数据驱动的方法相结合，以提高 AI 大模型学习的决策质量和可靠性。