**![2006099C[1]]() 拓展资源**

**1. 谓词逻辑**

**答：**

在命题逻辑中，研究的基本单位是命题，命题不可分，命题仅是有真值的一个陈述句。这样在命题演算的过程中，我们根本就不考虑命题的结构和成分，只是对命题之间的联系加以研究。从而要命题内部的特征命题逻辑是无能为力的。而事实上，在人类的思维过程，需要将命题内部的逻辑结构和成分作进一步分析。如著名的苏格拉底三段论：

所有的人都是要死的。

苏格拉底是人。

所以苏格拉底是要死的。

凭直觉这个论证过程是正确的，且上述三个命题之间有着密切的联系，但却无法用命题演算表达出来，因为它们都是。上述推理过程不仅仅与各命题有关，更与各命题的内部结构有关。因此有必要对命题内部的结构和成分加以深入研究，将命题演算扩展成谓词演算，对简单命题的成分、结构和命题间的共同特征等作分析。这就是谓词逻辑的目的。

谓词逻辑是在命题逻辑的基础上发展起来的，命题逻辑可看作是谓词逻辑的一种特殊形式。在谓词逻辑中，命题是用谓词表示的，一个谓词可分为谓词名和个体两个部分。个体表示某个独立存在的事物或者某个抽象的概念：谓词用于刻画个体的性质、状态或个体间的关系。例如，对于“小王和小李是朋友”这个命题可用谓词表示成Friend(wang,11)，其中“Friend”是谓词名，“Wang”和“Li”是个体，“Friend”刻画了“Wang”和“Li”的关系特征。

一阶谓词逻辑是一个高度形式化的系统，也是到目前为止能够表达人类思维活动规律的最精确的符号语言。这种符号语言与人类的自然语言比较接近，又可方便地存储到计算机中去被计算机做精确处理，因此它成为最早应用于人工智能领域来表示知识的一种逻辑系统。谓词逻辑不但适合于用与/或形（用“∧”和“∨”连接起来的公式）表示事物的状态、属性、概念等事实性知识，而且还可以用蕴涵式(P→Q)来表示事物间确定的因果关系。

**2. 数学命题在谓词逻辑中的符号化及形式推理**

利用谓词逻辑中的相关概念我们可以将自然数公理符号化：

A1：每一个自然数，有且仅有一个直接后继者。

A2：没有一个自然数使0是它的直接后继者。

A3：对任一个不是0的自然数，有且仅有一个直接先行者。

令谓词S(x)：x的直接后继者；P(x)：x的直接先行者；E(x,y)：x=y。则上述三个公理可符号化为：

A1：∀x∃y(E(y,S(x))∧(∀z(E(z,S(x))→E(y,z)))

A2：¬((∃x(E(0,S(x)))

A3：∀x(¬(E(0,x)→∃y(E(y,P(x))∧∀z(E(z,P(x))→E(z,y))))

谓词逻辑中的形式推理，即谓词演算，就是从给定的公理出发通过一些形式推理规则，得到一个形式公式，该公式必须是公理的逻辑结果。

**3. 谓词逻辑和机器证明**

机器证明是用电子计算机来完成数学命题的证明，是人工智能的一个重要研究领域。

在谓词逻辑中，利用谓词公式间的各种等价和永真蕴涵关系，通过一些推理规则，从一些已知的谓词公式推出另一些新的谓词公式。这就是谓词逻辑中的形式证明。

谓词逻辑中的结构严谨的形式证明，使得由公理系统中的公理和推理规则出发而求证定理的过程变得简单了。于是人们不禁要探究，是否有某种算法，按照这种算法就可以证明任何公理系统中的定理的正确性。若真的存在这种算法的话，将这种算法在计算机上实现后，则公理系统中的定理就可借助计算机证明，从而实现定理的自动证明。这就是所谓的机器证明。

德国数学家希尔伯特在1899年出版的《几何基础》中指出，初等几何中只涉及从属平行的定理可以实现证明的机械化，并提出了“希尔伯特机械化定理”。 1950年，波兰数理科学家塔尔斯基从理论上证明，初等代数和初等几何的定理可以机械化。1956年，美国的纽厄尔（A.Newell）和西蒙（H.Simon）从分析人类解答数学题的技巧入手, 经过反复的实验，认识到人类证明数学定理时，通过“分解”（把一个复杂问题分解为几个简单的子问题）和“代入”（利用已知常量代入未知的变量）等方法，用已知的定理、公理或解题规则进行试探性推理，直到所有的子问题最终都变成已知的定理或公理，从而解决整个问题。人类求证数学定理是一种启发式搜索，与电脑下棋的原理异曲同工。基于这样的研究成果，他们编制出了所谓的“逻辑理论机”（The Logic Theory Machine），即LT数学定理证明程序, 一举在计算机上证明了数学大师罗素和怀特海所著的《数学原理》第二章中的38个定理。这是机器证明的开始。1963年，经过改进的LT程序在一部更大的电脑上，最终完成了全部52条数学定理的证明。从手工证明到机器证明，是数学思想方法的重大飞跃。

1959年，美籍华人学者、洛克菲勒大学数学教授王浩用他首创的“王氏算法”，用一种相当原始的汇编语言写了两个证明程序，在一台速度不高的IBM704电计算机上用不到9分钟的时间就把这本数学史上视为里程碑的著作中全部（350条以上）定理，统统证明了一遍。

1965年美国数学家罗宾逊提出了与传统的演绎方法完全不同的归结法（Resolution Principle）（该原理的基本出发点是，要证明任何一个命题为真，都可以通过证明其否定为假来得到），他用归结法做到了谓词逻辑中永真式的自动证明。机器证明取得了重要的进展，国际上掀起了自动机器证明的高潮。

1971—1977年间，莱德索等人给出了分析拓扑学和集合论方面的一些著名定理的机器证明。1979年，波依尔和穆尔等人作出了递归函数方面的机器证明系统。

而机器证明最有名的例子，莫过于“四色问题”的证明。据说，“四色问题”最早是1852年一位21岁的大学生提出的数学难题：任何地图都可以用最多四种颜色着色，就能区分任何两相邻的国家或区域。这个看似简单的问题，就象“哥德巴赫猜想”一样，不知难倒了多少著名数学家和献身数学的业余爱好者，属于世界上最著名的数学难题之一。1976年6月，美国伊利诺斯大学的两位数学家沃尔夫冈·哈肯（W.Haken）和肯尼斯·阿佩尔（K.Apple）自豪地宣布，他们用计算机证明了这一定理。当“四色定理”被证明的消息传出后，许多大学的教师都纷纷中断讲课，打开香槟酒以示庆贺。在该定理被证明的所在地——伊利诺斯州乌班纳，连邮政局员工都欣喜若狂，他们在寄出的所有信件上都加盖了“四色是足够的”字样邮戳。

哈肯和阿佩尔攻克这一难题使用的方法仍然是前人提出的“穷举归纳法”，只是别人用的是手工计算，无论如何也不可能“穷举”所有的可能性。哈肯和阿佩尔编制出一种很复杂的程序，让3台IBM360计算机自动高速寻找各种可能的情况，并逐一判断它们是否可以被“归纳”。十几天后，共耗费了1200个机时，做完了200亿个逻辑判断，计算机终于证明了“四色定理”。这是机器证明首次解决传统人脑支配的手工证明所没有解决的重要难题。

二十世纪七十年代中期，我国数学家吴文俊引入了完全不同于前人的代数方法—“吴方法”，完成了初等几何定理的机械化证明。他的工作经出国留学生介绍到国外后，在国际学术界引起重大反响，国际《Artificial Intelligence》杂志曾以专集的形式发介绍他的研究工作。

从手工证明到机器证明，人类的数学思想方法产生了一个重大飞跃。

**4. 谓词逻辑与PROLOG语言**

逻辑程序设计将逻辑直接作为[程序设计语言](http://baike.baidu.com/view/128511.htm)并将计算作为受控推理的一种程序设计技术。

对于传统的程序设计来说，算法的逻辑意义往往被程序复杂的控制成分所掩盖，使程序的正确性难以得到证明。而且通常的高级[程序设计语言](http://baike.baidu.com/view/128511.htm)属于过程性语言，需要在程序执行前详细规定运行步骤。PROLOG是英文“Programming in Logic”的缩写,意为“逻辑程序设计”。设计逻辑程序语言的思想最早由英国人科瓦尔斯基（R.Kowalski）提出。1972年，法国马赛大学的科默寥尔（A.Clomerauer）及其研究小组设计成功PROLOG语言并在计算机上实现。1974年后科瓦尔斯基进一步阐明了PROLOG的理论基础，并系统地发展了逻辑程序设计的思想。科瓦尔斯基对传统的算法或对用通常高级语言编写的程序提出了一个著名的分析公式，即算法＝逻辑＋控制。其基本思想是要从根本上改变程序设计的方法：用户只需要编写程序的逻辑部分（逻辑程序设计之名由此而来），而系统中的[解释程序](http://baike.baidu.com/view/47200.htm)则实施控制部分的职能。

PROLOG以逻辑程序设计为基础，以处理一阶谓词演算为目的。它文法简洁，表达力丰富，具有独特的非过程型语言（一个语句就相当于过程语言的一个子程序而并非算法的一步），是一种具有推理功能的逻辑型语言。

PROLOG是一种说明性语言。这就是说，给出所需要的事实和规则，PROLOG将使用演绎推理求解编程问题。这与传统的过程性编程语言如C、BASIC和Pascal等形成了鲜明的对照。在过程性语言中，程序员必须提供一步一步的指令，准确地告诉计算机如何求解给定的问题。换句话说，程序员必须预先知道如何求解这个问题。相反，PROLOG程序员只需要提供对问题的描述和求解的基本规则。然后，PROLOG系统本身将确定如何找到一个解。

这样，现实世界中的问题只要能用谓词逻辑语言表示出来，就可以将它写成PROLOG程序，然后在计算机上实现：

现实世界问题谓词逻辑表示编写PROLOG程序计算机实现

因此，通过这样的方式可以用计算机的许多逻辑问题。

 目前，PROLOG语言已被广泛地应用于关系数据库、抽象问题求解、数理逻辑、公式处理、自然语言理解、专家系统以及人工智能的许多领域。

**5. 归结法（Resolution Principle）**

归结法是仅有一条推理规则的推理方法。归结法是与演绎法完全不同的，新的逻辑演算算法。它是一阶逻辑中最适合计算机进行推理的逻辑演算方法。

归结法基本原理：采用反证法或者称为反演推理方法，将待证明的表达式（定理）转换成为逻辑公式（谓词公式），然后再进行归结，归结能够顺利完成，则证明原公式（定理）是正确性的。归结证明法是某些领域中，用计算机证明定理的基础。

归结原理由J.A.Robinson于1965年提出，又称为消解原理。该原理是Robinson在Herbrand理论基础上提出的一种基于逻辑的、采用反证法的推理方法。由于其理论上的完备性，归结原理已成为机器定理证明的主要方法。

所谓定理证明的实质就是要对给出的（已知的）前提和结论，证明此前提推导出该结论这一事实是永恒的真理。这是非常困难的，几乎是不可实现的。
　　要证明在一个论域上一个事件是永真的，就要证明在该域中的每一个点上该事实都成立。很显然，论域是不可数时，该问题不可能解决。即使可数，如果该轮域是无限的，问题也无法简单地解决。

Herbrand采用了反证法的思想，将永真性的证明问题转化成为不可满足性的证明问题。Herbrand理论为自动定理证明奠定了理论基础，而Robinson的归结原理使得自动定理证明得以实现。从某种意义上讲大部分人工智能问题都可以转化为一个定理证明问题。

因此，归结推理方法在人工智能推理方法中有着很重要的历史地位。

若干个文字（变元或它的否定）组成的析取式称为子句，没有任何文字的子句称为空子句**□**，只有一个文字的子名称为单元子句。

将一个命题公式转换为它的合取范式，其每个合取式就是一个子句，所有这些子句组成的集合注是该公式的子句集。

归结式的定义：设C1和C2是子句集中的任意两个子句，如果C1中的文字L1与C2中的文字L2互补，那么可从C1和C2中分别消去L1和L2，并将C1和C2中余下的部分按析取关系构成一个新子句C12，则称这一个过程为归结，称C12为C1和C2的归结式，称C1和C2为C12的亲本子句。例如：子句C1=P∨A，　　　　　　　　　C2=¬P∨B中存在互补对P和¬P，可得归结式：C12=A∨B。没有互补对的两子句没有归结式。

而归结推理规则就指的是对两子句做归结，也即求归结式。

命题逻辑的归结法证明过程：命题逻辑的归结过程也就是推理过程，可以分为如下几个步骤：

1）建立待归结命题公式：首先根据反证法将所求证的问题转化成为命题公式，求证其是矛盾式（永假式）；

2）求取合取范式；

3）建立子句集；

4）归结。

归结法是在子句集S的基础上通过归结推理规则得到的，归结过程的最基本单元是得到归结式的过程。从子句集S出发，对S的子句间使用归结推理规则，并将所得归结式仍放入到S中（注意：此过程使得子句集不断扩大，是造成计算爆炸的根本原因），进而再对新子句集使用归结推理规则。重复使用这些规则直到得到空子句**□**。这便说明S是不可满足的，从而与S所对应的定理是成立的。

归结步骤：

1）对子句集中的子句使用归结规则；

2）归结式作为新子句加入子句集参加归结；

3）归结式为空子句**□**为止。

　　（证明完毕）

例 证明公式：(P→Q)→(¬Q→¬P)是永真式。

证明：根据归结原理将待证明公式转化成待归结命题公式：

¬((P→Q)→(¬Q→¬P)) ⇔ (P→Q)∧¬(¬Q→¬P))

公式的各项化为合取范式：(P→Q)∧¬ (¬Q→¬P)) ⇔ (¬P∨Q)∧¬Q∧P。

则得到该公式的子句集合为：{ ¬P∨Q,¬Q,P}

对子句集中的子句进行归结可得：

1）¬P∨Q

2）¬Q

3）P

4）Q 1），3）归结

5）□ 2），4）归结

由上可得原公式成立。

命题逻辑归结：从公式G=H1∧H2∧…∧Hn∧¬C出发，利用归结推理规则寻找矛盾，从而证明定理H1∧H2∧…∧Hn→C成立。命题逻辑归结过程：

1）将证明H1∧H2∧…∧Hn→C化成G=H1∧H2∧…∧Hn∧¬C的不可满足的证明；

2）将G化成合取范式；

3）求出G的子句集S；

4）使用归结法并把归结式送入到S中，直到得到空子句，定理得证。

谓词逻辑的归结过程与命题逻辑的归结过程相比，其基本步骤相同，但每步的处理对象不同。谓词逻辑需要把由谓词构成的公式集化为子句集，必要时在得到归结式前要进行置换和合一。

具体的谓词逻辑归结过程如下：

1）写出谓词关系公式；

2）化为SKOLEM标准形；

3）求取子句集S；

4）对S中可归结的子句做归结；

5）归结式仍放入S中，反复归结过程；

6）得到空子句；

7）命题得证。

例　“快乐学生”问题：假设任何通过计算机考试并获奖的人都是快乐的，任何肯学习或幸运的人都可以通过所有的考试，张不肯学习但他是幸运的，任何幸运的人都能获奖。求证：张是快乐的。

证明：

先将问题用谓词公式表示如下：

H1：“任何通过计算机考试并获奖的人都是快乐的”

　　　∀x((Pass(x,computer)∧Win(x,prize))→Happy(x))

H2：“任何肯学习或幸运的人都可以通过所有考试”

　　　∀x∀y(Study(x)∨Lucky(x)→Pass(x,y))

H3：“张不肯学习但他是幸运的”

　　　¬Study(zhang)∧Lucky(zhang)

H4：“任何幸运的人都能获奖”

　　　(∀x)(Luck(x)→Win(x,prize))

结论“张是快乐的”的否定

　　　¬Happy(zhang)

将上述谓词公式转化为子句集并进行归结如下：
　　 首先将每一个表示逻辑条件的谓词子句转换为子句集可以接受的Skolem标准形。

由H1可得

（1）¬Pass(x,computer)∨¬Win(x,prize)∨Happy(x)

由H2可得

（2）¬Study(y)∨Pass(y,z)

（3）¬Lucky(u)∨Pass(u,v)

由H3可得

（4）¬Study(zhang)

（5）Lucky(zhang)

由H4可得

（6）¬Lucky(w)∨Win(w,prize)

由结论可得

（7）¬Happy(zhang) 结论的否定

根据以上7条子句，归结如下：

（8）¬Pass(w,computer)∨Happy(w)∨¬Luck(w) （1），（6）归结，{w/x}

（9）¬Pass(zhang,computer)∨¬Lucky(zhang) （8），（7）归结，{zhang/w}

（10）¬Pass(zhang,computer) （9），（5）归结

（11）¬Lucky(zhang) （1），（3）归结，{zhang/u, computer/v}

（12）**□** （11），（5）归结