离散数学基础：数理逻辑导论

Fundamentals of Discrete Mathematics: Introduction to Mathematical Logics

周晓聪(isszxc@zsu.edu.cn)
中山大学计算机科学系, 广州 510275

2010 年 12 月 3 日

前言

数理逻辑也称为符号逻辑，从广义上理解，它包括五个部分的内容：**逻辑演算**、**证明论**、**公理集合论**、**递归论**和**模型论**。而从狭义上理解，数理逻辑则仅包括逻辑演算，进一步可分为**经典逻辑**和**非经典逻辑**。经典逻辑指命题逻辑、一阶谓词逻辑及其演算系统，非经典逻辑是对经典逻辑的扩充和修改，是现代数理逻辑研究的主要内容，例如有模态逻辑、直觉逻辑、多值逻辑、相干逻辑、模糊逻辑等等。我们这里仅涉及狭义数理逻辑内容的经典逻辑部分，包括命题逻辑及其演算系统和一阶谓词逻辑。

逻辑学是研究思维形式及其规律的学科，而**数理逻辑**则是采用数学的方法来研究思维形式及其规律的数学分支。或更简单地，逻辑学是研究推理是否有效，而数理逻辑则采用数学方法研究推理的有效性，即完全使用符号化的语言，根据公理化的基本思想建立一个形式系统，推理变成按照明确规则进行的符号改写。进一步，数理逻辑还从整体上研究这种形式系统的一致性、可靠性和完备性等。

传统的逻辑学起源于亚里思多德，现在见到的“形式逻辑”或“普通逻辑”著作，大多属于用自然语言表述的传统逻辑范围。而数理逻辑的最初设想源于17世纪的德国数学家莱布尼茨(G. W. Leibniz)，他认为**思维就是计算**，希望建立一个通用语言来进行推理，从而将推理变成按照明确规则进行的演算。1847年英国数学家和逻辑学家布尔(G.Boole)成功地建立了命题演算，即目前被广泛使用的布尔代数。1879年德国数学家和逻辑学家弗雷格(G. Frege)年建立了第一个谓词演算系统。到20世纪30年代，数理逻辑进入了一个新的发展时期，逻辑学不仅与数学相互渗透与结合，而且与其他科学技术相互渗透与结合，显示了逻辑学的实用意义。

数理逻辑研究的可计算性问题，特别是1931年哥德尔(K. Gödel)不完全性定理的提出以及递归函数的引入，导致了1936年图灵(A. Turing)提出通用机器的概念。图灵机是现代计算机的理想数学模型，因此数理逻辑是计算机科学的基础理论之一。数理逻辑在开关电路设计、自动控制、程序设计乃至软件开发中都有着十分广泛的应用。

本讲稿作为计算机专业本科一、二年级学生学习的数理逻辑导论课程使用的教材，只介绍有关命题逻辑和一阶谓词逻辑的基础知识，其中第一至四章属于命题逻辑，第五至七章属于一阶谓词逻辑，最后第八章作为选讲的内容简单介绍了一些非经典逻辑。我们在介绍命题逻辑和一阶谓词逻辑时采用的框架基本上相同：

(1) 首先介绍相关的基本概念：命题逻辑的基本概念是命题及各逻辑联结词（与、或、非、蕴含与等价）；一阶逻辑的基本概念是个体、谓词与量词等；

(2) 其次讨论自然语言命题在它们中的符号化，即分别讨论自然语言命题在命题逻辑和一阶逻辑中的符号化。注意，自然语言命题在逻辑中的符号化本身并不属于数理逻辑研究的内容，但是要将数理逻辑应用到具体应用领域，例如在程序中使用命题逻辑书写条件表达式，则需要将应用领域

的命题进行符号化。本讲稿比较详细地讨论有关符号化的内容，主要是希望读者领会符号化与非符号化之间的差别，并学会如何精确地、符号化地描述应用问题；

(3) 再次讨论了公式的语法和语义，即分别讨论了命题逻辑公式语法和语义的归纳定义，以及一阶逻辑公式语法和语义的归纳定义。其中一阶逻辑公式的语义定义比较难理解，初学者可不理会其严格定义，而仅仅直观上理解量词的含义即可；

(4) 然后讨论基于真值语义的等值演算，以及基于等值演算的范式，即分别讨论命题逻辑的等值演算和范式，以及一阶逻辑的等值演算和前束范式。等值演算讨论如何从语义上判断两个公式是否等价，而范式则给出了公式在语义上等价的一种标准形式，这种标准形式使得我们更容易讨论公式的语义性质。

(5) 最后讨论了自然推理系统，即分别讨论了命题逻辑的自然推理系统和一阶逻辑的自然推理系统。其中一阶逻辑自然推理系统中讨论与量词有关的派生规则一节，给出了一些一阶逻辑等值式的证明，这些证明对于初学者有一定难度，可作为选学的内容。对于命题逻辑的推理，第四章更严谨地定义了命题逻辑的演算系统，并讨论了一些元理论，这一章也可作为选学的内容。

与大多数离散数学教材中所讲述的数理逻辑部分相比，本讲稿的内容更有难度和深度，但与一些专业的数理逻辑教材相比，本讲稿的内容则浅一些，特别是在有关数理逻辑模型论方面的内容涉及得很少。因此本讲稿适合为本科低年级开设的数理逻辑课程，本讲稿使用备注形式给出的一些内容，初学者在第一次阅读时也可跳过。

学习本讲稿只需要一些有关集合、函数的基本知识，目前高中数学中的有关内容已经足够，不需要其他计算机专业课程作为先修课程。

学习数学课程（包括数理逻辑课程）的主要目的通常有两方面：一方面是为后续课程提供必要的数学知识准备；另一方面是训练学生的数学修养。

对于本讲稿提供的内容，对于计算机专业本科的许多后续课程，例如数字逻辑电路与设计、数据结构与算法、编译原理等，以及一些研究生课程，如程序设计理论、可计算性理论等提供了不可或缺的知识准备。

而另一方面，本讲稿给出的自然推理系统，在兼顾自然性的基础上，也强调了形式系统的思想，并将其与程序设计的相关思想紧密联系在一起，以期在提高学生逻辑思维严密性，以及强化学生逻辑证明技巧的基础上，能让学生初步领会计算机学科的一些基本思想，如形式化、公理化、可构造性、自顶向下分解、模块化等，这也是本讲稿与其他教材多从数学角度阐述离散数学知识的一个不同点。当然，是否做到还有待检验。

由于本讲稿直接用于教学，因此只给出了少量的习题，这些习题在本人的教学实践中都是要求学生作为书面作业完成的。现在市面上的离散数学教材很多，读者可从其他教材找到更多的习题作为练习。

目录

第一章　命题逻辑的基本概念

命题逻辑及其演算系统研究基于命题的推理演算。我们将命题逻辑及其演算系统的内容分为两大部分：

1. **命题逻辑**：这包括“命题逻辑的基本概念”、“命题逻辑的等值演算”以及“命题逻辑的自然推理”三章。这几章主要从真值语义的角度介绍命题逻辑，包括命题逻辑公式及其真值、基于真值语义的等值演算、以及基于命题逻辑的半形式化自然推理演算。

2. **命题演算**：这包括“命题逻辑的形式演算系统”一章，主要从形式化的角度介绍命题逻辑的演算系统。“命题逻辑的自然推理”侧重于推理的自然性，只是将利用自然语言进行的推理加以符号化。而“命题逻辑的形式演算系统”则侧重于构建一个形式系统，其推理是完全符号化的，本质上可与命题的真值语义无关。

这一章介绍命题逻辑的基本概念，包括命题的含义、命题的真值、自然语言命题的符号化，进一步给出命题逻辑公式的归纳定义，以及命题逻辑公式真值语义的归纳定义及相关概念。

1.1　命题及其符号化

1.1.1　命题的含义

实际上，在命题逻辑中无法给命题一个数学化的严格定义，因为命题是逻辑的最基本概念。我们只能给出命题的一个直观描述：**命题**(proposition)是**具有真假值但不能既真又假**的**陈述句**。具体来说，这有以下几个方面的含义：

1. 命题是表示判断的语句，判断是对某事物具有或不具有某属性的客观描述，因此命题必须是陈述句，不能是祈使句、感叹句或疑问句。从某种意义上来说，祈使句、感叹句或疑问句没有真假之分。例如下面的句子不是命题：

   - 这个小男孩多勇敢啊!（**感叹句**）
   - 乌鸦是黑色的吗？（**疑问句**）
   - 但愿中国队能取胜。（**祈使句**）
   - 请把门开一开!（**祈使句**）

2. 命题必须为真或为假，符合客观实事就为真，否则为假，二者必居其一。因此下面的句子不是命题：

   - $x + y > 10$（x和y**通常理解为变量，不能确定此句子的真假**）
   - 我写的这个句子是假的。（**这是一个语义悖论**）

备注 1.1.1 悖论是逻辑学研究的一个有趣的分支。简单地说，悖论是这样的句子，从该句子出发可合乎逻辑推出两个互为矛盾的结果。有各种形式的悖论，语义悖论是其中的一种，这种悖论与句子的语义有关，例如上面的句子:“我写的这个句子是假的”，这里所谓的“我写的这个句子”应该理解为指上面这个句子本身。从而若上面这个句子为假，则该句子所陈述的事情就为真，而若上面这个句子为真，则该句子所陈述的事情就为假，总是推出互为矛盾的结果。有兴趣的同学可查找相关资料，特别是一些有关逻辑学趣谈的书籍，进一步了解有关悖论的知识。

3. 在经典逻辑中，命题为真还是为假是命题所固有的性质与人们是否能够判断命题的真假无关，因此下面的句子在经典逻辑中都认为是命题，虽然人们现在还不能断定这些命题的真假，但它们确实具有惟一的真假值：

- 明年中秋节的晚上是晴天。
- 地球之外存在生物。
- 哥德巴赫猜想是正确的。

备注 1.1.2 在非经典逻辑（即对经典逻辑进行修改和扩充的逻辑）中，人们可能将命题这个概念作进一步的强化。例如，时态逻辑（模态逻辑的一种）专门研究与时间有关的命题，例如:“明年中秋节的晚上将是晴天”。而直觉逻辑（也称构造逻辑）则认为命题是（当前）能够证明为真或为假的陈述句，这样上面例子中的后两个句子就不认为是命题。

为了使用数学方法研究命题及其推理，将命题进行符号化十分重要。通常在命题逻辑中用小写符号表示命题，例如以p表示命题“雪是白的”这个命题，用q表示“北京是中国的首都”这个命题等。当p, q等表示任意命题时，我们称之为**命题变量**(proposition variable)，或者称为**命题变元**、**命题变项**等。

命题与命题变量的含义不同，命题指具体的具有真假值的陈述句，而命题变量本身不具有真假值，它代表任意的命题，只有当用具体命题代入命题变量时，命题变量才化为命题，方可确定其真假值。不过在命题逻辑中对命题与命题变量的处理原则是相同的，主要是关心命题的真假值，而对命题变量也主要是关心它所代入的具体命题的真假值，或者在多数时候，我们只关心对命题变量本身所赋予的真假值。

我们将命题或命题变量所具有的真假值统称为**真值**(truth value)。注意，**真值实际上包括真和假两个值**，通常用0表示假，1表示真，也有的教材使用F表示假(false)，而用T表示真(true)。

1.1.2 简单命题与复合命题

简单命题又称为**原子命题**是不包含与自身不同命题的命题，或者更直观地，从形式上说，简单命题只对一个事物的一个性质进行判断，属于简单的陈述句。例如，下面的命题都是简单命题：

- 雪是白的。
- 乌鸦是黑的。
- 我喜欢逻辑学。
- 我喜欢数学。

在命题逻辑中，认为简单命题不可再分割，因此将它定义为不包含与自身不同命题的命题。例如，简单命题“雪是白的”再分割为“雪”和“是白的”，则这两者都不是命题，因为它们根本不是陈述句，只是两个词语。后面我们将看到，谓词逻辑将对命题做进一步分割，例如简单命题“雪是白的”分割为“雪”和“是白的”，前者是**个体**，后者是**谓词**。

复合命题是指这样的命题：一是它包含与自身不同的命题作为**支命题**；二是它的真值由其支命题的真值惟一确定。例如下面的命题都是复合命题：

- 雪是白的，而乌鸦是黑的。
- 如果我喜欢逻辑学，那么我喜欢数学。

这些命题都含有支命题（上面用下划线给出），整个命题的真值由它的支命题惟一确定。那么是怎样确定的呢？这由复合命题中将支命题联系起来的**命题联结词**决定。上面例子中，没有加下划线的词语，像“而”、“如果…那么…”就是命题联结词。

实际上，在命题逻辑中只关心命题的真值，简单命题的真值由客观实事决定，而复合命题的真值则由简单命题的真值和命题联结词的性质决定。当然在逻辑中，我们不可能研究任意的客观实事，因此使用命题变量符号化简单命题，假定命题变量可代表任意的具体命题。同样，我们使用符号表示命题联结词，研究它们所表达的命题之间的真值关系。可以说，命题逻辑的核心是**研究命题联结词所表达的命题之间的真值关系，并通过这种真值关系来研究推理的有效性**。

1.1.3 命题联结词

命题联结词用来将简单命题组合成复合命题。在命题逻辑中，最常见的命题联结词有五个，其符号分别是：$\neg$、$\wedge$、$\vee$、$\rightarrow$和$\leftrightarrow$。下面我们讨论这五个命题联结词所表达的命题之间的真值关系，以及它们在自然语言中最常见的对应词语。

否定联结词$\neg$

否定(negation)联结词$\neg$是一个一元联结词，也就是它只作用到一个命题上面。命题p加上否定联结词就形成了一个新的命题，记作$\neg p$，这个新命题是原命题的**否定**，通常读作“**非**p”。

为了明确否定联结词所表达的命题间真值关系，根据命题变量p可代表任意命题，我们假定p可赋予任意的真值，然后给出每种情况下$\neg p$的真值，从而给出否定联结此所表达的命题间真值关系：

1. 若p的真值为0，则$\neg p$的真值为1；
2. 若p的真值为1，则$\neg p$的真值为0。

用称作**真值表**的表格形式可更清晰地描述否定联结词所表达的命题间真值关系：

p	$\neg p$
0	1
1	0

在自然语言中，通常用“并非”、“没有”、“不”等词语表示逻辑否定，例如：

- 并非癌症是遗传的。
- 张三昨天没有去看球赛。

- 我不喜欢数理逻辑学。

当然在实际生活中，人们表达否定还有许多方式，需要我们根据说话者的意思，乃至上下文情景进行分析和提炼。

合取联结词∧

合取(conjunction)联结词∧是一个二元联结词，它将两个命题p和q联结起来构成一个新的命题$p \wedge q$，这个新命题是原来两个命题的**合取**，通常简单地读作“p**与**q”。合取联结词所表达的命题间真值关系可用真值表描述如下：

p	q	$p \wedge q$
0	0	0
0	1	0
1	0	0
1	1	1

由于两个命题的真值取值组合共有四种，因此上面的真值表除表头之外还有四行，分别列出p和q取各种真值时，$p \wedge q$的真值。注意，**通常在列真值表时都按照二进制的编码顺序**，例如上面按$00, 01, 10, 11$的顺序，对命题变量进行赋值。通过观察上面的真值表，不难将合取联结词所表达的命题间真值关系概括为：

$$p \wedge q\text{的真值为}1 \text{ 当且仅当 } p\text{和}q\text{的真值都为}1$$

由于真值只有0和1两种取值，因此只要确定了$p \wedge q$何时取值为1，也就确定了$p \wedge q$的所有真值取值情况。

在自然语言中，通常使用“不仅…而且…”、“…并且…”、“…和…”、“…与…”等表示合取联结词，以断定几种事物情况同时存在，在普通逻辑学中，称这种判断为**联言判断**。例如：

- 小张不仅喜欢唱歌，而且喜欢跳舞。
- 小张喜欢唱歌，并且小李也喜欢唱歌。
- 教室里有10名男同学和10名女同学。

要注意的是，在自然语言中表示转折的一些词语，如“虽然…但是…”、“…而…”等，在命题逻辑中也抽象为合取联结词：

- 小张喜欢唱歌，而小李喜欢跳舞。
- 这台机器虽然质量很好，但是价钱也很贵。

在命题逻辑中，自然语言中这种表示转折的含义并不能精确地表达出来，只能表达两个支命题的真值与整个命题的真值之间的联系。例如上面的句子“这台机器虽然质量很好，但是价钱也很贵”显然是在“这台机器质量很好”和“这台机器价钱很贵”这两个支命题同时为真时才为真，因此“虽然…但是…”表达的命题之间的真值关系与合取联结词是相同的。

因此，我们应该知道，命题逻辑中的联结词是自然语言中一些表达句子之间联系的连词的**抽象**，这种抽象关注支命题与整个命题之间的真值关系，这些连词在自然语言中的某些含义便被忽略了。因此，**命题逻辑中的联结词与自然语言中的对应表达词语并不等同**，这一点请大家注意。

析取联结词∨

析取(disjunction)联结词∨也是一个二元联结词，它将两个命题p和q联结起来构成一个新的命题$p \vee q$，这个新命题称为原来两个命题的**析取**，通常简单地读作“p**或**q”。析取联结词所表达的命题间真值关系可用真值表描述如下：

p	q	$p \vee q$
0	0	0
0	1	1
1	0	1
1	1	1

通过观察上面真值表，不难将析取联结词所表达的命题间真值关系概括为：

$$p \vee q\text{的真值为}0 \text{ 当且仅当 } p\text{和}q\text{的真值都为}0$$

同样，由于真值只有0和1两种取值，因此只要确定了$p \vee q$何时取值为0，也就确定了$p \vee q$的所有真值取值情况。

在自然语言中，通常使用“或者…或者…”、“要么…要么…”、“…或…”等表示析取联结词，以表达某种选择，或断定几种可能事物情况至少存在一种情况，在普通逻辑学中，这种判断称为**选言判断**。例如：

- 小张喜欢唱歌或跳舞。
- 我今天要么去参加会议，要么在家里休息。

要注意的是，自然语言中这种表示选择的词语仔细分析有两种含义：一种是相容选择，一种是排斥选择。相容选择是指两种选择可能选择其中一种，也可能两者都可以选择，例如上面的句子“小张喜欢唱歌或跳舞”表达的是，可能小张喜欢唱歌，也可能喜欢跳舞，也可能两者都喜欢。排斥选择是指两种选择只能选择其中一种，例如上面的句子“我今天要么去参加会议，要么在家里休息”，显然两种情况只能有一种情况为真。

在逻辑里如果要进一步分析上面第二个句子，它应该为“我今天要么去参加会议（而不能在家休息），要么在家里休息（而不去参加会议）”，包含了否定、合取和析取等多个联结词。通过析取联结词的真值表我们可以看出，**析取联结词所表达的应该是相容选择**，但是在某些情况下，我们也可将自然语言中表示排斥选择的词语抽象为析取联结词，这要看求解问题的需要，我们后面会通过例子做进一步的说明。

蕴涵联结词→

蕴涵(implication)联结词→也是一个二元联结词，它将两个命题p和q联结起来构成一个新的命题$p \rightarrow q$，读作“p**蕴涵**q”，也可读作“**如果**p，**那么**q”，其中p称为蕴涵的**前件**，而q称为蕴涵的**后件**。蕴涵联结词也称为条件(condition)联结词。蕴涵联结词所表达的命题间真值关系可用真值表描述如下：

p	q	$p\rightarrow q$
0	0	1
0	1	1
1	0	0
1	1	1

通过观察上面真值表，不难将蕴涵联结词所表达的命题间真值关系概括为：

$$p\rightarrow q\text{的真值为}0\ \text{当且仅当}\ p\text{的真值为}1\text{而}q\text{的真值为}0$$

经典命题逻辑中的蕴涵称为**实质蕴涵**，其特点是强调只有当前件为真而后件为假时整个蕴涵式才为假，其他情况都为真，这意味着假的前件可蕴涵任何后件，而真的后件可被任何前件所蕴涵。在数学中也常常需要蕴涵的这种理解，例如，说A是B的子集的定义是：对任意的$x\in A$蕴涵$x\in B$，该定义使用了蕴涵联结词。众所周知，空集$\varnothing$是任意集合B的子集，这正是因为，对任意的x，都没有$x\in\varnothing$，所以$x\in\varnothing$蕴涵$x\in B$总成立，从而$\varnothing$是任意集合B的空集。

在自然语言中，最简单地使用“如果…那么…”、“如果…则（就）…”、“…蕴涵…”等表示蕴涵联结词，以断定事物情况之间的存在条件或事物之间的因果关系，在普通逻辑学中称这种判断为**假言判断**，例如：

- 如果天下雨，地就会湿。
- 自然数n能被4整除蕴涵它也能被2整除。

要注意的是，在自然语言中，句子“如果天下雨，地就湿”强调的是当天下雨时，地就湿，而没有断定天不下雨时的地面情况。将“如果p那么q”与$p\rightarrow q$等同起来容易导致初学者不理解为什么当p的真值为0时，$p\rightarrow q$的真值是真，因为自然语言中当p为假时，“如果p那么q”的含义是模糊的。

实际上，在普通逻辑学中，将事物情况之间的存在条件分为**充分条件**、**必要条件**和**充要条件**：

1. 说p是q的**充分条件**，如果有p，就必然有q，而没有p，是否有q不能确定。在自然语言中，“如果p那么q”、“有p就q”、“一旦p就q”、“假若p就q”等都表示p是q的充分条件；

2. 说p是q的**必要条件**，如果**没有**p，就一定没有q，而有p，是否有q不能确定。在自然语言中，“只有p才q”、“除非p不q”、“除非p才q”、“没有p就没有q”等都表示p是q的必要条件；

3. 说p是q的**充要条件**，如果有p就必然有q，而没有p，就必然没有q。在自然语言中，常用“p当且仅当q”表示p是q的充要条件。

由此我们知道，不管是充分条件的假言判断还是必要条件的假言判断，在数理逻辑中都是用蕴涵联结词来符号化：若p是q的充分条件，则符号化为$p\rightarrow q$；若p是q的必要条件，则符号化为$q\rightarrow p$。而充要条件则使用下面的等价联结词符号化。我们在下一节将进一步讨论自然语言命题在命题逻辑中的符号化。

等价联结词$\leftrightarrow$

等价联结词$\leftrightarrow$也是一个二元联结词，它将两个命题p和q联结起来构成一个新的命题$p\leftrightarrow q$，读作“p**等价于**q”，也可读作“p**当且仅当**q”。等价联结词也称为双条件(bicondition)联结词。等价联结词所表达的命题间真值关系可用真值表描述如下：

p	q	$p \leftrightarrow q$
0	0	1
0	1	0
1	0	0
1	1	1

通过观察上面真值表，不难将等价联结词所表达的命题间真值关系概括为：

$$p \leftrightarrow q\text{的真值为}1 \text{ 当且仅当 } p\text{和}q\text{的真值相同}$$

在自然语言中，通常使用“…当且仅当…”、“…等价于…”、“…的充要条件是…”等表示等价联结词，以表达两个事物情况的存在互为充要条件，例如：

- 自然数n是偶数当且仅当n能被2整除。
- 两个园的面积相等的充要条件是这两个园的半径相同。

小结

上面介绍了五个常用的命题联结词：¬、∧、∨、→以及↔。对这些联结词，我们要注意：

1. 命题联结词表达的是命题间的真值关系，读者应抓住这些联结词所表达真值关系的特点：

(1) $p \wedge q$**为真当且仅当**p**和**q**都为真**；

(2) $p \vee q$**为假当且仅当**p**和**q**都为假**；

(3) $p \rightarrow q$**为假当且仅当**p**为真且**q**为假**；

(4) $p \leftrightarrow q$**为真当且仅当**p**与**q**的真值相同**。

2. 命题联结词是自然语言中表示判断之间关系的一些词语的抽象（符号化），但与这些词语并不完全等同。例如：

(1) “虽然p但是q”被符号化为$p \wedge q$失去了其中的转折含义；

(2) “要么p要么q”可符号化为$p \vee q$但失去了其中排斥选择的含义；

(3) “如果p那么q”通常在自然语言中更强调p为真则q也为真的情况，而当p为假时，其含义可能是模糊的。

特别地，使用命题联结词联结起来的命题还可能是完全无关的，例如，设p表示“雪是白的”，q表示“天会下雨”，那么$p \rightarrow q$表示“如果雪是白的，那么天会下雨”，这种无意义的句子通常不会出现在自然语言中。这是因为在命题逻辑中，**只关心命题的真值以及命题之间的真值关系，而不关心命题本身的具体含义**。

1.1.4 自然语言命题的符号化

数理逻辑采用数学方法研究逻辑，第一步就是使用符号来代替具体的命题。因此数理逻辑本身是研究符号化之后的命题及命题之间真值关系，不过要将数理逻辑用于实际应用问题，例如后面我们利用命题逻辑来进行一些简单的推理，都需要将自然语言表示的命题进行符号化。在程序设计课程中，写出选择语句或循环语句中的条件表达式实际上也是将待求解问题的一些条件符号化程序设计语言中的表达式，其原理与这里讨论的自然语言命题的符号化是一致的。

通常在自然语言中一个句子不会太长，因此若要将是命题的自然语言句子符号化往往只涉及一到两个命题连结词。简单地说，可通过如下步骤将自然语言给出的句子符号化：

1. 首先，判定这个句子是否是命题逻辑中所研究的命题，排除一些不是陈述句的句子，以及一些不具有真假值的句子。

2. 其次，找出这个句子所包含的原子命题，这时要根据有关自然语言的一些常识分析句子的主语和谓语，有时还要显式给出句子中一些可能省略的主语。通常，如果这个句子只有一个主语和一个谓语，则本身是原子命题，否则是复合命题，一个主语及相应的谓语就是一个支命题。

3. 再次，将句子中不同的原子命题用不同的命题变量符号表示。在整个句子中可能同样的原子命题或多次出现，这时要注意用相同的命题变量符号表示。

4. 然后，分析句子中的连词所表达的逻辑含义，确定句子的整体结构，以及各支命题之间的逻辑关系。

5. 最后，使用合适的命题联结词符号化各支命题，并根据句子的结构，写出整个复合命题的符号化形式。

实际上，**将自然语言表示的命题符号化的关键有两点：一是找出其中的原子命题；二是分析支命题之间的逻辑关系**，特别是**分析支命题之间的逻辑关系可能是命题符号化的难点**。要化解这个难点有两个方法：一是**熟记一些常见连词的逻辑含义及其对应的命题联结词**；二是**可借助真值表来澄清支命题与整个命题之间的真值关系**。

下面通过例子来说明如何根据上面的步骤将自然语言表示的命题符号化。首先是简单的例子：

例子 1.1.3 判断下列句子哪些是命题？哪些是简单命题？哪些是复合命题？

1. 中国是社会主义国家。
2. 占世界人口四分之一的中国人是勤劳勇敢的。
3. $2x + y > 10$
4. 为什么青年人都喜欢流行音乐？
5. 请大家上课不要讲小话！
6. 小张与小李是要好的朋友。
7. 小张与小李都喜欢上网。
8. 我正在说假话。
9. 种瓜得瓜。
10. 郑州位于北京和广州之间。

分析：要判断一个句子是否是命题，首先要看它是否是陈述句，其次看它是否具有真假值。是否陈述句容易判断，是否具有真假词则可能要进行一些简单的分析：通常带有公认是变量的句子没有真假值而不是命题，被认为是悖论的句子也不是命题。

要判断命题是简单命题和复合命题通常比较简单，但要注意像“与”、“和”这样的词语不一定具有合取联结词的含义，而可能是表示事物之间的某种关系，因而不是复合命题而是简单命题。

解答：

1. 是命题，而且是简单命题。

2. 是命题，而且是简单命题，在我们的课程中，通常**不需要**将这个命题复杂化，例如认为它包含这样一些意思：“中国人占世界人口四分之一”、“中国人是勤劳的”、“中国是勇敢的”等。

3. 不是命题，因为其中含有变量x, y，从而不具有确定的真假值。

4. 不是命题，因为是一个疑问句，而不是陈述句；

5. 不是命题，因为是一个祈使句，而不是陈述句。

6. 是命题，而且是简单命题，因为它不能分解为“小张是要好的朋友”和“小李是要好的朋友”，这是一个表示事物之间关系的简单命题。

7. 是命题，而且是复合命题，因为可分解为“小张喜欢上网”和“小李喜欢上网”两个支命题。

8. 不是命题，这也是一个悖论。若认为是真，则因为句子说正在说假话；若认为是假，那么这个句子就是真的啦。此悖论与前面“我正在写的句子是假的”相同，都是说谎者型的语义悖论。

9. 是命题，而且是复合命题，因为其实际表达的意思是：“如果我们种瓜，那么我们得瓜”。

10. 是命题，而且是简单命题，同样这是表示事物间关系（具体说是位置关系）的简单命题。

通常，只涉及到否定、合取和析取联结词的命题符号化比较简单，因为在自然语言中表达这些联结词的词语意思都比较清楚。下面的例子给出了有关这三个联结词的符号化：

例子 1.1.4 将下列自然语言表示的命题在命题逻辑中符号化（主要取材于[1]，并稍有修改）：

1. 我们在学好逻辑学的同时，还应学好其他学科。
2. 我虽人到中年，但求知欲望并未减弱。
3. 液体沸腾的原因是湿度增高，或者是压力下降。
4. 要么他们来过，要么你们去过，二者必居其一。
5. 逆水行舟不进则退。
6. 将外国文化全盘否定或全盘吸收是不对的。
7. 对待外国文化的态度不是全盘否定或全盘吸收。

分析与解答：

1. 句子“我们在学好逻辑学的同时，还应学好其他学科”包含了两个原子命题（下面括号中按句意给出了原本省略的词语）：

$$p:\quad \text{我们（应）学好逻辑学} \qquad\qquad q:\quad \text{（我们）应学好其他学科}$$

而连词“在…的同时，还…”表明是同时给出这两个判断，因而要使用合取联结词将命题符号化：

$$p \wedge q$$

2. 句子“我虽人到中年，但求知欲望并未减弱”包含了两个原子命题：

$$p:\quad \text{我人到中年} \qquad\qquad q:\quad \text{（我的）求知欲望并未减弱}$$

而连词“虽…但…”同样表明应该使用合取联结词，因此该命题符号化为：

$$p \wedge q$$

注意，这里通常无需再将命题“我的求知欲望并未减弱”进一步分解为“r: 我的求知欲望减弱”，而符号化为$\neg\, r$。

在碰到带有否定意味的命题时，特别是像“…不是…”这样的句式时，例如，“2不是无理数”，将整个命题看作简单命题也并无不可。除非：(i). 句子特别强调其否定含义，例如像“并非……”这种

句式；(ii). 整个句子中还含有相同的而不带否定意味的支命题，例如句子同时包含“2不是无理数”和“2是无理数”这两个支命题。

当然，在本题中，令p表示“我人到中年”，q表示“我的求知欲望减弱”，而整个句子符号化为$p \wedge \neg q$也是完全正确的解答，关键是**要写清楚命题变量所代表的原子命题**。

3. 句子“液体沸腾的原因是湿度增高，或者是压力下降”包括两个原子命题：

p： 液体沸腾的原因是湿度增高　　q： （液体沸腾的原因）是压力下降

连词“或者”表明应该使用析取联结词将句子符号化为：

$$p \vee q$$

4. 句子“要么他们来过，要么你们去过，二者必居其一”包括两个原子命题：

p： 他们来过　　q： 你们去过

根据连词“要么…要么…”，以及句中的强调“二者必居其一”，我们应将句意理解为“要么他们来过而你们没去过，要么你们去过而他们没来过”，从而应将句子符号化为：

$$(p \wedge \neg q) \vee (q \wedge \neg p)$$

前面说过，选言判断中有相容选择和排斥选择之分，那么对于“要么…要么…”等这种选言判断是简单地用析取联结词符号化，还是用上面这种方式符号化呢？我们建议，**如果句子强调排斥选择，则用上面这种方式符号化，否则就简单地使用析取联结词符号化**。

有时选言判断中的两个（或多个）支命题本身不能同时为真，例如“小张生于1971年或生于1972年”中的两个支命题“小张生于1971年”和“（小张）生于1972年”按照常识不能同时为真，那么这时**更可以**简单地用析取联结词将其符号化即可，因为这时用$(p \wedge \neg q) \vee (q \wedge \neg p)$的形式来强调$p$和$q$已无必要。

有的教材或老师**可能不赞同**上述观点，可能碰到这种情况要求用排斥选择的形式符号化，为保险起见，读者在碰到这种题目时可以：(i). 用排斥选择的形式符号化，并指出是因为支命题不能同时为真；或者(ii). 用析取联结词符号化，并指出因为支命题不能同时为真，所以没有必要使用排斥选择的形式符号化来强调。

5. 句子“逆水行舟不进则退”中包含两个原子命题：

p： 逆水行舟（会）进　　q： 逆水行舟（会）退

句子中的“不进则退”是说明这两个情况必居其一，但进和退不能同时为真，因此我们仍可简单地使用析取联结词将其符号化为：

$$p \vee q$$

6. 对于句子“将外国文化全盘否定或全盘吸收是不对的”，我们可以认为“将外国文化全盘否定或全盘吸收”这个整体看作整个句子的主语，“是不对的”看作谓语，从而令p表示“将外国文化全盘否定或全盘吸收是对的”，整个句子符号化为$\neg p$，或者干脆说整个命题是简单命题，用命题变量p符号化即可。同样，也有人会不同意这种观点，认为这个句子应该提炼出下面两个原子命题：

p： （我们）将外国文化全盘否定　　q： （我们）将外国文化全盘吸收

然后整个句子符号化：

$$\neg\,(p \vee q)$$

这也是可接受的解答。读者在解答这种题目时**可根据自己的观点选择其中一种，并说明理由**。

7. 句子“对待外国文化的态度不是全盘否定或全盘吸收”的意思比较明确，应该提炼出两个原子命题：

p： 对待外国文化的态度是全盘否定　　q： 对待外国文化的态度是全盘吸收

从而整个句子符号化：

$$\neg\,(p \vee q)$$

通过上面的例子，我们看到：**自然语言是有歧义的，不同人的理解可能不同，在将自然语言命题符号化时也可能有不同的观点**，因此读者在符号化自然语言命题时应该注意：

1. **一定要给出句子中包含的原子命题**，并用合适的命题变量符号表示；
2. 必要时，**给出你为什么这样符号化的理由**。

下表总结了命题符号化时涉及否定、合取和析取三个联结词的典型句式以及注意事项：

典型句式	符号化	注意事项
并非p	$\neg\,p$	一些带有否定意味的简单命题可整个看作简单命题，除非句意强调，或同时有不带否定意味的同类命题。
不仅 p 而且 q 一边 p 一边 q 既 p 又 q 虽然 p 但是 q p 并且 q p 与 q p 和 q p 而 q	$p \wedge q$	有时“与”、“和”等连词会用于表示关系判断，从而是简单命题。
或者 p 或者 q 要么 p 要么 q 也许 p 也许 q 可能 p 可能 q p 或 q	$p \vee q$	(i). 如果句意强调，例如有“二者必居其一”之类的词语则使用下面排斥选择的符号化形式；(ii). 有时 p 和 q 本身就不能同时为真，这时可使用排斥选择的符号化形式，也可简单地使用析取并说明理由。
p 或 q 二者必居其一 p 或 q 二者不可兼得 只能是 p 或 q 之一	$(p \wedge \neg\, q) \vee (q \wedge \neg\, p)$	这些句子强调排斥选择，因此使用排斥选择的符号化形式。

自然语言中，表示条件或因果关系的词语更为丰富多彩，而且不同的词语所表示的条件不同，有些表示充分条件，有些表示必要条件，读者需要仔细分析。请看下面的例子：

例子 1.1.5 将下列自然语言表示的命题在命题逻辑中符号化（主要取材于[1]，并稍有修改）：

1. 如果看不到事物的否定方面，就不能科学地预见事物发展的方向。
2. 只有懂了事物的对立统一规律，才能懂得事物的发展。
3. 只要你努力，就会取得一定的成果。
4. 会休息的人，才会工作。
5. 不会休息的人，就不会工作。
6. 没有革命的理论，就没有革命的行动。
7. 理论一旦被群众掌握，就变成物质力量。
8. 哪里有他，哪里就有歌声。
9. 若要人不知，除非己莫为。
10. 除非他真心悔改，才能得到群众的谅解。

分析与解答：

1. 句子“如果看不到事物的否定方面，就不能科学地预见事物发展的方向”是最基本的表示**充分条件**的句型，有两个原子命题：

$$p:（我们）看不到事物的否定方面 \qquad q:（我们）不能科学地预见事物发展的方向$$

整个句子符号化为：

$$p \to q$$

这里同样**无需**引入原子命题r表示“我们能科学地预见事物发展的方向”而将句子符号化为$p \to \neg r$。

2. 句子“只有懂了事物的对立统一规律，才能懂得事物的发展”是最基本的表示**必要条件**的句型，也即“懂了事物的对立统一规律”是“懂得事物的发展”的必要条件，从另一角度说，句子的含义是：“如果（你）懂得事物的发展，就（表示你）懂了事物的对立统一规律”。句子有两个原子命题：

$$p:（你）懂了事物的对立统一规律 \qquad q:（你）懂得事物的发展$$

前者是后者的必要条件，即后者蕴涵前者，整个句子应该符号化为：

$$q \to p$$

3. 句子“只要你努力，就会取得一定的成果”也是表示**充分条件**，相当于“如果你努力，就会取得一定的成果”。句子有两个原子命题：

$$p:\ 你努力 \qquad q:（你）会取得一定的成果$$

整个句子符号化为：

$$p \to q$$

从现实意义上说，这个命题不为真。因为按照通常的理解，努力是取得一定成果的必要条件，即我们认为命题“只有你努力，才能取得一定的成果”是真命题，而上面这句话只是用来勉励人努力的，因为不努力更不能取得成果！

4. 句子“会休息的人，才会工作”表示**必要条件**，相当于“只有会休息的人，才会工作”，句子有两个原子命题：

$$p:（你是）会休息的人 \qquad q:（你）会工作$$

前者是后者的必要条件，即后者蕴涵前者，整个句子应该符号化为：

$$q \rightarrow p$$

5. 句子“不会休息的人，就不会工作”表示**充分条件**，相当于“如果你是不会休息的人，那么你就不会工作”。为了与上面对比，我们同样令：

p：（你是）会休息的人　　q：（你）会工作

则这个句子符号化为：

$$\neg p \rightarrow \neg q$$

注意，这里我们根据对自然语言的理解，“（你是）不会休息的人”等同于“并非（你是）会休息的人”，而“（你）不会工作”等同于“并非（你）会工作”，因此使用了否定联结词。

后面我们将看到，在命题逻辑中，$q \rightarrow p$和$\neg p \rightarrow \neg q$具有相同的真值。但在自然语言中，我认为这两者稍有差别。我们用类似真值表的形式讨论这两个句子的含义：

（你是）会休息的人	你会工作	会休息的人，才会工作	不会休息的人，就不会工作
假	假	真?	真△
假	真	假	假
真	假	真?	真?
真	真	真△	真?

上面的表格应该这样理解：当“（你是）会休息的人”为假，且“你会工作”也为假时，也即，你不会休息，而且也不会工作，那么与断言“会休息的人，才会工作”是否冲突呢？不冲突，所以我们在这一列标上真，与断言“不会休息的人，就不会工作”当然更不冲突，因此这一列也标上真。至于在“会休息的人，才会工作”这一列的真后面打个问号，是因为虽然“你不会休息，而且也不会工作”与该断言不冲突，但是该断言也没有正面断定这种情况，因此这时的真是有些模糊的。而断言“不会休息，就不会工作”则是对这种情况给予了正面肯定，这时的真是清楚的，所以上面以三角形标记加以强调。

其他的行也可类似地理解：当你不会休息，但你会工作，那么与断言“会休息的人，才会工作”是冲突的，与“不会休息的人，就不会工作”当然也是冲突的，所以这两个断言下面这一行都标记为假。而当你会休息，但你不会工作，与这两个断言虽然都不冲突，但它们都没有给予正面的肯定，所以我们用加问号的真标记。当你既会休息又会工作时，断言“会休息的人，才会工作”对这种情况给予了正面肯定，而另一个断言却没有给予正面肯定。

从上面的详细分析，我们可以知道，**虽然在逻辑上是相同的判断，但自然语言的不同表示有不同的强调和侧重**。自然语言的不同侧重点和某种模糊性，使得自然语言带有歧义，不同人可能有不同的理解。例如，有些初学者可能认为，“会休息的人，才会工作，不会休息的人，就不会工作”，因此“会休息的人”等价于“会工作的人”。我们可通过像上面列真值表的形式对于句意作深入的分析，而得到句子的正确理解。

6. 句子“没有革命的理论，就没有革命的行动”表示“革命的理论”是“革命的行动”的**必要条件**，或者说，“没有革命的理论”是“没有革命的行动”的充分条件，相当于“如果没有革命的理论，就没有革命的行动”。令：

p：（我们）有革命的理论　　q：（我们）有革命的行动

则整个句子符号化为：

$$\neg p \rightarrow \neg q$$

根据上面的讨论，$\neg p \rightarrow \neg q$与$q \rightarrow p$有相同的真值，也即上面的句子也相当于：“只有有革命的理论，才有革命的行动”，以及：“如果有了革命的行动，就（意味着）有革命的理论”。

7. 句子“理论一旦被群众掌握，就变成物质力量”也表示**充分条件**，相当于“如果理论被群众掌握，就变成物质力量”。句子有两个原子命题：

p： 理论被群众掌握　　q：（理论）变成物质力量

整个句子符号化为：

$$p \rightarrow q$$

8. 句子“哪里有他，哪里就有歌声”也表示**充分条件**，相当于“只要哪里有他，哪里就有歌声”。句子有两个原子命题：

p： 哪里有他　　q： 哪里有歌声

整个句子符号化为：

$$p \rightarrow q$$

9. 句子“若要人不知，除非己莫为”虽然短，但要理解“人不知”与“己莫为”之间的逻辑关系，需要做一些细致的分析。我们同样使用类似真值表的形式：

人不知	己莫为	若要人不知，除非己莫为	注释
假	假	真?	此句子没有给出“人知”且“己为”这种情况的正面断定
假	真	真?	同样，此句子也没有给出“人知”且“己莫为”这种情况的正面断定。此句子对于“人知”时的判断是模糊的，既没有断定一定是“己为”，也没有断定一定是“己莫为”
真	假	假△	此句子强调的是“人不知”且“己为”这种情况是不可能的
真	真	真	

所以整个句子相当于“只有己莫为，才能人不知”，表示“己莫为”是“人不知”的必要条件。令：

p： 人不知　　q： 己莫为

则整个句子符号化为：

$$p \rightarrow q$$

如果令s表示“人知”，而r表示“己为”，则句子符号化为：$\neg s \rightarrow \neg r$，这与$r \rightarrow s$等价，即相当于“如果已为，则人知”。

10. 通过上一题的分析，我们知道，“除非”实际上相当于“只有”，因此句子“除非他真心悔改，才能得到群众的谅解”相当于“只有他真心悔改，才能得到群众的谅解”，表示“他真心悔改”是“（他）得到群众谅解”的必要条件。令：

p：他真心悔改　　q：（他）得到群众的谅解

则整个句子符号化为：

$$q \rightarrow p$$

从上面的例子可以看出，自然语言（汉语）中表示条件的词语丰富多彩，有些表示充分条件，有些表示必要条件，读者要仔细分析，必要时可借助类似真值的形式进行分析。下面再看与蕴涵联结词有关的自然语言命题符号化例子：

例子 1.1.6 令p表示“今天下雨”，q表示“（今天）我上街”，符号化下列命题：

1. 如果今天不下雨，我就上街。
2. 因为今天下雨，所以我不上街。
3. 只要今天下雨，我就不上街。
4. 只有今天不下雨，我才上街。
5. 除非今天下雨，否则我上街。
6. 今天我上街，除非天下雨。

分析与解答：

1. “如果今天不下雨，我就上街”表示“今天不下雨”是“我上街”的充分条件，应符号化为：

$$\neg p \rightarrow q$$

2. “因为今天下雨，所以我不上街”表示“今天下雨”是“我不上街”的原因，应符号化为：

$$p \rightarrow \neg q$$

对于上面两个句子，我们看到，“如果今天不下雨，我就上街”，强调的是“今天不下雨”的情况，断定“天不下雨”和“我上街”这种情况，否定“天不下雨”和“我不上街”这种情况，而对于“今天下雨”会怎样，则是模糊的。而“因为今天下雨，所以我不上街”强调的是“今天下雨”的情况等等。

3. “只要今天下雨，我就不上街”表示“今天下雨”是“我不上街”的充分条件，应符号化为：

$$p \rightarrow \neg q$$

4. “只有今天不下雨，我才上街”表示“今天不下雨”是“我上街”的必要条件，应符号化为：

$$q \rightarrow \neg p$$

实际上，在逻辑上$p \rightarrow \neg q$和$q \rightarrow \neg p$具有相同的真值。

5. 句子“除非今天下雨，否则我上街”相当于“只有今天下雨，我才不上街”，表示“今天下雨”是“我不上街”的必要条件，因此该句子应该符号化为：

$$\neg q \rightarrow p$$

6. 句子“今天我上街，除非天下雨”同样相当于“只有今天下雨，我才不上街”，同样应该符号化为：

$$\neg q \to p$$

也可以说该句子的含义是“今天我没有上街，意味着天下雨了”，同样是符号化为$\neg q \to p$。

下标总结了有关蕴涵联结词和等价联结词的句式及其符号化：

典型句式	符号化	注释
如果 p 那么 q 如果 p 则 q 有 p 就有 q 一旦 p 就 q 假若 p 就 q 哪里 p 哪里就 q 只要 p 就 q	$p \to q$	这些句式表示p是q的充分条件。
因为 p 所以 q	$p \to q$	这表示p是q的原因。
只有 p 才 q 除非 p 才 q	$q \to p$	这些句式表示p是q的必要条件。
除非 p 否则 q	$\neg q \to p$	相当于：只有 p，才不 q。
p 除非 q	$\neg p \to q$	相当于：只有 q，才不 p。
不 p 不 q 没有 p 没有 q	$\neg p \to \neg q$	这些句式表示$\neg p$是$\neg q$的充分条件。
p 当且仅当 q p 的充要条件是 q p 等价于 q 如果 p 就 q，反之亦然	$p \leftrightarrow q$	

与等价联结词有关的符号化通常比较简单，这里不再给出例子。下面的例子综合了有关各联结词的符号化：

例子 1.1.7 将下列命题符号化（来自于[2]，有修改）：

1. 如果恐怖分子的要求能在规定期限内满足，则全体人质就能获释；否则，恐怖分子就要杀害人质，除非特种部队能实施有效的营救。

2. 如果小张在孩子落水的现场但没有参加营救，那么，或者他看到了孩子落水却假装没有看见，或者他确实不会游泳。

3. 如果光强调团结，不强调斗争，或者光强调斗争，不强调团结，就不能达到既弄清思想又团结同志的目的。

分析：正如前面所说，对于这些复杂句子的符号化，我们应该：(i). 通读句子，找出不同的原子命题以命题变量表示；(ii). 根据连词的逻辑含义，分析句子的结构，确定各支命题之间的逻辑关系；(iii). 符号化各支命题，并根据句子的结构，使用合适的命题联结词联结各支命题。

解答：

1. 如果恐怖分子的要求能在规定期限内满足，则全体人质就能获释；否则，恐怖分子就要杀害人质，除非特种部队能实施有效的营救。

读一遍句子，可发现该句子中的原子命题包括：

p： 恐怖分子的要求能在规定期限内满足　　q：全体人质能获释

r： 恐怖分子杀害人质　　s：特种部队能实施有效的营救

根据句子的结构，显然“否则”前面是一部分，而“否则”后面是一部分，前面应符号化为：$p \to q$，后面部分的含义是：“只有特种部队能实施有效的营救，恐怖分子才不能杀害人质”，应符号化为：$\neg\, r \to s$。而“否则”是否定p，表示“如果非p，则……”，整个句子的结构是：“如果p，则q；如果非p，则$\neg\, r \to s$”，因此整个句子应符号化为：

$$(p \to q) \wedge (\neg\, p \to (\neg\, r \to s))$$

2. 如果小张在孩子落水的现场但没有参加营救，那么，或者他看到了孩子落水却假装没有看见，或者他确实不会游泳。

读一遍句子，可发现该句子中的原子命题包括：

p： 小张在孩子落水的现场　　q：（小张）没有参加营救

r：（小张）看到了孩子落水　　s：（小张）假装没有看见（孩子落水）

t：（小张）确实不会游泳

显然整个句子的结构是：“如果…，那么，或者…或者…”，“如果”后面应符号化为：$p \wedge q$，前一个“或者”后面应符号化为：$r \wedge s$。注意，这里表示转折的“但”、“却”都应该使用合取联结词符号化。从而整个句子应符号化为：

$$(p \wedge q) \to ((r \wedge s) \vee t)$$

注意，这里没有使用否定联结词符号化“（小张）没有参加营救”以及“（小张）确实不会游泳”，因为我觉得上述解答就够了。当然，如果使用u表示“（小张）参加营救”，v表示“（小张）会游泳”，将整个句子符号化为：

$$(p \wedge \neg\, u) \to ((r \wedge s) \vee \neg\, v)$$

当然也是对的。

3. 如果光强调团结，不强调斗争，或者光强调斗争，不强调团结，就不能达到既弄清思想又团结同志的目的。

读一遍句子，可发现该句子中的原子命题包括：

p：（我们）强调团结　　q：（我们）强调斗争

r：（我们）达到弄清思想（的目的）　　s：（我们达到）团结同志的目的

显然句子的整体结构是:“如果…或者…，就不能……”，其中“如果”后面,“或者”前面应符号化为：$p \wedge \neg q$,“或者”后面应符号化为：$q \wedge \neg p$,“就不能”后面应符号化为：$r \wedge s$，从而整个句子应符号化为：

$$((p \wedge \neg q) \vee (q \wedge \neg p)) \rightarrow \neg (r \wedge s)$$

这里“（我们）强调团结”和“（我们）强调斗争”都同时出现了含义相反的同类命题，因此原子命题应使用肯定形式的命题，并使用否定联结词符号化其否定形式的命题。“就不能”中的“不能”**否定的是一个复合命题，也必须引入否定联结词**。

1.2 命题逻辑公式及其真值

上一节介绍了命题的含义、命题联结词以及自然语言命题的符号化，符号化得到的就是命题逻辑公式，这一节给出命题逻辑公式及其真值的严格定义。我们将所有命题逻辑公式的集合看作用于研究命题逻辑的形式语言（即符号化语言）记为$\mathfrak{L}_P$，严格定义命题逻辑公式，就是根据归纳定义的基本思想定义该语言的构造。

1.2.1 命题逻辑公式的定义

简单地说，定义一个语言大致包括以下几个方面：

1. 首先要确定这个语言的**符号表**，也即要确定这个语言中允许出现些什么符号。例如，英语允许使用二十六个字母及一些标点符号，C++语言允许使用二十六个字母、一些运算符、分界符和一些标点符号等。

2. 其次要定义怎样的符号串（也即句子）属于这个语言，也即要给出语言的**语法**(syntax)规则。例如，英语有英语的语法规则，C++语言也有自己的语法规则，通常来说，计算机程序设计语言的语法比较简单，可按照归纳定义的基本思想定义，而自然语言的语法比较复杂。

3. 给出语言的语法规则之后，进一步可研究句子所表示的含义，称为语言的**语义**(semantics)。通常使用数学化的方法研究计算机程序设计语言的语义，称为**形式语义**(formal semantics)。

命题逻辑的形式语言像计算机程序设计语言一样，属于人工语言，符号表比较简单，构造公式（即该语言的句子）的语法规则可使用归纳法定义，而公式的语义就是指公式的真值。下面我们首先给出命题逻辑形式语言$\mathfrak{L}_P$的符号表：

定义 1.2.1 语言$\mathfrak{L}_P$的**符号表**包括三类符号:

1. **小写英文字母**: 称为**命题变量**, 所有可能出现的命题变量构成的集合记为**Var**;
2. **命题联结词**: 包括$\neg, \wedge, \vee, \rightarrow, \leftrightarrow$;
3. **辅助符号**: 包括圆括号$(,)$。

进一步，我们使用归纳定义的基本思想给出命题逻辑公式的语法：

定义 1.2.2 语言$\mathfrak{L}_P$的元素（即句子）称为**命题逻辑公式**，简称**公式**，其归纳定义是:

1. **归纳基**: 命题变量是命题逻辑公式;
2. **归纳步**: 若A和B是命题逻辑公式，则

$$(\neg A) \qquad (A \wedge B) \qquad (A \vee B) \qquad (A \rightarrow B) \qquad (A \leftrightarrow B)$$

都是命题逻辑公式;

3. **最小化**: 语言$\mathfrak{L}_P$的所有命题逻辑公式通过有限次运用上面两个规则得到的。

简单地说，归纳定义一个集合（及其元素）的基本思想就是：

1. 首先，归纳基给出该集合的基本元素，例如上面命题变量是命题逻辑公式的基本元素。

2. 其次，归纳步给出由集合的已有元素构造其他元素的规则，例如当A和B是命题逻辑公式时，$(A \wedge B)$也是命题逻辑公式。注意这里A和B本身不是语言$\mathfrak{L}_P$所允许的符号，而是用来表示在这种构造中，A和B可用$\mathfrak{L}_P$中的任意命题逻辑公式代入。

3. 最后，最小化是指该集合的所有元素都通过有限次使用归纳基和归纳步得到。具体来说，对于命题逻辑公式，这意味着，任意命题逻辑公式要么是命题变量，要么具有$(\neg A), (A \wedge B), (A \vee B), (A \to B), (A \leftrightarrow B)$这五种形式之一，其中$A$和$B$是任意（已经得到且符合定义的）命题逻辑公式。“有限次使用”意味着在这五种形式的公式中，A和B比整个公式更简单，可通过更少步数得到。

例子 1.2.3 我们先看几个符合上述定义的命题逻辑公式的归纳构造过程：

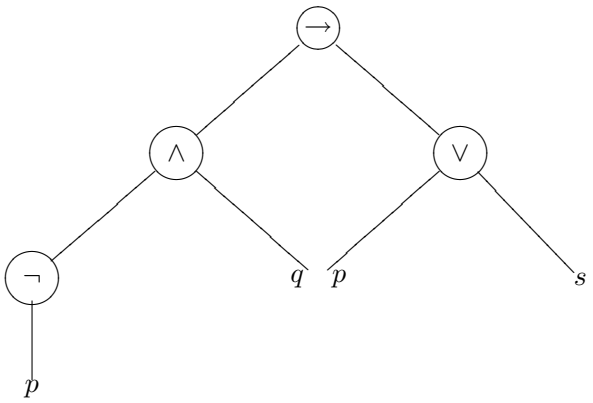
1. p是公式，上述定义的归纳基说明命题变量p是最基本的命题逻辑公式，构造该公式的步数可认为是0步。

2. $(\neg\, p)$是公式，具有形式$(\neg A)$，其中A用命题公式p代入，构造该公式的步数是1步。

3. $((\neg\, p) \wedge q)$是公式，具有形式$(A \wedge B)$，其中A用公式$(\neg\, p)$代入，B用q代入，A的构造步数是1步，而B的构造步数是0步，因此整个公式的构造步数是$1 + 0 + 1 = 2$步。

4. $(p \vee s)$是公式，具有形式$(A \vee B)$，其中A用公式p代入，B用s代入，A和B的构造步数是0步，整个公式的构造步数是$0 + 0 + 1 = 1$步。

5. $(((\neg\, p) \wedge q) \to (p \vee s))$是公式，具有形式$(A \to B)$，其中$A$用公式$((\neg\, p) \wedge q)$代入，而$B$用公式$(p \vee s)$代入，$A$的构造步数是2步，$B$的构造步数是1步，整个公式的构造步数是$2 + 1 + 1 = 4$步。该公式的归纳构造过程可用下面的树描述：



这种树称为公式的语法树，内部节点是命题联结词符号，叶子节点是命题变量符号，树的形式描述了公式的结构。在数据结构课程我们将知道，中序遍历上述语法树则得到公式$((\neg\, p) \wedge q) \to (p \vee s))$。

细心的读者不难看出，公式的构造步数实际上等于公式中所含命题联结词符号的个数。实际上，根据公式的归纳定义，我们也可归纳定义公式的构造步数（也即公式中所含命题联结词符号个数）：

定义 1.2.4 对任意命题逻辑公式A，其构造步数$\#(A)$归纳定义为：

1. 归纳基：若公式A是命题变量，则$\#(A)=0$；

2. 归纳步：(i) 若公式A具有形式$(\neg B)$，则$\#(A)=\#(B)+1$；(ii) 若公式具有形式$(B\wedge C),(B\vee C),(B\to C)$或$(B\leftrightarrow C)$之一，则$\#(A)=\#(B)+\#(C)+1$。

上述定义也体现了归纳定义的好处，即如果一个集合（像$\mathfrak{L}_P$）本身是归纳定义的，那么该集合（及其元素）的一些性质也可用归纳法加以定义。再例如，可用归纳法定义一个公式的**子公式**：

定义 1.2.5 对任意命题逻辑公式A，它的**子公式**归纳定义为：

1. 归纳基：若公式A是命题变量，则它**没有**子公式；

2. 归纳步：(i) 若公式A具有形式$(\neg B)$，则B是它的子公式；(ii) 若公式具有形式$(B\wedge C),(B\vee C),(B\to C)$或$(B\leftrightarrow C)$之一，则$B$和$C$都是它的子公式。

根据这个定义，$((\neg p)\wedge q)\to(p\vee s))$的子公式包括$p,q,s,(\neg p),(p\vee s),((\neg p)\wedge q)$，而$q\to p,p\wedge q$就不是它的子公式。

备注 1.2.6 实际上，归纳定义在程序设计语言的语法定义中也被普遍使用，例如，表达式的基本定义是：(i) **归纳基**：变量和常量是表达式；(ii) **归纳步**：如果A和B是表达式，则$(A+B)$是表达式，等等。

当然，在程序设计语言中，通常使用BNF定义语法规则，例如：表达式 ::= 常量|变量|(表达式+表达式)，等等，这是一种形式更为简单的归纳定义，其中“常量”、“变量”和“表达式”是非终结符号，这几个汉字本身不允许出现在表达式中，而$(,+,)$是终结符号，允许出现在表达式中。

类似地，命题逻辑公式也可用BNF的简单形式定义：

$$\text{公式} \quad ::= \quad \text{命题变量}|(\neg\text{公式})|(\text{公式}\wedge\text{公式})|\text{公式}\vee\text{公式}|\text{公式}\to\text{公式}|\text{公式}\leftrightarrow\text{公式}$$

正如可使用编译程序识别一个符号串是否是表达式，也可编写程序识别一个符号串是否是命题逻辑公式，而且通过这种识别过程可自动地构造公式的语法树，正如编译程序实际上也会为表达式构造语法树，以便生成计算表达式值的代码。

例子 1.2.7 下面看几个不符合定义1.2.2，从而不是命题逻辑公式的串：

1. (p)不是公式，因为不是命题变量，也不属于$(\neg A),(A\wedge B),(A\vee B),(A\to B),(A\leftrightarrow B)$五种形式的任意一种。

2. $(p\to q$不是公式，同样不是命题变量，也不属于定义中五种形式的任意一种，或者直观地说，定义中五种形式的圆括号都是配对的，而这个串中的圆括号不配对；

3. $p\wedge q$不是公式，同样不是命题变量，也不属于定义中五种形式的任意一种，或者直观地说，定义中五种形式的公式都用圆括号括住，而这个串不是命题变量，又不含圆括号。

上面的例子表明，如果严格按照定义1.2.2，串$p\wedge q$不是公式，但在实际应用中，我们常常将这个串看作合适的命题逻辑公式，这时因为定义1.2.2给出的是关于命题逻辑公式的严格定义，由于使用了归纳定义的思想，对于理解命题逻辑公式的结构十分有用，而且也便于使用归纳法进一步研究公式的性质，例如定义公式的子公式等。但在实际应用中，该定义过于严格，使得构造出的公式有太多圆括号而不简洁，因此常常像程序设计语言中的规定运算符的优先级一样，通过规定命题联结词之间的优先级来省略多余的括号使得公式看起来更为简洁。通常约定：

1. ¬的优先级最高，其次是∧，再次是∨，再次是→，最后是↔。因此：

$$(((\neg p) \wedge q) \rightarrow (p \vee s)) \qquad \text{可简写为} \qquad \neg p \wedge q \rightarrow p \vee s$$
$$p \wedge q \leftrightarrow s \rightarrow r \wedge q \qquad \text{相当于} \qquad ((p \wedge q) \leftrightarrow (s \rightarrow (r \wedge q)))$$

2. ∧, ∨, ↔都是从左至右结合，因此：

$$p \wedge q \wedge r \qquad \text{相当于} \qquad ((p \wedge q) \wedge r)$$
$$p \vee q \vee r \qquad \text{相当于} \qquad ((p \vee q) \vee r)$$
$$p \leftrightarrow q \leftrightarrow r \qquad \text{相当于} \qquad ((p \leftrightarrow q) \leftrightarrow r)$$

3. 但→**是从右至左结合**，因此：

$$p \rightarrow q \rightarrow r \qquad \text{相当于} \qquad (p \rightarrow (q \rightarrow r))$$

规定命题联结词之间的优先级与组合性可使得公式的书写大为简化，但是像公式

$$\neg p \wedge q \rightarrow p \vee s \qquad \text{和} \qquad p \wedge q \leftrightarrow s \rightarrow r \wedge q$$

省略太多圆括号而容易让人产生误会或不便于理解公式的结构，因此通常采用折衷的办法保留一些圆括号使得公式既简洁又清晰，例如上述公式通常写为：

$$(\neg p \wedge q) \rightarrow (p \vee s) \qquad \text{和} \qquad (p \wedge q) \leftrightarrow (s \rightarrow (r \wedge q))$$

为简便起见，我们以后采用折衷的方式书写公式，但在某些时候讨论命题逻辑公式的性质，如讨论公式的子公式，仍应该根据公式的归纳定义来确定，例如，$p \wedge q$和$r \wedge q$是公式$p \wedge q \leftrightarrow s \rightarrow r \wedge q$的子公式，而$q \leftrightarrow s$和$s \rightarrow r$就**不是**该公式的子公式，因为此公式的严格形式是$((p \wedge q) \leftrightarrow (s \rightarrow (r \wedge q)))$。

下面讨论公式的真值时也采用相同的处理方式，即使用归纳法定义公式的真值时是根据公式的严格语法形式，但后面举例给出的公式则采用简便形式，关键是**读者心中要知道每个公式的归纳构造过程**，不管给出这个公式时是采用严格语法形式，还是采用省略圆括号的简便形式。

1.2.2 命题逻辑公式的真值

定义公式的真值也使用归纳法，归纳基显然是要确定命题变量的真值，但我们知道命题变量是具体命题的抽象，具体命题的真值是根据是否符合客观实事而定，显然这里不能讨论具体命题的真值，因此我们假定有一个函数指派公式中出现的命题变量的真值，然后在此基础上归纳确定整个公式的真值，而这种归纳过程正体现了命题联结词的语义。我们先定义：

定义 1.2.8 **真值赋值函数**是具有形式t：**Var**→$\{0,1\}$的函数，它为每个命题变量p指派一个**真值**$t(p)\in\{0,1\}$。

这里**Var**是所有可能出现的命题变量构成的集合，但实际上，我们并不需要考虑所有命题变量，而是只考虑当前研究的公式中出现的所有命题变量，只要给出真值赋值函数t为这些命题变量指派的真值即可。

定义 1.2.9 给定真值赋值函数t：**Var**$\to\{0,1\}$，任意命题逻辑公式A在该真值赋值函数下的真值$t(A)\in\{0,1\}$归纳定义为：

1. **归纳基**：如果公式A是命题变量p，则$t(A)=t(p)$；
2. **归纳步**：
   (i). 如果公式A具有形式$(\neg B)$，则$t(A)=1$当且仅当$t(B)=0$；
   (ii). 如果公式A具有形式$(B\wedge C)$，则$t(A)=1$当且仅当$t(B)$和$t(C)$都等于1；
   (iii). 如果公式A具有形式$(B\vee C)$，则$t(A)=0$当且仅当$t(B)$和$t(C)$都等于0；
   (iv). 如果公式A具有形式$(B\to C)$，则$t(A)=0$当且仅当$t(B)=1$而$t(C)=0$；
   (v). 如果公式A具有形式$(B\leftrightarrow C)$，则$t(A)=1$当且仅当$t(B)=t(C)$。

根据前面的讨论，当A具有形式$(\neg B)$时，B的构造步数比A少，因此计算$t(A)$所引起的递归计算$t(B)$是计算步数更少的公式，从而这种递归计算一定会终止，这说明上述定义的$t(A)$是合适的。

例子 1.2.10 研究公式$A=(\neg\, p\wedge q)\to(p\vee s)$在真值赋值函数$t$：**Var**$\to\{0,1\}$下的真值。由于$A$中出现的命题变量包括$p,q,s$，定义真值赋值函数$t$时，我们只需给出$t(p),t(q),t(s)$。假定：

$$t(p)=1\qquad t(q)=0\qquad t(s)=1$$

则公式A的真值$t(A)$递归计算如下：

1. 公式A具有形式$(B\to C)$，其中$B=(\neg\, p\wedge q),C=(p\vee s)$，因此需要递归计算$t(B)$及$t(C)$；
2. 公式B具有形式$(B_1\wedge B_2)$，其中$B_1=(\neg\, p),B_2=q$，因此需要递归计算$t(B_1)$及$t(B_2)$；
3. 公式B_1具有形式$(\neg\, B_{11})$，其中$B_{11}=p$，因此需要递归计算$t(B_{11})$；
4. 公式$B_{11}=p$，因此$t(B_{11})=t(p)=1$；
5. 从而由定义，$t(B_1)=1$当且仅当$t(B_{11})=0$，而$t(B_{11})=1$，因此$t(B_1)=0$；
6. 公式$B_2=q$，因此$t(B_2)=t(q)=0$；
7. 从而由定义，$t(B)=1$当且仅当$t(B_1)$和$t(B_2)$都等于1 ，而$t(B_1)=0$，因此$t(B)=0$；
8. 公式(C)具有形式$(C_1\vee C_2)$，其中$C_1=p,C_2=s$，因此需要递归计算$t(C_1)$和$t(C_2)$；
9. 公式$C_1=p$，因此$t(C_1)=t(p)=1$；
10. 公式$C_2=s$，因此$t(C_2)=t(s)=1$；
11. 从而由定义，$t(C)=0$当且仅当$t(C_1)$和$t(C_2)$都等于0，而$t(C_1)=1$，因此$t(C)=1$；
12. 从而由定义，$t(A)=0$当且仅当$t(B)=1$且$t(C)=0$，而现在$t(C)=1$，因此$t(A)=1$。

具有递归程序编写经验的读者容易看出，很容易编写一个递归程序实现上述计算过程，特别是在得到公式A的语法树之后，上述计算过程实际上是对该语法树作**后序遍历**。

上面完全用文字叙述的计算过程看起来有些乱，可用表格的形式更简洁地计算公式A的真值：

p	q	s	$\neg\, p$	$(\neg\, p\wedge q)$	$(p\vee s)$	$(\neg\, p\wedge q)\to(p\vee s)$
1	0	1	0	0	1	1

上述表格的第一行列出了上述递归计算过程中需要计算的公式，其中最前面三列按字母顺序给出公式A中出现的命题变量，最后一列是公式A本身，中间都是在计算公式A的真值时需要递归计算的**子公式，按照上述计算过程中能够算出真值的先后顺序列出**。

上述表格的第二行在命题变量下面给出真值赋值函数t所指定的该命题变量的真值，各子公式下给出在递归计算过程中得到的该子公式的真值，实际上是该子公式在真值赋值函数t下的真值，最后一列是公式A的真值。

下面的例子进一步说明如何使用表格形式递归计算公式的真值：

例子 1.2.11 给定真值赋值函数$t:\mathbf{Var}\to\{0,1\}$，其中$t(p)=0,t(q)=1,t(r)=0,t(s)=1$，计算以下公式的真值：

$$A=(p\wedge q)\leftrightarrow(s\to(r\wedge q))$$

解答：根据公式A的语法结构，在递归计算该公式的真值时，需要计算真值的子公式按顺序分别为：$(p\wedge q),(r\wedge q),(s\to(r\wedge q))$，因此用表格形式计算公式$A$的真值如下：

p	q	r	s	$(p\wedge q)$	$(r\wedge q)$	$(s\to(r\wedge q))$	$(p\wedge q)\leftrightarrow(s\to(r\wedge q))$
0	1	0	1	0	0	0	1

使用上述表格形式计算公式A的真值的好处除了简洁之外，列出所有需要计算的子公式真值也使得我们每次只需关注一个联结词，例如，在计算子公式$r\wedge q$的真值时，我们只要盯住p和q的真值，再根据$\wedge$的定义确定该子公式的真值，而在计算子公式$(s\to(r\wedge q))$，我们只要盯住s和$(r\wedge q)$的真值，再根据$\leftrightarrow$的定义确定该子公式的真值，这样的计算不容易出错。

在命题逻辑中，往往不仅关心公式在某个真值赋值函数下的真值，而是研究公式在任意的真值赋值函数下的真值。由于真值的取值只有0和1两个值，因此当公式中出现n个命题变量时，（影响该公式真值的）可能的真值赋值函数个数只有2^n个，例如只有两个命题变量p和q时，真值赋值函数有$2^2=4$个：

$$t_0:\ t_0(p)=0\quad t_0(q)=0\qquad\qquad t_1:\ t_1(p)=0\quad t_1(q)=1$$
$$t_2:\ t_2(p)=1\quad t_2(q)=0\qquad\qquad t_3:\ t_3(p)=1\quad t_3(q)=1$$

对于这些真值赋值函数，如果假定命题变量符号总是按字母顺序，那么可使用二进制编码表示真值赋值函数的值，例如，t_0是00，t_1是01，t_2是10，t_3是11等。

可将上述计算公式在某个指定真值赋值函数下的真值的表格形式推广到计算公式在任意真值赋值函数下的真值：

例子 1.2.12 计算公式$A=(p\wedge q)\to(p\vee q)$在任意真值赋值函数下的真值。

解答：根据公式A的语法结构，需要计算真值的子公式按照顺序有$(p\wedge q)$和$(p\vee q)$，因此可使用如下表格形式计算A在任意真值赋值函数下的真值：

p	q	$(p\wedge q)$	$(p\vee q)$	$(p\wedge q)\to(p\vee q)$
0	0	0	0	1
0	1	0	1	1
1	0	0	1	1
1	1	1	1	1

上述表格的第二行开始按照二进制编码的顺序列出所有真值赋值函数为命题变量p和q指定的真值，这些行其他列的计算与前面相同。

这种计算公式在任意真值赋值函数下真值的表格称为该公式的**真值表**。在给出公式的真值表时，读者，特别是初学者应该：

1. 真值表的列应该**按字母顺序**列出公式包含的命题变量，然后按顺序**列出需要计算真值的子公式**，而不要无序地列出命题变量，或只列出命题变量和公式本身；

2. 真值表的行应该**按二进制编码顺序**列出所有可能的真值赋值函数为命题变量指定的真值；

例子 1.2.13 公式$A = (\neg p \wedge q) \to (p \vee s)$的真值表如下：

p	q	s	$\neg p$	$(\neg p \wedge q)$	$(p \vee s)$	$(\neg p \wedge q) \to (p \vee s)$
0	0	0	1	0	0	1
0	0	1	1	0	1	1
0	1	0	1	1	0	0
0	1	1	1	1	1	1
1	0	0	0	0	1	1
1	0	1	0	0	1	1
1	1	0	0	0	1	1
1	1	1	0	0	1	1

公式A含有三个变量，可能的真值赋值函数有$2^3 = 8$个，因此它的真值表除第一行之外还有八行，应该按照二进制编码的顺序列出，即$000, 001, 010, 011, 100, 101, 110, 111$。由于有比较多的真值赋值函数，因此一行一行计算有些慢，这时可在按**二进制编码的顺序**列出行，且按**字母顺序**列出命题变量的情况下，根据命题联结词特点**一列一列地**计算：

1. 对于子公式$\neg p$，这时应该看到各行对命题变量p赋值的顺序是00001111，因此根据$\neg$的特点，马上可写出该子公式下各行的真值应该是11110000；

2. 对于子公式$(\neg p \wedge q)$，注意到$\neg p$的真值是11110000，根据$\wedge$的特点，该子公式下后四行的真值肯定是0000，而前四行的真值与q相同，再注意到各行对命题变量q赋值的顺序是00110011，马上可得到该子公式下各行的真值是00110000；

3. 对于子公式$(p \vee s)$，注意到各行对p的赋值顺序是00001111，根据$\vee$的特点，该子公式下的后四行的真值必是1111，而前四行的真值与s相同，再注意到各行对s的赋值顺序是01010101，马上可得到该子公式下各行的真值是01011111；

4. 最后对于公式本身，根据$\to$的特点，只有当前件真而后件假时，该公式才为假，因此只要找到子公式$(\neg p \wedge q)$为1而$p \vee s$为0的行在最后一列写上0，其他行写上1即可。

总的来说，可使用以下技巧快速且不出错地构造公式的真值表：

1. 首先应保证真值表的**行按二进制编码顺序**，**列按命题变量的字母顺序**，并**列出需要计算的子公式**。按固定的顺序使得我们**可利用命题变量的赋值特点**，列出子公式使得我们**每次只关注一个命题联结词**；

2. 然后**一列一列**地计算子公式以及整个公式的真值，并在计算过程中利用命题联结词的特点：

(i). 如果当前列的子公式具有形式$(\neg B)$，则各行写上与B的真值（这时应该已经计算了B的真值）分别相反的值；

(ii). 如果当前列的子公式具有形式$(B \wedge C)$，则**找到B和C的真值都为1的行**，在当前列的这些行写上1，剩下的行直接写上0即可；

(iii). 如果当前列的子公式具有形式$(B \vee C)$，则**找到B和C的真值都为0的行**，在当前列的这些行写上0，剩下的行直接写上1即可；

(iv). 如果当前列的子公式具有形式$(B \to C)$，则**找到B的真值为1，而C的真值为0的行**，在当前列的这些行写上0，剩下的行直接写上1即可；

(v). 如果当前列的子公式具有形式$(B \leftrightarrow C)$，则找到B和C的真值相同的行，在当前列的这些行写上1，其他行写上0。

例子 1.2.14 构造公式$A = ((p \wedge q) \to r) \leftrightarrow (\neg r \to (p \vee q))$的真值表。

解答：根据公式A的语法结构，需要计算的子公式按顺序包括$(p \wedge q), ((p \wedge q) \to r), \neg r, (p \vee q), (\neg r \to (p \vee q))$，因此构造出的真值表如下：

p	q	r	$(p \wedge q)$	$((p \wedge q) \to r)$	$\neg r$	$(p \vee q)$	$(\neg r \to (p \vee q))$	A
0	0	0	0	1	1	0	0	0
0	0	1	0	1	0	0	1	1
0	1	0	0	1	1	1	1	1
0	1	1	0	1	0	1	1	1
1	0	0	0	1	1	1	1	1
1	0	1	0	1	0	1	1	1
1	1	0	1	0	1	1	1	0
1	1	1	1	1	0	1	1	1

永真式、矛盾式和可满足式

根据公式在任意真值赋值函数下的真值，可对公式进行分类：

定义 1.2.15

1. 如果公式A在任意真值赋值函数下的真值都为1，则称公式A为**永真式**(tautology)，也称为**重言式**；
2. 如果公式A在任意真值赋值函数下的真值都为0，则称公式A为**矛盾式**(contradiction)；
3. 如果公式A**不是矛盾式**，则称公式A为**可满足式**(satisfiable formula)。

可以通过公式的真值表判定公式的类型：从真值表看，(i) 永真式的真值表的最后一列（即该公式本身所在的一列）全为1；(ii) 矛盾式的真值表的最后一列全为0；(iii) 可满足式的真值表的最后一列**存在**1。注意，**永真式也是可满足式**，有时为了区分，可强调地说某公式是非永真式的可满足式。

例子 1.2.16

1. 例子1.2.12给出的公式$(p \wedge q) \to (p \vee q)$是永真式；
2. 例子1.2.13给出的公式$(\neg p \wedge q) \to (p \vee s)$和例子1.2.14给出的公式$((p \wedge q) \to r) \leftrightarrow (\neg r \to (p \vee q))$都是非永真式的可满足式；
3. 公式$\neg(p \to q) \wedge q$是矛盾式，它的真值表如下：

p	q	$(p \to q)$	$\neg(p \to q)$	$\neg(p \to q) \wedge q$
0	0	1	0	0
0	1	1	0	0
1	0	0	1	0
1	1	1	0	0

作 业

作业 1.1 判断下列语句是否是命题，并对命题确定其真值：

(1) 火星上有生命存在.

(2) 12是质数。

(3) 香山比华山高。

(4) $x+y=2$。

(5) 这盆茉莉花真香!

(6) 结果对吗?

(7) 这句话是错的。

(8) 假如明天是星期天，那么学校放假。

作业 1.2 令p表示今天很冷，q表示正在下雪，将下列命题符号化：

(1) 如果正在下雪，那么今天很冷。

(2) 今天很冷当且仅当正在下雪。

(3) 正在下雪的必要条件是今天很冷。

用自然语言叙述下列公式：

$$\neg(p\wedge q)\qquad \neg\, p\vee\neg\, q\qquad p\to q\qquad \neg\, p\vee q\qquad \neg\neg\, p\qquad \neg\, p\leftrightarrow q$$

作业 1.3 将下列命题符号化：

(1) 他个子高且很胖。

(2) 他个子高但不很胖。

(3) 并非他个子高或很胖。

(4) 他个子不高也不胖。

(5) 他个子高或者他个子矮而很胖。

(6) 他个子矮或他不很胖都是不对的。

(7) 如果水是清的，那么或者张三能见到池底或者他是个近视眼。

(8) 如果嫦娥是虚构的，而如果圣诞老人也是虚构的，那么许多孩子受骗了。

作业 1.4 针对严格符合定义1.2.2的公式，使用归纳法证明公式中左圆括号的数目与公式中联结词的数目相同，同样右圆括号的数目也与公式中联结词的数目相同。

作业 1.5 给定真值赋值函数$t:\mathbf{Var}\to\{0,1\}$，其中$t(p)=t(q)=0,t(r)=t(s)=1$，确定下列公式在真值赋值函数t下的真值：

(1). $p\vee(q\wedge r)$　　(2). $(p\leftrightarrow r)\wedge(\neg\, q\vee s)$

(3). $(\neg\, p\wedge\neg\, q\wedge r)\leftrightarrow(p\wedge q\wedge\neg\, r)$　　(4). $(\neg\, r\wedge s)\to(p\wedge\neg\, q)$

作业 1.6 构造下列公式的真值表，并判断该公式的类型（永真式、非永真式的可满足式还是矛盾式），注意在列真值表时，行要按二进制编码顺序，前几列要按命题变量的字母顺序，并要给出需要计算的子公式：

(1). $(p\to\neg\, p)\wedge(p\wedge q)$　　(2). $(p\to q)\to(\neg\, q\to\neg\, p)$

(3). $(p\wedge r)\leftrightarrow(\neg\, p\wedge\neg\, q)$　　(4). $((p\to q)\wedge(q\to r))\to(p\to r)$

第二章　命题逻辑的等值演算

上一章给出了命题逻辑公式及其真值的严格定义，这一章在此基础上，讨论如何判断两个公式（在任意的真值赋值函数下都）具有相同的真值，为此引入命题逻辑公式的等值演算，并讨论命题逻辑联结词之间的真值关系，联结词的完备集，以及具有规范化形式的公式，即范式，和如何得到与某公式具有相同真值的范式。

2.1　等值演算基础

如果两个公式在任意的真值赋值函数下都具有相同的真值就称为两个等值的公式。两个公式等值，意味着，从真值语义的角度看，它们是相同的，或说等价的。等值演算是判断两个公式是否等值的一种方法，它从一些基本等值式出发，利用一些规则对公式进行变形，从而判断两个公式是否等值。这一节先讨论公式等值的含义，然后讨论一些基本的等值式，并给出等值演算的基本规则。

2.1.1　公式等值的含义

定义 2.1.1 *如果公式A和B在任意的真值赋值函数$t:\mathbf{Var}\to\{0,1\}$都有$t(A)=t(B)$，则称A与B等值*(A is logically equivalent to B)，*记为*$A\Leftrightarrow B$。

A与B等值，意味着，从讨论公式真值的角度看，这两个公式是等价的。根据定义，显然最简单的方法可列出两个的真值表，通过真值表（的最后一列）比较这两个公式是否等值。不过，根据等价联结词的定义，容易证明：

定理 2.1.2 *公式A和B等值当且仅当公式$A\leftrightarrow B$是永真式。*

证明　若A与B等值，则对任意的真值赋值函数t，有$t(A)=t(B)$，根据等价联结词的定义有$t(A\leftrightarrow B)=1$，从而根据永真式的定义，$A\leftrightarrow B$是永真式。而若$A\leftrightarrow B$是永真式，则对任意的真值赋值函数t有$t(A\leftrightarrow B)=1$，而根据等价联结词的定义，这意味着$t(A)=t(B)$，因此A与B等值。□

根据上述定义，我们就不需要使用两个真值表来判断公式A与B是否等值，而只要构造公式$A\leftrightarrow B$的真值表，就可以判断是否有$A\Leftrightarrow B$。顺便提一下，虽然$\Leftrightarrow$与命题联结词$\leftrightarrow$有密切联系，但$\Leftrightarrow$不是命题逻辑公式的联结词，本身不能出现在命题逻辑公式中，而且$A\Leftrightarrow B$与$A\leftrightarrow B$的含义也完全不同，前者意味着*后者是永真式*。

例子 2.1.3

1. 公式$p\to q$与$\neg p\vee q$等值，即$(p\to q)\Leftrightarrow(\neg p\vee q)$。因为$(p\to q)\leftrightarrow(\neg p\vee q)$的真值表如下：

p	q	$(p \to q)$	$\neg p$	$(\neg p \vee q)$	$(p \to q) \leftrightarrow (\neg p \vee q)$
0	0	1	1	1	1
0	1	1	1	1	1
1	0	0	0	0	1
1	1	1	0	1	1

2. 公式$A = p \leftrightarrow q$与公式$B = (p \to q) \wedge (q \to p)$等值，即$(p \leftrightarrow q) \Leftrightarrow ((p \to q) \wedge (q \to p))$。因为$(p \leftrightarrow q) \leftrightarrow ((p \to q) \wedge (q \to p))$的真值表如下：

p	q	$A = (p \leftrightarrow q)$	$(p \to q)$	$(q \to p)$	B	$A \leftrightarrow B$
0	0	1	1	1	1	1
0	1	0	1	0	0	1
1	0	0	0	1	0	1
1	1	1	1	1	1	1

2.1.2 等值演算

不难看到，当命题逻辑公式所含的命题变量很多时（例如超过4个），公式的真值表会比较复杂，使用真值表判断公式是否等值有些困难。等值演算提供另外一种判断公式是否等值的方法，**它以一些基本等值式为基础，利用一些规则对两个公式进行变形，并在变形的过程中保持变形前后公式的等值性，最后如果两个公式能够变形为形式上相同的公式，则它们等值**。

基本等值式

为了判断两个公式是否等值，我们先使用真值表的方法证明一些简单但常用的等值式，将这些等值式称为**基本等值式**。下表列出了这些基本等值式：

名称	基本等值式模式
双重否定律	$A \Leftrightarrow \neg\neg A$
幂等律	$A \Leftrightarrow A \wedge A$　　$A \Leftrightarrow A \vee A$
交换律	$A \wedge B \Leftrightarrow B \wedge A$　　$A \vee B \Leftrightarrow B \vee A$
结合律	$(A \wedge B) \wedge C \Leftrightarrow A \wedge (B \wedge C)$　　$(A \vee B) \vee C \Leftrightarrow A \vee (B \vee C)$
分配律	$A \vee (B \wedge C) \Leftrightarrow (A \vee B) \wedge (A \vee C)$　　$A \wedge (B \vee C) \Leftrightarrow (A \wedge B) \vee (A \wedge C)$
吸收律	$A \wedge (A \vee B) \Leftrightarrow A$　　$A \vee (A \wedge B) \Leftrightarrow A$
德摩根律	$\neg(A \wedge B) \Leftrightarrow (\neg A \vee \neg B)$　　$\neg(A \vee B) \Leftrightarrow (\neg A \wedge \neg B)$
零律	$A \wedge 0 \Leftrightarrow 0$　　$A \vee 1 \Leftrightarrow 1$
同一律	$A \wedge 1 \Leftrightarrow A$　　$A \vee 0 \Leftrightarrow A$
矛盾律	$A \wedge \neg A \Leftrightarrow 0$
排中律	$A \vee \neg A \Leftrightarrow 1$
蕴涵等值式	$A \to B \Leftrightarrow \neg A \vee B$
等价等值式	$A \leftrightarrow B \Leftrightarrow (A \to B) \wedge (B \to A)$

对于上表需要说明的是：

1. 上面给出的都是**等值式模式**，所谓模式是指本身并不是等值式，只有将其中的A, B, C等变量代入任意的命题逻辑公式之后才真正成为命题逻辑的等值式。上表中的A, B, C都可代入任意的命题逻辑公式，而0代表任意的矛盾式，1代表任意的永真式。**一个公式与0等值就是表示这个公式是矛盾式，一个公式与1等值就是表示这个公式是永真式**。

2. 基本等值式实际上给出了命题联结词的一些基本性质：(i) 双重否定律给出了否定联结词的基本性质；(ii) 幂等律、交换律、结合律、零律、同一律给出了合取联结词和析取联结词的基本性质；(iii) 分配律、吸收律给出了合取联结词和析取联结词之间的关系；(iv) 德摩根律给出了否定联结词与合取、析取联结词之间的关系；(v) 排中律给出了最基本的永真式，矛盾律给出了最基本的矛盾式；(vi) 蕴涵等值式和等价等值式给出了蕴涵联结词和等价联结词与否定、合取以及析取联结词之间的关系。

3. 上面的幂等律、交换律、结合律、分配律、吸收律、德摩根律、零律、同一律都有两个等值式，但实际上只要记住其中一个即可。这些等值式都只涉及到否定、合取、析取联结词以及0和1，只要将左边等值式中的$\wedge$换成$\vee$，$\vee$换成$\wedge$，0换成1，1换成0就可得到右边的等值式。通常将右边的等值式称为左边等值式的**对偶式**。

等值演算规则

等值演算就是从已有的等值式推演出另外一些等值式的过程，这种过程是以上面的基本等值式模式为基础，利用一些规则对公式进行变形，因此等值演算最基本的一条规则是：

代入实例规则：基本等值式模式的任意代入实例都是等值式。所谓基本等值式模式的代入实例，就是将基本等值式模式中的A, B, C等符号用任意的命题逻辑公式代入后得到等值式。

其次，公式等值本身是一个等价关系，因此有以下规则：

1. **自反规则**：对任意的命题逻辑公式A，都有$A \Leftrightarrow A$；
2. **对称规则**：对任意的命题逻辑公式A, B，若$A \Leftrightarrow B$，则$B \Leftrightarrow A$；
3. **传递规则**：对任意的命题逻辑公式A, B, C，若$A \Leftrightarrow B$且$B \Leftrightarrow C$，则$A \Leftrightarrow B$。

上述规则都是等值演算的基本规则，但等值演算过程中最重要的规则是置换规则：

等值置换规则：设Φ是任意命题逻辑公式，A是Φ的子公式（参见定义1.2.5），B是与A等值的公式，则将Φ中的子公式A**置换成**B得到的公式Φ'与Φ等值。

等值置换规则中的将Φ中的子公式A**置换成**与A等值的公式B是将Φ中的子公式A整个换成公式B，例如$(p \to q)$是公式$(p \to q) \to r$的子公式，而根据蕴涵等值式，$(p \to q)$与$(\neg p \vee q)$等值，用$(\neg p \vee q)$置换$(p \to q) \to r$中的子公式$(p \to q)$得到$(\neg p \vee q) \to r$。等值置换规则断定$(p \to q) \to r \Leftrightarrow (\neg p \vee q) \to r$。

备注 2.1.4 等值置换规则的正确性建立在以下事实的基础上：

1. 若$A \Leftrightarrow A'$，则$(\neg A) \Leftrightarrow (\neg A')$；
2. 若$A \Leftrightarrow A'$且$B \Leftrightarrow B'$，则：$(A \wedge B) \Leftrightarrow (A' \wedge B')$、$(A \vee B) \Leftrightarrow (A' \vee B')$、$(A \to B) \Leftrightarrow (A' \to B')$以及$(A \leftrightarrow B) \Leftrightarrow (A' \leftrightarrow B')$

上述事实很容易利用公式等值的定义证明，例如若$A \Leftrightarrow A'$且$B \Leftrightarrow B'$，则对任意的真值赋值函数t都有$t(A) = t(A')$和$t(B) = t(B')$，从而不难由$\wedge$的定义（即$t(A \wedge B) = 1$当且仅当$t(A) = t(B) = 1$）得到$t(A \wedge B) = t(A' \wedge B')$，这表明$(A \wedge B) \Leftrightarrow (A' \wedge B')$。

根据上述事实，以及子公式的归纳定义，可对置换规则中的公式Φ实施归纳法证明等值置换规则的正确性。有兴趣的读者可进一步查找相关资料了解其详细证明，此处不再做更深入的讨论。

在实际应用中，利用等值置换规则对公式进行变形是等值演算中使用得最多的情况，这时通常上述等值置换规则中的Φ是需要进行变形的公式，A是Φ的子公式，通过使用合适的基本等值式模式的代入实例找到与A等值的公式B，用B置换Φ的A达到变形的目的。

当我们要证明公式Φ与Ψ等值，可通过使用等值置换规则连续对Φ变形直到得到Ψ，或者连续对Ψ变形直到得到Φ，或者对Φ和Ψ进行变形直到都得到另一中间公式Θ，这里都隐含地使用了公式等值本身是等价关系这个特性。

例子 2.1.5 利用等值演算证明：$(p \to (q \to r)) \Leftrightarrow ((p \wedge q) \to r)$。

分析：对于含有蕴涵联结词的公式，通常利用蕴涵等值式将其变形为只含有否定、析取和合取联结词的公式，这是因为基本等值式中关于这三个联结词的等值式最多，容易再进一步利用其他基本等值式。后面将看到规范化形式的公式，即范式，也是只含有否定、析取和合取联结词的公式。

对于公式$(p \to (q \to r))$，$(q \to r)$是它的子公式，而$(q \to r) \Leftrightarrow (\neg q \vee r)$是蕴涵等值式模式的代入实例，公式$(p \to (q \to r))$中的子公式$(q \to r)$置换成$(\neg q \vee r)$后得到公式$(p \to (\neg q \vee r))$，根据等值置换规则$(p \to (q \to r))$等值于$(p \to (\neg q \vee r))$，这样我们就将公式$(p \to (q \to r))$变形为$((p \to (\neg q \vee r))$。

下面证明中的每一步对是类似这样的变形，其中等值置换公式是必用的，每一步变形的不同之处在于使用的基本等值式，因此下面证明的注释中只列出所使用的基本等值式。由于对哪个子公式进行置换通常也是显而易见，因此也通常不指明置换所针对的子公式。

证明

$$
\begin{aligned}
(p \to (q \to r)) &\Leftrightarrow (p \to (\neg q \vee r)) && // \text{蕴涵等值式}\\
&\Leftrightarrow (\neg p \vee (\neg q \vee r)) && // \text{蕴涵等值式}\\
&\Leftrightarrow ((\neg p \vee \neg q) \vee r)) && // \vee\text{的结合律}\\
&\Leftrightarrow (\neg(p \wedge q) \vee r) && // \text{德摩根律}\\
&\Leftrightarrow ((p \wedge q) \to r) && // \text{蕴涵等值式}
\end{aligned}
$$

□

例子 2.1.6 证明下述等值式：

1. $((p \wedge q) \to r) \Leftrightarrow ((p \to r) \vee (q \to r))$
2. $((p \vee q) \to r) \Leftrightarrow ((p \to r) \wedge (q \to r))$
3. $(p \to (q \wedge r)) \Leftrightarrow ((p \to q) \wedge (p \to r))$
4. $(p \to (q \vee r)) \Leftrightarrow ((p \to q) \vee (p \to r))$

证明 下面只证明1和3，2和4的证明留给读者作为练习。

1.

$$
\begin{aligned}
((p \wedge q) \rightarrow r) &\Leftrightarrow (\neg (p \wedge q) \vee r) && // \text{蕴涵等值式}\\
&\Leftrightarrow ((\neg p \vee \neg q) \vee r) && // \text{德摩根律}\\
&\Leftrightarrow (\neg p \vee r \vee \neg q) && // \vee\text{的结合律、交换律}\\
&\Leftrightarrow (\neg p \vee r \vee \neg q \vee r) && // \vee\text{的幂等律、交换律}\\
&\Leftrightarrow ((\neg p \vee r) \vee (\neg q \vee r)) && // \vee\text{的结合律}\\
&\Leftrightarrow ((p \rightarrow r) \vee (\neg q \vee r)) && // \text{蕴涵等值式}\\
&\Leftrightarrow ((p \rightarrow r) \vee (q \rightarrow r)) && // \text{蕴涵等值式}
\end{aligned}
$$

3.

$$
\begin{aligned}
(p \rightarrow (q \wedge r)) &\Leftrightarrow (\neg p \vee (q \wedge r)) && // \text{蕴涵等值式}\\
&\Leftrightarrow ((\neg p \vee q) \wedge (\neg p \vee r)) && // \vee\text{对}\wedge\text{分配律}\\
&\Leftrightarrow ((p \rightarrow q) \wedge (\neg p \vee r)) && // \text{蕴涵等值式}\\
&\Leftrightarrow ((p \rightarrow q) \wedge (p \rightarrow r)) && // \text{蕴涵等值式}
\end{aligned}
$$

□

小结

等值演算用于判断两个公式等值，或说证明一些等值式，它以一些基本等值式（模式）为基础，利用代入实例规则、公式等值是等价关系，以及等值演算规则对公式进行变形，从而判断两个公式是否等值。对于等值演算，我们要注意：

1. 只能针对公式的子公式使用置换规则，例如不能将公式$p \rightarrow q \rightarrow r$利用$p \rightarrow q$置换成$(\neg p \vee q) \rightarrow r$，公式$p \rightarrow q$不是$p \rightarrow q \rightarrow r$的子公式。公式$p \rightarrow q \rightarrow r$的严格形式是$(p \rightarrow (q \rightarrow r))$。这也提醒我们在书写公式的时候注意使用圆括号。

2. 对于初学者，等值演算的每一步最好只针对一个子公式使用等值置换规则。在逐渐熟练之后也可将一些简单的演算步骤进行合并，特别是幂等律、交换律、结合律等可与其他基本等值式合并使用，因为这些等值式比较容易理解和掌握，而且又很常用，每次使用都分离出来会使得演算过程冗长。

3. **等值演算的每一步都应该写上所使用的基本等值式作为注释**，这对于初学者尽快掌握等值演算及基本等值式很有帮助。在使用等值置换规则将公式的子公式替换成等值的公式时，原则上都应该利用前面所给出的基本等值式，通常不需要利用已经证明过的比较复杂的等值式。

可以说，**等值演算的核心是等值置换规则的使用**，读者应该深入理解等值置换规则的含义。等值演算除了证明等值式外，也是求与一个公式等值的范式的基本方法。在数字逻辑与电路设计课程中，命题逻辑的等值演算也用于对一些公式进行化简（变形），求与原公式等值的，但更适合进行电路设计的逻辑公式。

在某些教材，等值演算还用于一些实际问题的解决，但根据作者的观察，等值演算用于这些解决一些实际问题时，往往过于复杂而不实用，不如直观的逻辑推理简明，因此这里就不打算举例说明等值演算在这方面的应用，有兴趣的读者可参考有关教材，例如[3]。

2.2 联结词的完备集

联结词的完备集是讨论命题逻辑中至多需要哪些命题联结词，这当然是在某个背景下而讨论的，是说**在要表达命题逻辑中的真值函数时至多需要哪些命题逻辑联结词**。注意，这里说的真值函数与前面说的真值赋值函数完全不同：

定义 2.2.1 **真值函数**是$\{0,1\}$集上的$n(n\geq 1)$元函数，形式为$f:\{0,1\}^n\to\{0,1\}$，这里$\{0,1\}^n$是n个$\{0,1\}$集的笛卡尔积。

也就是说，n元真值函数有n个自变量，每个自变量的可能取值都是0或1，而且真值函数的值也是0或1。每个命题逻辑公式都可看作是一个真值函数的表达式，例如，公式$p\to(q\wedge r)$可看作一个三元真值函数$f(p,q,r)=p\to(q\wedge r)$，正如在数学分析中所讨论的实数上函数$f(x,y)=x^2+y^2$，x^2+y^2就是二元函数$f(x,y)$的表达式。

联结词的完备集就是**研究至多需要哪些命题联结词（即逻辑运算符）可表达所有可能的真值函数**，这里的表达当然是在等值的意义下而言，即认为相互等值的公式表达的是同样的真值函数，例如上面的三元真值函数$f(p,q,r)=p\to(q\wedge r)$也可用$f(p,q,r)=\neg p\vee(q\wedge r)$表示。

研究联结词的完备集的意义在于：(i) 在需要的时候可使用比较少的命题联结词来构造公式，只要这些命题联结词构成完备集就可在等值的意义上表达所有可能的真值函数，从而既满足研究的需要，又能简化公式的结构；(ii) 通过研究联结词的完备集可进一步理解各命题联结词之间的关系。

命题逻辑的联结词存在完备集的前提是，对于任意的自然数$n\geq 1$，所有可能的真值函数只有2^{2^n}个。例如，对于$n=1$，可能的一元真值函数有$2^2=4$个，如下表所示：

p	$f_0^{(1)}$	$f_1^{(1)}$	$f_2^{(1)}$	$f_3^{(1)}$
0	0	0	1	1
1	0	1	0	1

一元真值函数有一个自变量，使用p表示，p的取值有0和1两种，而不同的一元真值函数在p的这两种取值下，函数值可分别为0和1，组合起来有2^2个不同的一元真值函数，上表从左到右，按列读的话，刚好也是按二进制编码顺序，即$00,01,10,11$。

由此，不难推测当$n=2$时，有两个自变量，取值组合有四种：$00,01,10,11$，而不同的二元真值函数的值的取值组合有$0000,0001,0010,\cdots$等16种，从而有16个不同的二元真值函数，如下表所示：

p	q	$f_0^{(2)}$	$f_1^{(2)}$	$f_2^{(2)}$	$f_3^{(2}$	$f_4^{(2)}$	$f_5^{(2)}$	$f_6^{(2)}$	$f_7^{(2)}$
0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1

p	q	$f_8^{(2)}$	$f_9^{(2)}$	$f_{10}^{(2)}$	$f_{11}^{(2}$	$f_{12}^{(2)}$	$f_{13}^{(2)}$	$f_{14}^{(2)}$	$f_{15}^{(2)}$
0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1

下面定义联结词的完备集：

定义 2.2.2 设Ω是一些命题联结词构成的集合。称Ω是**联结词的完备集**，如果对任意的n元($n \geq 1$)真值函数f都可使用只含Ω中联结词的公式A表达，即A至多含n个命题变元$p_1, p_2, \cdots, p_n$，且A在任意真值赋值函数$t : \mathbf{Var} \to \{0,1\}$下的真值$t(A)$等于$f(t(p_1), t(p_2), \cdots, t(p_n))$。

定理 2.2.3 联结词集$\{\neg, \wedge, \vee, \to\}$是联结词的完备集。

证明 该定理的详细证明不要求读者掌握。简单地来说，要证明该集合是联结词的完备集，即要证明对任意的n元真值函数，都可用只含$\neg, \wedge, \vee, \to$这几个联结词的公式表达。对n实施数学归纳法：

1. **归纳基**：当$n = 1$时，一元真值函数如上表所示有四个，它们可用只含$\neg, \wedge$和$\vee$的公式表达：

$$f_0^{(1)} = p \wedge \neg p \qquad f_1^{(1)} = p \qquad f_2^{(1)} = \neg p \qquad f_3^{(1)} = p \vee \neg p$$

2. **归纳步**：设$n = k$时，命题成立，即对任意的k元真值函数都可用只含$\neg, \wedge, \vee, \to$的公式表达，考虑任意的$k+1$元真值函数$f(x_1, x_2, \cdots, x_k, x_{k+1})$，定义如下两个$k$元真值函数：

$$f_0(x_1, x_2, \cdots, x_k) = f(x_1, x_2, \cdots, x_k, 0) \qquad f_1(x_1, x_2, \cdots, x_k) = f(x_1, x_2, \cdots, x_k, 1)$$

按照归纳假设，这两个公式可用只含上述四个联结词的公式表达，设f_0用A_0表达，f_1用A_1表达，且A_0和A_1都只含k个命题变量$p_1, \cdots, p_k$，设p_{k+1}是一个新的命题变量符号，不难证明f可用$(\neg\, p_{k+1} \to A_0) \wedge (p_{k+1} \to A_1)$表达，从而命题对$n = k+1$时成立。

□

例子 2.2.4 对于一些二元真值函数，我们给出它在完备集$\{\neg, \wedge, \vee, \to\}$中的公式表示：

$$f_0^{(2)} = p \wedge \neg p \qquad f_1^{(2)} = p \wedge q \qquad f_2^{(2)} = \neg(p \to q)$$
$$f_4^{(2)} = \neg(q \to p) \qquad f_6^{(2)} = (p \wedge \neg q) \vee (q \wedge \neg p) \qquad f_{14}^{(2)} = \neg(p \wedge q)$$

推论 2.2.5 以下联结词集都是完备集：

1. $S_1 = \{\neg, \wedge\}$
2. $S_2 = \{\neg, \vee\}$
3. $S_3 = \{\neg, \to\}$
4. $S_4 = \{\neg, \wedge, \vee\}$
5. $S_5 = \{\neg, \wedge, \vee, \to, \leftrightarrow\}$

证明 S_1是完备集，因为含有联结词$\vee, \to$的公式都可用等值的且只含$\neg$和$\wedge$的公式表示：

$$(A \vee B) \Leftrightarrow \neg(\neg A \wedge \neg B) \qquad (A \to B) \Leftrightarrow (\neg\, A \vee B) \Leftrightarrow \neg(A \wedge \neg B)$$

S_2是完备集，因为含有联结词$\wedge, \to$的公式都可用等值的且只含$\neg$和$\vee$的公式表示：

$$(A \wedge B) \Leftrightarrow \neg(\neg A \vee \neg B) \qquad (A \to B) \Leftrightarrow (\neg\, A \vee B)$$

S_3是完备集，因为含有联结词$\wedge, \vee$的公式都可用等值的且只含$\neg$和$\to$的公式表示：

$$(A \wedge B) \Leftrightarrow \neg(\neg A \vee \neg B) \Leftrightarrow \neg(A \to \neg B) \qquad (A \vee B) \Leftrightarrow (\neg A \to B)$$

显然S_4和S_5是完备集，因为$\{\neg, \wedge\}$已经是完备集。 □

上述推论证明实际上给出了**各联结词之间的关系**，请读者务必记熟，因为各种不同的联结词完备集各有用处：(i) 命题逻辑公式的范式使用完备集$\{\neg,\wedge,\vee\}$；(ii) 命题逻辑的有些演算系统使用完备集$\{\neg,\rightarrow\}$。简单来说，完备集$\{\neg,\wedge,\vee\}$十分常用，在表示公式方面十分方便，在数字逻辑电路设计中通常也是使用这个完备集；而完备集$\{\neg,\rightarrow\}$在推理方面十分有用，通常用于命题逻辑的形式演算系统中。

并不是联结词的任意集合都是完备集，例如：

定理 2.2.6 联结词集$\{\wedge,\vee,\rightarrow,\leftrightarrow\}$不是完备集。

证明 总取0值的真值函数$f_0^{(n)}$不能由只含$\wedge,\vee,\rightarrow,\leftrightarrow$的命题公式表达，这样的命题公式当所有命题变元都赋值为1时真值为1，不可能为0，因为$1\wedge 1=1, 1\vee 1=1, 1\rightarrow 1=1$以及$1\leftrightarrow 1=1$。 □

2.3 命题逻辑公式的范式

我们知道，命题逻辑公式中互相等值的公式很多，具有各种各样的形式，而范式(normal form)就是某种意义下的规范化形式的公式，更明确地，是容易从形式上判断类型（即是永真式、矛盾式还是可满足式）的公式。判断公式的类型很重要，公式等值可利用永真式证明，而命题逻辑的推理规则也建立在永真式的基础上。

为介绍命题逻辑公式的范式，我们逐步引入文字、简单析取式、简单合取式、析取范式、合取范式、极小项、极大项、主析取范式、主合取范式等概念。

2.3.1 析取范式和合取范式

定义 2.3.1 命题变量或者命题变量否定称为**文字**(literal)。

例子 2.3.2 $p,\neg p,q,\neg q,r,\neg r$等都是文字，而$\neg(p\vee q), p\wedge q$等都不是文字。

备注 2.3.3 从某种意义上说，文字是构成命题逻辑公式最简单的东西。命题变量与命题变量的否定都称为文字表明当否定联结词作用在命题变量时，实际上也是认为十分简单的公式，这也是我们在命题符号化时对一些带否定意味的命题不强求使用否定联结词的原因之一。

定义 2.3.4 仅由有限个文字的析取构成的公式称为**简单析取式**；仅由有限个文字的合取构成的公式称为**简单合取式**。

例子 2.3.5 下面公式都是简单析取式：

$$p \qquad p\vee q \qquad \neg p\vee q \qquad \neg p\vee\neg q\vee r \qquad p\vee\neg q\vee r$$

下面公式都是简单合取式：

$$p \qquad p\wedge q \qquad \neg p\wedge\neg q \qquad \neg p\wedge r\wedge\neg r \qquad p\wedge q\wedge\neg r$$

下面公式都不是简单析取式：

$$\neg(p\wedge q) \qquad p\wedge(q\vee r) \qquad p\wedge\neg(q\wedge r) \qquad p\rightarrow q$$

下面公式都不是简单合取式：

$$\neg(p \vee q) \qquad p \vee (q \wedge r) \qquad p \vee \neg(q \vee r) \qquad p \to q$$

简单析取式和简单合取式都可从形式上简单地判定类型：

定理 2.3.6 一个简单析取式是永真式当且仅当它同时含一个命题符号及其否定；一个简单合取式是矛盾式当且仅当它同时含一个命题符号及其否定。

证明 由排中律，$A \vee \neg A$是永真式，而由矛盾律，$A \wedge \neg A$是矛盾式。排中律和矛盾式分别给出了永真式和矛盾式的最基本形式。 □

例子 2.3.7 简单析取式$p \vee q \vee \neg p$是永真式，而$p \vee q \vee r$不是永真式；简单合取式$p \wedge q \wedge \neg p$是矛盾式，而$p \wedge q \wedge r$不是矛盾式。

定义 2.3.8 有限个简单合取式的析取构成的公式称为**析取范式**(disjunctive normal form)；有限个简单析取式的合取构成的公式称为**合取范式**(conjunctive normal form)。

例子 2.3.9 下面公式是析取范式：

$$p \qquad p \vee r \qquad p \wedge q \qquad (p \wedge q) \vee r \qquad (p \wedge q) \vee (q \wedge r)$$

下面公式是合取范式：

$$p \qquad p \vee r \qquad p \wedge q \qquad (p \vee q) \wedge r \qquad (p \vee q) \wedge (q \vee r)$$

下面公式都不是析取范式：

$$(p \vee q) \wedge r \qquad (p \vee q) \wedge (q \vee r) \qquad (p \vee q) \to r \qquad (p \to q) \vee r$$

下面公式都不是合取范式：

$$(p \wedge q) \vee r \qquad (p \wedge q) \vee (q \wedge r) \qquad (p \vee q) \to r \qquad (p \to q) \vee r$$

上面提到，容易判断简单合取式是否是矛盾式，容易判断简单析取式是否是永真式。我们也很容易判断析取范式和合取范式的类型：

定理 2.3.10 析取范式是矛盾式当且仅当它的每个简单合取式都是矛盾式；合取范式是永真式当且仅当它的每个简单析取式都是永真式。

例子 2.3.11 析取范式$(p \wedge q \wedge \neg p) \vee (q \wedge \neg q \wedge r)$是矛盾式，而$(p \wedge r) \vee (q \wedge r)$不是矛盾式；合取范式$(p \vee \neg p \vee r) \wedge (q \vee r \vee \neg q)$是永真式，而$(p \vee r) \wedge (q \vee r)$不是永真式。

定理 2.3.12 每个命题逻辑公式都存在与之等值的析取范式，也存在与之等值的合取范式。

这里不证明这个定理，实际上，后面将给出用真值表求与一个公式等值的主析取范式和主合取范式的方法，而主析取范式和主合取范式也分别是析取范式和合取范式，这也就间接证明了上述定理。这里先介绍使用等值演算的方法求与一个等值的析取范式和合取范式的启发式步骤：

1. 使用**蕴涵等值式**和**等价等值式**：

$$(A \to B) \Leftrightarrow (\neg A \vee B) \qquad (A \leftrightarrow B) \Leftrightarrow (A \to B) \wedge (B \to A) \Leftrightarrow (\neg A \vee B) \wedge (\neg B \vee A)$$

消除公式中的蕴涵联结词$\to$和等价联结词$\leftrightarrow$，使得公式只含$\neg, \wedge$和$\vee$；

2. 利用**双重否定律**和**德摩尔根律**：

$$\neg\neg A \Leftrightarrow A \qquad \neg(A \wedge B) \Leftrightarrow (\neg A \vee \neg B) \qquad \neg(A \vee B) \Leftrightarrow (\neg A \wedge \neg B)$$

将否定连结词移到命题变量前，并消除多余的否定联结词；

3. 利用**分配律**，其中求与公式等值的析取范式使用$\wedge$对$\vee$的分配律，而求与公式等值的合取范式则使用$\vee$对$\wedge$的分配律：

$$A \wedge (B \vee C) \Leftrightarrow (A \wedge B) \vee (A \wedge C) \qquad \text{// 求与公式等值的析取范式}$$
$$A \vee (B \wedge C) \Leftrightarrow (A \vee B) \wedge (A \vee C) \qquad \text{// 求与公式等值的合取范式}$$

4. 通过以上几步通常已经得到析取范式或合取范式，但对析取范式还可进一步使用矛盾律**消除是矛盾式的简单合取式**，而对合取范式还可进一步使用排中律**消除是永真式的简单析取式**，并根据幂等律，消除重复的简单合取式，或重复的简单析取式，从而简化公式；

5. 最后，为使公式更容易理解，以及为下面求主析取范式和主合取范式准备，根据交换律，将析取范式中的每个简单合取式，或将合取范式的每个简单析取式中的**命题变量都按字母顺序排列**。

例子 2.3.13 使用等值演算方法求与公式$(p \to q) \leftrightarrow r$等值的析取范式和合取范式。

解答：先求与该公式等值的析取范式：

$$\begin{aligned}
& (p \to q) \leftrightarrow r && \text{// 使用蕴涵等值式消除}\to \\
\Leftrightarrow\ & (\neg p \vee q) \leftrightarrow r && \text{// 使用等价等值式消除}\leftrightarrow \\
\Leftrightarrow\ & (\neg(\neg p \vee q) \vee r) \wedge (\neg r \vee (\neg p \vee q)) && \text{// 德摩根律将否定移到命题变量前} \\
\Leftrightarrow\ & ((p \wedge \neg q) \vee r) \wedge (\neg r \vee (\neg p \vee q)) && \text{// 使用}\wedge\text{对}\vee\text{的分配律} \\
\Leftrightarrow\ & ((p \wedge \neg q) \wedge (\neg r \vee \neg p \vee q)) \vee \\
& (r \wedge (\neg r \vee \neg p \vee q)) && \text{// 再使用}\wedge\text{对}\vee\text{的分配律} \\
\Leftrightarrow\ & (p \wedge \neg q \wedge \neg r) \vee (p \wedge \neg q \wedge \neg p) \vee (p \wedge \neg q \wedge q) \vee \\
& (r \wedge \neg r) \vee (r \wedge \neg p) \vee (r \wedge q) && \text{// 消除是矛盾式的简单合取式} \\
\Leftrightarrow\ & (p \wedge \neg q \wedge \neg r) \vee (\neg p \wedge r) \vee (q \wedge r) && \text{// 将命题变量按字母顺序排列}
\end{aligned}$$

再求与该公式等值的合取范式：

$$\begin{aligned}
& (p \to q) \leftrightarrow r && \text{// 前几步与上面相同} \\
\Leftrightarrow\ & (\neg p \vee q) \leftrightarrow r \\
\Leftrightarrow\ & (\neg(\neg p \vee q) \vee r) \wedge (\neg r \vee (\neg p \vee q)) \\
\Leftrightarrow\ & ((p \wedge \neg q) \vee r) \wedge (\neg r \vee (\neg p \vee q)) && \text{// 使用}\vee\text{对}\wedge\text{的分配律} \\
\Leftrightarrow\ & (p \vee r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee \neg r)
\end{aligned}$$

显然，与一个公式等值的合取范式和析取范式都不惟一，例如上面求析取范式的过程中，在消除是矛盾式的简单合取式之前已经得到了析取范式，后面都是对析取范式进行等值演算，同样在求合取范式过程中也会出现这种情况。为了得到具有惟一形式的与公式等值的规范化形式公式，进一步引入主析取范式和主合取范式。

2.3.2 主析取范式和主合取范式

首先引入极小项和极大项的概念：

定义 2.3.14 在含有n命题变量的**简单合取式**中，若每个命题变量和它的否定不同时出现，而二者之一必出现且仅出现一次，且按照预定的顺序，第i个命题变量或它的否定出现在从左算起的第i位上，称这样的简单合取式为**极小项**。

类似地，在含有n命题变量的**简单析取式**中，若每个命题变量和它的否定不同时出现，而二者之一必出现且仅出现一次，且按照预定的顺序，第i个命题变量或它的否定出现在从左算起的第i位上，称这样的简单析取式为**极大项**。

上述定义中，所谓的预定顺序通常就是命题变量的字母顺序，或命题变量下标顺序。本质上来说，不管是什么顺序，只要是在求与任何公式的等值的主析取范式或主合取范式之前确定的一个顺序即可。我们下面将这种顺序理解为命题变量的字母顺序。

由于每个命题变量在极小项中只能以原形或否定形式出现且仅出现一次，因而n个命题变量只能产生2^n个不同的极小项，例如当$n=1$时，只有p和$\neg p$两个极小项，当$n=2$时只有四个极小项：

$$\neg p \wedge \neg q \qquad \neg p \wedge q \qquad p \wedge \neg q \qquad p \wedge q$$

而且这些极小项有一个特点，只有当：(i) 如果出现的是命题变量本身，则对其赋值为1，且(ii) 如果出现的是命题变量的否定，则对其赋值为0时，该极小项的真值才为1。例如极小项$\neg p \wedge q$只有在对p赋值为0，而q赋值为1时的真值才为1。

我们把这种使得极小项真值为1的命题变量的赋值作为极小项的编码，例如极小项$\neg p \wedge q$的编码就是01。或者从极小项的形式上看，**只要出现命题变量本身就用1编码，出现命题变量的否定就用0编码**，按照字母顺序排列这些二进制数就得到该极小项的编码。进一步使用小写的m字母加上该极小项编码的十进制值为下标的名称命名该极小项，例如$\neg p \wedge q$就命名为m_1。

极大项也有类似性质，当$n=1$时，也只有p和$\neg p$两个极大项，当$n=2$时只有四个极大项：

$$\neg p \vee \neg q \qquad \neg p \vee q \qquad p \vee \neg q \qquad p \vee q$$

极大项的特点是，只有当：(i) 如果出现的是命题变量本身，则对其赋值为0，且(ii) 如果出现的是命题变量的否定，则对其赋值为1时，该极大项的真值才为0。例如极大项$\neg p \vee q$只有在对p赋值为1，而q赋值为0时的真值才为0。

同样将这种使得极大项真值为0的命题变量的赋值作为极大项的编码，例如极大项$\neg p \vee q$的编码就是10。或者从极大项的形式上看，**只要出现命题变量本身就用0编码，出现命题变量的否定就用1编码**，按照字母顺序排列这些二进制数就得到该极大项的编码。进一步使用大写的M字母加上该极大项编码的十进制值为下标的名称命名该极大项，例如$\neg p \wedge q$就命名为M_2。

按照这种编码和命名规则，下表列出了只有两个命题变量p和q构成的极小项和极大项：

极小项	成真赋值的编码	名称	极大项	成假赋值的编码	名称
$\neg p \wedge \neg q$	0 0	m_0	$p \vee q$	0 0	M_0
$\neg p \wedge q$	0 1	m_1	$p \vee \neg q$	0 1	M_1
$p \wedge \neg q$	1 0	m_2	$\neg p \vee q$	1 0	M_2
$p \wedge q$	1 1	m_3	$\neg p \vee \neg q$	1 1	M_3

下表则列出了有三个命题变量p, q和r的极小项和极大项：

极小项	成真赋值的编码	名称	极大项	成假赋值的编码	名称
$\neg p \wedge \neg q \wedge \neg r$	0 0 0	m_0	$p \vee q \vee r$	0 0 0	M_0
$\neg p \wedge \neg q \wedge r$	0 0 1	m_1	$p \vee q \vee \neg r$	0 0 1	M_1
$\neg p \wedge q \wedge \neg r$	0 1 0	m_2	$p \vee \neg q \vee r$	0 1 0	M_2
$\neg p \wedge q \wedge r$	0 1 1	m_3	$p \vee \neg q \vee \neg r$	0 1 1	M_3
$p \wedge \neg q \wedge \neg r$	1 0 0	m_4	$\neg p \vee q \vee r$	1 0 0	M_4
$p \wedge \neg q \wedge r$	1 0 1	m_5	$\neg p \vee q \vee \neg r$	1 0 1	M_5
$p \wedge q \wedge \neg r$	1 1 0	m_6	$\neg p \vee \neg q \vee r$	1 1 0	M_6
$p \wedge q \wedge r$	1 1 1	m_7	$\neg p \vee \neg q \vee \neg r$	1 1 1	M_7

请读者务必根据上表熟记极小项和极大项的形式。虽然极小项$\neg p \wedge \neg q$和$\neg p \wedge \neg q \wedge \neg r$都命名为$m_0$，但很容易从公式所包含的命题变量个数区分。从上表不难看出，极小项与极大项有如下关系：

定理 2.3.15 设m_i和M_i是由命题变量$p_1, p_2, \cdots, p_n$构成的极小项和极大项，则：$\neg m_i \Leftrightarrow M_i$，且$\neg M_i \Leftrightarrow m_i$。

定义 2.3.16 若由n个命题变量构成的析取范式中所有的简单合取式都是极小项，则该析取范式称为**主析取范式**；若由n个命题变量构成的合取范式中所有的简单析取式都是极大项，则该合取范式称为**主合取范式**。

定理 2.3.17 任何命题公式都存在着与之等值且惟一的主析取范式；任何命题公式也都存在与之等值的且惟一的主合取范式。

证明 任何命题公式都可以构造它的真值表，如果该命题公式存在n个命题变量$p_1, p_2, \cdots, p_n$，则它的真值表有2^n行，将真值为1的行所对应的对命题变量$p_1, p_2, \cdots, p_n$的赋值按预定顺序排列成一个二进制数，将这个二进制数作为极小项的编码，该公式必等值于由这些极小项的析取构成的主析取范式，因为该公式的真值为1当且仅当对命题变量$p_1, p_2, \cdots, p_n$按其中一个极小项的编码赋值。另一方面，由于极小项的真值为1当且仅当按照其编码对其中的命题变量赋值，因此该公式的主析取范式只能含有这些极小项，因此在不计较极小项的顺序（或严格按极小项编码顺序）的情况下，与一个公式等值的主析取范式惟一。

类似地，将真值为0的行所对应的对命题变量$p_1, p_2, \cdots, p_n$的赋值按预定顺序排列成一个二进制数作为极大项的编码，该公式必等值于有这些极大项的合取构成的主合取范式，因为该公式的真值为0当且仅当对命题变量$p_1, p_2, \cdots, p_n$按其中一个极大项的编码赋值。同样，由于极大项的真值为0当且仅当按照其编码对其中的命题变量赋值，因此该公式的主合取范式也只能含有这些极大项，因此与一个公式等值的主合取范式惟一（按极大项编码顺序）。 □

上述定理的证明表明我们可利用真值表求一个公式的主析取范式和主合取范式：

例子 2.3.18 使用真值表求与公式$(p \to q) \leftrightarrow r$等值的主析取范式和主合取范式。

解答：我们构造该公式的真值表如下：

p	q	r	$(p \to q)$	$(p \to q) \leftrightarrow r$
0	0	0	1	0
0	0	1	1	1
0	1	0	1	0
0	1	1	1	1
1	0	0	0	1
1	0	1	0	0
1	1	0	1	0
1	1	1	1	1

根据该真值表，公式$(p \to q) \leftrightarrow r$的主析取范式由编码为$001, 011, 100, 111$的极小项构成，即：

$$(p \to q) \leftrightarrow r \Leftrightarrow m_1 \vee m_3 \vee m_4 \vee m_7$$

而该公式的主合取范式由编码为$000, 010, 101, 110$的极大项构成，即：

$$(p \to q) \leftrightarrow r \Leftrightarrow M_0 \wedge M_2 \wedge M_5 \wedge M_6$$

在容易构造公式的真值表时（例如命题变量个数少于等于3个），这种方法求与公式等值的主析取范式和主合取范式极为方便，而且这种方法也给出了一个公式的主析取范式和主合取范式之间的关系，即包含n个命题变量的公式，其等值的主析取范式和主合取范式的编码是互补的，即**它的主析取范式和主合取范式的编码合在一起必构成从0到$(2^n - 1)$这2^n个数的所有二进制编码**。特别地，

1. 对于永真式，其真值表的每行真值都为1，从而**永真式的主析取范式包括所有的极小项**；由于其真值表没有一行的真值是0，我们将**永真式的主合取范式特别地记为1**；

2. 对于矛盾式，其真值表的每行真值都为0，从而**矛盾式的主合取范式包括所有的极大项**；由于其真值表没有一行的真值是1，我们将**矛盾式的主析取范式特别地记为0**；

但在命题变量个数多，不容易构造公式的真值表时，这种方法不适合用于求公式等值的主析取范式和主合取范式，这时我们仍需借助等值演算方法。实际上，对于含有n命题变量$p_1, p_2, \cdots, p_n$的公式A，可使用前面所说的等值演算法求出与其等值的一个析取范式A'，然后再在A'的基础上求与A等值的主析取范式：

1. 对于A'中的所有简单合取式做这样的扩展工作：设A_i是A'所含的简单合取式，且A_i既不含命题变量p_j，也不含它的否定$\neg p_j$，则将A_i扩展成如下形式：

$$A_i \Leftrightarrow A_i \wedge 1 \Leftrightarrow A_i \wedge (p_j \vee \neg p_j) \Leftrightarrow (A_i \wedge p_j) \vee (A_i \wedge \neg p_j)$$

继续这个过程直到所有简单合取式都含有所有命题变量或它的否定。

2. 上述过程会产生重复的极小项，消除重复，并将极小项按编码排序则得到与该公式等值的主析取范式；

利用主析取范式与主合取范式的特点，得到与公式A等值的主析取范式之后可很容易地得到与之等值的主合取范式，当然也可类似地直接从与公式A等值的合取范式开始扩展得到与其等值的主合取范式，这时对与公式A等值的合取范式中的简单析取式A_i做如下扩展：

$$A_i \Leftrightarrow A_i \vee 0 \Leftrightarrow A_i \vee (p_j \wedge \neg p_j) \Leftrightarrow (A_i \vee p_j) \wedge (A_i \vee \neg p_j)$$

例子 2.3.19 使用等值演算方法求与公式$(p \to q) \leftrightarrow r$等值的主析取范式和主合取范式。

解答：

1. 前面已经得到一个与该公式等值的析取范式：

$$(p \to q) \leftrightarrow r \Leftrightarrow (p \wedge \neg q \wedge \neg r) \vee (\neg p \wedge r) \vee (q \wedge r)$$

$(p \wedge \neg q \wedge \neg r)$已经是极小项，编码是100，即$m_4$，无需再扩展，而$(\neg p \wedge r)$可作如下扩展：

$$(\neg p \wedge r) \Leftrightarrow (\neg p \wedge r) \wedge (q \vee \neg q) \Leftrightarrow (\neg p \wedge q \wedge r) \vee (\neg p \wedge \neg q \wedge r)$$

即$(\neg p \wedge\ r)$扩展得到两个极小项，其编码分别是011和001，即m_3和m_1。而$(q \wedge r)$可作如下扩展：

$$(q \wedge r) \Leftrightarrow (q \wedge r) \wedge (p \vee \neg p) \Leftrightarrow (p \wedge q \wedge r) \vee (\neg p \wedge q \wedge r)$$

即$(q \wedge r)$扩展得到两个极小项，其编码分别是111和011，即m_7和m_3，消除重复，得到：

$$(p \to q) \leftrightarrow r \Leftrightarrow m_1 \vee m_3 \vee m_4 \vee m_7$$

2. 前面也已经得到一个与该公式等值的合取范式：

$$(p \to q) \leftrightarrow r \Leftrightarrow (p \vee r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee \neg r)$$

将$(p \vee r)$扩展可得到：

$$(p \vee r) \Leftrightarrow (p \vee q \vee r) \wedge (p \vee \neg q \vee r) \Leftrightarrow M_0 \wedge M_2$$

而将$(\neg q \vee r)$扩展可得到：

$$(\neg q \vee r) \Leftrightarrow (p \vee \neg q \vee r) \wedge (\neg p \vee \neg q \vee r) \Leftrightarrow M_2 \wedge M_6$$

而$(\neg p \vee q \vee \neg r)$是极大项$M_5$，因此与该公式等值的主合取范式是：

$$(p \to q) \leftrightarrow r \Leftrightarrow M_0 \wedge M_2 \wedge M_5 \wedge M_6$$

实际上，根据极小项和极大项的特点，我们**无需真的**对简单合取式进行扩展，而只要弄清楚二进制编码即可。例如，假设公式含有p, q, r三个命题变量，与某公式（如上述公式）等值的析取范式中含有简单合取式$(q \wedge r)$，这个简单合取式扩展后的极小项有哪些呢？

我们可这样计算，根据p, q, r的字母顺序，这个简单合取式可编码为(-11)，其中$(-)$留给此简单合取式中不出现命题变量p。对这个编码，将$-$取遍所有的二进制数即可得011和111，因此简单合取式$(q \wedge r)$扩展得到的极小项必是m_3和m_7。

这种方式的扩展可推广到公式含有多个命题变量的情况，从而使得等值演算方法求与公式等值的主析取范式或主合取范式可用于含有多个命题变量的公式。例如，设某公式含有p, q, r, s, t五个命

题变量，通过等值演算法得到与该公式等值的析取范式中含有简单合取式$(q \wedge r)$，这个简单合取式扩展后的极小项有哪些呢？

注意这时有五个变量，如果按照上述方法利用等值演算一步一步去扩展复杂而容易出错。但我们可这样计算，根据p,q,r,s,t的字母顺序，这个简单合取式可编码为$(-11--)$，其中$(-)$按顺序留给不在该简单合取式中出现的命题变量p,s,t。在这三个位置取遍所有可能的二进制数，得到$01100, 01101, 01110, 01111, 11100, 11101, 11110, 11111$这八个二进制编码，因此简单合取式$(q \wedge r)$扩展成由$p,q,r,s,t$构成的极小项必然是这八个二进制编码所对应的极小项，即$m_{12}, m_{13}, m_{14}, m_{15}, m_{28}, m_{29}, m_{30}, m_{31}$。后面在介绍主范式的应用时将使用这个方法求一个含有五个变量公式的主析取范式。

2.3.3 主范式的应用

求与公式等值的主析取和主合取范式有许多应用：

1. **判断两公式是否等值**：由于与公式等值的主析取范式和主合取范式都惟一，因此要判断两个公式是否等值，只要分别求出与这两个公式等值的主析取范式或主合取范式，如果它们的主析取范式或主合取范式相等就等值，否则就不等值；

2. **判断公式的类型**：从与公式等值的主析取范式或主合取范式容易看出该公式是否是永真式、矛盾式还是可满足式。从主析取范式角度看：(i) 永真式的主析取范式包括所有可能的极小项，(ii) 矛盾式的主析取式范式是$\mathbf{0}$，不包括任何极小项，(iii) 非永真式的可满足式的主析取范式介于这两者之间。从主合取范式角度看：(i) 永真式的主合取范式是$\mathbf{1}$，不包括任何极大项，(ii) 矛盾式的主合取式范式包括所有可能的极小项，(iii) 非永真式的可满足式的主合取范式介于这两者之间。

3. **给出公式的成真赋值和成假赋值**：实际上，**与公式等值的主析取范式和主合取范式与公式的真值表有严格的对应关系**，因此从主析取范式或主合取范式，很容易判定对公式中所含命题变量怎样的赋值使得公式的真值为真，怎样的赋值会使得公式的真值为假。这在公式包含的命题变量多，不方便构造公式真值表时十分有用，可以使用等值演算法求出与公式的主析取范式或主合取范式，从而判断公式何时为真，何时为假，并由此解决一些实际应用问题。

前面已经举例说明如何使用等值演算法证明公式等值以及如何使用构造真值表法判断公式类型，利用主范式求解这两类问题的方法也比较简单，这里不再举例。下面举例说明如何利用主范式（或者更广义地，利用等值演算方法）求解实际应用问题，这些例子都来自[3]第二章的习题（略有修改）。

例子 2.3.20 一个排队线路，输入为A,B,C，其输出分别是F_A, F_B, F_C。该线路的功能是，在同一时间只能有一个信号通过，若同时有两个或两个以上信号申请输出时，则按A,B,C的顺序输出，写出F_A, F_B, F_C在联结词完备集$\{\neg, \vee\}$中的表达式。

分析：首先应该理解题意，抓住问题的关键。这里的关键是弄清楚F_A, F_B, F_C何时有信号输出，我们当F_A有信号输出时，它对应的公式的真值为1，否则为0，F_B, F_C类似。同样地，当输入端A有信号输入时理解将其赋值为1，否则赋值为0，B,C的类似，也即，题目中的原子命题是$A:A$端有信号输入，$B:B$端有信号输入和$C:C$端有信号输入，用A,B,C写出F_A, F_B, F_C的表达式（这里为了与题意一致，采用大写字母表示原子命题）。

根据该线路的功能，同一时间只能有一个信号通过，也即当只有A端有信号输入时，F_A有信号

输出，当只有B端有信号输入时，F_B有信号输出，当只有C端有信号输入时，F_C有信号输出。而当有两个或两个以上信号申请输出，也即当A,B,C三个端子同时有两个或三个信号输入是，按A,B,C的顺序输出，即例如，若A,B端同时有信号输入，则F_A有信号输出，B,C端同时有信号输入，则F_B有信号输出等等。

根据上述分析，我们可列出F_A,F_B,F_C的真值表，然后写出它们的主析取范式，通过等值演算变形成在联结词完备集$\{\neg,\vee\}$中的表达式。

解答：令A表示A端有信号输入，B表示B端有信号输入，C表示C端有信号输入，根据题意，列出F_A,F_B,F_C的真值表如下：

A	B	C	F_A	F_B	F_C	注释
0	0	0	0	0	0	都没有信号输入，就都不输出
0	0	1	0	0	1	只有C有信号输入
0	1	0	0	1	0	只有B有信号输入
0	1	1	0	1	0	B优先输出
1	0	0	1	0	0	只有A有信号输入
1	0	1	1	0	0	以下都是A优先输出
1	1	0	1	0	0	
1	1	1	1	0	0	

根据上表，容易写出F_A,F_B,F_C的表达式：

$$F_A = A \qquad F_B = \neg A \wedge B \qquad F_C = \neg A \wedge \neg B \wedge C$$

根据德摩根律，容易将上述表达式变形为完备集$\{\neg,\vee\}$中的表达式：

$$F_A \Leftrightarrow A \qquad F_B \Leftrightarrow \neg(A \vee \neg B) \qquad F_C = \neg(A \vee B \vee \neg C)$$

备注 2.3.21 通常在数字逻辑电路与设计课程中，设计组合电路时与上述解题思路相同：根据需要确定线路的功能，根据线路功能确定输出端与输入端之间逻辑关系，这时往往使用真值表可容易地根据线路功能表达这种逻辑关系，然后再化简（数字逻辑电路与设计课程中有专门的化简方法，称为**卡诺图法**），这种化简通常是根据最基本的门电路元件而进行，例如，在数字逻辑电路设计中，“与非门”或“或非门”十分常见，因此通常化简为只含$\neg,\wedge$或只含$\neg,\vee$的表达式。

例子 2.3.22 某公司要从赵、钱、孙、李、周五名新毕业的大学生中选派一些人出国学习。根据他们的工作需要，选派必须满足以下条件：

(1) 若赵去，则钱也去；
(2) 李、周两人中必有一人去；
(3) 钱、孙两人中去且去一人；
(4) 孙、李两人同去或同不去；
(5) 若周去，则赵、钱也同去。

请问，公司应如何选派他们出国？

分析：对于这种应用问题，首先要抓住问题的关键。这里问题的关键是派谁出国学习，因此应该使用原子命题表示选派某某出国，例如用p表示“选派赵出国学习”等，然后使用这些原子命题符

号化选派所应满足的条件，而最后的选派方案应该同时满足这些条件，也即最后选派方案对应的公式应该是这些条件的合取，从而可通过确定该公式的成真赋值，即哪些变量赋值为真（相当于相应地选派某人出国学习）会使得公式为真而确定选派方案。

解答：令p,q,r,s,t分别表示赵、钱、孙、李、周出国学习，则题中的选派条件可符号化为：

(1) 若赵去，钱也去：

$$p \to q \Leftrightarrow (\neg p \vee q)$$

(2) 李、周两人必有一人去：

$$r \vee t$$

(3) 钱、孙两人去且仅去一人：

$$(q \wedge \neg r) \vee (\neg q \wedge r) \Leftrightarrow (q \vee r) \wedge (\neg q \vee \neg r)$$

(4) 孙、李两人同去或同不去：

$$(r \wedge s) \vee (\neg r \wedge \neg s) \Leftrightarrow (r \vee \neg s) \wedge (\neg r \vee s)$$

(5) 若周去，则赵、钱也同去：

$$t \to (p \wedge q) \Leftrightarrow (\neg t \vee p) \wedge (\neg t \vee q) \Leftrightarrow (p \vee \neg t) \wedge (q \vee \neg t)$$

由于选派方案对应的公式是这些条件的合取，因此我们将上述条件都变形为简单析取式，以方便求与公式等值的合取范式。选派方案所对应的公式是：

$$F = (\neg p \vee q) \wedge (r \vee t) \wedge (q \vee r) \wedge (\neg q \vee \neg r) \wedge (r \vee \neg s) \wedge (\neg r \vee s) \wedge (p \vee \neg t) \wedge (q \vee \neg t)$$

可通过求F的主合取范式确定F的成真赋值。因为上式已经合取范式，因此只要将其中的每个简单析取式扩展为极大项：

(1) 考虑$(\neg p \vee q)$，按p,q,r,s,t的顺序，它的编码应该是$(10---)$，从而它扩展出的极大项的编码应该是：

$$10000 \quad 10001 \quad 10010 \quad 10011 \quad 10100 \quad 10101 \quad 10110 \quad 10111$$

也就是说$(\neg p \vee q)$扩展得到极大项是M_{16}至M_{23}。

(2) 考虑$r \vee t$，它的编码应该是$(--0-0)$，从而它扩展出的极大项的编码应该是：

$$00000 \quad 00010 \quad 01000 \quad 01010 \quad 10000 \quad 10010 \quad 11000 \quad 11010$$

也就是说$r \vee t$扩展得到的极大项是$M_0, M_2, M_8, M_{10}, M_{16}, M_{18}, M_{24}, M_{26}$。

(3) $q \vee r$的编码是$(-00--)$，它扩展出的极大项的编码应该是：

$$00000 \quad 00001 \quad 00010 \quad 00011 \quad 10000 \quad 10001 \quad 10010 \quad 10011$$

也即它扩展得到的极大项是$M_0, M_1, M_2, M_3, M_{16}, M_{17}, M_{18}, M_{19}$。

(4) $\neg q \vee \neg r$的编码是$(-11--)$，扩展得到的极大项的编码应该是：

$$01100 \quad 01101 \quad 01110 \quad 01111 \quad 11100 \quad 11101 \quad 11110 \quad 11111$$

即为M_{12}至M_{15}，以及M_{28}至M_{31}。

(5) $(r \vee \neg s)$的编码是$(--01-)$，扩展得到的极大项的编码应该是：

$$00010 \quad 00011 \quad 01010 \quad 01011 \quad 10010 \quad 10011 \quad 11010 \quad 11011$$

即为$M_2, M_3, M_{10}, M_{11}, M_{18}, M_{19}, M_{26}, M_{27}$。

(6) $(\neg r \vee s)$的编码是$(--10-)$，扩展得到的极大项的编码应该是：

$$00100 \quad 00101 \quad 01100 \quad 01101 \quad 10100 \quad 10101 \quad 11100 \quad 11101$$

即为$M_4, M_5, M_{12}, M_{13}, M_{20}, M_{21}, M_{28}, M_{29}$。

(7) $(p \vee \neg t)$的编码是$(0---1)$，扩展得到的极大项的编码应该是：

$$00001 \quad 00011 \quad 00101 \quad 00111 \quad 01001 \quad 01011 \quad 01101 \quad 01111$$

即为$M_1, M_3, M_5, M_7, M_9, M_{11}, M_{13}, M_{15}$，

(8) $(q \vee \neg t)$的编码是$(-0--1)$，扩展得到的极大项的编码应该是：

$$00001 \quad 00011 \quad 00101 \quad 00111 \quad 10001 \quad 10011 \quad 10101 \quad 10111$$

即为$M_1, M_3, M_5, M_7, M_{17}, M_{19}, M_{21}, M_{23}$。

因此F的主合取范式包含了除M_6及M_{25}以外的所有极大项，从而F的主析取范式为$m_6 \vee m_{25}$，即它的成真赋值是00110和11001，即r, s为真或p, q, t为真，最后得到选派方案为：**“孙、李去”**或**“赵、钱、周去”**。

备注 2.3.23 上述扩展使用手工完成还是显得比较复杂，但由于这种扩展十分规则，我们可编写一个程序来自动完成上述扩展，程序的功能应该将简单析取式（或简单合取式）扩展成极大项（或极小项），输入是简单析取式（或简单合取式）的编码，输出是扩展出的极大项（或极小项）的编码。进一步，可编写程序自动将合取范式扩展成主合取范式，析取范式扩展成主析取范式。实际上，我已经编写了这样一个Java程序。

作业

作业 2.1 使用等值演算方法证明下列等值式（注意写清楚所使用的基本等值式）：

(1) $p \to (q \to p) \Leftrightarrow \neg\, p \to (p \to \neg\, q)$

(2) $\neg(p \leftrightarrow q) \Leftrightarrow (p \vee q) \wedge \neg(p \wedge q) \Leftrightarrow (p \wedge \neg q) \vee (\neg p \wedge q)$

(3) $(p \to (q \vee r)) \Leftrightarrow (p \wedge \neg\, q) \to r$

(4) $((p \wedge q) \to r) \wedge (q \to (s \vee r)) \Leftrightarrow (q \wedge (s \to p)) \to r$

作业 2.2 对任意的公式A, B, C，如果$A \vee C \Leftrightarrow B \vee C$，是否有$A \Leftrightarrow B$？如果$A \wedge C \Leftrightarrow B \wedge C$是否有$A \Leftrightarrow B$？如果$\neg A \Leftrightarrow \neg B$，是否有$A \Leftrightarrow B$？为什么？

作业 2.3 求解下面有关联结词完备集的问题：

(1) 将公式$(p \to (q \wedge r)) \vee p$化成与之等值的且仅含$\neg$和$\wedge$的公式；

(2) 将公式$(p \to (q \wedge \neg p)) \wedge q \wedge r$化成与之等值的且仅含$\neg$和$\vee$的公式；

(3) 将公式$(p \to (q \wedge \neg p)) \wedge q \wedge r$化成与之等值的且仅含$\neg$和$\to$的公式；

(4) 定义联结词“与非”↑和“或非”↓：对任意的公式A和B，

$$A \uparrow B \Leftrightarrow \neg(A \wedge B) \qquad A \downarrow B \Leftrightarrow \neg(A \vee B)$$

在数字逻辑电路设计中经常使用与非门和或非门，不难证明$\{\uparrow\}$是联结词的完备集，而且$\{\downarrow\}$也是联结词的完备集。试将公式$p \to (\neg p \to q)$化成与之等值的且仅含$\{\uparrow\}$的公式，同样把该公式化成与之等值的且仅含$\{\downarrow\}$的公式。

作业 2.4 使用等值演算方法求与下面公式等值的析取范式和合取范式（请注意写清楚等值演算中所用的基本等值式）：

(1) $(p \wedge q) \vee \neg(p \wedge q)$

(2) $p \to (\neg p \wedge q \wedge r)$

(3) $\neg(p \vee \neg q) \wedge (s \to t)$

作业 2.5 使用等值演算方法或构造真值表法求与下面公式等值的主析取范式和主合取范式（请注意，如使用等值演算法请写清楚所用的基本等值式，如使用构造真值表法请列出构造过程中的子公式）：

(1) $(\neg p \vee \neg q) \to (p \leftrightarrow q)$

(2) $(p \to (q \wedge r)) \wedge (\neg p \to (\neg q \wedge \neg r))$

(3) $p \to ((q \wedge r) \to s)$

作业 2.6 某电路有一个灯泡和三个开关A, B, C。已知在且仅在下述四种情况下灯亮：

(a) C的扳键向上，A, B的扳键向下；

(b) A的扳键向上，B, C的扳键向下；

(c) B, C的扳键向上，A的扳键向下；

(d) A, B的扳键向上，C的扳键向下。

设F为1表示灯亮，p, q, r分别表示A, B, C的扳键向上（也即$\neg p, \neg q, \neg r$分别表示A, B, C的扳键向下）。

(1) 求F的主析取范式；

(2) 在联结词完备集$\{\neg, \wedge\}$上构造F；

(3) 在联结词的完备集$\{\neg, \to\}$上构造F。

作业 2.7 A, B, C, D四人要派两个人出差，按下述三个条件有几种派法？如何派？

(a) 若A去，则C和D中要去且仅去一人；

(b) B和C不能都去；

(c) 若C去，则D不能去。

第三章　命题逻辑的自然推理

在前面讨论命题逻辑公式及其真值，以及命题逻辑公式的等值演算基础上，这一章讨论基于命题逻辑的自然推理。**推理**(inference)，简单地说，就是从一些称为**前提**的判断（命题）合乎逻辑地得到一个称为**结论**的判断（命题）。

在命题逻辑中，使用命题逻辑公式及其之间的关系表示推理的形式结构。自然推理是判定这种推理形式是否有效的一种方法，它的基本思想是确定一些推理规则，这些规则具有**保真性**，也就是说，依据这些规则，从真前提只会推出真结论。因此，从所要判定的推理的前提出发，根据自然推理给出的规则，如果能推出预期的结论，就说明该推理有效。当然，如果不能如此推出预期的结论，尚不能就此断定此推理是无效的，要断定推理的无效，需要用其他方法。

自然推理与下一章要讨论的命题演算的公理系统所讨论的推理的不同之处，在于自然推理的推理依据只有推理规则，没有公理，更符合人们日常思维的自然习惯，因此称为自然推理。而命题演算的公理系统通常从较少的公理出发，使用较少的规则，通过推演公理系统所定义的定理研究推理形式的有效性。

3.1　推理的形式结构及其有效性

我们先定义推理的形式结构及其有效性：

定义 3.1.1 推理是由一组有限的公式构成的前提推出一个结论，设前提包括$A_1, A_2, \cdots, A_n$，而结论是A，则将从前提$A_1, A_2, \cdots, A_n$推出结论A这个推理的**形式结构**记为：

$$A_1, A_2, \cdots, A_n \vdash A$$

称推理$A_1, A_2, \cdots, A_n \vdash A$是**有效的**当且仅当命题逻辑公式$(A_1 \wedge A_2 \wedge \cdots \wedge A_n) \to A$是**永真式**。

注意，有的教材将推理的形式结构记为$A_1 \wedge A_2 \wedge \cdots \wedge A_n \Rightarrow A$，这里为方便与后面所讨论的公理化命题演算系统对比而采用上述记号。

根据合取联结词的结合律和交换律，推理$A_1, A_2, \cdots, A_n \vdash A$给出前提的顺序不影响它的有效性，又由合取联结词的幂等律，重复的前提也没有意义，因此实际上可将前提看作一个集合，称为**前提集**，并常用大写的希腊字母命名前提集而简化推理的形式结构，例如令$\Gamma = \{A_1, A_2, \cdots, A_n\}$，从而将$A_1, A_2, \cdots, A_n \vdash A$简记为$\Gamma \vdash A$。

当前提用集合形式给出时，通常仍用逗号表示前提集的并，即通常用$\Gamma, \Delta \vdash A$表示$\Gamma \cup \Delta \vdash A$，并在没有特别说明时约定这时有$\Gamma \cap \Delta = \varnothing$。进一步用$\Gamma, B \vdash A$表示$\Gamma \cup \{B\} \vdash A$，并在没有特别说明时约定这时有$B \notin \Gamma$。注意，我们用**大写希腊字母表示前提集，而大写英文字母表示公式**。

3.2 自然推理系统

按照推理有效性的定义，当前提$\Gamma=\{A_1,A_2,\cdots,A_n\}$时，推理$\Gamma\vdash A$的有效性可以通过构造命题逻辑公式$(A_1\wedge A_2\wedge\cdots\wedge A_n\rightarrow A)$的真值表而判定。但我们知道，当公式中含有太多命题变量时，构造真值表的方法往往显得复杂，而自然推理系统则提供了判定推理有效性更方便的方法。

自然推理系统提供一系列的**推理规则**用于构造一个**证明序列**来验证某个推理是否有效。简单地说，证明序列是一系列的推理形式结构：

$$\begin{array}{c}\Gamma_1\vdash A_1\\ \Gamma_2\vdash A_2\\ \vdots\quad\vdots\\ \Gamma_n\vdash A_n\end{array}$$

序列中的每一个推理形式结构$\Gamma_i\vdash A_i(1\leq i\leq n)$都是通过排在它前面的推理形式结构（即下标范围为1至$i-1$的推理形式结构）根据自然推理系统的规则而得到的，整个证明序列称为**推理$\Gamma_n\vdash A_n$的证明**。当然，第一个推理形式结构$\Gamma_1\vdash A_1$由是自然推理系统的规则直接得到。实际上，对任意的$1\leq i\leq n$，证明序列：

$$\begin{array}{c}\Gamma_1\vdash A_1\\ \Gamma_2\vdash A_2\\ \vdots\quad\vdots\\ \Gamma_i\vdash A_i\end{array}$$

都是推理$\Gamma_i\vdash A_i$的证明。

备注 3.2.1 大多数离散数学教材在讲述命题逻辑的自然推理时，常采用省略前提的证明序列：

$$\begin{array}{c}A_1\\ A_2\\ \vdots\\ A_n\end{array}$$

这里$A_i(1\leq i\leq n)$都是公式，或者是推理$B_1,B_2,\cdots,B_m\vdash B$中的前提（即存在$1\leq j\leq m$使得$A_i=B_j$，或者是由排在它前面的公式利用自然推理系统的规则得到的，而最后的$A_n=B$。

但实际上，在这个证明序列中A_i不是永真式，只有$B_1\wedge B_2\wedge\cdots\wedge B_m\rightarrow A_i$才可能是永真式。所谓“可能”是说，在整个推理中，直观地看A_i实际上可能由不同的前提推出，而不一定是由$B_1,B_2,\cdots,B_n$推出。也就是说，上述证明序列隐藏了前提的变化，在逻辑上可以说是不严格的。

因此我们觉得应该将每一次推理的前提都写出来，从而明确前提的变化，而且在概念上更简单地，证明序列就是一系列推理形式结构$\Gamma_i\vdash A_i$，直观地看，这时A_i都是由Γ_i推出，或更准确地，若$\Gamma_i=\{B_{1i},B_{2i},\cdots,B_{mi}\}$，则$B_{1i}\wedge B_{2i}\wedge\cdots\wedge B_{mi}\rightarrow A_i$就一定是永真式。当然，这两种形式的证明序列并无本质的差别，后面的例子将说明这一点。

下面将首先讨论自然推理系统的推理规则，然后严格定义证明序列，推理的证明，以及推理的有效性等。

3.2.1 自然推理系统的基本规则

推理规则用于构造推理的证明序列。前面看到，证明序列的第一个推理形式应该由自然系统的最基本规则得到，这个最基本规则称为前提引入规则：

定理 3.2.2 **前提引入规则**：对任意的前提集Γ和公式A，若$A\in\Gamma$，则 $\Gamma\vdash A$ 是有效的推理。

直观地说，前提引入规则是说，如果推理的结论在前提集中，当然前提集可以合乎逻辑地推出结论。上面是用定理的形式给出这个规则，因为该规则断定，当$A\in\Gamma$时，$\Gamma\vdash A$是有效的推理，也即直观地，前提集Γ可合乎逻辑地推出A。若$\Gamma=\{A_1,\cdots,A,\cdots,A_n\}$，显然

$$(A_1\wedge\cdots\wedge A\wedge\cdots\wedge A_n)\to A$$

是永真式，因此这个定理很容易证明。

下面要讨论的规则比上述基本规则复杂，通常是断定，对任意前提集$\Gamma_1,\Gamma_2,\cdots,\Gamma_n,\Gamma$和任意公式$A_1,A_2,\cdots,A_n,A$，若$\Gamma_1\vdash A_1,\Gamma_2\vdash A_2,\cdots,\Gamma_n\vdash A_n$是有效的推理，则$\Gamma\vdash A$也是有效的推理。为简单起见，通常用下面的形式给出规则：

$$\frac{\Gamma_1\vdash A_1\quad\Gamma_2\vdash A_2\quad\cdots\quad\Gamma_n\vdash A_n}{\Gamma\vdash A}$$

并把$\Gamma_1\vdash A_1,\Gamma_2\vdash A_2,\cdots,\Gamma_n\vdash A_n$称为**该规则的前提**，而$\Gamma\vdash A$称为**该规则的结论**。

上述前提引入规则可写成如下形式：

$$\text{前提引入}\ \frac{}{\Gamma,A,\Delta\vdash A}$$

其中Γ,Δ任意的前提集，A是任意的公式，这里使用Γ,A,Δ的形式强调A属于该推理形式结构的前提集。从形式上看，这个规则本身没有前提，因此视为最基本的规则，相对于公理化命题演算系统中的公理。

自然推理系统之所以自然，就是它有许多规则，而且这些规则与命题联结词密切相关，符合人们日常使用这些联结词进行推理的方式。具体来说，自然推理系统对于每个命题联结词都相应地有**引入**和**消除**规则，下面一一介绍这些规则。

定理 3.2.3 **否定引入规则**和**否定消除规则**：对任意的前提集Γ和任意公式A,B有：

$$\text{否定引入}\ \frac{\Gamma\vdash A}{\Gamma\vdash\neg\neg A}\qquad\qquad\text{否定消除}\ \frac{\Gamma,\neg A\vdash B\qquad\Gamma,\neg A\vdash\neg B}{\Gamma\vdash A}$$

证明 由于我们主要是关心规则的使用，而非规则正确性的证明，因此这里只以否定引入规则简单解释一下如何证明规则的正确性，及规则正确性的含义，后面给出规则时，我们都将不再证明。

设$\Gamma=\{A_1,A_2,\cdots,A_n\}$，要证明否定引入规则的正确性，实际上就是要证明当$(A_1\wedge A_2\wedge\cdots\wedge A_n)\to A$是永真式时，$(A_1\wedge A_2\wedge\cdots\wedge A_n)\to\neg\neg A$也是永真式，也即从逻辑上来说，规则的正确性就是规则的**保真性**：**若规则前提都是永真式，则规则结论也是永真式**。这种保真性又将保证使用规则得到的推理形式结构$\Gamma\vdash A$的**保真性**：**若前提集Γ中的所有公式真，则结论A也真**。

实际上，要证明$(A_1 \wedge A_2 \wedge \cdots \wedge A_n) \to A$是永真式时，$(A_1 \wedge A_2 \wedge \cdots \wedge A_n) \to \neg\neg A$也是永真式并不难。对任意的真值赋值函数$t : \mathbf{Var} \to \{0,1\}$，为简单起见，记：

$$t(\Gamma) = t(A_1 \wedge A_2 \wedge \cdots \wedge A_n)$$

$t(\Gamma)$不是0就是1。当$t(\Gamma) = 0$时，根据蕴涵联结词的定义，

$$t((A_1 \wedge A_2 \wedge \cdots \wedge A_n) \to \neg\neg A) = 1$$

而当$t(\Gamma) = 1$时，由$t((A_1 \wedge A_2 \wedge \cdots \wedge A_n) \to A) = 1$，得$t(A) = 1$，从而由否定联结词的含义$t(\neg\neg A) = 1$，从而也有

$$t((A_1 \wedge A_2 \wedge \cdots \wedge A_n) \to \neg\neg A) = 1$$

因此$(A_1 \wedge A_2 \wedge \cdots \wedge A_n) \to \neg\neg A$是永真式。 □

否定引入规则的特点是：**规则前提中不出现否定联结词，但是规则结论中有否定联结词**；而否定消除规则的特点是：**规则前提中出现了否定联结词，规则结论可消除其中的（某些）否定联结词**。根据这个特点，如何使用这两个规则就比较清楚：**当要得到带有否定联结词的推理形式时就使用否定引入联结词，当要消除推理形式中某些否定联结词时就使用否定消除规则**。从后面的例子可以看到，这是比较符合人们日常的推理方式的，而且下面要介绍的推理规则都有类似的特点。

否定引入规则的直观含义很简单，就是如果前提集Γ能推出A，当然它也能推出$\neg\neg A$。而否定消除规则的直观含义则与通常所说反证法相似：要证明Γ推出A，可假设A不成立，即将$\neg A$作为附加的前提，由原有的前提集Γ以及$\neg A$，既可推出B，又可推出$\neg B$，就意味着推出了矛盾，从而假设A不成立是不对的，因此由前提集Γ可推出A。所以，后面又称否定消除规则为**反证律**。

备注 3.2.4 为了统一起见，这里为每个联结词都给出了引入和消除规则，但实际上，否定引入规则可不作为基本规则，而作为否定消除规则的派生规则，后面将说明派生规则的含义及派生规则的证明。

定理 3.2.5 **合取引入规则**和**合取消除规则**：对任意的前提集Γ和任意公式A, B有

$$\text{合取引入}\ \frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B} \qquad\qquad \text{合取消除}\ \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A} \qquad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}$$

合取引入规则的直观含义是，如果Γ既能推出A又能推出B，则它能推出$A \wedge B$。而合取消除规则的直观含义则与之对应，如果从Γ能推出$A \wedge B$，则它既能推出A又能推出B。合取消除规则又称为**化简律**。

合取消除规则有两种形式：一种是从$A \wedge B$得到A，一种是从$A \wedge B$得到B。虽然说，根据等值演算我们知道$A \wedge B$等值于$B \wedge A$，但为了使得证明序列的构造过程的严格化、机械化和形式化，暂不涉及等值演算在构造证明序列过程中的应用，所以这里给出两种形式的化简规则。后面会通过例子进一步解释证明序列的构造过程。

定理 3.2.6 **析取引入规则**和**析取消除规则**：对任意的前提集Γ和任意公式A, B, C有

$$\text{析取引入}\ \frac{\Gamma \vdash A}{\Gamma \vdash A \vee B} \qquad \frac{\Gamma \vdash B}{\Gamma \vdash A \vee B} \qquad\qquad \text{析取消除}\ \frac{\Gamma \vdash A \vee B \qquad \Gamma, A \vdash C \qquad \Gamma, B \vdash C}{\Gamma \vdash C}$$

析取引入规则的直观含义是，如果Γ能推出A，则对任意的公式B，Γ能推出A或B。同理，若Γ能推出B，则对任意公式A，Γ都能推出A或B。这里析取引入规则同样有两种形式。析取引入规则又称为**附加律**。析取消除规则比较复杂，其直观含义是，如果Γ本身能推出A或B，而且Γ加上前提A能推出C，Γ加上前提B也能推出C，这意味着Γ直接就可推出C。

例如这样一个推理：假若我今天要么参加乒乓球比赛，要么参加羽毛球比赛，而参加乒乓球比赛我能得奖，参加羽毛球比赛我也能获奖，则我今天肯定获奖。这个推理就利用了析取消除规则：参加乒乓球比赛为p，参加羽毛球比赛为q，而能得奖为r，则$p \vee q$为真，且$p \to r$和$q \to r$都为真，则可得到r为真。这用推理形式结构表示就相当于从$\vdash p \vee q$，$p \vdash r$及$q \vdash r$得到$\vdash r$，即析取消除规则中Γ用空集带入，A用p代入，B用q代入，C用r代入的结果。后面将进一步讨论证明序列的构造以及自然语言推理的符号化与有效性验证。

直接使用析取引入和析取消除规则往往需要更多技巧，在实际应用时，更多地是使用由它们（当然再加上其他一些规则）派生的**析取三段论规则**。

定理 3.2.7 **蕴涵引入规则**和**蕴涵消除规则**：对任意的前提集Γ和任意公式A, B有

$$\text{蕴涵引入}\ \frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B} \qquad \text{蕴涵消除}\ \frac{\Gamma \vdash A \to B \qquad \Gamma \vdash A}{\Gamma \vdash B}$$

蕴涵引入规则的直观含义是，如果Γ加上前提A能推出B，则Γ本身可推出$A \to B$，这在某种意义上说出了推理形式结构的前提与蕴涵式前件之间的密切联系。这也给出了人们日常生活中常见的一种推理方式，**附加前提法**：即如果要证明的结论是蕴涵式$A \to B$，通常将其前件A作为附加前提，通过证明由Γ加上A可推出B，而证明Γ可推出$A \to B$。

蕴涵消除规则的直观含义是，如果Γ可推出A蕴涵B，且Γ可推出A，则Γ可推出B，简单地说，就是由A蕴涵B，以及A可推出B。这在某种意义上说出了“推出”和“蕴涵”之间的密切联系，因此是逻辑推理中最重要的推理规则，有许多别名，在普通逻辑学中称为**假言推理规则**，而在公理化命题演算系统中称为**分离规则**。

定理 3.2.8 **等价引入规则**和**等价消除规则**：对任意的前提集Γ和任意公式A, B有

$$\text{等价引入}\ \frac{\Gamma \vdash A \to B \qquad \Gamma \vdash B \to A}{\Gamma \vdash A \leftrightarrow B} \qquad \text{等价消除}\ \frac{\Gamma \vdash A \leftrightarrow B}{\Gamma \vdash A \to B} \qquad \frac{\Gamma \vdash A \leftrightarrow B}{\Gamma \vdash B \to A}$$

上述规则给出了等价联结词的基本性质。等价引入规则的直观含义是，如果Γ可推出A蕴涵B，也可推出B蕴涵A，则可推出A与B等价。等价消除规则的含义则与之对应，如果Γ可推出A与B等价，则可推出A蕴涵B和B蕴涵A。

为方便证明序列的构造，最后还引入一条与联结词无关的规则，称为**弱化规则**：

定理 3.2.9 **弱化规则**：对任意的前提集Γ和任意公式A, B有：

$$\text{弱化规则}\ \frac{\Gamma \vdash A}{\Gamma, B \vdash A}$$

弱化规则的直观含义是，如果Γ已经能推出A，那么再多加一个前提B，照样能推出A。之所以称为弱化规则，是从某种角度上来说，多一个前提进行推理当然比少一个前提进行推理要简单一些，因而弱化了推理。

下表总结了上述规则，其中规则名称中还列出了该规则的一些别名：

规则名称	规则模式	规则名称	规则模式
前提引入	$\dfrac{}{\Gamma, A, \Delta \vdash A}$	弱化规则	$\dfrac{\Gamma \vdash A}{\Gamma, B \vdash A}$
否定引入	$\dfrac{\Gamma \vdash A}{\Gamma \vdash \neg\neg A}$	否定消除 反证律	$\dfrac{\Gamma, \neg A \vdash B \qquad \Gamma, \neg A \vdash \neg B}{\Gamma \vdash A}$
合取引入	$\dfrac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B}$	合取消除 化简律	$\dfrac{\Gamma \vdash A \wedge B}{\Gamma \vdash A} \qquad \dfrac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}$
析取引入 附加律	$\dfrac{\Gamma \vdash A}{\Gamma \vdash A \vee B} \qquad \dfrac{\Gamma \vdash B}{\Gamma \vdash A \vee B}$	析取消除	$\dfrac{\Gamma \vdash A \vee B \qquad \Gamma, A \vdash C \qquad \Gamma, B \vdash C}{\Gamma \vdash C}$
蕴涵引入	$\dfrac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}$	蕴含消除 假言推理 分离规则	$\dfrac{\Gamma \vdash A \to B \qquad \Gamma \vdash A}{\Gamma \vdash B}$
等价引入	$\dfrac{\Gamma \vdash A \to B \qquad \Gamma \vdash B \to A}{\Gamma \vdash A \leftrightarrow B}$	等价消除	$\dfrac{\Gamma \vdash A \leftrightarrow B}{\Gamma \vdash A \to B} \qquad \dfrac{\Gamma \vdash A \leftrightarrow B}{\Gamma \vdash B \to A}$

总的来说，自然推理系统共提供十二条规则，其中前提引入规则和弱化规则与联结词无关，其他是个规则分别是命题联结词$\neg, \wedge, \vee, \to, \leftrightarrow$的引入和消除规则。要提醒读者的是，上面给出这些规则都强调了其中的前提集Γ，及公式A, B, C都是任意的，在真正利用这些规则构造证明序列时，应用具体的前提集和具体的公式代入。因此，与前面的基本等值式一样，上述规则都是规则模式，在真正使用的时候都需要隐含地**代入规则**。

3.2.2 证明序列的构造

给出自然推理系统的推理规则之后，我们可以给出证明序列的严格定义：

定义 3.2.10 **证明序列**是推理形式结构的有穷序列：

$$\begin{aligned}
&\Gamma_1 \vdash A_1\\
&\Gamma_2 \vdash A_2\\
&\vdots \quad \vdots\\
&\Gamma_n \vdash A_n
\end{aligned}$$

其中$\Gamma_1 \vdash A_1$是前提引入规则的代入实例，也即$A_1 \in \Gamma_1$，而对于任意的$2 \le i \le n$满足：

(1) $\Gamma_i \vdash A_i$是前提引入规则的代入实例，即$A_i \in \Gamma_i$，或者，

(2) $\Gamma_i \vdash A_i$是由排在它前面的推理形式结构根据自然推理系统的除前提引入规则之外的十一条规则之一得到的，即存在$j_1, j_2, \cdots, j_m$，这里对任意的$1 \le k \le m$都有$1 \le j_k < i$，使得：

$$\frac{\Gamma_{j_1} \vdash A_{j_1} \qquad \Gamma_{j_2} \vdash A_{j_2} \qquad \cdots \qquad \Gamma_{j_m} \vdash A_{j_m}}{\Gamma_i \vdash A_i}$$

是这十一条规则中某一条规则的代入实例。

我们称上述证明序列是推理$\Gamma_n \vdash A_n$的有效性的**证明**。相应地，如果推理$\Gamma \vdash A$存在它的有效性的证明，则称它是**有效的推理**。

简单地，要在自然推理系统中判断推理$\Gamma \vdash A$是否有效，就是要构造一个证明序列使得该序列的最后一个推理形式结构就是$\Gamma \vdash A$。下面的例子给出了一个推理的证明序列：

例子 3.2.11 $p \to q, q \to r \vdash p \to r$ 是有效的推理，证明它的证明序列如下：

(1)	$p \to q,\ q \to r,\ p \vdash p$	// 前提引入
(2)	$p \to q,\ q \to r,\ p \vdash p \to q$	// 前提引入
(3)	$p \to q,\ q \to r,\ p \vdash q$	// (1), (2)蕴涵消除
(4)	$p \to q,\ q \to r,\ p \vdash q \to r$	// 前提引入
(5)	$p \to q,\ q \to r,\ p \vdash r$	// (3), (4)蕴涵消除
(6)	$p \to q,\ q \to r \vdash p \to r$	// (5)蕴含引入

这个证明序列严格符合证明序列的定义3.2.10。设$\Gamma = \{p \to q,\ q \to r\}$，则：

(1) 证明序列的第一个推理形式结构是$\Gamma, p \vdash p$，它是前提引入规则的代入实例，因为$p \in \Gamma \cup \{p\}$；

(2) 同样第二个推理形式结构$\Gamma, p \vdash p \to q$也是前提引入规则的代入实例，因为$p \to q \in \Gamma$；

(3) 第三个推理形式结构$\Gamma, p \vdash q$，它是由排在它前面的两个推理形式结构(1)和(2)根据蕴消除规则得到，因为：

$$\frac{\Gamma, p \vdash p \to q \qquad \Gamma, p \vdash p}{\Gamma, p \vdash q}$$

是蕴涵消除规则

$$\frac{\Delta \vdash A \to B \qquad \Delta \vdash A}{\Delta \vdash B}$$

的代入实例，其中Δ用前提集$\Gamma \cup \{p\}$代入，而A用p代入，B用q代入。

注意，因为前面规则中的Γ, A, B, C都分别代表任意的前提集和公式，因此改用其他符号并无关系，这里为方便起见，上面给出蕴涵消除规则时用Δ而不是原先的Γ。

这里的关键是，在考虑规则的代入实例时，应该用具体的前提集代入规则中某个表示前提集的符号的**所有出现**，也应该用具体的公式代入规则中某个表示公式的符号的**所有出现**，不能将规则中的相同的符号用不同的前提集或公式集代入。例如，上面用$\Gamma \cup \{p\}$代入Δ，就应该代入上述规则中前提和结论中出现的每一个Δ，同样，用p代入A，就应该代入前提和结论中出现的每一个A。

(4) 第四个推理形式结构$\Gamma, p \vdash q \to r$也是前提引入规则的代入实例，因为$q \to r \in \Gamma$；

(5) 第五个推理形式结构$\Gamma, p \vdash r$是通过推理形式结构(3)和(4)根据蕴涵消除规则得到，因为：

$$\frac{\Gamma, p \vdash q \to r \qquad \Gamma, p \vdash q}{\Gamma, p \vdash r}$$

是蕴涵消除规则的代入实例，其中用$\Gamma \cup \{p\}$代入规则中的Δ，用q代入A，用r代入B。

(6) 第五个推理形式结构$\Gamma \vdash p \to r$是通过推理形式结构(5)根据蕴涵引入规则得到，因为：

$$\frac{\Gamma, p \vdash r}{\Gamma \vdash p \to r}$$

是蕴涵引入规则

$$\frac{\Delta, A \vdash B}{\Delta \vdash A \to B}$$

的代入实例，由Γ代入Δ，p代入A，r代入B而得到。

简单地说，按照证明序列中推理形式结构本身的排列顺序，第一个推理形式结构必须是前提引入规则的代入实例，而从第二个开始，可能是前提引入规则的代入实例，也可能是其他十一个规则中某个规则的**结论的代入实例**，而且排在它前面的推理形式结构中**存在该规则所有前提的代入实例**。当然该规则作为一个整体，**规则前提和规则结论中的相同符号要用相同的前提集或公式代入**，也正是通过这一点，证明序列中各个推理形式结构之间具有了密切的联系。

但是对于如何构造证明一个推理的有效性的证明序列来说，通常并不是按照证明序列中推理形式结构本身的排列顺序来思考的，而是按照其相反的顺序来思考。例如对于上述例子中的证明序列，如果从第一个推理形式结构$p \to q,\ q \to r, p \vdash p$看，初学者很难明白为什么将它排在第一位，它与要验证的推理$p \to q,\ q \to r \vdash p \to r$有什么关系？这两个推理形式结构的前提集为什么不相同？

实际上，上述证明序列是按照**倒推**的方式构造的。因为要验证的推理是$p \to q,\ q \to r \vdash p \to r$，因此它的证明序列最后一个推理形式结构必须是它本身，也就是说，首先我们有如下的证明序列：

$$(?) \quad p \to q,\ q \to r \vdash p \to r$$

这个推理形式结构的最大特点是什么？它的结论是一个蕴涵式。前面所给出的规则中，哪一个规则的结论是蕴涵式？只有蕴涵引入规则，因此这个推理形式结构必然是由蕴涵引入规则得到的，对比蕴涵引入规则

$$\frac{\Delta, A \vdash B}{\Delta \vdash A \to B}$$

的结论，很容易看到，需要使用$\Gamma = \{p \to q,\ q \to r\}$代入$\Delta$，$p$代入$A$，$r$代入$B$而得到上面推理形式结构，根据这个代入，为了使用蕴涵引入规则，证明序列的前面需要有该规则前提的代入实例，因此我们有了如下的证明序列：

$$\begin{array}{ll} (?) & p \to q,\ q \to r, p \vdash r \\ (?) & p \to q,\ q \to r \vdash p \to r \qquad // \text{上一个推理形式结构使用蕴涵引入规则} \end{array}$$

现在我们需要验证$p \to q,\ q \to r, p \vdash r$，再看它的结论$r$出现在前提什么地方？出现在蕴涵式$q \to r$的后件，因此很容易想到使用蕴涵消除规则来得到这个推理形式结构，对比蕴涵消除规则

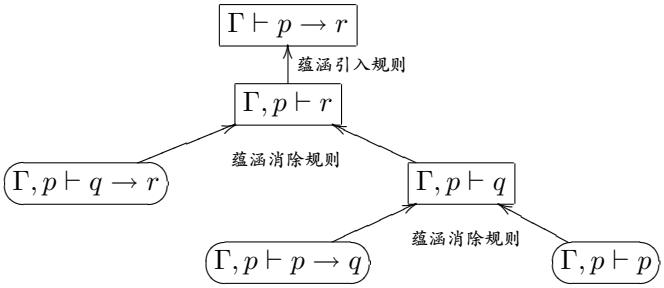
$$\frac{\Delta \vdash A \to B \qquad \Delta \vdash A}{\Delta \vdash B}$$

的结论，很容易看到，Δ应该用$p \to q,\ q \to r, p = \Gamma \cup \{p\}$代入，$B$用$r$代入，结合该规则的前提知道$A$应该用$q$代入。根据这个代入，为了使用蕴涵消除规则，证明序列的前面需要有该规则两个前提的代入实例，因此我们有了如下的证明序列：

$$\begin{array}{ll} (?) & p \to q,\ q \to r,\ p \vdash q \\ (?) & p \to q,\ q \to r,\ p \vdash q \to r \\ (?) & p \to q,\ q \to r, p \vdash r \qquad // \text{上两个推理形式结构使用蕴涵消除规则得到} \\ (?) & p \to q,\ q \to r \vdash p \to r \qquad // \text{上一个推理形式结构使用蕴涵引入规则} \end{array}$$

现在对于最前面两个推理形式结构，显然第二个是前提引入规则的代入实例，因此只要验证$p \to q,\ q \to r,\ p \vdash q$，通过与上面类似的分析，就不难得到上述例子中给出的整个证明序列，再整理一下它们的顺序，加上编号，写出注释即可。

显然上述构造证明序列的过程是一个**分析**过程，**从要验证的推理形式结构出发，分析它应该由什么规则得到，再由该规则的前提，分析如何得到这些前提，一步一步进行逆推，直到最后需要的都是前提引入规则的代入实例**。实际上，这也是一个**自顶向下**分解过程，从要验证的推理形式结构出发，根据所要用的规则进行分解，直到最后要使用的是前提引入规则。例如上述证明序列的构造可用如下的分解树描述（下面同样令$\Gamma=\{p\to q,\ q\to r\}$）：



上图中，根是要验证的推理，叶子节点都是前提引入规则的代入实例，内部节点是在验证推理时需要由规则得到的中间推理形式结构。在实际应用中，也有用如下方式来给出构造验证推理的证明序列的过程，或者直接用如下形式验证推理的有效性：

$$\dfrac{\dfrac{\dfrac{\Gamma,p\vdash p\to q \qquad \Gamma,p\vdash p}{\Gamma,p\vdash q}\ \text{蕴涵消除} \qquad \Gamma,p\vdash q\to r}{\Gamma,p\vdash r}\ \text{蕴涵消除}}{\Gamma\vdash p\to r}\ \text{蕴涵引入}$$

从上述证明序列的构造我们可以看到，在自然推理系统中利用规则验证推理的有效性与人们日常生活中的一些逻辑推理是十分相似的，这也是自然推理系统之所以称为“自然推理”的原因。

备注 3.2.12 上面详细介绍了有前提形式的证明序列的构造，如果证明序列采用下面省略前提的形式：

$$\begin{array}{l} A_1\\ A_2\\ \vdots\\ A_n \end{array}$$

这时在构造证明序列时**只能采用规则前提和结论都不改变推理形式结构本身的前提的规则**。对于上面给出的十二条规则：

(1). **前提引入规则**仍然适用，即如果某公式属于要验证推理的前提集中，则可直接出现在证明序列中。实际上，这种形式的证明序列，第一个公式一定是要验证推理的前提中的某个公式；

(2). 这种形式的证明序列**不会用到弱化规则**；

(3). 对于**否定引入、合取引入、合取消除（化简律）、析取引入（附加律）、蕴涵消除（假言推理、分离规则）、等价引入和等价消除**这几个规则，它们的前提和结论的推理形式结构本身的前都集相同，都是Γ。当采用省略前提形式的证明序列时，约定**上述证明序列中的公式总是由要验证的推理的**

前提集推出，从而这些规则可仅仅看作推理形式结构本身的结论之间的推出。例如，对于合取引入规则

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B}$$

规则前提和结论中的推理形式结构都只涉及到Γ这个前提集，而且没有变化，因此可看作如下的规则：

$$\frac{A \qquad B}{A \wedge B}$$

其直观含义是，如果（从要验证的推理形式结构的前提集）得到A，又得到B，那么我们可以得到$A \wedge B$。其他没有改变前提集的规则也可作类似的解读。

(4). 对于否定消除（反证律）、析取消除和蕴涵引入规则这些涉及的前提集变化的规则，则在这种证明序列中通常不使用析取消除规则，而蕴涵引入和否定引入规则都要使用附加前提证明法，蕴涵引入将要验证的推理结论的蕴涵式的前件作为附加前提引入，而否定引入则将要验证的推理结论的否定作为附加前提引入而推出矛盾，在这两种情况下，证明序列中都要对这种附加前提做特别的说明。

当采用省略前提形式的证明序列时，验证推理$p \to q,\ q \to r \vdash p \to r$的证明序列如下：

(1)	p	// 附加前提引入
(2)	$p \to q$	// 前提引入
(3)	q	// (1), (2)蕴涵消除
(4)	$q \to r$	// 前提引入
(5)	r	// (3), (4)蕴涵消除
(6)	$p \to r$	// 消除(1)引入的附加前提

这种证明序列从形式上来看稍简单一些，但是却不够严谨。比较这个证明序列和前面带前提集的证明序列，本质上是一样的。有兴趣的读者可参考耿素云等编写的教材[3]作进一步的比较和学习。

下面再使用一个例子说明证明序列的构造：

例子 3.2.13 验证推理$q \to p,\ q \leftrightarrow s,\ s \leftrightarrow t,\ t \wedge r \vdash p \wedge q$的有效性。

分析：要验证的推理的结论部分是合取式，因此容易想到，最后应该使用合取引入规则，也即，最后应该是如下的推理形式结构序列：令$\Gamma = \{q \to p,\ q \leftrightarrow s,\ s \leftrightarrow t, t \wedge r\}$，

(?)	$\Gamma \vdash p$	
(?)	$\Gamma \vdash q$	
(?)	$\Gamma \vdash p \wedge q$	// 前两个推理形式结构使用合取引入规则

如何从Γ推出p? 我们看到Γ中$q \to p$，因此根据蕴涵消除规则，我们可有如下的证明序列：

(?)	$\Gamma \vdash q$	
(?)	$\Gamma \vdash q \to p$	// 前提引入
(?)	$\Gamma \vdash p$	// 前两个推理形式结构使用蕴涵消除规则
(?)	$\Gamma \vdash p \wedge q$	// 合取引入

如何从Γ推出q? 我们看到q还出现在前提集的公式$q \leftrightarrow s$中，而根据等价消除规则，这使得Γ 可推出$q \to s$和$s \to q$，显然我们需要后面一个公式再利用蕴涵消除规则得到q，从而得到证明序列：

(?) $\Gamma \vdash s$

(?) $\Gamma \vdash q \leftrightarrow s$ // 前提引入

(?) $\Gamma \vdash s \to q$ // 上一个推理形式结构使用等价消除规则

(?) $\Gamma \vdash q$ // 蕴涵消除

(?) $\Gamma \vdash q \to p$ // 前提引入

(?) $\Gamma \vdash p$ // 蕴涵消除

(?) $\Gamma \vdash p \wedge q$ // 合取引入

现在问题变成如何由Γ推出s，同样看到s还出现在$s \leftrightarrow t$中，这可得到$t \to s$，而由合取消除规则从前提$t \wedge r$可得到t，因此最后的证明序列应该是：

(1) $\Gamma \vdash t \wedge r$ // 前提引入

(2) $\Gamma \vdash t$ // (1)合取消除

(3) $\Gamma \vdash s \leftrightarrow t$ // 前提引入

(4) $\Gamma \vdash t \to s$ // (3)等价消除

(5) $\Gamma \vdash s$ // (2),(4)蕴涵消除

(6) $\Gamma \vdash q \leftrightarrow s$ // 前提引入

(7) $\Gamma \vdash s \to q$ // (6)等价消除

(8) $\Gamma \vdash q$ // (5),(7)蕴涵消除

(9) $\Gamma \vdash q \to p$ // 前提引入

(10) $\Gamma \vdash p$ // (8),(9)蕴涵消除

(11) $\Gamma \vdash p \wedge q$ // (8),(10)合取引入

备注 3.2.14 由于上面例子的证明序列中所有推理形式结构的前提集都是Γ，因此若采用省略前提集的证明序列与上面的证明序列基本相同：

(1) $t \wedge r$ // 前提引入

(2) t // (1)合取消除

(3) $s \leftrightarrow t$ // 前提引入

(4) $t \to s$ // (3)等价消除

(5) s // (2),(4)蕴涵消除

(6) $q \leftrightarrow s$ // 前提引入

(7) $s \to q$ // (6)等价消除

(8) q // (5),(7)蕴涵消除

(9) $q \to p$ // 前提引入

(10) p // (8),(9)蕴涵消除

(11) $p \wedge q$ // (8),(10)合取引入

总的来说，证明序列用于验证某个推理的有效性，它是从前提引入规则的代入实例出发，一步一步对自然推理系统的规则进行代入，从规则前提的代入实例得到规则结论的代入实例，最后得到所要验证的推理。由于**自然推理系统的规则都是保真的**，即若前提都对应永真式，则规则的结论也对应永真式，从而得到所要验证的推理对应永真式，也即该推理是有效的。

构造验证推理有效性的证明序列通常采用自顶向下的分析方法：分析待验证的推理的结论，根据它的特点确定应使用的规则，根据该规则的前提确定要进一步验证的推理，如此逆推，直到所有要验证的推理都是前提引入规则的代入实例。在构造证明序列时，我们应该**写出每一步所采用的推理规则，并标出该规则所依赖的前提的编号**，同时读者还应该清楚是**如何对规则进行代入而得到每一步的推理形式结构的**。

3.2.3 自然推理的派生规则

前面提到，自然推理系统的十二条基本规则中有些规则，如析取消除规则比较难以使用，因此需要由这些基本规则派生一些更为实用的规则。这里首先想明确；两个问题：(i). 为什么称前面十二条规则为**基本规则**，而这一节介绍的规则为**派生规则**？(ii). 怎样保证派生规则的正确性？

实际上，前面给出的十二条规则有十条都是常见命题联结词的引入和消除规则，体现了这些命题联结词在推理方面的最基本性质，而且它们都只涉及一个联结词，这个联结词还只在一处出现（除否定消除规则外），下面的派生规则则可能涉及多个联结词，从这一点可以看出，将命题联结词的引入和消除规则作为基本规则是有道理的。

前提引入规则毋庸置说是最基本的规则，它说的是进行推理的出发点，即从前提出发进行推理。弱化规则实际上则体现了经典命题逻辑推理的基本特性：**命题逻辑推理是单调推理**，即增加新的前提之后，原来已经推出的结论仍有效。与之相对的是**非单调推理**：增加新的前提之后可能推翻原来推出的结论。后面我们将看到，弱化规则实际上可只用在派生规则正确性的证明中，在具体的推理的有效性验证中，我们总是可是事先确定所有需要的前提，从而无需使用弱化规则。

备注 3.2.15 弱化规则本身实际上也可作为派生规则。上述自然推理系统实际上是以形式化方法归纳定义了哪些推理形式结构是有效的推理，归纳基是前提引入规则，而否定消除规则、合取引入和消除规则、析取引入和消除规则、蕴含引入和消除规则、等价引入和消除规则都是归纳步，即如果我们得到这些规则前提中的推理形式结构（的代入实例）是有效的，则其结论（的代入实例）也是有效的推理。最后，所有有效的推理都是这样得到的（最小化）。利用这个归纳定义，可以归纳地证明弱化规则的保真性。

进一步，正如根据命题公式语法归纳定义可给出每一个命题公式的语法树，利用这个归纳定义，也可将每个证明序列转换成一棵树，树叶都是前提引入规则，而对于每个内部节点，以它的所有儿子为前提，它自己为结论则构成了某个规则的代入实例，最后，树根就是要验证的推理（即证明序列的最后一个推理形式结构）。因此，也可以说，对于自然推理系统而言，一个推理形式结构是有效的推理当且仅当存在一棵以它为根的这种证明树。

我们知道，**规则的正确性即它的保真性**，即如果前提都是永真式，那么结论也是永真式。例如

对于析取消除规则：

$$\frac{\Gamma \vdash A \vee B \qquad \Gamma, A \vdash C \qquad \Gamma, B \vdash C}{\Gamma \vdash C}$$

设$\Gamma = \{A_1, A_2, \cdots, A_n\}$，不难验证当：

$$(A_1 \wedge A_2 \wedge \cdots \wedge A_n) \to (A \vee B)$$
$$(A_1 \wedge A_2 \wedge \cdots \wedge A_n \wedge A) \to C$$
$$(A_1 \wedge A_2 \wedge \cdots \wedge A_n \wedge B) \to C$$

都是永真式时，$(A_1 \wedge A_2 \wedge \cdots \wedge A_n) \to C$也是永真式。可以通过这种方式证明基本规则的正确性，但对于派生规则，我们可以根据基本规则的正确性，使用类似构造证明序列的方式验证其正确性。设有派生规则：

$$\frac{\Gamma_1 \vdash A_1 \qquad \Gamma_2 \vdash A_2 \qquad \cdots \qquad \Gamma_n \vdash A_n}{\Gamma \vdash A}$$

如果存在推理形式结构的序列：

$$\Delta_1 \vdash B_1$$
$$\Delta_2 \vdash B_2$$
$$\vdots \quad \vdots$$
$$\Delta_m \vdash B_m$$

使得$\Delta_m = \Gamma$, $B_m = A$，而且满足，对任意的$1 \le i \le m$有下列情况之一成立：

1. 存在$1 \le j \le n$使得$\Delta_i = \Gamma_j$且$B_i = A_j$，即该推理形式结构是上述派生规则的前提之一；
2. $B_i \in \Delta_i$，即它是前提引入规则的代入实例；
3. $\Delta_i \vdash B_i$由排在它前面的推理形式结构根据**基本规则或已经证明过的派生规则**得到，即存在$j_1, j_i, \cdots, j_k$，这里对任意的$1 \le p \le k$有$1 \le j_p < i$，使得

$$\frac{\Delta_{j_1} \vdash B_{j_1} \qquad \Delta_{j_2} \vdash B_{j_2} \qquad \cdots \qquad \Delta_{j_k} \vdash B_{j_k}}{\Delta_i \vdash B_i}$$

是某个基本规则或已经证明过的派生规则的代入实例。

注意，因为派生规则本身实际上也是规则模式，所以**这时的代入不是以具体的前提集或公式代入**，而仍然是以代表前提集的符号代入基本规则或已证明过的派生规则中出现的前提集，以代表公式的符号代入基本规则或已证明过的派生规则中出现的公式。但这与以具体前提集或公式代入本质上是相同的，也都要用同样的符号代入某个符号的所有出现。

下面在给出和验证派生规则的正确性时都使用这种证明序列，某个派生规则一旦验证是正确之后，在后面就可以用于验证其他派生规则或具体的推理。

例子 3.2.16 第一个重要的派生规则与例子3.2.11中所验证的推理十分相似，我们称其为**传递规则** 或**假言三段论**：

$$\text{传递规则} \qquad \frac{\Gamma \vdash A \to B \qquad \Gamma \vdash B \to C}{\Gamma \vdash A \to C}$$

验证它的证明序列如下：

(1) $\Gamma \vdash A \to B$ // 待验证规则的前提
(2) $\Gamma, A \vdash A \to B$ // (1)弱化规则
(3) $\Gamma, A \vdash A$ // 前提引入规则
(4) $\Gamma, A \vdash B$ // (2),(3)蕴涵消除
(5) $\Gamma \vdash B \to C$ // 待验证规则的前提
(6) $\Gamma, A \vdash B \to C$ // (5)弱化规则
(7) $\Gamma, A \vdash C$ // (4),(6)蕴涵消除
(8) $\Gamma \vdash A \to C$ // (7)蕴涵引入

这个证明序列与例子3.2.11的证明序列也十分相似，只是在例子3.2.11使用的是具体前提集和具体公式的证明序列，而这里是前提集符号和公式符号的证明序列，但本质上是相同的，构造的思路也相同，例如第二个推理形式结构使用弱化规则引入A的原因就是规则结论中含有蕴涵式$A \to C$。

这个证明序列与例子3.2.11所用证明序列的主要区别是弱化规则的使用。例子3.2.11的前提集含有公式$p \to q$，因此在前提集上附加公式p就可直接得到q。但对于上述派生规则而言，我们只有$\Gamma \vdash A \to B$，这并不意味着$A \to B$在前提集Γ中，而要推出公式B，必须先得到A，这只能在前提中引入。但是注意下面不是蕴涵消除规则的代入形式：

$$\frac{\Gamma \vdash A \to B \qquad \Gamma, A \vdash A}{\Gamma \vdash B}$$

因为这使得蕴涵消除规则

$$\frac{\Delta \vdash C \to D \qquad \Delta \vdash C}{\Delta \vdash D}$$

中的前提集Δ要使用不同的符号代入。因此这里我们需要使用**弱化规则**使得前提集相同，从而利用蕴涵消除规则下面的代入形式来构造证明序列：

$$\frac{\Gamma, A \vdash A \to B \qquad \Gamma, A \vdash A}{\Gamma, A \vdash B}$$

读者应该从这个例子进一步理解如何对规则进行代入，**既要能熟练地使用具体的前提集和公式代入，又要能使用前提集符号和公式符号进行代入**。另一方面，读者也应领会弱化规则的使用：**弱化规则常常用于改变前提集以满足使用其他规则的需要**。

例子 3.2.17 使用传递规则，我们可得到验证例子3.2.13中的推理$q \to p,\ q \leftrightarrow s,\ s \leftrightarrow t,\ t \wedge r \vdash p \wedge q$的另一个证明序列，令$\Gamma = \{q \to p,\ q \leftrightarrow s,\ s \leftrightarrow t,\ t \wedge r\}$：

(1) $\Gamma \vdash q \leftrightarrow s$ // 前提引入
(2) $\Gamma \vdash s \to q$ // (1)等价消除
(3) $\Gamma \vdash s \leftrightarrow t$ // 前提引入
(4) $\Gamma \vdash t \to s$ // (3)等价消除

(5) $\Gamma \vdash t \to q$ // (2),(4)传递规则
(6) $\Gamma \vdash t \wedge r$ // 前提引入
(7) $\Gamma \vdash t$ // 合取消除
(8) $\Gamma \vdash q$ // (5),(7)蕴涵消除
(9) $\Gamma \vdash q \to p$ // 前提引入
(10) $\Gamma \vdash p$ // (8),(9)蕴涵消除
(11) $\Gamma \vdash p \wedge q$ // (8),(10)合取引入

例子 3.2.18 第二个重要的派生规则是**双重否定律**：

$$\text{双重否定律} \quad \frac{\Gamma \vdash \neg\neg A}{\Gamma \vdash A}$$

验证它的证明序列如下：

(1) $\Gamma \vdash \neg\neg A$ // 待验证规则的前提
(2) $\Gamma, \neg A \vdash \neg\neg A$ // (1)弱化规则
(3) $\Gamma, \neg A \vdash \neg A$ // 前提引入规则
(4) $\Gamma \vdash A$ // (2),(3)否定消除

对于这个证明序列，为什么一开始引入$\neg A$作为前提呢？因为规则前提$\Gamma \vdash \neg\neg A$本身有否定联结词，因此应该可以想到使用否定消除规则，从而可以得到应该在前提中引入$\neg A$。注意最后规则结论的得到是利用否定消除规则

$$\frac{\Delta, \neg B \vdash C \qquad \Delta, \neg B \vdash \neg C}{\Delta \vdash B}$$

的如下代入形式：

$$\frac{\Gamma, \neg A \vdash \neg A \qquad \Gamma, \neg A \vdash \neg\neg A}{\Gamma \vdash A}$$

其中Δ用Γ代入，B用A代入，C用$\neg A$代入。

备注 3.2.19 否定引入规则也可作为否定消除规则的派生规则：

$$\text{否定引入} \quad \frac{\Gamma \vdash A}{\Gamma \vdash \neg\neg A}$$

验证它的证明序列如下：

(1) $\Gamma, \neg\neg\neg A \vdash \neg\neg\neg A$ // 前提引入规则
(2) $\Gamma, \neg\neg\neg A \vdash \neg A$ // (1),上面的双重否定律
(3) $\Gamma \vdash A$ // 待验证规则的前提
(4) $\Gamma, \neg\neg\neg A \vdash A$ // (3)弱化规则
(5) $\Gamma \vdash \neg\neg A$ // (2),(4)否定消除

例子 3.2.20 第三个重要的派生规则仍然与否定联结词有关，其本质含义与否定消除类似，我们称为**归谬律**：

$$\text{归谬律}\quad \frac{\Gamma, A \vdash B \qquad \Gamma, A \vdash \neg B}{\Gamma \vdash \neg A}$$

验证它的证明序列如下：

(1)	$\Gamma, A \vdash B$	// 待验证规则的前提
(2)	$\Gamma \vdash A \to B$	// (1)蕴涵引入
(3)	$\Gamma, \neg\neg A \vdash A \to B$	// (2)弱化规则
(4)	$\Gamma, \neg\neg A \vdash \neg\neg A$	// 前提引入规则
(5)	$\Gamma, \neg\neg A \vdash A$	// (4)双重否定律
(6)	$\Gamma, \neg\neg A \vdash B$	// (3), (5)蕴涵消除
(7)	$\Gamma, A \vdash \neg B$	// 待验证规则的前提
(8)	$\Gamma \vdash A \to \neg B$	// (7)蕴涵引入
(9)	$\Gamma, \neg\neg A \vdash A \to \neg B$	// (8)弱化规则
(10)	$\Gamma, \neg\neg A \vdash \neg B$	// (5), (9)蕴涵消除
(11)	$\Gamma \vdash \neg A$	// (6),(10)否定消除

构造上述证明序列的思路是，因为最后要得到$\Gamma \vdash \neg A$，仍考虑使用否定消除规则，则需要从前提$\Gamma, \neg\neg A$推出矛盾，当然这个矛盾也只能是同时推出B和$\neg B$。因为规则前提给出了Γ, A可推出B和$\neg B$，而我们知道A与$\neg\neg A$等值，因此直觉上可换成$\Gamma, \neg\neg A$也可同时推出B和$\neg B$，这可利用双重否定律实现。

可能有的读者想根据否定消除规则：

$$\frac{\Gamma, \neg A \vdash B \qquad \Gamma, \neg A \vdash \neg B}{\Gamma \vdash A}$$

使用A代入上述$\neg A$而得到

$$\frac{\Gamma, A \vdash B \qquad \Gamma, A \vdash \neg B}{\Gamma \vdash \neg\neg A}$$

但这不是代入，**代入只能用公式去代入表示公式的符号，而不能用公式或表示公式的符号去代入公式**。可以看到，上述形式的结论$\Gamma \vdash \neg\neg A$不可能是$\Gamma \vdash A$的代入实例。而用$\neg A$去代入否定消除规则中的A就没有问题，我们将得到：

$$\frac{\Gamma, \neg\neg A \vdash B \qquad \Gamma, \neg\neg A \vdash \neg B}{\Gamma \vdash \neg A}$$

这也正是上面证明序列最后一步所使用的代入形式。但这也不能简单地根据A与$\neg\neg A$等值直接由$\Gamma, \neg\neg A \vdash B$得到$\Gamma, A \vdash B$，因为没有这种规则。

因此，**证明序列的构造应该严格根据规则，按照代入的方式来进行**，而不能凭想像，或凭我们知道的公式之间的等值来对规则做随意的变换，公式之间的等值只能启发我们去构造证明序列，而不能乱用。

我们目前**没有**这样的规则：由公式A与B等值，就可从$\Gamma, A \vdash C$得到$\Gamma, B \vdash C$，或者从$\Gamma \vdash A$就可得到$\Gamma \vdash B$。虽然这种规则是可以证明的（但证明十分复杂，而且要建立在对证明序列构造的归纳定义基础上），但引入这种规则，在证明序列构造中随意使用等值公式替换，就会因为与公式等值的公式有无数多，就很难让初学者找到如何构造证明序列的思路与方向，从而更容易出错。

理解这种严格根据规则，按照代入的方式构造证明序列的方法对于计算机专业的学生来说也十分重要，因为这与程序的执行过程本质上是一样的：程序的执行就是按照预定的基本指令一步一步对内存存放的二进制数据进行严格、机械化的变换。或者可从另一角度理解上述构造证明序列的方式与程序执行方式的相同性，即**我们完全可以编写一个程序，验证证明序列的构造是否是按照规则进行的**，因为构造证明序列过程中对规则的代入也是严格和机械化的过程。

例子 3.2.21 下面再给出一个与否定联结词有关的派生规则，称为**矛盾律**：

$$\text{矛盾律}\quad \frac{\Gamma \vdash A \qquad \Gamma \vdash \neg A}{\Gamma \vdash B}$$

矛盾律的直观含义就是，如果前提集Γ既能推出A又能推出$\neg A$，即A的否定，那么Γ能推出任意的公式B，也就是说矛盾的前提（假的前提）能推出任何东西，这是经典命题逻辑的一个基本特点。验证这个推理的证明序列很简单：

(1)	$\Gamma \vdash A$	// 待验证规则的前提
(2)	$\Gamma, \neg B \vdash A$	// (1)弱化规则
(3)	$\Gamma \vdash \neg A$	// 待验证规则的前提
(4)	$\Gamma, \neg B \vdash \neg A$	// (3)弱化规则
(5)	$\Gamma \vdash B$	// (2),(4)否定消除

至此为此，我们给出了有关否定联结词的一些派生规则，包括双重否定律、反证法和矛盾律，这些是在实际推理中使用得比较多的推理规则。前面提到，析取消除规则是一个比较难以运用的规则，通常是使用与析取联结词有关的派生规则，特别是析取三段论，下面我们介绍这个派生规则：

例子 3.2.22 下面给出**析取三段论**规则的两种基本形式：

$$\text{析取三段论}\quad \frac{\Gamma \vdash A \vee B \qquad \Gamma \vdash \neg A}{\Gamma \vdash B} \qquad \frac{\Gamma \vdash A \vee B \qquad \Gamma \vdash \neg B}{\Gamma \vdash A}$$

这两个规则需然形式上不同，但其含义是一样的，即，如果前提集Γ能推出A或B，而Γ又能推出A的否定，那么Γ就能推出B，或者Γ能推出B的否定，那么Γ就能推出A。或者更直观地说，如果知道Γ能推出A或B之一成立，而又知道Γ不能推出其中的一个（即Γ能推出它的否定），那么Γ肯定能推出另外一个。

之所以给出两个析取三段论规则，是因为到目前为止还没引入下面的派生规则：

$$\text{析取交换律}\quad \frac{\Gamma \vdash A \vee B}{\Gamma \vdash B \vee A}$$

当然这个很容易使用下面的证明序列验证析取交换律：

(1) $\Gamma, A \vdash A$ // 前提引入规则
(2) $\Gamma, A \vdash B \vee A$ // (1)析取引入
(3) $\Gamma, B \vdash B$ // 前提引入规则
(4) $\Gamma, B \vdash B \vee A$ // (3)析取引入
(5) $\Gamma \vdash A \vee B$ // 待验证规则的前提
(6) $\Gamma \vdash B \vee A$ // (2),(4),(5)析取消除

这个证明序列给出了使用析取消除规则的方法，我们要由$\Gamma \vdash A \vee B$得到$\Gamma \vdash B \vee A$，可考虑通过Γ分别附加A和B这两个前提是否能推出$B \vee A$，然后运用析取消除规则来得到。注意上面最后一步是对析取消除规则

$$\frac{\Delta \vdash C \vee D \qquad \Delta, C \vdash E \qquad \Delta, D \vdash E}{\Delta \vdash E}$$

用Γ代入其中的Δ，用A代入其中C，用B代入D，用$B \vee A$代入E而得到。

有了这个使用析取消除规则的经验，我们可来验证析取三段论。已知$\Gamma \vdash A \vee B$和$\Gamma \vdash \neg A$，要得到$\Gamma \vdash B$，那么要消除$A \vee B$中的析取联结词，我们需要考虑是否有$\Gamma, A \vdash B$和$\Gamma, B \vdash B$，后一个显然，前一个呢？不难看到由$\Gamma \vdash \neg A$可得到$\Gamma, A \vdash \neg A$，而显然又有$\Gamma, A \vdash A$，从而由矛盾律Γ, A可推出任意公式，当然包括B，因此验证第一个析取三段论的证明序列如下：

(1) $\Gamma \vdash \neg A$ // 待验证规则的前提
(2) $\Gamma, A \vdash \neg A$ // (1)弱化规则
(3) $\Gamma, A \vdash A$ // 前提引入规则
(4) $\Gamma, A \vdash B$ // (2),(3)矛盾律
(5) $\Gamma, B \vdash B$ // 待验证规则的前提
(6) $\Gamma \vdash A \vee B$ // 待验证规则的前提
(7) $\Gamma \vdash B$ // (4),(5),(6)析取消除

有了第一个析取三段论规则，再利用析取交换律，很容易验证第二析取三段论规则：

(1) $\Gamma \vdash A \vee B$ // 待验证规则的前提
(2) $\Gamma \vdash B \vee A$ // (1)析取交换律
(3) $\Gamma \vdash \neg B$ // 待验证规则的前提
(4) $\Gamma \vdash A$ // (2),(3)第一个析取三段论

总而言之，析取三段论是揭示这样一个推理规律：当从一个前提能推出两个东西至少有一个成立，那么否定其中一个，就可以肯定另外一个。后面不必再分是否定第一个析取分支还是否定第二个析取分支，直接在注释中写析取三段论即可。

读者可能要问，为什么我们一定给出两个析取三段论，而不强调析取交换律呢？或者干脆认为$A \vee B$与$B \vee A$等值呢？前面已经给出了不用等值的理由，而不强调析取交换律是因为我们认为析

取和合取联结词的交换律、结合律、分配律等，主要从运算的性质刻画这些联结词的性质，与推理的直接关系不大，而析取三段论则与推理的关系更为密切，所以强调上述两个析取三段论，而不强调析取交换律。实际上，在实际推理时，很少使用析取交换律。

前面提到，构造证明序列的过程就像计算机程序执行的过程，这里借这个例子再介绍程序设计中广泛使用的一个思想在构造证明序列中的使用。我们知道，模块化是程序设计的基本方法之一，在程序设计语言中引入函数、过程或方法都是模块化思想的体现，而**模块化思想在构造证明序列中的体现就是派生规则的使用**。

我们可以从这样的角度理解派生规则的使用：**在构造证明序列时使用派生规则相当于调用验证派生规则正确性的证明序列**。例如上面在验证第一个析取三段论得到第四个推理形式结构时使用了矛盾律：

$$
\begin{array}{lll}
 & \vdots & \\
(2) & \Gamma, A \vdash \neg A & // \ (1)\text{弱化规则} \\
(3) & \Gamma, A \vdash A & // \ \text{前提引入规则} \\
(4) & \Gamma, A \vdash B & // \ (2),(3)\text{矛盾律} \\
 & \vdots &
\end{array}
$$

而验证矛盾律的证明序列是这样的（为了避免混淆，下面改变了其中使用的符号）：

$$
\begin{array}{ll}
(1) & \Delta \vdash C \qquad // \ \text{待验证规则的前提} \\
(2) & \Delta, \neg D \vdash C \qquad // \ (1)\text{弱化规则} \\
(3) & \Delta \vdash \neg C \qquad // \ \text{待验证规则的前提} \\
(4) & \Delta, \neg D \vdash \neg C \qquad // \ (3)\text{弱化规则} \\
(5) & \Delta \vdash D \qquad // \ (2),(4)\text{否定消除}
\end{array}
$$

在上面的证明序列中使用矛盾律，可以理解为用(Γ, A)去代入验证矛盾律的证明序列中的Δ，用A代入其中的C，用B代入其中的D，调用验证矛盾律的证明序列，或者说可将上面的证明序列展开为：

$$
\begin{array}{lll}
 & \vdots & \\
(2) & \Gamma, A \vdash \neg A & // \ (1)\text{弱化规则} \\
(3) & \Gamma, A \vdash A & // \ \text{前提引入规则} \\
(4.1) & \Gamma, A \vdash A & // \ (3) \\
(4.2) & \Gamma, A, \neg B \vdash A & // \ (4.1)\text{弱化规则} \\
(4.3) & \Gamma, A \vdash \neg A & // \ (2) \\
(4.4) & \Gamma, A, \neg B \vdash \neg A & // \ (4.3)\text{弱化规则} \\
(4) & \Gamma, A \vdash B & // \ (4.2),(4.4)\text{否定消除} \\
 & \vdots &
\end{array}
$$

对比程序中的函数调用，可以看到，**代入从某种意义上相当于参数传递**，即用实际参数代入形式参数。例如上面用(Γ, A)代入Δ等。由这一点，读者更应理解为什么只能用公式代入到符号，而不能用公式代入到公式，这与程序中不能用表达式作为形式参数的道理相同。

读者也应该理解证明序列中表示前提集和公式的符号是可以一致地改用其他符号，正如我们上面改变了前面验证矛盾律的证明序列中使用的符号，这与程序中，形式参数的名称可以在函数内部统一第改用其他标识符的道理相同。我们希望**通过将证明序列的构造与程序执行过程相联系而加深读者对这两方面的理解**。

下面回到对派生规则的介绍，前面给出了有关蕴涵联结词的派生规则，即传递规则，下面的**拒取式**也是十分常用的与蕴涵联结词有关的派生规则：

例子 3.2.23 **拒取式**，又称为**假言易位**规则既与蕴涵联结词有关，也与否定联结词有关：

$$\text{拒取式}\quad \frac{\Gamma \vdash A \to B \qquad \Gamma \vdash \neg B}{\Gamma \vdash \neg A}$$

拒取式的直观含义是，如果前提集能推出A蕴涵B，但又能推出B的否定，那么可以推出A的否定。例如说，如果天下雨，地就湿，现在地没湿，那么可推出天没下雨。验证该推理的证明序列如下：

(1)	$\Gamma \vdash A \to B$	// 待验证规则的前提
(2)	$\Gamma, A \vdash A \to B$	// (1)弱化规则
(3)	$\Gamma, A \vdash A$	// 前提引入规则
(4)	$\Gamma, A \vdash B$	// (2),(3)蕴涵消除规则
(5)	$\Gamma \vdash \neg B$	// 待验证规则的前提
(6)	$\Gamma, A \vdash \neg B$	// (5)弱化规则
(7)	$\Gamma \vdash \neg A$	// (4),(6)归谬律

构造该证明序列的关键是，规则结论$\Gamma \vdash \neg A$存在否定联结词，而前面的归谬律的结论也是存在否定联结词，因此应该想到使用归谬律，从而应该将A作为附加前提推出矛盾，显然只能尝试由Γ, A推出B和$\neg B$，循着这个思路容易得到上述证明序列。

至此，我们介绍了所有常用的派生规则，下表总结了这些派生规则的名称及模式：

规则名称	规则模式	规则名称	规则模式
双重否定律	$\dfrac{\Gamma \vdash \neg\neg A}{\Gamma \vdash A}$	归谬律	$\dfrac{\Gamma, A \vdash B \qquad \Gamma, A \vdash \neg B}{\Gamma \vdash \neg A}$
矛盾律	$\dfrac{\Gamma \vdash A \qquad \Gamma \vdash \neg A}{\Gamma \vdash B}$		
析取三段论	$\dfrac{\Gamma \vdash A \vee B \qquad \Gamma \vdash \neg B}{\Gamma \vdash A}$	析取三段论	$\dfrac{\Gamma \vdash A \vee B \qquad \Gamma \vdash \neg A}{\Gamma \vdash B}$
拒取式 假言易位	$\dfrac{\Gamma \vdash A \to B \qquad \Gamma \vdash \neg B}{\Gamma \vdash \neg A}$	传递规则 假言三段论	$\dfrac{\Gamma \vdash A \to B \qquad \Gamma \vdash B \to C}{\Gamma \vdash A \to C}$

3.2.4 自然推理系统小结

至此我们给出自然推理系统的常用规则与证明序列的构造方法，在进一步举例说明自然推理系统如何验证推理的有效性之前，先对这一节的内容作一个小结。

对于自然推理系统的常用规则，我们一共给出了十二条基本规则和六条派生规则：

1. 基本规则包括：**前提引入、弱化规则、否定引入、否定消除、合取引入、合取消除、析取引入、析取消除、蕴涵引入、蕴涵消除、等价引入、等价消除**，后面我们将看到，在验证具体的推理时，弱化规则是常用的，否定引入和析取消除规则也是不常用的，因此常用的只有九条规则；

2. 派生规则包括：**双重否定律、归谬律、矛盾律、析取三段论、假言易位和传递规则**，其中矛盾律也比较少用。

自然推理系统的所有规则都有**保真性**，即如果规则的所有前提对应永真式，则其结论也对应永真式。规则前提和结论都由推理形式结构组成。若$\Gamma = \{A_1, A_2, \cdots, A_n\}$，说推理形式结构$\Gamma \vdash A$对应永真式，是指$(A_1 \wedge A_2 \wedge \cdots \wedge A_n) \rightarrow A$是永真式，从而也是指推理形式结构，或直接说推理$\Gamma \vdash A$是**有效的推理**。规则的保真性也是规则的正确性，基本规则的正确性可直接证明，而派生规则的正确性可使用证明序列的形式由基本规则导出。

验证推理有效的**证明序列**是推理形式结构序列，**每个推理形式结构要么是前提引入规则的代入实例，要么是排在它前面的推理形式结构根据某个推理规则得到的**，也即，存在排在它前面的一些推理形式结构作为规则前提的代入实例，而该推理形式结构同时是该规则结论的代入实例。这种代入是严格的、机械的，可以编写一个程序自动验证证明序列对规则的代入是否正确。代入的关键是，**规则中同一个前提集符号或公式符号在规则前提和结论的所有出现都需要使用相同的前提集或公式代入**。

对于证明序列的构造读者还要领会两个与计算机学科密切相关的思想：

1. **证明序列的构造通常采用自顶向下的分析方法**：分析待验证的推理的结论，根据它的特点确定应使用的规则，根据该规则的前提确定要进一步验证的推理，如此逆推，直到所有要验证的推理都是前提引入规则的代入实例。自顶向下分析方法是计算机科学中编写程序求解问题的基本方法。

2. **派生规则的使用相当于调用验证派生规则的证明序列**：这种调用与程序中函数的调用极为相似。原则上，所有的证明序列的构造都可只使用十二条基本规则，派生规则的使用都可通过代入扩展成基本规则的使用，这种扩展与用实际参数代入形式参数的过程类似，因此代入只允许用前提集或公式代入前提集符号或公式符号，而不允许用公式代换公式。

最后在构造证明序列时，读者应该**写出每一步所采用的推理规则，并标出该规则所依赖的前提的编号**，同时还应该清楚是**如何对规则进行代入而得到每一步的推理形式结构的**。

3.3 自然推理系统应用举例

这一节进一步举例说明如何使用自然推理系统的规则，包括基本规则和派生规则，来验证推理的有效性。首先是一些纯形式化的推理，然后是一些简单的实际生活中推理的验证。

首先我们讨论教材[4]的第2.9节所给出的例子(教材第33页至第35页)：

例子 3.3.1 验证推理$p \to q, q \to r, p \vdash r$的有效性的证明序列如下：

(1) $p \to q, q \to r, p \vdash p$ // 前提引入

(2) $p \to q, q \to r, p \vdash p \to q$ // 前提引入

(3) $p \to q, q \to r, p \vdash q$ // (1),(2)蕴涵消除（分离规则）

(4) $p \to q, q \to r, p \vdash q \to r$ // 前提引入

(5) $p \to q, q \to r, p \vdash r$ // (3),(4)蕴涵消除（分离规则）

不难运用上面介绍的分析方法得到上述证明序列，而且读者也很容易看出这里给出的证明序列与教材所给出的证明序列本质上是一样的，虽然教材是采用省略前提形式的证明序列。

注意，教材是说r是$p \to q, q \to r, p$的**逻辑推论**，这与有的教材说可从前提$p \to q, q \to r, p$**有效地推出**或**合乎逻辑地推出**r，以及与我们上面说$p \to q, q \to r, p \vdash r$是**有效的推理**，都具有相同的含义。

例子 3.3.2 验证推理$c \vee d, (c \vee d) \to \neg e, \neg e \to (a \wedge \neg b), (a \wedge \neg b) \to r \vee s \vdash r \vee s$的有效性的证明序列如下；令$\Gamma = \{c \vee d, (c \vee d) \to \neg e, \neg e \to (a \wedge \neg b), (a \wedge \neg b) \to r \vee s\}$，

(1) $\Gamma \vdash (c \vee d) \to \neg e$ // 前提引入

(2) $\Gamma \vdash \neg e \to (a \wedge \neg b)$ // 前提引入

(3) $\Gamma \vdash (c \vee d) \to (a \wedge \neg b)$ // (1),(2)传递规则（假言三段论）

(4) $\Gamma \vdash (a \wedge \neg b) \to r$ // 前提引入

(5) $\Gamma \vdash (c \vee d) \to r$ // (3),(4)传递规则

(6) $\Gamma \vdash (c \vee d)$ // 前提引入

(7) $\Gamma \vdash r$ // (5),(6)蕴含消除

上面为了与教材保持一致也是采用传递规则（即假言三段论），实际上直接每一步使用蕴含消除规则也可以。

例子 3.3.3 验证推理$p \vee q, p \to r, q \to s \vdash s \vee r$的有效性。

分析：

粗看对于如何构造验证上述推理有效性的证明序列有些难入手，教材使用了等值置换，将析取都变换成蕴涵来进行推理。但是，在我们上面介绍的自然推理系统中没有引入与等值置换有关的规则，那么要构造验证该推理有效性的证明序列就需要对其作进一步的分析。

首先不难想到，虽然推理的结论含有析取，但是如果想直接使用析取的引入肯定很困难，因为该推理的前提集要推出s或r直观上看是不可能的。因此我们需要分析前提集的特点，前提集的最显著特点是含有析取式，因此我们想到要使用析取消除规则来构造证明序列，这时就应该将结论看作一个整体，若令$\Gamma = \{p \vee q, p \to r, q \to s\}$，根据析取消除规则

$$\frac{\Gamma \vdash A \vee B \qquad \Gamma, A \vdash C \qquad \Gamma, B \vdash C}{\Gamma \vdash C}$$

显然A应该用p代入，而B要用q代入，C要用$s \vee r$代入。因此我们要考察推理$\Gamma, p \vdash s \vee r$和$\Gamma, q \vdash s \vee r$的有效性。想到这里，再由前提集含有$p \to r$和$q \to s$，我们知道前提集$\Gamma, p$是可以推出$r$的，而$\Gamma, q$是可以推出$s$的，因此得到下面的证明序列：

解答：证明序列如下：令$\Gamma=\{p\vee q, p\to r, q\to s\}$，

(1) $\Gamma\vdash p\to r$ // 前提引入
(2) $\Gamma, p\vdash p\to r$ // (1)弱化规则
(3) $\Gamma, p\vdash p$ // 前提引入
(4) $\Gamma, p\vdash r$ // (2),(3)蕴涵消除
(5) $\Gamma, p\vdash s\vee r$ // (4)析取引入
(6) $\Gamma\vdash q\to s$ // 前提引入
(7) $\Gamma, q\vdash q\to s$ // (6)弱化规则
(8) $\Gamma, q\vdash q$ // 前提引入
(9) $\Gamma, q\vdash s$ // (7),(8)蕴涵消除
(10) $\Gamma, q\vdash s\vee r$ // (9)析取引入
(11) $\Gamma\vdash p\vee q$ // 前提引入
(12) $\Gamma\vdash s\vee r$ // (5),(10),(11)析取消除

注意：推理$p\vee q, p\to r, q\to s\vdash s\vee r$称为**构造性二难推理**，很容易将上面的证明序列加以变化得到如下派生规则的证明序列：

$$\frac{\Gamma\vdash A\vee B \qquad \Gamma\vdash A\to C \qquad \Gamma\vdash B\to D}{\Gamma\vdash C\vee D}$$

该规则的直观含义是，如果前提集Γ能推出A或B之一成立，而Γ可推出A蕴涵C，又可推出B蕴涵D，那么Γ就能推出C或D之一成立。与之相应的还有所谓的**破坏性二难推理**：

$$\frac{\Gamma\vdash \neg C\vee\neg D \qquad \Gamma\vdash A\to C \qquad \Gamma\vdash B\to D}{\Gamma\vdash \neg A\vee\neg B}$$

该规则的直观含义是，如果前提集Γ可推出A蕴涵C，又可推出B蕴涵D，而且能推出C的否定或D的否定之一成立，那么Γ就可推出A或C的否定之一成立。验证破坏性二难推理的证明序列也与上面的证明序列相似，只是将蕴涵消除规则的使用换成假言易位规则即可，具体的证明序列构造留给读者作为练习。

正如在实际应用中，析取消除规则是不容易掌握的，构造性二难推理和破坏性二难推理的使用更难，所以我们上面没有将它们列为自然推理系统的常用派生规则。

例子 3.3.4 验证推理$p\to(q\to s), \neg r\vee p, q\vdash r\to s$的有效性。

解答：

推理结论含有蕴涵式$r\to s$，因此第一反应就应想到将其前件r作为附加前提来推出s。既然将r作为了附加前提就要用，那么就要用在$\neg r\vee p$中，从而得到p，然后再根据$p\to(q\to s)$得到$q\to s$，前提集中又含有q，这样就可以得到s了。这样我们得到如下的证明序列：令$\Gamma=\{p\to(q\to s), \neg r\vee p, q\vdash r\to s\}$，

(1) $\Gamma, r\vdash r$ // 前提引入
(2) $\Gamma, r\vdash \neg\neg r$ // (1)否定引入

(3) $\Gamma, r \vdash \neg r \vee p$ // 前提引入
(4) $\Gamma, r \vdash p$ // (2),(3)析取三段论
(5) $\Gamma, r \vdash p \to (q \to s)$ // 前提引入
(6) $\Gamma, r \vdash q \to s$ // (4),(5)蕴涵消除
(7) $\Gamma, r \vdash q$ // 前提引入
(8) $\Gamma, r \vdash s$ // (6),(7)蕴涵消除
(9) $\Gamma \vdash r \to s$ // (6),(7)蕴涵消除

注意，在上述证明序列中，由$\Gamma, r \vdash r$和$\Gamma, r \vdash \neg r \vee p$还不能直接根据前面给出析取三段论规则

$$\frac{\Delta \vdash A \vee B \qquad \Delta \vdash \neg A}{\Delta \vdash B}$$

直接得到$\Gamma, r \vdash p$，因为不能用r代入$\neg A$，所以上面先使用否定引入得到$\Gamma, r \vdash \neg\neg r$，再使用上述析取三段论规则就可以得到$\Gamma, r \vdash p$，这时是使用$\neg r$代入$A$。

当然根据析取三段论的实质，即对于析取式，否定一个分支就可肯定另外一个分支，再加上根据双重否定律也可认为A是$\neg A$的否定，我们可引入析取三段论的如下变体（读者在构造证明序列时也可使用析取三段论的变体形式，而注释仍然说是使用析取三段论）：

析取三段论 $\dfrac{\Delta \vdash \neg A \vee B \qquad \Delta \vdash A}{\Delta \vdash B}$ $\qquad$ $\dfrac{\Delta \vdash A \vee \neg B \qquad \Delta \vdash B}{\Delta \vdash A}$

例子 3.3.5 验证推理$\neg(p \to q) \to \neg(r \vee s), (q \to p) \vee \neg r, r \vdash p \leftrightarrow q$有效性的证明序列如下：
令$\Gamma = \{\neg(p \to q) \to \neg(r \vee s), (q \to p) \vee \neg r, r\}$，

(1) $\Gamma \vdash r$ // 前提引入
(2) $\Gamma \vdash r \vee s$ // (1)析取引入
(3) $\Gamma \vdash \neg(p \to q) \to \neg(r \vee s)$ // 前提引入
(4) $\Gamma \vdash \neg\neg(p \to q)$ // (2),(3)假言易位
(5) $\Gamma \vdash p \to q$ // (4)双重否定律
(6) $\Gamma \vdash \neg\neg r$ // (1)否定引入
(7) $\Gamma \vdash (q \to p) \vee \neg r$ // 前提引入
(8) $\Gamma \vdash q \to p$ // (6),(7)析取三段论
(9) $\Gamma \vdash p \leftrightarrow q$ // (5),(8)等价引入

上述证明序列用到了假言易位规则

假言易位（拒取式） $\dfrac{\Gamma \vdash A \to B \qquad \Gamma \vdash \neg B}{\Gamma \vdash \neg A}$

这个规则的实质是，由蕴涵式后件的否定可推出蕴涵式前件的否定，同样根据任意公式与它的否定互为否定这一点，我们也可得到假言易位规则的一些变体（同样，读者在使用时仍可只说是使用假言易位规则）：

假言易位 $\dfrac{\Gamma \vdash \neg A \to B \qquad \Gamma \vdash \neg B}{\Gamma \vdash A}$ $\qquad$ $\dfrac{\Gamma \vdash A \to \neg B \qquad \Gamma \vdash B}{\Gamma \vdash \neg A}$

$$\frac{\Gamma \vdash \neg A \to \neg B \qquad \Gamma \vdash B}{\Gamma \vdash A}$$

有了这些假言易位规则的变体，上述证明序列中的推理形式结构(4)可以省略，即可直接根据(2)和(3)使用假言易位规则得到(5)。

可利用自然推理系统中构造证明序列的方法验证一些数学和日常生活中的推理：

例子 3.3.6 在自然推理系统中验证下述推理的正确性：

若数a是实数，则它不是有理数就是无理数。若a不能表示成分数，则它不是有理数。a是实数且它不能表示成分数。所以a是无理数。

分析：在本课程中遇到的数学和日常生活中的推理通常都具有与上面类似的简单形式，碰到这类问题，我们**首先要将整个推理进行符号化**，此时应注意：

(1) 首先应该**分清楚这个推理的前提和结论**。简单地说，在这种题目中，词语“所以”或“因此”前面的句子都是给出推理的前提，而“所以”或“因此”以后的句子就是结论；

(2) 在**对句子符号化时应该遵循前面对自然语言命题进行符号化的方法**，即简单地说应该：(i) 先提炼原子命题，用命题变量符号表示；(ii) 然后分析句子的结构，使用合适的命题联结词符号化整个句子。

解答：首先通读句子得到如下的原子命题：

$$p：a\text{是实数} \qquad q：a\text{是有理数} \qquad r：a\text{是无理数} \qquad s：a\text{不能表示成分数}$$

从而前提可符号化为：

1. “若数a是实数，则它不是有理数就是无理数”符号化为$p \to (q \vee r)$；
2. “a不能表示成分数，则它不是有理数”符号化为$s \to \neg q$；
3. “a是实数且它不能表示成分数”符号化为$p \wedge s$。

显然结论应符号化为r，因此要验证的推理是：$p \to (q \vee r), s \to \neg q, p \wedge s \vdash r$，很容易构造验证该推理有效性的证明序列：令$\Gamma = \{p \to (q \vee r), s \to \neg q, p \wedge s \vdash r\}$，

(1)	$\Gamma \vdash p \wedge s$	// 前提引入
(2)	$\Gamma \vdash p$	// (1)合取消除
(3)	$\Gamma \vdash p \to (q \vee r)$	// 前提引入
(4)	$\Gamma \vdash q \vee r$	// (2),(3)蕴涵消除
(5)	$\Gamma \vdash s$	// (1)合取消除
(6)	$\Gamma \vdash s \to \neg q$	// 前提引入
(7)	$\Gamma \vdash \neg q$	// (5),(6)蕴涵消除
(8)	$\Gamma \vdash r$	// (4),(7)析取三段论

例子 3.3.7 在自然推理系统中验证下述推理的正确性：

如果工资提高，或者物价提高，则将有通货膨胀。如果通货膨胀，则或者国家将采取紧缩政策，或者人民将遭受损失。如果人民遭受损失，改革就会失去人心。国家将不采取紧缩正则，并且改革不会失去人心。因此，物价不会提高。

解答：通读句子，可提炼出如下的原子命题：

p：工资提高	q：物价提高	r：通货膨胀
s：国家采取紧缩政策	t：人民遭受损失	u：改革会失去人心

而整个推理的前提可符号化为：

(1) 句子“如果工资提高，或者物价提高，则将有通货膨胀”可符号化为：$(p \vee q) \to r$；

(2) 句子“如果通货膨胀，则或者国家将采取紧缩政策，或者人民将遭受损失”可符号化为：$r \to (s \vee t)$；

(3) 句子“如果人民遭受损失，改革就会失去人心”可符号化为：$t \to u$；

(4) 句子“国家将不采取紧缩正则，并且改革不会失去人心”可符号化为：$\neg s \wedge \neg u$。

显然结论应该符号化为$\neg q$。因此要验证的推理是：

$$(p \vee q) \to r,\ r \to (s \vee t),\ t \to u,\ \neg s \wedge \neg u \vdash \neg q$$

验证该推理有效性的证明序列如下：令$\Gamma = \{(p \vee q) \to r,\ r \to (s \vee t),\ t \to u,\ \neg s \wedge \neg u\}$，

(1)	$\Gamma, q \vdash q$	// 前提引入
(2)	$\Gamma, q \vdash p \vee q$	// (1)析取引入
(3)	$\Gamma, q \vdash (p \vee q) \to r$	// 前提引入
(4)	$\Gamma, q \vdash r$	// (2),(3)蕴涵消除
(5)	$\Gamma, q \vdash r \to (s \vee t)$	// 前提引入
(6)	$\Gamma, q \vdash s \vee t$	// (4),(5)蕴涵消除
(7)	$\Gamma, q \vdash \neg s \wedge \neg u$	// 前提引入
(8)	$\Gamma, q \vdash \neg s$	// (7)合取消除
(9)	$\Gamma, q \vdash t$	// (6),(8)析取三段论
(10)	$\Gamma, q \vdash t \to u$	// 前提引入
(11)	$\Gamma, q \vdash u$	// (9),(10)蕴涵消除
(12)	$\Gamma, q \vdash \neg u$	// (7)合取消除
(13)	$\Gamma \vdash \neg q$	// (11),(12)归谬律

构造该证明序列的思路是：当待验证推理的结论是否定式时，我们应该首先考虑使用归谬律，即引入结论被否定的部分作为附加前提一起推出矛盾，即既推出A又推出$\neg A$，且这时应该考察前提集中的公式，如果前提集中有形式为$\neg A$的公式，那么应该考虑推出A，例如上面前提集Γ中含有$\neg s \wedge \neg u$，因此可考虑由Γ, q推出u（上面的证明序列就是这样），也可考虑由Γ, q推出s（这可得到另一个证明序列）。

如何从Γ, q推出u呢？再注意到u出现在公式$t \to u$中，因此只要得到t即可，再看t出现在$r \to (s \vee t)$，因此如果能得到r就可得到$s \vee t$，而$\neg s \wedge \neg u$可得到$\neg s$，再用析取三段论就可得到t。现在问题变成如何得到r，显然r出现$(p \vee q) \to r$中，只要有$p \vee q$即可得到r，而$p \vee q$可由q通过析取引入规则得到，从而最终形成了上述证明序列。

练习 3.3.8 对于上述例子，考虑通过验证推理$\Gamma, q \vdash s$的有效性来构造推理$\Gamma \vdash \neg q$的证明序列。

例子 3.3.9 在自然推理系统中验证如下推理的正确性：

张大侠参与作案，只有在下述情况下才有可能：或者他受到胁迫，或者他既不明真相，又不愿意告发朋友。如果他不愿意告发朋友，则他一定明白真相。因此，如果不是他受到胁迫，他就决不可能参与作案。

解答：通读句子，可提炼出如下的原子命题：

p：张大侠参与作案　　q：（张大侠）他受到胁迫

r：（张大侠）他明白真相　　s：（张大侠）他不愿意告发朋友

而整个推理的前提可符号化为：

(1) 句子“张大侠参与作案，只有在下述情况下才有可能：或者他受到胁迫，或者他既不明真相，又不愿意告发朋友”的句式是“只有…张大侠才参与作案”，因此应该符号化为：$p \to (q \vee (\neg r \wedge s))$；

(2) 句子“如果他不愿意告发朋友，则他一定明白真相”可符号化为：$s \to r$；

结论“如果不是他受到胁迫，他就决不可能参与作案”则应该符号化为$\neg q \to \neg p$。因此，要验证的推理是：

$$p \to (q \vee (\neg r \wedge s)), s \to r \vdash \neg q \to \neg p$$

验证该推理有效性的证明序列如下：令$\Gamma = \{p \to (q \vee (\neg r \wedge s)), s \to r\}$，

(1) $\Gamma, \neg q, p \vdash p$　　// 前提引入

(2) $\Gamma, \neg q, p \vdash p \to (q \vee (\neg r \wedge s))$　　// 前提引入

(3) $\Gamma, \neg q, p \vdash q \vee (\neg r \wedge s)$　　// (1),(2)蕴涵消除

(4) $\Gamma, \neg q, p \vdash \neg q$　　// 前提引入

(5) $\Gamma, \neg q, p \vdash \neg r \wedge s$　　// (3),(4)析取三段论

(6) $\Gamma, \neg q, p \vdash s$　　// (5)合取消除

(7) $\Gamma, \neg q, p \vdash s \to r$　　// 前提引入

(8) $\Gamma, \neg q, p \vdash r$　　// (6),(7)蕴涵消除

(9) $\Gamma, \neg q, p \vdash \neg r$　　// (5)合取消除

(10) $\Gamma, \neg q \vdash \neg p$　　// (8),(9)归谬论

(11) $\Gamma \vdash \neg q \to \neg p$　　// (10)蕴涵引入

构造上述证明序列的思路是：由于待验证推理的结论是蕴涵式$\neg q \to \neg p$，所以我们将$\neg q$作为附加前提来验证推理$\Gamma, \neg q \vdash \neg p$的正确性，而这个推理的结论是否定式，因此又将$p$再作为附加前提来推出矛盾，即由前提集$\Gamma \cup \{\neg q, p\}$来推出矛盾。

显然这时考虑由$\Gamma, \neg q, p$来推出$\neg q$和q的思路是不正确的，因为$\neg q$是引入的附加前提，我们应该优先考虑Γ中的否定式，即应该考虑由$\Gamma, \neg q, p$来推出$\neg r$和r。这样由p及$p \to (q \vee (\neg r \wedge s))$可得到$q \vee (\neg r \wedge s)$，再由$\neg q$及析取三段论可得到$\neg r$和$s$，再由$s$和$s \to r$就可得到$r$，从而推出矛盾。上述证明序列的构造正体现了这一思路。

作业

作业 3.1 试仿照否定引入规则正确性的证明，证明析取消除规则

$$\frac{\Gamma \vdash A \vee B \qquad \Gamma, A \vdash C \qquad \Gamma, B \vdash C}{\Gamma \vdash C}$$

的正确性。

作业 3.2 试给出证明破坏性二难推理规则

$$\frac{\Gamma \vdash \neg C \vee \neg D \qquad \Gamma \vdash A \to C \qquad \Gamma \vdash B \to D}{\Gamma \vdash \neg A \vee \neg B}$$

正确性的证明序列。

作业 3.3 试根据推理有效性的含义及派生规则正确性的含义证明：对任意的前提集Γ和公式$A_1, A_2, \cdots, A_n$，若$A_1, A_2, \cdots, A_n \vdash A$是有效的推理，则下述派生规则

$$\frac{\Gamma \vdash A_1 \qquad \Gamma \vdash A_2 \quad \cdots \quad \Gamma \vdash A_n}{\Gamma \vdash A}$$

是正确的。

作业 3.4 在自然推理系统中验证下述推理的有效性：

(1) $q \to p,\ q \leftrightarrow s,\ s \leftrightarrow t,\ t \wedge r \vdash p \wedge q$；

(2) $(p \vee q) \to (r \wedge s), (s \vee t) \to u \vdash p \to u$；

(3) $p \to \neg q, \neg r \vee q, r \wedge \neg s \vdash \neg p$；

(4) $p \to (q \to r), r \to \neg r, s \to p, t \to q \vdash s \to \neg t$。

作业 3.5 在自然推理系统中验证下述推理的有效性：

(1) 只要张三曾到过受害者房间并且11点以前没有离开，则张三犯了谋杀罪。张三曾到过受害者房间。如果张三在11 点以前离开，则看门人会看见他。看门人没有看见他。所以，张三犯了谋杀罪。

(2) 在大城市球赛中，如果北京队第三，那么上海队第二意味者天津队第四。沈阳队不是第一或北京队第三。上海队第二。因此，如果沈阳对第一，则天津队第四。

(3) 如果法官是公正严明的，则就应当宣判张大侠有罪，除非现有的证据尚不充分。如果法官是公正严明的，则不会不认定现有的证据是充分的，除非这些证据有假。证据都没有假。因此，如果法官是公正严明的，就应当宣判张大侠有最。

第四章　命题逻辑的演算系统

命题逻辑的核心是基于命题之间的真值关系，也即命题联结词的性质研究推理的有效性，上一章讨论了命题逻辑的自然推理系统，它从前提出发，根据一些基本规则对推理形式结构进行变换，以验证推理的有效性。命题演算系统则使用更为抽象的方法研究推理的有效性：**命题演算系统是纯粹符号化的形式系统，甚至不涉及命题的真值，而将推理看作纯粹符号的变换**。

我们知道，使用数学方法研究逻辑的第一步是符号化，将具体命题使用符号，即命题变量表示，在命题逻辑中只关心赋予这些符号的真值，而不关心这些符号所代表的具体命题。命题演算系统作为形式系统则更进一步：**符号就是符号**，不具有任何含义，连符号是否具有真值也不涉及，任何对符号含义的解释都是形式系统以外的内容，是**形式系统的模型**。

计算机也是这样的形式系统：所处理的所有符号都是二进制代码串，从计算机的角度看，这些二进制串只是纯粹的符号，计算机系统本身不理解这些二进制串的含义，所有的含义都是程序员或计算机用户所赋予的。

这一章我们首先简单介绍形式系统的基本含义，然后简单讨论一个公理化的命题演算系统。

4.1　形式系统的基本概念

前面我们已经接触到形式系统，因为数理逻辑是用数学化方法研究逻辑，就是为逻辑构建一个形式系统来研究推理的有效性。例如，前面在定义命题逻辑公式时，我们已经指出，所有命题逻辑公式构成一个**形式语言**。实际上，任何一个形式系统都有自己的形式语言。

对于形式系统，我们研究它的三个方面内容：

1. **形式语法**部分：定义形式系统所使用的形式语言，在确定**符号表**的基础上给出构造句子的语法规则。简单地说，**形式语法部分就是确定形式系统要研究怎样的句子（即哪些是符合语法的符号串）**？

2. **形式演算**部分：确定形式系统的内定理。内定理是形式系统的某些句子，这些句子我们认为是好的、有用的，或者说是真的。形式演算通常是预先确定一些真的句子（称为公理），再确定从已有定理得到新的定理的一些规则。**从公理出发，根据规则确定内定理的过程就是形式系统的演算**。简单地说，**形式演算部分就是确定哪些句子是形式系统好的、有用的或真的句子**。

3. **形式语义**部分：给出一个模型，解释形式语言句子的含义，这是应用形式系统的基础。形式语法和形式演算部分不涉及句子的含义，但形式系统总是客观世界某个问题的抽象，要应用形式系统，那么还是需要讨论形式系统的语义。

在定义一个形式系统时，必须严格给出它的形式语法，这是形式系统的最基础部分。通常形式演算部分也是必须的，但是有时可能是严格定义公理和规则，有时没有严格给出公理和演算规则。

形式语义讨论形式系统的模型，可以说是形式系统本身以外的内容，但它是应用形式系统的基础，并且通过形式系统的语义模型可从整体上研究形式系统的性质，这种性质相对于形式系统的内定理而言是形式系统的**元理论**。

计算机是数据处理的机器，根据程序将输入变换成输出。它从某种意义上说是一个形式系统，二进制编码是它的形式语言，没有公理，但程序是它的演算规则，或者说计算机的机器指令是它对表示输入数据的二进制串变换成表示输出数据的二进制串的演算规则。

上一章讨论的命题逻辑的自然推理系统从某种意义上来说也是一个形式系统，所有命题逻辑公式构成它的形式语言。虽然我们没有严格定义它的公理和演算规则，但读者不难想到前提引入规则相当于公理，它是构造证明序列的出发点，其他规则都是构造证明序列的演算规则，有效的推理可看作自然推理系统的内定理。

与计算机系统类比，可以说，证明序列就是程序：证明序列根据规则将前提引入规则的代入实例变换成所要验证的推理；程序根据指令将输入数据变换成输出数据。计算机系统的内定理是什么？可以说是“从某输入可以正确产生某输出”，程序是对这种定理的证明。

形式系统的构造通常采用**公理化**的方法，公理化方法的关键是找出一些基本的东西，以及从基本东西构造复杂东西的方法。前面看到，所有命题逻辑公式构成的形式语言的定义采用了公理化方法，或更具体地，采用了归纳法定义，归纳基是命题变量，即最基本的命题逻辑公式，归纳步利用命题联结词给出了命题逻辑公式的构造。

形式演算部分更是常常采用公理化方法定义：给出一些公理作为最基本的内定理，以及从已有定理得到新的定理的规则。命题逻辑的自然推理系统中，前提引入规则给出的推理形式结构可看作是最基本的内定理，其他规则从已有的（有效的）推理得到新的（有效的）推理。

这一章所讨论的命题逻辑的公理化演算系统更是使用公理化方法定义，与上一章讨论的自然推理系统的主要区别在于：更强调是纯粹符号的演算，不再从真值的角度定义什么是有效的推理，而是根据公理化方法，确定一些公理以及规则，再归纳定义内定理。虽然内定理本质上也是永真式，但从形式系统本身来看，完全没有“真值”这个概念。对公式赋予真值，是对命题演算系统的一种解释，是给出命题演算系统形式语义的一种方法。上一章没有像大多数教材在自然推理部分引入等值置换规则也正是希望读者逐步领会形式系统的基本思想。

这一章的公理化演算系统与上一章自然推理系统的另一个区别是：为了突出形式系统的特点，公理化演算系统不使用太多的命题联结词，只利用完备集$\{\neg, \rightarrow\}$定义其形式语言。进一步，公理化演算系统也只有极少数公理和规则，因此内定理的证明比自然推理系统的验证推理有效性的证明序列构造更需要技巧，从而在这个意义上，公理化演算系统不像自然推理系统，自然推理系统由于使用很多规则，规则都是联结词的引入和消除而更接近人们日常生活中的推理。

下一节将按照公理化方法，严格并且完整地给出命题逻辑的公理化演算系统，包括它的形式语言和形式演算部分，而再下一节则简单讨论该系统的形式语义模型以及一些与形式系统元理论有关的基本知识。

4.2 命题逻辑的公理化演算系统

我们将下面所要讨论的公理化演算系统记为$\mathcal{P}$。下面分几个小节给出它的定义：第一小节定义它的形式语法，即给出它的符号表，以及它所研究的公式的归纳定义，这一小节是前面命题逻辑公

式定义的简化；第二小节给出它的公理、推理规则，以及内定理的定义，从而给出它的形式演算部分；第三小节使用例子说明它的一些内定理的证明，并介绍在证明内定理时所要使用的一些基本定理，并与上一章所讨论的自然推理系统进行简单的比较。

4.2.1 公理化演算系统的公式

系统$\mathcal{P}$只使用否定联结词和蕴涵联结词：

定义 4.2.1 *系统$\mathcal{P}$的形式语言的**符号表**包括三类符号：*

1. **小写英文字母**：*称为**命题变量**，所有可能出现的命题变量构成的集合记为$\mathbf{Var}$；*
2. **命题联结词**：*包括$\neg, \rightarrow$；*
3. **辅助符号**：*包括圆括号$(,)$。*

定义 4.2.2 *系统$\mathcal{P}$的（**合式**）**公式**(well-formed formula)归纳定义为：*

1. **归纳基**：*命题变量是公式；*
2. **归纳步**：*若A和B是公式，则$(\neg A)$和$(A \rightarrow B)$也是公式；*
3. **最小化**：*所有公式通过有限次运用上面两个规则得到的。*

记系统$\mathcal{P}$的所有（合式）公式构成的集合为$\mathfrak{Form}(\mathcal{P})$。

我们知道，$\{\neg, \rightarrow\}$是联结词的完备集，因此从理论上来说，使用否定联结词和蕴涵联结词就够了。根据前面的讨论，否定联结词必不可少，而蕴涵联结词与推理的关系最为密切，因此在研究命题演算系统时通常采用这两个联结词构成的完备集。实际上，我们可利用联结词之间的关系，引入其它联结词：

$$A \wedge B \stackrel{\text{def}}{=} \neg(A \rightarrow \neg B) \qquad A \vee B \stackrel{\text{def}}{=} \neg A \rightarrow B$$

$$A \leftrightarrow B \stackrel{\text{def}}{=} \neg((A \rightarrow B) \rightarrow \neg(B \rightarrow A))$$

注意，这里**没有等值**这个概念，上面使用的是符号$\stackrel{\text{def}}{=}$，意思是，符号串$A \wedge B$是公式$\neg(A \rightarrow \neg B)$的**语法意义上的缩写**，其他的联结词定义含义相同。通过这种缩写，也可在公理化演算系统中研究涉及合取、析取以及等价联结词的推理，但下面我们只考虑含有否定和蕴涵联结词的公式。

我们也可以通过规定$\neg$的优先级比$\rightarrow$高，且$\rightarrow$从右至左结合来简化公式的书写方法，省略一些不必要的圆括号。

4.2.2 公理演算系统的内定理

这一节讨论命题逻辑的公理化演算系统$\mathcal{P}$的形式演算部分。形式演算部分的核心是**要确定系统系统的哪些公式是内定理**。内定理可认为是在形式系统内部认为是好的、有用的或者说真的公式，之所以称为内定理是相对于形式系统的元理论而言的。**命题演算系统的内定理本质上就是也是永真式，但不是从真值的角度定义，而是从纯粹符号的角度使用公理化方法定义**。由于公理化命题演算系统的内定理通常是用蕴涵式，而蕴涵式与推理的关系密切，因此实际上这些内定理给出了基于命题逻辑进行推理的基本性质。

根据公理化方法，要确定形式系统的内定理是先确定形式系统的一些公式作为公理，即基本的内定理，然后确定一些从已有定理得到新的定理的方法，即规则。形式系统的演算就是从公理出发，

根据规则得到内定理的过程。从应用形式系统的角度而言，形式系统为求解某些问题而对现实世界的抽象，问题所处的现实世界是形式系统的模型。公理通常是形式系统模型中一些不言自明的事实，而规则通常刻画了求解问题的一般方法。

定义 4.2.3 *系统$\mathcal{P}$有三条公理（模式）：对任意的公式A,B,C，下面三个公式（的代入实例）是公理：*

$$(A1).\ \ (A\to(B\to A))$$
$$(A2).\ \ (A\to(B\to C))\to((A\to B)\to(A\to C))$$
$$(A3).\ \ ((\neg B\to\neg A)\to(A\to B))$$

同样，A,B,C这些符号本身不属于系统$\mathcal{P}$的符号表，它们用于代表任意的公式，因此上面给出的是公理模式，必须使用具体的公式代入其中的A,B,C才能得到具体的公理。同样，这里的代入也要**使用同一个公式代入某个符号在公理模式中的所有出现**。例如下面的公式：

$$((p\to q)\to(r\to(p\to q)))$$

是公理$(A1)$的代入实例，用$p\to q$代入其中的符号A，而r代入其中的B。

直观地看，这几个公理（模式）给出了基于命题逻辑推理的基本特点：

(1) 公理$(A1)$ $(A\to(B\to A))$的直观含义可这样理解：如果已经得到A，那么A可由任意的公式B蕴涵（推出），这体现了命题逻辑推理的基本特点：**真的命题可由任何命题推出**；

(2) 公理$(A2)$ $(A\to(B\to C))\to((A\to B)\to(A\to C))$的直观含义是：如果$A$蕴涵$B\to C$，那么由$A$蕴涵$B$可推出$A$蕴涵$C$，实际上这是**假言三段论（传递规则）的公理化表示**，即A蕴涵B，B蕴涵C，那么可得到A蕴涵C；

(3) 公理$(A3)$. $((\neg B\to\neg A)\to(A\to B))$的直观含义是：由$B$的否定蕴涵$A$的否定可得到$A$蕴涵$B$的否定，显然这是**假言易位规则公理化表示**，体现了蕴涵与否定之间的基本推理关系。

下面给出系统$\mathcal{P}$的内定理的定义，并引出得到内定理的证明序列及规则：

定义 4.2.4 *系统$\mathcal{P}$的**内定理**(internal theorem)归纳定义为：*

1. **归纳基**：*公理$(A1),(A2),(A3)$的任意代入实例都是内定理；*
2. **归纳步**：*如果公式A和公式$(A\to B)$都是内定理，则B也是内定理；*
3. **最小化**：*所有内定理都是通过上述两个规则有限步得到。*

记系统$\mathcal{P}$的所有内定理构成的集合为$\Im\mathfrak{h}(\mathcal{P})$，显然它是所有合式公式构成的集合$\Im\mathfrak{h}(\mathcal{P})$的子集。

命题的公理化演算系统的一个优势就是可以很简洁地给出它的内定理的归纳定义。自然推理系统也可从纯粹符号的角度给出**有效推理**的归纳定义，但比较麻烦，不如从真值的角度定义更为自然。很容易看出，**系统$\mathcal{P}$的内定理都是命题逻辑的永真式**，因为公理的代入实例都是永真式，而当A和$A\to B$是永真式时，B也是永真式。不过，**说内定理是永真式是给出了形式系统的语义模型**，在下面推演内定理时，**完全不需要涉及永真式这个语义概念，而可以通过纯粹符号的变换得到内定理**。

根据内定理的定义，公式$(p\to(q\to p))$是内定理，它是公理$(A1)$的代入实例。为了得到更多的内定理，仅仅根据上述归纳定义有些困难，因此同样，我们可引入证明序列的概念，通过构造证明序列来证明和得到更多的内定理：

定理 4.2.5 如果公式A存在如下的有穷公式序列，则A是内定理：

$$\begin{array}{l} A_1 \\ A_2 \\ \vdots \\ A_n \end{array}$$

其中$A_n = A$，且满足对任意的$1 \le i \le n$有下列两种情况之一成立：

(1) A_i是公理$(A1), (A2), (A3)$中的某一个公理的代入实例；

(2) 存在$1 \le j, k < i$，使得$A_k = (A_j \to A_i)$，这时称公式A_i是由排在它前面的两个公式A_j和A_k根据**分离规则**得到的。

称上面的这种序列为内定理A的**证明序列**。

证明 很容易根据内定理的定义，通过证明上述证明序列中的任意公式$A_i (1 \le i \le n)$都是内定理的方式证明$A = A_n$是内定理。对证明序列实施归纳法即可：

1. **归纳基**：根据给出证明序列的条件，A_1, A_2必然是公理的代入实例，从而是内定理；

2. **归纳步**：对任意的A_i，如果是公理的代入实例则当然是内定理，否则是由排在它前面的两个公式A_j, A_k使用分离规则得到的，根据归纳假设，A_j, A_k都是内定理，从而由内定理定义的归纳步得到A_i也是内定理。

□

也就是说，在证明内定理时（或说构造内定理的证明序列时），只需要使用两个规则：

1. **代入规则**：公理的代入实例是内定理；

2. **分离规则**：如果A和$(A \to B)$是内定理，则B也是内定理，可用下面的方式表示：

$$\text{分离规则} \quad \frac{A \qquad A \to B}{B}$$

例子 4.2.6 公式$p \to p$是系统$\mathcal{P}$的内定理，可使用如下的证明序列证明：

(1)	$(p \to ((p \to p) \to p)) \to ((p \to (p \to p)) \to (p \to p))$	// 公理$(A2)$
(2)	$(p \to ((p \to p) \to p))$	// 公理$(A1)$
(3)	$(p \to (p \to p)) \to (p \to p)$	// (1),(2)分离规则
(4)	$(p \to (p \to p))$	// 公理$(A1)$
(5)	$p \to p$	// (3),(4)分离规则

注意，上述证明序列的公式(1)是公理$(A2)$

$$(A \to (B \to C)) \to ((A \to B) \to (A \to C))$$

的代入实例，其中用$p \to p$代入B，用p代入A，同样用p代入C。公式(2)是公理$(A1)$

$$(A \to (B \to A))$$

的代入实例，其中用p代入A，而用$p \to p$代入B。

由于公理化演算系统的公理规则很少，因此要构造证明内定理的证明序列就需要熟练的技巧而缺乏比较清晰的启发式思路，这与自然推理系统的证明序列常常可使用自顶向上分解的分析思路而得到就大不相同。但从另一角度看，公理化演算系统更能体现公理化的思想，更容易使用归纳严格定义，更容易使用归纳证明有关证明序列，乃至整个形式系统的一些元性质。

例如，我们先定义如何用一个公式去代入另一个公式中命题变量的所有出现：

定义 4.2.7 设公式A是含有命题变量p的公式，用B代入A中命题变量p的所有出现得到的公式记为$A[B/p]$，可归纳定义为：

1. **归纳基**：如果公式A是命题变量p，则$p[B/p] = B$；如果公式A是命题变量$q(q \neq p)$，则$q[B/p] = q$；

2. **归纳步**：如果公式A具有形式$(A_1 \to A_2)$，则$A[B/p] = (A_1[B/p] \to A_2[B/p])$。

实际上，用公式代入另一个公式中的命题变量和用公式代入公理模式中的表示任意公式的符号本质上是相同的，例如，对于公式

$$p \to (q \to p)$$

可用公式$(p \to q)$代入其中的p从而得到

$$(p \to q) \to (q \to (p \to q))$$

这和将公理模式$A \to (B \to A)$用公式$(p \to q)$代入其中的A，q代入其中的B的代入方法本质上是相同的：

$$(p \to q) \to (q \to (p \to q))$$

从这个角度看，**具体公式中的命题变量和公理模式中代表任意公式的符号具有类似的含义**：前面我们说命题变量是原子命题的抽象，实际上说命题变量是任意命题的抽象也是可以的，因为命题逻辑真正关心的不是命题变量的具体含义，而只关心它的真值。不过为了在形式上区别起见，当命题变量代表任意命题时，我们使用代表任意公式的符号来表示，这样得到类似公理模式的**公式模式**，例如$A \to A$是公式模式，而$p \to p$是具体公式，是这个公式模式的代入实例。

在理解了命题变量符号与代表任意公式的符号之间的关系之后，我们可根据内定理的归纳定义证明下面的定理：

定理 4.2.8 设公式A是含有命题变量p的（具体）公式，若公式A是内定理，则对任意公式B，用B代入公式A的命题变量p的所有出现得到的公式$A[B/p]$也是内定理。

证明 根据内定理的归纳定义，对内定理A实施归纳法：

1. **归纳基**：若A是某个公理的代入实例，则可证明$A[B/p]$也是这个公理的代入实例。例如，若A是公理$(A1)$的代入实例，即存在公式A_1, A_2使得

$$A = (A_1 \to (A_2 \to A_1))$$

从而根据$A[B/p]$的定义不难得到

$$A[B/p] = (A_1[B/p] \to (A_2[B/p] \to A_1[B/p]))$$

这就说明$A[B/p]$也是公理$(A1)$的代入实例。对于其他公理两个公理也可得到类似结论。因此当A是某个公理的代入实例时，$A[B/p]$也是该公理的代入实例，从而$A[B/p]$是内定理。

2. **归纳步**：若A是由于存在内定理$C \to A$，及内定理C而使用分离规则得到的，则根据**归纳假设**有$(C \to A)[B/p] = (C[B/p] \to A[B/p])$和$C[B/p]$是内定理，从而显然有$A[B/p]$也是内定理。

□

这个定理告诉我们，当证明了公式$p \to p$是内定理之后，可以得到，对任意的公式A，$A \to A$的代入实例都是内定理，或者说，我们得到公式模式$A \to A$（的任意代入实例）是内定理。更一般地，我们可针对公式模式讨论证明序列的构造，只要将针对具体公式的证明序列中的命题变量符号（在整个证明序列中）统一地用表示任意公式的符号代入即可。例如，可以说下面的证明序列证明了公式模式$A \to A$是内定理：

(1) $(A \to ((A \to A) \to A)) \to ((A \to (A \to A)) \to (A \to A))$ // 公理$(A2)$

(2) $(A \to ((A \to A) \to A))$ // 公理$(A1)$

(3) $(A \to (A \to A)) \to (A \to A)$ // (1),(2)分离规则

(4) $(A \to (A \to A))$ // 公理$(A1)$

(5) $A \to A$ // (3),(4)分离规则

只是这时在考虑公理的代入实例时，是使用公式模式代入公理中代表任意公式的符号，例如上述证明序列的公式模式(1)是公理$(A2)$

$$(A \to (B \to C)) \to ((A \to B) \to (A \to C))$$

的代入实例，其中用公式模式$A \to A$代入B，用A代入A，同时也用A代入C。

因此，根据定理4.2.8，在构造内定理的证明序列时，使用公式模式和使用具体公式本质上并无差别，因此下一节全部针对公式模式进行讨论，不再区分公式与公式模式。

4.2.3 公理化演算系统的内定理证明

这一节我们给出系统$\mathcal{P}$的更多内定理及其证明序列，并讨论一些与内定理证明有关的性质。首先根据定理4.2.8，我们可利用已经证明过的内定理构造证明序列：

推论 4.2.9 若公式A存在如下序列，则也是内定理：

$$A_1$$
$$A_2$$
$$\vdots$$
$$A_n$$

其中$A_n = A$，且满足对任意的$1 \le i \le n$有下列两种情况之一成立：

(1) A_i是**某个（已证明的）内定理**的代入实例；

(2) A_i由排在它前面的两个公式A_j和$A_k = A_j \to A_i$根据**分离规则**得到。

对比自然推理系统派生规则的使用，读者不难想到，已证明内定理的使用本质上也可看作是将证明该内定理的证明序列使用合适的符号进行代入后补充到当前内定理的证明序列中。读者可仿照上一章对派生规则使用的说明，通过例子验证这一点。

为了更容易地构造证明序列，我们将公理$(A2)$所表达的推理传递性用下面的引理给出：

引理 4.2.10 对任意公式A, B, C，如果$A \to B$和$B \to C$都是内定理，则$A \to C$也是内定理。

证明 当$A \to B$和$B \to C$是内定理时，可用如下的证明序列验证$A \to C$是内定理：

(1)	$(B \to C) \to (A \to (B \to C))$	// 公理$(A1)$
(2)	$B \to C$	// 已知是内定理
(3)	$A \to (B \to C)$	// (1),(2)分离规则
(4)	$(A \to (B \to C)) \to ((A \to B) \to (A \to C))$	// 公理$(A2)$
(5)	$(A \to B) \to (A \to C)$	// (3),(4)分离规则
(6)	$A \to B$	// 已知是内定理
(7)	$A \to C$	// (5),(6)分离规则

□

在后面构造证明序列时，我们将上述定理的使用称为**传递规则**，读者可将这种使用看作在作合适的符号代入后，把上述证明序列补充到所要构造的证明序列中。

例子 4.2.11 公式（模式）$\neg A \to (A \to B)$是内定理，验证它的证明序列如下：

(1)	$\neg A \to (\neg B \to \neg A)$	// 公理$(A1)$
(2)	$((\neg B \to \neg A) \to (A \to B))$	// 公理$(A3)$
(3)	$\neg A \to (A \to B)$	// (1),(2)传递规则

该公式的直观含义是，如果A为假（即A的否定真），则A可推出任意公式B，这是命题逻辑推理的基本特点之一。

例子 4.2.12 公式$\neg\neg A \to A$是内定理，验证它的证明序列如下：

(1)	$(\neg\neg A) \to (\neg A \to \neg\neg\neg A)$	// $\neg A \to (A \to B)$的代入实例
(2)	$((\neg A \to \neg\neg\neg A) \to (\neg\neg A \to A))$	// 公理$(A3)$
(3)	$(\neg\neg A) \to (\neg\neg A \to A))$	// (1),(2)传递规则
(4)	$(\neg\neg A \to (\neg\neg A \to A)) \to ((\neg\neg A \to \neg\neg A) \to (\neg\neg A \to A))$	// 公理$(A2)$
(5)	$(\neg\neg A \to \neg\neg A) \to (\neg\neg A \to A)$	// (3),(4)分离规则
(6)	$(\neg\neg A \to \neg\neg A)$	// $A \to A$的代入实例
(7)	$\neg\neg A \to A$	// (5),(6)分离规则

例子 4.2.13 公式$A \to \neg\neg A$是内定理，验证它的证明序列如下：

(1)	$(\neg\neg\neg A) \to (\neg A)$	// $\neg\neg A \to A$的代入实例
(2)	$((\neg\neg\neg A \to \neg A) \to (A \to \neg\neg A))$	// 公理$(A3)$
(3)	$A \to (\neg\neg A)$	// (1),(2)分离规则

至此我们得到了公理化演算系统中的双重否定律，即A可推出$\neg\neg A$，$\neg\neg A$也可推出A，与自然推理系统比较，发现这里用于证明双重否定律的证明序列复杂而且需要十分熟练的技巧才能得到。相比较而言，自然推理系统中的推理形式都是带有前提集的推理，而内定理的证明相当于无前提的推理，前者简单而后者要困难许多。

从计算机实现的角度看，我们都可编写程序自动验证自然推理系统的证明序列，以及公理化演算系统的证明序列。但是不难想到，自然推理系统的程序可能编写起来更为困难一些，因为它涉及太多的规则。但从自动构造证明序列的角度看，对于自然推理系统，我们可发现更多的启发式规则来辅助程序进行推理，而公理化演算系统的证明序列构造可以说很难用计算机自动实现。

人工智能有称为**自动定理证明**的专门分支研究与使用计算机辅助进行定理证明的相关问题，其中教材[4]介绍的**归结推理法**及**王浩算法**就是该领域的研究成果，由于授课时数限制，这里不作介绍，有兴趣的读者可根据教材及相关参考文献自己学习。

在公理化演算系统中也可引入带前提集的推理：

定义 4.2.14 设$\Gamma=\{A_1,A_2,\cdots,A_n\}$是有穷个公式的集合，$A$是任意公式，称前提集$\Gamma$（合乎逻辑地）**推出**结论$A$，记为$\Gamma\vdash A$，如果存在公式序列：

$$\begin{array}{c}B_1\\B_2\\\vdots\\B_m\end{array}$$

使得$A=B_m$，且对任意的$1\le i\le m$满足：

1. B_i是公理$(A1),(A2),(A3)$之一的代入实例；
2. 存在$1\le j\le n$使得$B_i=A_j$，即B_i属于Γ；
3. 存在$1\le j,k<i$使得$B_k=B_j\to B_i$，即B_i是由B_j和B_k使用分离规则得到。

我们将上述公式序列称为$\Gamma\vdash A$的**证明序列**。

与内定理的证明序列相比，带前提的推理$\Gamma\vdash A$允许使用Γ中的公式参与证明序列的构造。显然当Γ等于空集$\varnothing$时，$\varnothing\vdash A$表明A就是内定理，这时通常省略前面的空集符号，直接记为$\vdash A$。

备注 4.2.15 在自然推理系统中，我们将推理的形式结构就定义为$\Gamma\vdash A$，这时严格地说，$\Gamma\vdash A$并不表明Γ能合乎逻辑地推出A，只有说$\Gamma\vdash A$是有效的推理时，才能断定Γ能合乎逻辑地推出A。不过在自然推理系统中，我们也只有当$\Gamma\vdash A$是有效的推理时，才使用这样的符号，因此自然推理系统的$\Gamma\vdash A$的含义我们也可理解为与公理化演算系统中的含义相同：即，$\Gamma\vdash A$就表明Γ能合乎逻辑地推出A。类似地，前面我们要说“证明A是内定理”，后面则只说“证明$\vdash A$”即可。

对于前提集的书写，我们采用与自然推理系统相同的处理方式，即省略表示集合的花括号，而且使用逗号表示前提集的并。例如，当$\Gamma=\{A_1,A_2,\cdots,A_n\}$时，$\Gamma\vdash A$直接写为$A_1,A_2,\cdots,A_n\vdash A$。$\Gamma,B\vdash A$表示$\Gamma\cup\{B\}\vdash A$等。

例子 4.2.16 $A, B \to (A \to C) \vdash B \to C$的证明序列如下：

(1)	A	// 前提
(2)	$A \to (B \to C)$	// 公理$(A1)$
(3)	$B \to A$	// (1),(2)分离规则
(4)	$B \to (A \to C)$	// 前提
(5)	$(B \to (A \to C)) \to ((B \to A) \to (B \to C))$	// 公理$(A2)$
(6)	$(B \to A) \to (B \to C)$	// (4),(5)分离规则
(7)	$B \to C$	// (3),(6)分离规则

不难看到，仅仅使用前提、公理以及分离规则构造带前提推理的证明序列仍然十分困难。于公理化演算系统有一条十分有用的关于所构造的证明序列性质的定理（不是内定理），即**演绎定理**：

定理 4.2.17 **演绎定理**：$\Gamma, A \vdash B$当且仅当$\Gamma \vdash A \to B$；特别地，$A \vdash B$当且仅当$\vdash A \to B$。

在自然推理系统中，由$\Gamma, A \vdash B$得到$\Gamma \vdash A \to B$是蕴涵引入定理，而且也很容易利用蕴涵消除规则（及弱化规则）证明：$\Gamma \vdash A \to B$是有效的，则$\Gamma, A \vdash B$也是有效的。但在公理化演算系统，演绎定理的证明十分复杂，因此这里不作详细介绍。

演绎定理使得在构造证明序列时可灵活地变换前提集，从而不仅简化带前提推理的证明序列的构造，也可简化内定理的证明序列的构造，而且将带前提推理与内定理十分紧密地联系在一起。为明确前提的变化，与自然推理系统类似，后面的证明序列也采用带有前提的序列：

$$\Gamma_1 \vdash B_1$$
$$\Gamma_2 \vdash B_2$$
$$\vdots$$
$$\Gamma_m \vdash B_m$$

这时证明序列要满足的条件是：对任意的$1 \le i \le m$，$\Gamma_i \vdash B_i$满足下列条件之一：

1. $B_i \in \Gamma_i$，即B_i属于前提集Γ_i；
2. B_i是公理或已证明的内定理的代入实例；
3. 存在$1 \le j, k < i$使得$\Gamma_i = \Gamma_j = \Gamma_k$且$B_k = B_j \to B_i$，这时仍称由分离规则得到。

注意，**分离规则必须在前提集相同的情况下才能使用**。

最后，为与自然推理系统对比，我们在公理化演算系统中证明自然推理系统中几个与否定与蕴涵联结词有关的推理规则。前面已经证明了双重否定律、传递规则，以及矛盾律（参见例子4.2.11证明的内定理$\neg A \to (A \to B)$），假言易位相当于公理$(A3)$，因此下面考虑自然推理系统的否定引入律，归谬律以及假言易位规则的变体。

对于自然推理系统的否定引入规则

$$\frac{\Gamma, A \vdash B \qquad \Gamma, A \vdash \neg B}{\Gamma \vdash \neg A}$$

其本质的含义是，如果前提集Γ, A可推出B和$\neg B$，则Γ可推出$\neg A$。

为了将其翻译成公理化演算系统中的带前提推理，不能针对任意的前提集Γ，实际上可针对空的前提集来进行推理，即若由A可推出B和$\neg B$，那么$\neg A$就是内定理。进一步，根据演绎定理，$A \vdash B$相当于$\vdash A \to B$，因此上述规则在公理化演算系统可解读为:“如果$A \to B$和$A \to \neg B$是内定理，则$\neg A$是内定理”，再翻译成带前提的推理就是要证明：

$$A \to B,\ A \to \neg B \vdash \neg A \text{ 或内定理形式：} \quad \vdash (A \to B) \to ((A \to \neg B) \to \neg A)$$

自然推理系统的矛盾律

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash \neg A}{\Gamma \vdash B}$$

按照上面的翻译方法，相当于要证明：$A, \neg A \vdash B$，或内定理形式：$\vdash A \to (\neg A \to B)$。这正是例子4.2.11已经证明的内定理。对于自然推理系统的归谬律：

$$\frac{\Gamma, \neg A \vdash B \qquad \Gamma, \neg A \vdash \neg B}{\Gamma \vdash A}$$

翻译到公理化演算系统就是要证明：$\neg A \to B, \neg A \to \neg B \vdash A$，或内定理形式：

$$\vdash (A \to B) \to ((\neg A \to \neg B) \to A)$$

对于自然推理系统的假言易位规则：

$$\frac{\Gamma \vdash A \to B \qquad \Gamma \vdash \neg B}{\Gamma \vdash \neg A}$$

翻译到公理化演算系统就是要证明：$A \to B, \neg B \vdash \neg A$，或内定理形式：$\vdash (A \to B) \to (\neg B \to \neg A)$。而假言易位规则的变体：

$$\frac{\Gamma \vdash \neg A \to \neg B \qquad \Gamma \vdash B}{\Gamma \vdash A}$$

则相当于：$\neg A \to \neg B, B \to A$，或内定理形式：$\vdash (\neg A \to \neg B) \to (B \to A)$，显然这就是公理$(A3)$。因此下面我们主要是证明：

$$A \to B,\ A \to \neg B \vdash \neg A$$
$$\neg A \to B, \neg A \to \neg B \vdash A$$
$$A \to B, \neg B \vdash A$$

例子 4.2.18 首先可容易地证明 $A \to B \vdash \neg\neg A \to \neg\neg B$：

(1) $A \to B, \neg\neg A \vdash \neg\neg A$ // 前提

(2) $A \to B, \neg\neg A \vdash \neg\neg A \to A$ // 内定理$\neg\neg A \to A$

(3) $A \to B, \neg\neg A \vdash A$ // (1),(2)分离规则

(4) $A \to B, \neg\neg A \vdash A \to B$ // 前提

(5) $A \to B, \neg\neg A \vdash B$ // (3),(4)分离规则

(6) $A \to B, \neg\neg A \vdash B \to \neg\neg B$ // 内定理$B \to \neg\neg B$

(7) $A \to B, \neg\neg A \vdash \neg\neg B$ // (5),(6)分离规则

(8) $A \to B \vdash \neg\neg A \to \neg\neg B$ // (7)演绎定理

读者可以看到，在前面证明了内定理$\neg\neg A \to A$的情况下，很容易地由$\Gamma \vdash \neg\neg A$得到$\Gamma \vdash A$。一般地，我们可根据下面的引理简化证明序列的构造：

引理 4.2.19 如果有$\vdash A \to B$，则对任意的前提集Γ有：若$\Gamma \vdash A$则$\Gamma \vdash B$。

例子 4.2.20 在上面例子的基础上，可利用公理$(A3)$证明$A \to B, \neg B \vdash \neg A$：

(1) $A \to B \vdash A \to B$ // 前提

(2) $A \to B \vdash \neg\neg A \to \neg\neg B$ // 例子4.2.18

(3) $A \to B \vdash (\neg\neg A \to \neg\neg B) \to (\neg B \to \neg A)$ // 公理(A3)

(4) $A \to B \vdash \neg B \to \neg A$ // (2),(3)分离规则

(5) $A \to B, \neg B \vdash \neg A$ // (4)演绎定理

例子 4.2.21 为证明否定消除规则，先利用矛盾律$\vdash A \to (\neg A \to B)$证$A \to B, A \to \neg B, A \vdash C$：

(1) $A \to B, A \to \neg B, A \vdash A$ // 前提

(2) $A \to B, A \to \neg B, A \vdash A \to B$ // 前提

(3) $A \to B, A \to \neg B, A \vdash B$ // (1),(2)分离规则

(4) $A \to B, A \to \neg B, A \vdash B \to (\neg B \to C)$ // 矛盾律

(5) $A \to B, A \to \neg B, A \vdash \neg B \to C$ // 矛盾律

(6) $A \to B, A \to \neg B, A \vdash A \to \neg B$ // 前提

(7) $A \to B, A \to \neg B, A \vdash \neg B$ // (1),(6)分离规则

(8) $A \to B, A \to \neg B, A \vdash C$ // (5),(7)分离规则

实际上这是矛盾律的推广形式：即如果前提A能推出B和$\neg B$，那么A能推出任何公式C。

例子 4.2.22 为证明否定消除规则，再利用矛盾律证明$\neg A \to A \vdash A$：

(1) $\neg A \to A \vdash \neg A \to (A \to \neg(\neg A \to A))$ // 矛盾律

(2) $\neg A \to A \vdash (\neg A \to (A \to \neg(\neg A \to A)))$
$\to ((\neg A \to A) \to (\neg A \to \neg(\neg A \to A)))$ // 公理$(A2)$

(3) $\neg A \to A \vdash (\neg A \to A) \to (\neg A \to \neg(\neg A \to A))$ // (1),(2)分离规则

(4) $\neg A \to A \vdash (\neg A \to A)$ // 前提

(5) $\neg A \to A \vdash \neg A \to \neg(\neg A \to A)$ // (3),(4)分离规则

(6) $\neg A \to A \vdash (\neg A \to \neg(\neg A \to A)) \to ((\neg A \to A) \to A)$ // 公理$(A3)$

(7) $\neg A \to A \vdash (\neg A \to A) \to A$ // (5),(6)分离规则

(8) $\neg A \to A \vdash A$ // (4),(7)分离规则

例子 4.2.23 在上面两个例子的基础上，可证明否定消除规则$\neg A \to B, \neg A \to \neg B \vdash A$：

(1) $\neg A \to B, \neg A \to \neg B, \neg A \vdash A$ // 例子4.2.21

(2) $\neg A \to B, \neg A \to \neg B \vdash \neg A \vdash A$ // (1)演绎定理

(3) $\neg A \to B, \neg A \to \neg B \vdash (\neg A \vdash A) \to A$ // 例子4.2.22

(4) $\neg A \to B, \neg A \to \neg B \vdash A$ // (2),(3)分离规则

例子 4.2.24 最后，利用否定消除规则的内定理形式

$$\vdash (\neg\neg A \to \neg\neg B) \to ((\neg\neg A \to \neg\neg\neg B) \to \neg A)$$

以及双重否定律可证明归谬律$A \to B, A \to \neg B \vdash \neg A$：

(1) $A \to B, A \to \neg B \vdash$
$(\neg\neg A \to \neg\neg B) \to ((\neg\neg A \to \neg\neg\neg B) \to \neg A)$ // 否定消除规则

(2) $A \to B, A \to \neg B \vdash A \to B$ // 前提

(3) $A \to B, A \to \neg B \vdash \neg\neg A \to \neg\neg B$ // 例子4.2.18

(4) $A \to B, A \to \neg B \vdash (\neg\neg A \to \neg\neg\neg B) \to \neg A$ // (1),(3)分离规则

(5) $A \to B, A \to \neg B \vdash A \to \neg B$ // 前提

(6) $A \to B, A \to \neg B \vdash \neg\neg A \to \neg\neg\neg B$ // 例子4.2.18

(7) $A \to B, A \to \neg B \vdash \neg A$ // (4),(6)分离规则

与自然推理系统一样，上面例子中的证明序列都是严格按照代入规则、分离规则以及已经证明过的内定理进行构造，这种构造是纯粹符号化的变换，与公式是否具有真值没有任何关系。自然推理系统中的推理有效性的定义虽然建立在永真式的基础上，但是实际上也可使用归纳法定义而不涉及公式的真值，只是这种定义比较复杂而我们没有采用而已。自然推理系统的规则多，因此证明序列的构造比较简单，而公理化演算系统的规则少，因此证明序列的构造需要洞察力以及技巧。

4.3 形式系统的元理论

这一节我们简单介绍与形式系统元理论有关的内容。所谓**元理论**(meta theory)是指在形式系统本身之上的理论，或者说是关于形式系统本身的理论。严格地说，形式系统内部的理论主要是形式语法部分的合式公式的构造以及内定理的证明，除此以外的内容都可以说属于形式系统的元理论。

形式系统的元理论可分为两个大的方面：

1. **证明论**方面：研究形式语言语法构造以及内定理证明的一些证明论性质，例如可否判定一个符号串属于形式语言，可否判定一个公式是内定理，可否判定一个公式序列是某内定理合法的证明序列，能否自动构造内定理的证明序列等等问题都属于形式系统元理论研究的内容之一；

2. **模型论**方面：研究形式系统的语义解释，并探讨语义模型与形式语法及形式演算部分之间的联系，例如形式系统的一致性、完全性、公理的独立性等问题也都属于形式系统元理论研究的重要内容。

对于命题演算系统而言：(i) 可以机械地判定一个符号串是否属于形式语言合法的句子；(ii) 可以机械地判定一个公式序列是否是合法的证明序列。人工智能的分支－－自动定理证明则对证明一个公式是否是内定理，以及如何由计算机辅助完成内定理的证明作了更为深入的研究。

这一节所要讨论的命题演算系统元理论主要是讨论命题演算系统的语义模型，及命题演算系统的一致性与完全性，至于演算系统的上述可判定性，以及公理的独立性等问题我们不再介绍，有兴趣的读者可参考相关文献资料。

4.3.1 公理化演算系统的形式语义

形式系统的语义首先要给出形式语法部分所定义的每个语法成份的解释，并且探讨这种解释与形式演算部分之间的关系。对形式系统每个语法成份的解释构成了形式系统的语义模型，此模型首先是确定一个**论域**(domain)，即形式系统的所有语法成份均被解释为该论域中的元素。论域上有一定的结构，此结构与形式系统的语法规则（也就是形式系统构造语法成份的规则）相对应。

上面给出的命题逻辑公理化演算$\mathcal{P}$系统的基本语法成份是命题逻辑公式。前几章所讨论的公式真值则给出了公式的一种解释，这种解释以$\mathbf{bool}=\{0,1\}$为论域，所有公式都解释为论域的0或1值，即公式的真值。公式的解释（真值）是由公式的语法构造所确定。公式的语法构造是以命题变量符号为基本符号，以联结词为构造规则进行构造，对于公理化演算系统而言，命题逻辑公式的归纳定义是：

1. **归纳基**：命题变量符号是公式；
2. **归纳步**：如果A,B是公式，则$(\neg A)$和$(A\to B)$是公式。

我们前几章已经看到，公式的语义解释，即公式的真值定义也与上述定义对应：首先确定命题变量符号的真值，即给定一个真值赋值函数t，然后根据联结词的性质归纳定义公式在t下的真值$t(A)$：

1. **归纳基**：若公式A是命题变量符号p，则$t(A)=t(p)$；
2. **归纳步**：如果A具有形式$(\neg B)$，则$t(A)=1$当且仅当$t(B)=0$；如果公式A具有形式$(B\to C)$，则$t(A)=0$当且仅当$t(B)=1$而且$t(C)=0$。

简单地说，命题逻辑的公理化演算系统的基本语法成份是公式，该演算系统的语义模型就是要给出公式的解释，以$\mathbf{bool}=\{0,1\}$为论域的模型把公式的真值看作公式的解释。上述公式真值的归纳定义实际上基于**bool**上的真值函数（参见联结词完备集一节的定义），或者说基于**bool**上的布尔操作而构成的布尔代数。进一步可概括地说**布尔代数就是命题演算系统的语义模型**。

4.3.2 公理化演算系统的一致性与完备性

形式系统元理论研究的主要内容之一就是探讨形式系统语义模型与形式系统本身，特别是形式系统演算部分之间的关系，其中最重要的关系是**一致性**(consistence)与**完备性**(completeness)。

演算系统的一致性

一致性又称为**可靠性**、**无矛盾性**、**和谐性**以及**相容性**，直观地看，就是要求演算系统本身不会存在逻辑矛盾，这是对一个形式系统的基本要求。

形式系统的一致性可以从几个方面定义：

1. **一致性的古典定义**：古典定义也就是最直观的定义，即要求演算系统不会推出矛盾的结论。符号化一点地说，演算系统的**古典一致性**是要求**演算系统不能既推出公式A，又推出公式A的否定作为它的内定理**。

2. **一致性的语法定义**：一致性也可纯粹从形式系统本身的角度定义，即不涉及形式系统的语义模型的角度定义。演算系统的**语法一致性**是要求**不是所有的公式都是演算系统的内定理**。显然如果演算系统不是语法一致的，也就是说所有公式都是内定理，那么它也就是不是古典一致的，而对于公理化演算系统$\mathcal{P}$而言，我们知道矛盾的前提可推出任何结论，因此如果它不是古典一致的，即能推出A及A的否定，那么能推出任何公式作为内定理，从而就不是语法一致的。

3. **一致性的语义定义**：通常，形式系统的古典一致性和语法一致性是比较难以研究和证明的，因此往往是通过探讨形式系统的语法和演算部分与它的语义模型之间的关系来研究形式系统的一致性。直观地说，形式系统的语义一致性就是要求在形式系统内部认为好的公式都是语义模型认为好的公式。具体到命题演算系统而言，**语义一致性**就是要求**所有内定理都是永真式**。

形式系统的语法一致性又称为**内在无矛盾性**，语义一致性又称为**外在无矛盾性**，这是将形式系统的语法和演算部分看作形式系统内部（或说形式系统本身）的内容，而形式系统的语义模型看作形式系统外部的内容。

形式系统的语义一致性，从形式系统的角度看又称为形式系统的**可靠性**，意味着形式系统不会得到语义模型所不需要的东西。从某种意义上说，形式系统的语义模型往往对应着客观世界，是我们要求解的问题域，形式系统是这个问题域的抽象。

形式系统的语义一致性，从语义模型的角度看又称为语义模型的**合理性**(soundness)，意味着对形式系统的语法成份所作的解释与它的演算部分是相容的、一致的，从而表明这种语义解释本身是合理的。

显然，对于上面所讨论的公理化演算系统$\mathcal{P}$而言，它是古典一致的，也是语法一致的，相对与上述真值语义模型，也是语义一致的。很容易看到，$\mathcal{P}$的所有公理（的代入实例）都是永真式，而分离规则则保证从永真式只能得到永真式，即若A和$A \to B$都是永真式的话，那么B必然是永真式，因此$\mathcal{P}$的所有内定理必然都是永真式，这就意味着它是语义一致的，从而这也意味着不是永真式的公式不是它的内定理，从而它是语法一致的，也是古典一致的。

演算系统的完备性

形式系统的完备性(completeness)又翻译成为**完全性**，是形式系统元理论研究的另一个重要问题，但相对于一致性而言，形式系统的完备性不是一定要满足的性质。从另一个角度说，一致性是对形式系统的基本要求，而完备性是对形式系统的更高要求，也是一个更难研究和证明的性质。

形式系统的完备性也可从几个方面定义：

1. **完备性的古典定义**：古典完备性也是最简单和直观的完备性定义，即要求对任意的公式A，A和它的否定至少有一个是形式系统的内定理。这个要求是很强的，通常大多数形式系统都满足这个定义，例如公理化演算系统$\mathcal{P}$的所有内定理都是永真式，这意味着非永真式的可满足式及其否定都不是内定理，因而$\mathcal{P}$不是古典完全的。

2. **完备性的语法定义**：完备性也可纯从形式系统内部的角度进行定义，形式系统的**语法完备性**是指对任意不是内定理的公式A，如果将A增加为形式系统的公理，则该形式系统就不再是一致的。

3. **完备性的语义定义**：完备性的语义定义是从形式系统内部和形式系统的语义模型之间关系的角度定义。形式系统的语义完备性与形式系统的语义一致性相对应，直观地说，就是语义模型认为好的东西都是形式系统的演算部分能推出的东西。具体到命题演算系统而言，**语义完备性**就是指所有的永真式都是演算系统的内定理，或者说，形式系统能推出所有的永真式作为它的内定理。

对于上面所讨论的公理化演算系统$\mathcal{P}$而言，它不是古典完备的，但是**是语义完备和语法完备的**，要证明一点，比证明它的一致性要难，此处不再作深入的讨论，有兴趣的读者可参考相关文献资料，大多数专门讨论数理逻辑的书籍都会讨论命题演算系统的完备性，如[5, 6, 7, 8, 2]等，有关命题演算系统元理论的更多内容都可参考这些文献。

作业

作业 4.1 谈谈你对形式系统的理解。

作业 4.2 命题逻辑的自然推理系统也是一个命题演算系统，请仿照上面定义公理化演算系统的方式，考虑自然推理系统的内定理是什么？并给出它的归纳定义。

作业 4.3 试证明公式$(\neg\neg A \to \neg\neg B) \to (A \to B)$是系统$\mathcal{P}$的内定理。

作业 4.4 谈谈你对形式系统的一致性和完备性的理解。

第五章　一阶逻辑的基本概念

前面几章讨论了命题逻辑的基本概念、命题逻辑的等值演算以及基于命题逻辑的推理，我们看到，命题逻辑揭示了逻辑推理的一些基本特点，如假言推理、析取三段论、假言三段论、假言易位、归谬律等，可用于解决数学乃至日常生活中的一些应用问题。但是命题逻辑还比较简单，因而在表达推理方面仍存在能力不足的问题，例如，下面是日常生活中认可的常见推理：

所有人都要呼吸；张三是人；所以，张三要呼吸

如果在命题逻辑中对此推理进行分析，则：(i)“所有人都要呼吸”是一个原子命题，用p表示；(ii)“张三是人”也是原子命题，用q表示；(iii)“张三要呼吸”也是原子命题，用r表示，则上述推理用命题逻辑公式表示为$p \wedge q \rightarrow r$，这个公式不是永真式，因此不是公理化演算系统的内定理，也就不会是自然推理系统中的有效推理，也即**在命题逻辑中上述推理不是有效的推理**。

上述例子充分说明了命题逻辑的不足，关键在于命题逻辑只能将上述例子中的三个句子符号化成三个独立的原子命题，而不能揭示句子之间的密切联系。要揭示这种联系，我们需要**对原子命题的结构作进一步的分析，谓词逻辑正是基于这一点而建立起来的，它将原子命题进一步分解为个体与谓词等部分，并引入函数、量词等概念，对原子命题作更为精细的符号化**，从而对逻辑推理的性质作更深入的研究。

我们介绍谓词逻辑的内容框架与命题逻辑基本一致，首先介绍有关谓词逻辑，主要是一阶谓词逻辑的基本概念，并讨论自然语句在一阶谓词逻辑中的符号化，然后定义一阶谓词逻辑公式的语法和语义解释，并在此基础讨论一阶谓词逻辑公式的等值演算，最后讨论基于一阶谓词逻辑的推理。

5.1　个体、谓词与量词

谓词逻辑将原子命题的结构作进一步的分析，引入个体、谓词、函数以及量词等概念，对原子命题作更精细的符号化。我们知道，原子命题是一个陈述句，对某个或某些实体是否具有某种性质或关系进行判断。在谓词逻辑中，**将原子命题所判断的对象或说实体**称为**个体**，**将原子命题中所断定的性质或关系**称为**谓词**。

例如，对于命题“张三是人”，其中的“张三”就是个体，而“…是人”是对个体性质的判断，是谓词。在谓词逻辑中，用小写字母表示个体，特别地，用字母表前面的字符，如a, b, c等表示像“张三”这样的具体的个体，称为**个体常量**，而用字母表后面的字符，如x, y, z等表示任意的个体，称为**个体变量**。在谓词逻辑中，用大写字母，特别是字母表中间的大写字母，如F, G, H等表示谓词，例如用$F(x)$符号化“…是人”这个谓词，其中的x是个体变量，表示这个谓词可作用于任何一个个体。这时的个体变量符号x只是占位，表明谓词F是对**一个个体**的性质判断。

如果谓词是对几个个体之间的关系进行判断，则使用几个不同的个体变量符号表明，例如命题“张三与李四是同学”中的“张三”和“李四”毫无疑问都是个体，而“…与…是同学”则是谓词，它对两个个体之间的关系进行判断，通常用$G(x,y)$这种形式对其进行符号化。

因此在谓词逻辑中对自然语言命题进行符号化时，**先分清其中的个体与谓词，具体的个体用个体常量符号表示，任意的个体或不确定的个体用个体变量表示，而谓词用大写字母表示，并用个体变量符号标记它是对一个个体的性质判断，还是对多个个体之间的关系的判断**。因此：

(1) 命题“张三是人”在谓词逻辑中进行符号化时，首先用个体常量符号a表示“张三”，用$F(x)$表示谓词“x是人”，整个命题符号化为$F(a)$；

(2) 命题“张三与李四是同学”在谓词逻辑中进行符号化时，首先用个体常量符号a表示“张三”，b表示“李四”，用$G(x,y)$表示谓词“x与y是同学”，整个命题符号化为$G(a,b)$。

如果谓词是对一个个体的性质进行判断，就称为**一元谓词**，如果它对n个个体之间的关系进行判断，就称为**n元谓词**，显然一元谓词是n元谓词的一种特殊情况，在大多数情况，我们碰到的都是一元、二元谓词，三元及三元以上的谓词比较少见。我们在对谓词进行符号化时，使用个体变量符号进行占位以清楚地表明谓词的元数。

为了精确符号化像“所有的人都要呼吸”这种命题，谓词逻辑引入**量词**指明谓词所判断的个体范围，“所有的人都要呼吸”是对人这样的个体的**全体**进行判断，我们将用到**全称量词**，而“有的人会画画”是对人这样的个体的**部分**进行判断，我们将用到**存在量词**。

在符号化这种涉及到量词的命题之前，我们先要明确一个所谓**个体域**这样的概念：**个体域是某类个体全体的总称，是代表这类个体的个体变量的取值范围**。例如，前面谈到的“人”就是一个个体域，“张三”、“李四”都是其中的个体。

对于句子“所有的人都要呼吸”，显然“…要呼吸”是一个一元谓词，可用符号$F(x)$表示，句子的主语“所有的人”并不是特指某个具体的个体，因此不能用个体常量符号表示，也不是表示任意的某个个体，也不能用个体变量符号表示，而是指人这种个体的全体，因此我们用全称量词来符号化，将整个句子符号化为$\forall xF(x)$，可读作：“所有的x，$F(x)$”，或：“任意的x，$F(x)$”。这里$\forall x$中的x称为**指导变元**，$\forall x$表明谓词$F(x)$是对其中个体变量x的取值范围，即x的个体域的全部个体进行判断。

对于句子“有的人会画画”，显然“…会画画”是一个一元谓词，可用符号$G(x)$表示，句子的主语“有的人”是指人这种个体的部分，我们用存在量词符号化，将整个句子符号化为$\exists xG(x)$，读作：“存在x，$G(x)$”。同样，这里$\exists x$中的x也是**指导变元**，$\exists x$表明谓词$G(x)$是对其中个体变量x的个体域的部分个体进行判断。

上面两个句子涉及的个体域都是“人”，但有些命题可能会涉及到不同的个体域，例如命题“兔子比乌龟跑得快”中的“兔子”和“乌龟”显然并不是指某个具体的兔子和某个具体的乌龟，而是指“所有的兔子都比所有的跑得快”，这里就涉及到两个个体域，一个是“兔子”，一个是“乌龟”，如果用$H(x,y)$表示“x比y跑得快”这个二元谓词，那么这个句子可符号化为$\forall x\forall yH(x,y)$。

但是用$\forall x\forall yH(x,y)$符号化“兔子比乌龟跑得快”这个句子之后，我们每次提到它都要说明其中的x的个体域是“兔子”，y的个体域是“乌龟”，否则它就不符合这个句子的本意，例如x的个体域不能是“乌龟”，y的个体域不能是“兔子”。每次要指明个体变量的个体域显然很麻烦，因此引入了所谓**全总域**的概念：**全总域就是正在研究的问题范围内的所有个体全体**。后面如果没有特别说明，**所有个体变量的取值范围都是全总域**。

将所有个体变量的取值范围都确定为全总域之后，句子“所有的人都要呼吸”不能再符号化

为$\forall x F(x)$，因为这时它表示对全总域的任意个体都要呼吸，显然不合句意。这时需要引入所谓**特征谓词**来描述句子是对全总域的哪一类个体进行判断。例如，句子“所有的人都要呼吸”是对人这种个体进行判断，因此要引入谓词$M(x)$表示“x是人”，并把此句理解为：“对任意的个体x，如果x是人，那么x要呼吸”，从而符号化为$\forall x(M(x) \to F(x))$，这里谓词$F(x)$仍表示“x要呼吸”。

类似地，对于句子“有的人会画画”，同样要引入特征谓词$M(x)$表示“x是人”，并把句子理解为“存在某个（不确定的）个体x，x是人而且x会画画”，从而将句子符号化为$\exists x(M(x) \wedge G(x))$，这里$G(x)$仍表示“$x$会画画”。

备注 5.1.1 **特征谓词**仍是谓词，是在引入全总域之后，通常句子是对全总域的某一类个体的全体或部分进行性质或关系进行判断，从而需要使用谓词表示所判断的这一类个体的基本特征，这种谓词被称为**特征谓词**。特征谓词实际上是在全总域的范围内概括出句子真正针对的个体域。特征谓词所描述的个体域，通常是由句子中像“所有的”、“任意的”、“有的”、“存在”等这种词语所修饰的名词给出。

在假定所有个体变量的取值范围都是全总域的情况下，命题在谓词逻辑的符号化需要使用特征谓词，这时需要正确分析**特征谓词句子所判断性质或关系的谓词之间的逻辑关联**，下一节我们会就命题在谓词逻辑的符号化做更深入的讨论。

最后，在谓词逻辑中还可能需要使用**函数**，**函数用于将基本的个体转换为复杂的个体**。例如，对于句子“π的平方是无理数”，显然“x是无理数”是谓词，用$W(x)$表示，“π的平方”可直接符号化为个体常量a，但为更精确地表达句意，通常引入函数$f(x)$表示x的平方，而用个体常量b表示个体π，从而整个句子符号化为$W(f(b))$。简单地说，**在谓词逻辑中引入函数是为了能够描述复杂的个体**。

小结

为精确符号化自然语言表示的命题，谓词逻辑对命题的结构作进一步的分析，引入了个体、谓词、量词以及函数等概念：

1. **个体**与**谓词**：个体是命题所判断的对象或实体；谓词是命题所断定的性质或关系。
2. **个体常量**与**个体变量**：个体常量是指具体的、确定的个体；个体变量是指任意的或不确定的个体。
3. **谓词的元数**：谓词的元数描述谓词是对个体的性质进行判断，还是对个体之间的关系进行判断。一元谓词对个体的性质进行判断，而n元谓词对n个个体之间的关系进行判断。
4. **全称量词**与**存在量词**：当句子对某一类个体的全体或者部分的性质或关系进行判断时，需要使用量词符号化句子。当句子是对某类个体的全体进行判断时使用全称量词，而当句子是对某类个体的部分进行判断时使用存在量词。
5. **个体域**：因为量词是对某一类个体的全体或者部分进行判断，因此要确定这类个体的范围，这个范围就是个体域，也是**量词指导变元的取值范围**。
6. **全总域**与**特征谓词**：为了统一所有个体变量的取值范围而引入全总域的概念，全总域是指研究范围内所有个体的全体；在假定个体变量的取值范围是全总域的情况下，特征谓词用于概括出句子所判断的某类个体的基本特征。
7. **函数**：函数用于描述复杂的个体如何由基本的个体变换而得到。**函数作用于个体仍然是个体，而谓词作用于个体则形成命题**。

5.2 自然语言命题的符号化

5.2.1 不涉及量词的命题的符号化

这一节我们讨论在谓词逻辑中对自然语言命题进行符号化。首先如果命题中没有“所有的”、“任意的”、“有的”、“存在”等词汇表示对某类个体的全体或部分的性质或关系进行判断，那么它的符号化不需要使用量词，只需要提炼出个体和谓词即可：

例子 5.2.1 将下列句子在谓词逻辑中进行符号化：

1. 中国是社会主义国家。
2. $2x + y > 10$
3. 小张与小李是要好的朋友。
4. 小张与小李都喜欢上网。
5. 郑州位于北京和广州之间。

解答：

1. 句子“中国是社会主义国家”中个体是“中国”，用个体变量符号c表示，“x是社会主义国家”是谓词，用$S(x)$表示，整个句子符号化为：$S(c)$。

2. 命题逻辑中不认为“$2x + y > 10$”是命题，因为其中含有公认是变量的符号，不具有确定的真假值。在谓词逻辑中，可将x, y理解为个体变量符号，$2x$中含有乘法符号，2是个体，$+$也是函数符号，而$>$是谓词，10是个体常量，可认为$2x + y > 10$就是符号化了的命题，可以在谓词逻辑中对其进行研究。但与命题逻辑一样，$2x + y > 10$本身也不具有确定的真值，只有在对个体变量符号x, y指派了具体的个体后，该公式才有确定的真值。参见后面对谓词逻辑公式语义解释的讨论。

3. 句子“小张与小李是要好的朋友”中的“小张”和“小李”都是个体，分别用a, b表示，“x与y是要好的朋友”是二元谓词，用$G(x, y)$表示，整个句子符号化为：$G(a, b)$。

4. 句子“小张与小李都喜欢上网”中的“小张”和“小李”同样都是个体，分别用a, b表示，而“x喜欢上网”是句子中的谓词，是一个一元谓词，用$H(x)$表示，整个句子符号化为：$H(a) \wedge H(b)$。

5. 句子“郑州位于北京和广州之间”中的“郑州”、“北京”以及“广州”都是个体，分别用a, b, c表示，句子中的谓词应该是“x位于y和z之间”，是一个三元谓词，用$T(x, y, z)$表示，整个句子符号化为：$T(a, b, c)$。

简单地说，如果命题只是对具体个体的性质或者关系进行判断，那么在谓词逻辑中的符号化不需要使用量词，只需要将具体个体用个体常量符号表示，并根据句子的含义提取其中的谓词，用谓词符号表示，谓词作用于个体常量用于符号化命题逻辑中认为是原子命题的句子，如果是复合命题，则与命题逻辑一样，分析支命题之间的逻辑关系并用合适的命题逻辑联结词表示。

5.2.2 涉及量词的基本命题的符号化

在谓词逻辑中，我们主要关心哪些涉及到量词的命题的符号化，这些命题才体现谓词逻辑的特点。如果命题是对某类个体的全体，或者对某类个体中不确定的个体进行性质或关系判断，那么这个命题的符号化需要使用量词。在普通逻辑中，将这种命题分为全称判断和特称判断。

全称判断是对某类个体的全体具有或不具有某种性质的判断，进一步可分为全称肯定判断和全称否定判断。全称肯定判断的标准形式是：“所有S都是P”，例如：“所有人都（是）要呼吸（的）”。当

个体变量的取值范围是全总域时，全称肯定判断理解为：“对任意的个体x，如果x是S，那么x是P（具有性质P）”，这里“x是S”就是**特征谓词**，假设我们就用谓词符号$S(x)$表示，“x是P”是对这类个体性质的进一步描述，也用谓词符号$P(x)$表示，那么全称肯定判断符号化为：$\forall x(S(x) \to P(x))$。

全称否定判断的标准形式是：“**所有S都不是P**”，例如，“所有的天鹅都不是黑的”。同样，全称否定判断应理解为：“对任意的个体x，如果x是S，则x不是P（不具有性质P）”，因而它的符号化应该是：$\forall x(S(x) \to \neg P(x))$。

特称判断是对某类个体至少存在一个个体具有或不具有某种性质的判断，进一步也可分为**特称肯定判断**和**特称否定判断**。特称肯定判断的标准形式是：“**有的S是P**”，例如：“有的人（是）会画画（的）”。特称肯定判断应理解为：“（至少）存在某个个体x，x是S，而且x是P（具有性质P）”，因此应该符号化为：$\exists x(S(x) \wedge P(x))$。

特称否定判断的标准形式是：“**有的S不是P**”，例如：“有的人不（是）会画画（的）”。特称否定判断应理解为：“（至少）存在某个个体x，x是S，而且x不是P（不具有性质P）”，因此应该符号化为：$\exists x(S(x) \wedge \neg P(x))$。

这里需要强调的是，**在全称判断中，特征谓词和其他谓词（对所判断个体性质的进一步描述）之间的关系是逻辑蕴涵关系**，而**在特称判断中，特征谓词和其他谓词之间的关系是逻辑与关系**。

例如，句子“所有人都要呼吸”中，特征谓词“$M(x)$: x是人”与谓词“$F(x)$: x要呼吸”之间是逻辑蕴涵关系，即句子的含义是：“对任意的x，如果x是人，那么x要呼吸”，整个句子符号化为：$\forall x(M(x) \to F(x))$，而**不能符号化为**：$\forall x(M(x) \wedge F(x))$。

而句子“有的人会画画”中，特征谓词“$M(x)$: x是人”与谓词“$H(x)$: x会画画”之间是逻辑与关系，即句子的含义是：“至少存在某个个体x，x是人，而且x会画画”，整个句子符号化为：$\exists x(M(x) \wedge H(x))$，而**不能符号化为**：$\exists x(M(x) \to H(x))$。

究其原因是由于**全称判断没有断定具有特征谓词所描述性质的个体的存在性**，而**特称判断断定了具有特征谓词所描述性质的个体的存在**。

例如，“所有人都要呼吸”没有断定“人”这种个体的存在，因此当完全不存在“人”这种个体时，这个句子仍然是真的。再例如，著名的牛顿定律“所有不受外力作用的物体都保持匀速直线运动”仅仅断定：对所有物体而言，如果它不受外力作用，那么它保持匀速直线运动；至于不受外力作用的物体是否存在，这个定律没有断定。实际上，这种物体是不存在的，但这个定律是真的。我们知道，逻辑蕴涵在前件假时，整个命题为真，这符合全称判断中特征谓词与其他谓词之间的关系，即当特征谓词所断定的个体不存在时，句子为真，如果使用逻辑与表达特征谓词与其他谓词之间的关系，则当特征谓词所断定的个体不存在时，句子为假，就不符合全称判断句子的含义。

对应地，“有的人会画画”就断定了“人”这种个体的存在，而且它首先断定至少有一个“人”这种个体，而且具有“会画画”这种进一步的性质。因此这时特征谓词“人”与“会画画”之间就是逻辑与关系，如果使用蕴涵，则当完全不存在“人”这种个体时，就不符合句子本身的含义。

例子 5.2.2 将下列涉及量词的基本命题在谓词逻辑中进行符号化（下面的句子来自[2]）：

1. 所有商品都是有价值的。
2. 有的官员是清廉的。
3. 所有迷信都不是科学。
4. 有的新闻报导不是真实的。

解答：

1. 句子“所有商品都是有价值的”中的特征谓词是：$S(x)$: x是商品，另外，我们用$J(x)$表示“x是有价值的”，则整个句子符号化为：$\forall x(S(x) \to J(x))$。

2. 句子“有的官员是清廉的”中的特征谓词是：$G(x)$: x是官员，另外，用$Q(x)$表示“x是清廉的”，则整个句子符号化为：$\exists x(G(x) \wedge Q(x))$。

3. 句子“所有迷信都不是科学”中的特征谓词是：$M(x)$: x是迷信，另外用$K(x)$表示“x是科学”，则整个句子符号化为：$\forall x(M(x) \to \neg K(x))$。

4. 句子“有的新闻报导不是真实的”中的特征谓词是：$W(x)$: x是新闻报道，另外用$Z(x)$表示“x是真实的”，则整个句子符号化为：$\exists x(W(x) \wedge \neg Z(x))$。

5.2.3 涉及重叠量词的命题的符号化

上一小节讨论的命题都只涉及一个量词，是基本的全称判断、肯定判断等。这一节讨论一些复杂的命题，需要使用两个或两个以上量词进行符号化：

例子 5.2.3 将下列自然语言命题符号化：

1. 兔子都比乌龟跑得快。
2. 有的兔子比有的乌龟跑得快。
3. 存在比所有乌龟跑得快的兔子。
4. 任意兔子都比某个乌龟跑得快。

解答：上述句子都是对两类个体之间关系的判断，由于个体变量的取值范围都是全总域，我们需要引入两个特征谓词描述句子所断定个体的基本特征：令$T(x)$表示x是兔子，$W(x)$表示x是乌龟。上述句子所判断的关系都是“…比…跑得快”，这是一个二元谓词，我们用$K(x,y)$表示x比y跑得快。简言之，我们有如下谓词：

$$T(x): \ x\text{是兔子}$$
$$W(x): \ x\text{是乌龟}$$
$$K(x,y): \ x\text{比}y\text{跑得快}$$

有了这些谓词之后，我们可将上述句子作下述符号化：

1. 句子“兔子都比乌龟跑得快”的含义是“任意兔子都比任意乌龟跑得快”，更准确地说，该句子的含义是：“对任意的x和任意的y，如果x是兔子而且y是乌龟，则x比y跑得快”，从而句子应该符号化为：

$$\forall x \forall y(T(x) \wedge W(y) \to K(x,y))$$

2. 句子“有的兔子比有的乌龟跑得快”的含义更准确地说是：“存在某个x，存在某个y，x是兔子，y是乌龟，而且x比y跑得快”，因此句子符号化为：

$$\exists x \exists y(T(x) \wedge W(y) \wedge K(x,y))$$

3. 句子“存在比所有乌龟跑得快的兔子”的含义更准确地是：“存在某个x，x是兔子，而且对任意的y，如果y是乌龟，则x比y跑得快”，从而句子应该符号化为：

$$\exists x(T(x) \wedge \forall y(W(y) \to K(x,y)))$$

4. 句子“任意兔子都比某个乌龟跑得快”的含义更准确地说是:“对任意的x，如果x是兔子，则存在某个y，y是乌龟而且x比y跑得快”，从而句子应该符号化为：

$$\forall x(T(x) \to \exists y(W(y) \wedge K(x,y)))$$

上述句子都是对两类个体之间的关系进行判断，而且是对这两类个体的全体或部分之间的关系进行判断，因此需要使用两个量词来分别描述句子中对每一类个体的约束。对于这种句子的符号化，我们可以像例子所示，先在假定个体变量的取值范围是全总域的情况下，增加对特征谓词的叙述，并详细的给出句子的含义，然后在此含义的基础上进行符号化。

细心的读者会发现上述例子中前两个句子的符号化与后两个句子的符号化稍有不同，前两个句子的两个量词是紧挨在一起的，而后两个句子的两个量词是分开的。实际上，前两个句子我们也可使用量词分开的形式进行符号化，例如“兔子都比乌龟跑得快”，我们可以这样理解:“对任意的x，如果x是兔子，则对任意的y，如果y是乌龟，则x比y跑得快”，从而将句子符号化为：

$$\forall x(T(x) \to \forall y(W(y) \to K(x,y)))$$

后面我们将看到，这个公式与$\forall x\forall y(T(x) \wedge W(y) \to K(x,y))$等值，也即具有相同的真值。再例如“有的兔子比有的乌龟跑得快”可理解为:“存在某个x，x是兔子，而且存在某个y，y是乌龟，而且x比y跑得快”，从而将句子符号化为：

$$\exists x(T(x) \wedge \exists y(W(y) \wedge K(x,y)))$$

同样，这个公式与$\exists x\exists y(T(x) \wedge W(y) \wedge K(x,y))$等值。

实际上，我们认为，所有涉及到量词的命题都可使用量词分开的形式进行符号化，把每个量词与它所约束的特征谓词所描述的个体类紧密地放在一起，而将对该个体类的性质或该个体类与其他个体类之间的关系抽象为另外的谓词，特征谓词与这些谓词之间的逻辑关系则把握这样的原则：**全称量词约束的特征谓词与其他谓词（乃至公式的其他部分）的逻辑关系是蕴涵**，而**存在量词约束的特征谓词与其他谓词（乃至公式其他部分）的逻辑关系是合取**。

进一步，我们认为当不同的个体类采用不同的量词约束时，将量词与特征谓词紧密结合，不同的量词分开形式的符号化更为自然的。而当不同的个体类采用相同的量词约束时，将所有的量词放在公式的最前面则可能更为自然。

5.2.4 涉及量词的复合命题的符号化

前面讨论的命题都是命题逻辑中的简单命题，即使上一节所列出的涉及多个量词的命题在命题逻辑中仍只能视为简单命题，例如,“有的兔子比乌龟跑得快”这个句子在命题逻辑中只能看作简单命题。

从本质上来说，谓词逻辑对命题逻辑所作的扩充主要是针对简单命题，对简单命题的结构做进一步的分解，用个体、谓词、量词等符号更精确地描述简单命题。至于将简单命题组合起来构成复杂命题的逻辑联结词仍与命题逻辑一样，通常使用否定、析取、合取、蕴涵以及等价等联结词。

例子 5.2.4 将下列自然语言命题符号化（下面的句子来自[2]）：

1. 没有不透风的墙。

2. 我的矛能刺穿天下所有的盾，而我的盾天下所有的矛都不能刺穿。

3. 如果甲班有学生考试作弊，那么甲班所有学生都不能获得本年度的奖学金。

4. 每个自然数都有自然数比它大，但没有最大的自然数。

解答：

1. 句子“没有不透风的墙”的含义是“并非有的墙是不透风的”，因此其中的特征谓词是：$Q(x):x$是墙，另外用$T(x)$表示x是透风的，则整个句子符号化为：

$$\neg\exists x(Q(x)\wedge\neg T(x))$$

从自然语言的角度看，这个句子也相当于：“所有的墙都是透风的”，符号化则为：$\forall x(Q(x)\rightarrow T(x))$，后面我们将看到这两个谓词逻辑公式也是等值的。

2. 句子“我的矛能刺穿天下所有的盾，而我的盾天下所有的矛都不能刺穿”有两个支命题，用“而”联结在一起。前一个支命题“我的矛能刺穿天下所有的盾”的更准确含义是：“对任意的个体x，如果x是盾，则我的矛能刺穿x”，其中“我的矛”是具体的个体，用个体常量符号a表示，特征谓词是：$D(x):x$是盾，“x能刺穿y”是一个二元谓词，用$C(x,y)$表示，从而整个支命题符号化为：

$$\forall x(D(x)\rightarrow C(a,x))$$

同理，支命题“我的盾天下所有的矛都不能刺穿”的更准确含义是：“对任意的个体x，如果x矛，则x不能刺穿我的盾”，其中“我的盾”是具体个体，用个体常量符号b表示，特征谓词是：$M(x):x$是矛，该支命题同样使用了二元谓词“x能刺穿y”，需要同样用$C(x,y)$表示，从而整个支命题符号化为：

$$\forall x(M(x)\rightarrow\neg C(x,b))$$

从而整个命题符号化为：

$$\forall x(D(x)\rightarrow C(a,x))\wedge\forall x(M(x)\rightarrow\neg C(x,b))$$

3. 句子“如果甲班有学生考试作弊，那么甲班所有学生都不能获得本年度的奖学金”也有两个支命题，用“如果…那么…”联结在一起。前一个支命题是“甲班有学生考试作弊”，更准确的含义是：“存在某个个体x，x是甲班学生，而且x考试作弊”，因此特征谓词是$J(x):x$是甲班学生，另外有谓词：$K(x):x$考试作弊，该支命题符号化为：

$$\exists x(J(x)\wedge K(x))$$

而支命题“甲班所有学生都不能获得本年度的奖学金”的准确含义是：“对任意的个体x，如果x是甲班学生，那么x不能获得本年度的奖学金”，因此特征谓词仍然是$J(x):x$是甲班学生，另外有谓词：$H(x):x$获得本年度的奖学金，该支命题符号化为：

$$\forall x(J(x)\rightarrow\neg H(x))$$

从而整个命题符号化为：

$$\exists x(J(x)\wedge K(x))\rightarrow\forall x(J(x)\rightarrow\neg H(x))$$

4. 句子“每个自然数都有自然数比它大，但没有最大的自然数”也有两个支命题，用“但”联结在一起，是逻辑与关系。前一个支命题“每个自然数都有自然数比它大”的更准确含义是：“对任意的个体x，如果x是自然数，那么存在个体y，y是自然数，而且y比x大”，因此其中有特征谓词：$N(x):x$是自然数，还有二元谓词：$G(x,y):x$比y大，整个支命题符号化为：

$$\forall x(N(x)\rightarrow\exists y(N(y)\wedge G(y,x)))$$

支命题“没有最大的自然数”的含义是“不存在比所有自然数都大的自然数”，更准确地说是：“不存在个体x，x是自然数，而且满足：对任意的个体y，如果y是自然数，则x比y大”，因此其中使用的谓词与前一个支命题相同，而符号化为：

$$\neg\exists x(N(x)\wedge\forall y(N(y)\rightarrow G(x,y)))$$

从而整个命题符号化为：

$$\forall x(N(x)\rightarrow\exists y(N(y)\wedge G(y,x)))\wedge\neg\exists x(N(x)\wedge\forall y(N(y)\rightarrow G(x,y)))$$

对于上面的例子，读者**要注意其中个体变量符号的使用**。例如，第二个命题我们符号化为：

$$\forall x(D(x)\rightarrow C(a,x))\wedge\forall x(M(x)\rightarrow\neg C(x,b))$$

两个支命题的量词指导变元使用了相同的个体变量符号，这是可以的，但读者应该知道，前一支命题中$D(x)$的x与后一支命题中$M(x)$的x**实际上并无关系**。下一节会讨论量词的**辖域**(scope)或称**作用域**，我们将看到，前一个支命题中的量词$\forall x$的作用域仅仅是$(D(x)\rightarrow C(a,x))$，也就是说其中的x只在这个范围内有效，与后一个支命题中$M(x)$的x无关。实际上，我们也可采用不同的指导变元，而将该命题符号化为：

$$\forall x(D(x)\rightarrow C(a,x))\wedge\forall y(M(y)\rightarrow\neg C(y,b))$$

上述两个公式我们可认为**在语法上等价**。

所谓在语法上等价，与公式等值不太一样：公式等值往往是两个公式在语法形式上不一样，但是在考虑其真值时总是相同，涉及到公式的语义解释（即公式的真值）；而语法上等价是指两个公式在语法形式上虽略有不同，但不需要考虑公式的真值，从语法的角度看就是一样的。**语法等价比公式在语义上等值更强，也即语法等价的公式肯定是等值的，但等值的公式不一定语法上等价**。下一节讨论的**自由变元替换规则**和**约束变量改名规则**可将一个公式变换成在语法上与它等价的另一形式的公式。

上面例子的其他命题也在不同的支命题中采用了相同的量词指导变元，读者通过下一节的学习之后都要能理解每个指导变元的作用域，分清楚个体变量符号之间的关系：哪些个体变量符号为同一个量词所约束（从而实际上代表相同的个体），而哪些个体变量符号为不同的量词所约束（从而即使采用相同的个体变量符号实际上也是代表不同的个体）。

在上面例子中，读者还需要注意的问题是，我们在给出谓词符号时使用了个体变量符号，例如，例子中说用$C(x,y)$表示x能刺穿y。读者应该理解这里的x和y是占位用的，为了叙述方便也为了给出谓词的元数而使用的，与在公式中使用的指导变元或其他的个体变量符号完全没有关系。这正如C++中的函数原型说明中的形式参数，它与函数实现中使用的形式参数实际上是无关的，可以采用不同的标识符。

最后上述例子的第四个句子，其中只抽取出一个特征谓词：$N(x)$表示x是自然数，那么我们可假定所有变量的个体域都是自然数，也即全总域是自然数集，从而简化句子的符号化：

$$\forall x\exists yG(y,x)\wedge\neg\exists x\forall yG(x,y)$$

上述公式的含义是：“对任意的自然数x，存在自然数y使得y大于x，而且（但是）不存在自然数x，对所有的自然数y，x都大于y”。这里的符号化只要将前面公式中的特征谓词去掉，特征谓词与其他谓词之间的逻辑联结词去掉即可。

5.2.5 小结

这一节讨论了各种自然语言命题在谓词逻辑中的符号化，读者应该理解：

1. 谓词逻辑对命题逻辑中的**原子命题**（简单命题）的结构作了进一步的分解，引入个体、谓词以及量词等对原子命题作进一步的刻画。而**在谓词逻辑中处理复合命题中各支命题之间的逻辑关系与命题逻辑完全相同**。

2. 如果命题不涉及量词，也就是说**命题只是对某个具体的个体的性质或某些具体的个体之间的关系进行判断**，那么在谓词逻辑中的符号化的关键是抽象出其中的具体个体和谓词：具体个体用个体常量符号表示，谓词用谓词符号表示，并使用个体变量符号作为占位符给出该谓词所表达的含义。**整个命题则用谓词作用于个体而符号化**。

3. 如果命题涉及量词，也就是说**命题针对某类个体的全体或部分（某个或某些不确定个体）的性质，或者针对某几类个体的全体或部分之间的关系进行判断**，那么：(i) 若所有个体变量的取值范围是全总域，则应注意抽取句子中的特征谓词，并把握特征谓词与其他谓词之间的关系：**全称判断通常是逻辑蕴涵关系，特称判断通常是逻辑与关系**。(ii) 若句子中所有个体变量的取值范围相同，只抽象出一个特征谓词，那么可在说明个体变量的个体域的前提下，将特征谓词去掉，将特征谓词与其他谓词之间的逻辑关系去掉而得到比较简单的符号化形式。

4. 如果命题涉及到多个量词，这通常是对某几类个体的全体或部分之间的关系进行判断的命题，那么最好采用不同量词分开，而将量词与其约束的特征谓词紧密放在一起的方式进行符号化。

5. 最后，在谓词逻辑中符号化自然语言命题，特别是符号化涉及到量词的命题，关键在于根据句子的含义，假定变量的取值范围是全总域，插入对特征谓词的描述，明确对量词的使用，以更清楚、更准确的方式描述句子，在此描述的基础上进行符号化。

总之，这一节使得我们对谓词逻辑公式有了初步的感性认识，下一节将严格定义什么是谓词逻辑公式，并讨论像约束变量、自由变量、约束变量改名、自由变量替换、量词（包括指导变元）的作用域等与谓词逻辑公式语法有关的内容，并解释为什么我们课程学习的是**一阶谓词逻辑**。而在再下一节，我们将进一步讨论谓词逻辑公式的语义解释，即如何确定谓词逻辑公式的真值。

5.3 一阶谓词逻辑公式的定义

谓词逻辑公式的语法要比命题逻辑公式的语法复杂很多，因为它涉及许多语法成份，如个体变量、个体常量、谓词、量词、函数等。首先我们要将本课程所讨论的谓词逻辑公式限制为**一阶谓词逻辑公式**，或简称**一阶公式**。所谓的“一阶”是指**研究的谓词是一阶的而不是高阶的，只能对个体变量使用量词而不能对其他语法成份使用量词**。

谓词可分为**一阶谓词**和**高阶谓词**。我们知道，谓词是个体性质或个体之间关系的抽象，例如句子“玫瑰花是红色的”，其中“…是红色的”就是谓词，这种对个体性质或关系描述的谓词是一阶谓词。但是性质和关系本身还有性质，例如“红色”这种性质还有“鲜艳”、“明亮”、“暗淡”等性质，再例如，关系也有自反性、对称性、传递性等。因此，表达性质和关系的谓词具有层次，这种层次称为谓词的**阶**：**只刻画个体性质或关系的谓词是一阶谓词，从而公式的形式看，一阶谓词只作用到个体变量或个体常量上**。

当我们说存在某些个体具有“红色”这种性质时，这是对个体变量使用量词或说进行量化；而当我们说某些性质具有“鲜艳”这种性质是，就是对谓词变量进行量化了。当我们**涉及到谓词的谓词，或者对谓词变量进行量化时，就进入了高阶谓词逻辑的范围了**。高阶谓词逻辑的许多问题可以化为一阶谓词逻辑。**本课程只讨论一阶逻辑**。

概括地说，一阶谓词逻辑，就是**其中的谓词都是一阶谓词，只作用于个体变量或个体常量，其中的量词也只刻画个体变量的量化**。

5.3.1 一阶公式的符号

与定义命题逻辑公式一样，我们先确定一阶公式的符号表。正如前面所说，一阶公式有许多语法成份，首先我们可以将这些语法成份（语法符号）分为两类：

1. **非逻辑符号**：包括个体常量符号、函数符号以及谓词符号，这些符号与一阶公式的应用领域密切相关，而与一阶公式本身的逻辑性质关系不大。我们要将一阶谓词逻辑用于某个应用领域，必须对该领域所涉及的个体、个体之间的函数关系、个体的性质和个体之间的关系进行符号化；

2. **逻辑符号**：包括个体变量符号、量词符号、逻辑联结词符号以及辅助符号（如圆括号、逗号），这些符号与一阶公式的应用领域无关，或者说在任何应用领域都需要使用这些符号，而我们在数理逻辑课程中对一阶谓词逻辑的研究，也主要是研究与这些符号，特别是与量词和逻辑联结词有关的性质，这些性质我们称为一阶公式的逻辑性质，是一阶谓词逻辑用于任何领域所具有的通用性质。

注意，这里我们将个体常量和个体变量符号分在不同的符号类，这表明，在谓词逻辑中，个体常量和个体变量之间的区别是十分重要的，这是因为，量词可作用于个体变量，而不能用于个体常量，这给个体常量和个体变量之间带来本质的区别。在命题逻辑中，命题变量和命题常量实际上没有本质差别，因为在命题逻辑中不能对命题变量本身作任何的变换。同样，在一阶谓词逻辑中，谓词常量和谓词变量符号、函数常量和函数常量符号之间也没有本质差别，因为一阶谓词逻辑不涉及对谓词变量的量化，也不涉及对函数符号的量化。

任何对一阶公式的研究都是建立在一组非逻辑符号的基础上，即我们假定从应用领域中抽象出了一些个体常量符号、函数符号以及谓词符号：

定义 5.3.1 *假定从应用领域中抽象出了如下符号：*

1. **个体常量符号**：*用$a, b, c, \cdots$等排在字母表前面的小写字母表示，记所有个体常量符号构成的集合为$\mathcal{C}$；*

2. **函数符号**：*用$f, g, h, \cdots$等排在字母表中间的小写字母表示，并且每个函数符号具有**元数**，表明它能作用于几个个体。记所有函数符号构成的集合为$\mathcal{F}$；*

3. **谓词符号**：*用$F, G, H, \cdots$等排在字母表中间的大写字母表示，同样每个谓词符号也具有**元数**，表明它能作用于几个个体。记所有谓词符号构成的集合为$\mathcal{P}$。*

所有上述符号构成了一阶公式的**非逻辑符号集**。个体常量符号集$\mathcal{C}$和函数符号集$\mathcal{F}$可以是空集，但谓词符号集$\mathcal{P}$不能为空集，即**至少有一个谓词符号**。

定义 5.3.2 除非逻辑符号集之外，一阶公式还使用如下的**逻辑符号**：

1. **个体变量符号**：用$x, y, z, \cdots$等排在字母表后面的小写字母表示，记所有个体变量符号构成的集合为$\mathbf{Var}$；
2. **量词符号**：包括全称量词符号$\forall$，和存在量词符号$\exists$；
3. **逻辑联结词符号**：与命题逻辑相同，包括$\neg, \wedge, \vee, \rightarrow, \leftrightarrow$五个符号；
4. **辅助符号**：包括圆括号和逗号。

这里要强调的是，非逻辑符号是应用领域中具体个体、函数和谓词的抽象，非逻辑符号集的不同导致一阶公式具有不同的表达能力和应用范围。逻辑符号是所有一阶公式必须具有的符号，体现了一阶逻辑用于任何应用领域时的共性，数理逻辑课程研究的就是一阶公式中与逻辑符号有关的性质。

5.3.2 一阶公式的归纳定义

确定一阶公式的非逻辑符号集和逻辑符号集之后，我们使用归纳法严格定义一阶公式。这分三个层次进行，分别是**项**，**原子公式**和**一般公式**。注意，在下面定义的一阶公式，以及后面谈到的一阶公式都是在给定非逻辑符号集$\mathcal{C} \cup \mathcal{F} \cup \mathcal{P}$的基础上进行讨论的。

定义 5.3.3 **项**(term)归纳定义为：

1. **归纳基**：任意的个体常量符号$c \in \mathcal{C}$和任意的个体变量符号$x \in \mathbf{Var}$都是项；
2. **归纳步**：若$t_1, t_2, \cdots, t_n$是项，$f \in \mathcal{F}$是n元函数符号，则$f(t_1, t_2, \cdots, t_n)$是项；
3. **最小化**：所有项都是通过有限次运用上面两步得到的。

我们将所有项构成的集合记为Term。

实际上，一阶公式的项是个体的抽象：个体常量和个体变量表示基本的个体，而函数作用于已有的项（个体）构成了复杂的个体。

定义 5.3.4 **一阶逻辑公式**，简称**一阶公式**，归纳定义为：

1. **归纳基**：如果$t_1, t_2, \cdots, t_n$是项，$F \in \mathcal{P}$是n元谓词符号，则$F(t_1, t_2, \cdots, t_n)$是公式，并且称为**原子公式**；
2. **归纳步**：(i) 如果A, B是公式，则$(\neg A), (A \wedge B), (A \vee B), (A \rightarrow B), (A \leftrightarrow B)$是公式；(ii) 如果$A$是公式，$x \in \mathbf{Var}$是个体变量符号，则$\forall x A$和$\exists x A$是公式。
3. **最小化**：所有一阶公式都是通过有限次运用上述方式得到的。

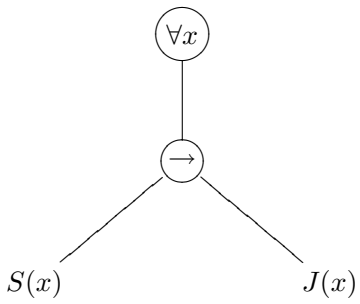
我们将所有一阶构成的集合记为Form。

对于此定义，首先我们进一步看到谓词符号与函数符号的区别，谓词作用于项（个体）得到公式，而且是原子公式，相当于命题逻辑中的原子命题，而函数作用于项（个体）仍然得到项。谓词符号是研究一阶逻辑公式必不可少的，而函数符号则不是必不可少的。由于谓词的必要性，有些讨论一阶逻辑公式的教材甚至将“相等”（用符号“=”表示）这个谓词作为构造一阶逻辑公式的基本符

号，因为这个谓词在绝大多数应用领域中都需要。本教程没有将它作为基本符号，只是假定至少有一个谓词符号。

我们可根据上述定义分析任何一阶公式的语法结构：

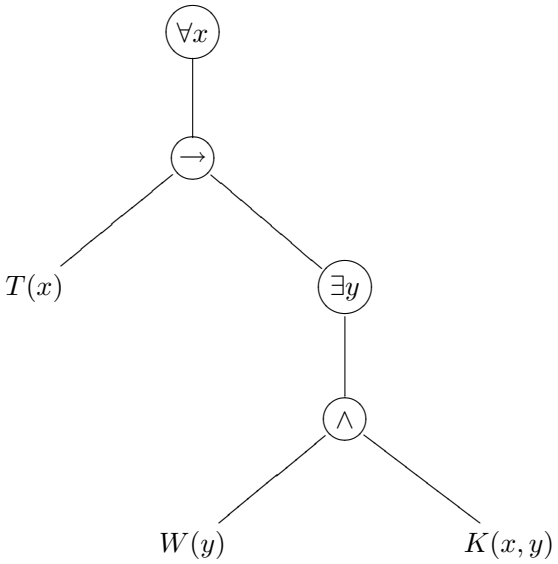
例子 5.3.5 一阶公式$\forall x(S(x) \to J(x))$的语法结构可用下图表示：



为简便起见，上述语法树没有对公式中原子公式以及项的语法做进一步的分解，因此树叶就是原子公式。语法树的内部节点给出了上述归纳定义中归纳步所给出的构造公式的方式，其中使用逻辑联结词构造公式的方式与命题逻辑完全相同，只是在涉及量词符号时才有所不同。

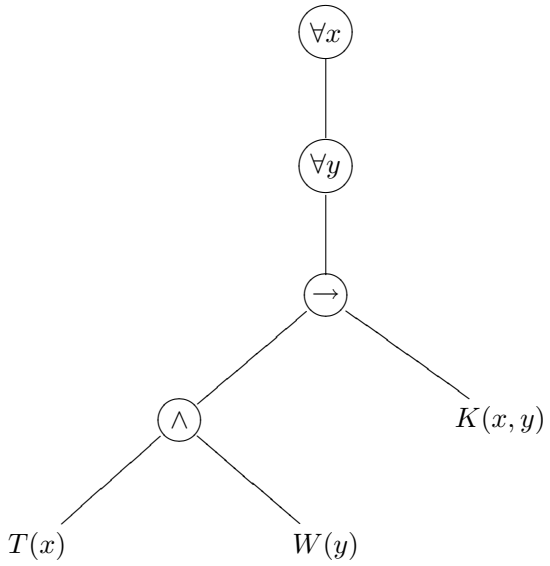
上图中我们在语法树的内部节点给出了量词符号以及它的指导变元，然后该节点有一个分支，给出了这个量词所约束的公式，实际上，该节点的子树就是这个量词（包括指导变元）的**辖域**。这里，我们要强调的是，**量词符号与其指导变元是密不可分的整体，在一阶公式中不能单出现量词符号，而个体变量符号要么作为量词的指导变元，要么出现在公式的项中**。

例子 5.3.6 一阶公式$\forall x(T(x) \to \exists y(W(y) \wedge K(x,y)))$的语法结构可用下图表示：



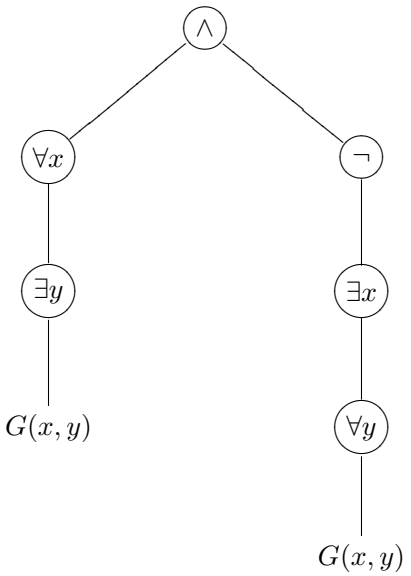
上述公式中，$T(x), W(y)$和$K(x,y)$都是原子公式，$W(y)$和$K(x,y)$用合取联结起来得到$(W(y) \wedge K(x,y))$，量词$\exists y$作用在该公式上得到$\exists(W(y) \wedge K(x,y))$，而公式$T(x)$和这个公式用蕴涵联结起来得到$(T(x) \to \exists(W(y) \wedge K(x,y)))$，然后量词$\forall x$作用在这个公式上得到题目中的公式。

例子 5.3.7 一阶公式$\forall x \forall y((T(x) \wedge W(y)) \to K(x,y))$的语法结构可用下图表示：



上述公式中，同样$T(x), W(y)$和$K(x,y)$是原子公式，$T(x)$和$W(y)$用合取联结起来得到$(T(x) \wedge W(y))$，然后与$K(x,y)$使用蕴涵联结起来得到$((T(x) \wedge W(y)) \rightarrow K(x,y))$，量词$\forall y$先作用在这个公式上得到$\forall x((T(x) \wedge W(y)) \rightarrow K(x,y))$，而量词$\forall x$再作用在这个公式上得到题目中的公式。

例子 5.3.8 一阶公式$\forall x \exists y G(x,y) \wedge \neg \exists x \forall y G(x,y)$的语法结构可用下图表示：



上述公式已经省略其中一些不必要的圆括号。与命题逻辑公式相同，我们可通过定义优先级与组合性省去一些不必要的括号：

1. **量词具有最高优先级**，因此公式$\forall x G(x,y) \rightarrow K(x)$相当于

$$((\forall x G(x,y)) \rightarrow K(x))$$

而不是$\forall x(G(x,y) \rightarrow K(x))$；

2. 逻辑联结词的优先级按顺序排依次是：$\neg, \wedge, \vee, \rightarrow, \leftrightarrow$，这与命题逻辑相同；

3. $\wedge, \vee, \leftrightarrow$是从左至右结合，而$\rightarrow$是从右至左结合，因此$\forall x(T(x) \wedge W(y) \wedge K(x,y))$相当于

$$\forall x((T(x) \wedge W(y)) \wedge K(x,y))$$

而$\forall x(T(x) \rightarrow W(y) \rightarrow K(x,y))$相当于

$$\forall x(T(x) \rightarrow (W(y) \rightarrow K(x,y)))$$

对于省略一些圆括号的公式，读者也必须能够清楚它的语法结构，必要时画出公式的语法图：

练习 5.3.9 请画出以下一阶公式的语法树：

(1) $\neg\exists x(Q(x) \wedge \neg T(x))$

(2) $\forall x(D(x) \rightarrow C(a,x)) \wedge \forall x(M(x) \rightarrow \neg C(x,b))$

(3) $\forall x(N(x) \rightarrow \exists y(N(y) \wedge G(y,x))) \wedge \neg\exists x(N(x) \wedge \forall y(N(y) \rightarrow G(x,y)))$

与命题逻辑公式一样，我们可使用归纳法定义一阶公式的**子公式**，读者也可通过一阶公式语法结构的分析了解它的子公式。有关子公式的严格定义留给涂着。

练习 5.3.10 请参考命题逻辑公式子公式的定义，给出一阶公式的子公式的归纳定义。

5.3.3 约束变量与自由变量

上一小节使用归纳法给出了一阶公式的定义，对于每个一阶公式我们都可画出它的语法树。为了进一步了解一阶公式的语法结构，我们还需要理解一些概念，特别是**要区分公式中的约束变量和自由变量**：

定义 5.3.11 对于公式$\forall xA$，其中公式A称为量词$\forall x$的**辖域**(scope)，或称作用域，而公式A中出现的x都称为个体变量符号x的**约束出现**。类似地，对于公式$\exists xA$，A为量词$\exists x$的辖域，A中出现的x也称为x的约束出现。

设公式A是含有个体变量符号x的公式，如果x不出现在量词$\forall x$或$\exists x$的辖域中，则称x在A中**自由出现**。

设公式A是含有个体变量符号x的公式，如果x在A中**都是约束出现**的，则x称为公式A的**约束变量**(bounded variable)。如果x不是公式A的约束变量，即x在A中的**某处出现是自由出现**，则x称为公式A的**自由变量**(free variable)。

例子 5.3.12

1. 公式$\forall x(T(x,y) \rightarrow K(z))$中量词$\forall x$的辖域是$(T(x,y) \rightarrow K(z))$：

$$\forall x \underbrace{(T(x,y) \rightarrow K(z))}_{\forall x \text{ 的辖域}}$$

其中出现的x是约束出现，因此x**是公式的约束变量**，而y没有出现在量词$\forall y$或$\exists y$的辖域中，因此y**是公式的自由变量**，类似的，z**也是公式的自由变量**：

$$\begin{array}{cccccc} \text{指导变元} & \text{约束出现} & & \text{自由出现} & & \text{自由出现} \\ \downarrow & \downarrow & & \downarrow & & \downarrow \\ \forall\ x(T(& x, & & y) \rightarrow K(& & z)) \end{array}$$

2. 公式$\forall x T(x,y) \to \exists y G(x,y)$中的量词$\forall x$的辖域是$T(x,y)$，因此$T(x,y)$中出现的$x$是约束出现，而$y$是自由出现；量词$\exists y$的辖域是$G(x,y)$，因此$G(x,y)$中出现的$y$是约束出现，而$x$是自由出现。对于整个公式而言，**$x,y$都是公式的自由变量**。

	指导变元	约束出现	自由出现	指导变元	自由出现	约束出现
	↓	↓	↓	↓	↓	↓
$\forall$	$xT($	$x,$	$y) \to \exists$	$yG($	$x,$	$y)$

3. 公式$\forall x(T(x,y) \to \exists y G(x,y,z))$中的量词$\forall x$的辖域是$(T(x,y) \to \exists y G(x,y,z))$：

$$\forall x \underbrace{(T(x,y) \to \exists y G(x,y,z))}_{\forall x \text{ 的辖域}}$$

因此$T(x,y)$中出现的x是约束出现，$G(x,y,z)$中出现的x也是约束出现。量词$\exists y$的辖域是$G(x,y,z)$，因此$T(x,y)$中出现的y是自由出现，而$G(x,y,z)$中出现的y是约束出现。$G(x,y,z)$中出现的z是自由出现：

	指导变元	约束出现	自由出现	指导变元	约束出现	约束出现	自由出现
	↓	↓	↓	↓	↓	↓	↓
$\forall$	$x(T($	$x,$	$y) \to \exists$	$yG($	$x,$	$y,$	$z))$

对于整个公式而言，**x是公式的约束变量，而y和z是公式的自由变量**。

4. 公式$\forall x \exists y(F(x,y) \wedge G(y,z)) \vee \exists x H(x,y,z)$中量词$\forall x$的辖域是$\exists y(F(x,y) \wedge G(y,z))$，而量词$\exists y$的辖域是$(F(x,y) \wedge G(y,z))$：

$$\forall x \underbrace{\exists y \overbrace{(F(x,y) \wedge G(y,z))}^{\exists y \text{ 的辖域}}}_{\forall x \text{ 的辖域}} \vee \exists x H(x,y,z)$$

量词$\exists x$的辖域是$H(x,y,z)$，因此$F(x,y)$中的x和y都是约束出现的，而$G(y,z)$中的y是约束出现，z是自由出现，$H(x,y,z)$中的x是约束出现，而y和z都是自由出现：

	指导变元	指导变元	约束出现	约束出现	约束出现	自由出现	指导出现	约束出现	自由出现	自由出现
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
$\forall$	$x\exists$	$y(F($	$x,$	$y) \wedge G($	$y,$	$z)) \vee \exists$	$xH($	$x,$	$y,$	$z)$

对于整个公式而言，**x是公式的约束变量，而y和z是公式的自由变量**。

读者通过上述例子，**对任意的一阶公式，应该弄清楚每个量词的辖域，每个个体变量符号的出现是指导变元，还是约束出现，还是自由出现**。实际上，量词辖域的概念与程序设计语言中的标识符作用域概念极为类似，我们可以将量词指导变元看作程序中函数的形式参数，量词（指导变元）的辖域相当于函数体，我们知道，形式参数的作用域也局限于函数体，约束出现的个体变量它也局限于它所在的量词辖域。从某种意义上说，可将自由出现的个体变量的作用范围理解为整个公式，相当于程序中的全局变量。

与程序中标识作用域可以嵌套和屏蔽一样，我们所定义的一阶公式也允许量词辖域的嵌套，例如，$\forall x(T(x) \wedge \exists x(G(x) \vee H(x)))$也是合法的一阶公式，这里量词$\forall x$的辖域是$(T(x) \wedge \exists x(G(x) \vee H(x))$，但是由于量词$\exists x$的指导变元也是$x$，而$\exists x$的辖域是$(G(x) \vee H(x))$，实际上，在子公式$(G(x) \vee H(x))$中，

量词$\forall x$中的指导变元的作用被屏蔽了，因此这个子公式中出现的x与$T(x)$中的x是本质上不同的两个变量。

备注 5.3.13 教材[4]p59在定义一阶公式$\forall xA$和$\exists xA$时，要求x在A中是自由变量，实际上这种要求是不必要的，也容易使得定义不够严谨。教材实际上没有严格定义什么是自由变量，因为只有当定义了一阶公式之后，才能定义一阶公式的自由变量，因此在定义一阶公式的时候就要求x在A中是自由变量，在某种意义上，有循环定义的嫌疑。

我们在定义$\forall xA$和$\exists xA$时，不要求x是A的自由变量，因此公式$\forall x(T(x) \wedge \exists xH(x))$是合法的一阶公式，$\forall xT(y)$也是合法的一阶公式，虽然$x$不在$T(y)$中出现，后面我们将看到这个公式等值于$T(y)$。甚至$\forall x\exists x(T(x) \to H(x))$也是合法的公式，这时$\forall x$的辖域是$\exists x(T(x) \to H(x))$，而$\exists x$的辖域是$(T(x) \to H(x))$，在这个子公式中，屏蔽了量词$\forall x$的指导变元的作用，后面我们将看到这个公式等值于$\exists x(T(x) \to H(x))$。当然，当$x$是$A$的自由变量时，$\forall xA$和$\exists xA$是最常用，也最容易理解的形式。

这里顺便提请读者注意，是说x是A的自由变量，当形成公式$\forall xA$或$\exists xA$之后，对于$\forall xA$或$\exists xA$整个公式而言，x就是它的约束变量了。

量词辖域的嵌套，乃至不同量词使用相同的指导变元，个体变量符号既有自由出现也有约束出现往往给读者带来一些迷惑，我们认为当一个一阶公式满足如下条件时是最清晰和最容易理解的：

1. **每个变量符号在公式中要么自由出现要么约束出现，没有既自由出现又约束出现的情况**；
2. **所有量词后面采用的指导变元符号互不相同**。

实际上，量词后面的约束变元只在它的辖域里有意义，处于其辖域以外的同名变元与该约束变元实际上无关，因此在各个量词后面使用不同的指导变元是必要的，而且也是可以的。任何公式都可通过使用**约束变量改名规则**和**自由变量替换规则**得到满足上述条件而在语法上等价的公式。

定义 5.3.14 **约束变量改名规则**：对于公式A，对其中某个量词的指导变元x，连同该量词的辖域中相同变元符号x的所有出现都用**不在公式A中出现的个体变元符号**代替得到的公式A'与A在**语法上等价**。

自由变元替换规则：对于公式A，将其中某个个体变量符号x的**所有自由出现的地方**都使用**不在公式A中出现的个体变元符号**代替得到的公式A'与A在**语法上等价**。

备注 5.3.15 上面为了简单起见，在约束变量改名和自由变元替换规则中，都使用不在A中出现的变元符号去代替原有的变量符号x。实际上，对于约束变量改名规则而言，对公式A中某个量词的指导变元，连通该量词辖域中x的所有出现都使用**不在该量词辖域中出现的变元符号**代替得到的公式A'也与A在语法上等价。这很容易理解，因为替换只是发生在量词的辖域中。

对于自由变元替换规则，对公式A中自由出现的变元符号x，可使用满足如下条件的变元符号y去代替x的所有自由出现的地方，得到的公式A'与A也语法等价：(1) y不在公式A中自由出现；而且(2) x的每个自由出现的地方都不在量词$\forall y$或$\exists y$的辖域中。这里第(1)个条件是避免将$A(x,y)$替换成$A(y,y)$（显然从直观上看这两者不等价），而第(2)个条件是避免将$\exists yA(x,y)$替换成$\exists yA(y,y)$（显然从直观上看这两者也不等价）。

例子 5.3.16 使用约束变量改名规则和自由变量替换规则变换下列公式，使得其满足上述两个条件而更清晰、更容易理解：

1. $\exists x(P(x) \vee R(x)) \to \forall x(P(x) \wedge Q(x))$
2. $\forall x(P(x) \leftrightarrow Q(x)) \wedge \exists x R(x) \vee S(x)$
3. $\forall x P(x) \wedge \exists x Q(x) \vee (\forall x P(x) \to Q(x))$

解答：首先确定量词的辖域，然后确定变元符号的约束出现和自由出现，然后再进行变换：

1. 对于公式$\exists x(P(x) \vee R(x)) \to \forall x(P(x) \wedge Q(x))$，量词$\exists x$的辖域是$(P(x) \vee R(x))$，而量词$\forall x$的辖域是$(P(x) \wedge Q(x))$，为了使得不同量词的指导变元不同，我们将$\forall x(P(x) \wedge Q(x))$中的$x$使用约束变量改名规则改名为$y$，而得到：

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

2. 对于公式$\forall x(P(x) \leftrightarrow Q(x)) \wedge \exists x R(x) \vee S(x)$，量词$\forall x$的辖域是$(P(x) \leftrightarrow Q(x))$，而$\exists x$的辖域是$R(x)$，$S(x)$中的$x$是自由出现的，为了满足上述两个条件，我们将$\exists x R(x)$中的$x$使用约束变量改名规则改名为$y$，而将$S(x)$中的$x$使用自由变量替换规则替换为$z$，得到：

$$\forall x(P(x) \leftrightarrow Q(x)) \wedge \exists y R(y) \vee S(z)$$

3. 对于公式$\forall x P(x) \wedge \exists x Q(x) \vee (\forall x P(x) \to Q(x))$，前一个量词$\forall x$的辖域是$P(x)$，而$\exists x$的辖域是$Q(x)$，后一个量词$\forall x$的辖域是$P(x)$，最后$Q(x)$中$x$是自由出现的，为了满足上述两个条件我们将$\exists x Q(x)$中的$x$使用约束变量改名规则改名为$y$，将后一个$\forall x P(x)$中的$x$改名为$z$，而将$Q(x)$中的$x$使用自由变量替换规则替换为$u$，得到：

$$\forall x P(x) \wedge \exists y Q(y) \vee (\forall z P(z) \to Q(u))$$

实际上，公式中的自由变量是十分重要的，我们在用A, B, C等大写符号命名一个公式时，通常还给出该公式出现的自由变量，例如公式：

$$\forall x(F(x, y) \to G(y, z) \to H(z, u)$$

中的自由变量包括y, z, u，因此可用$A(y, z, u)$命名。

当用大写字母以及变量符号的形式$A(x_1, x_2, \cdots, x_n)$命名公式（代表公式）时，表示该公式中**至少有**$x_1, x_2, \cdots, x_n$这些自由变量，有时我们还可能特别说明该公式**只有**$x_1, x_2, \cdots, x_n$这n个自由变量。例如，在大多数情况下，$A(x, y)$表示x和y都是该公式的自由变量，通常该公式还可能有其他的自由变量，除非特别声明它只有自由变量x和y。

细心的读者还可能注意到，当我们使用$A(x, y, z)$的形式命名公式，代表公式时，如谓词符号$F(x, y)$的形式一致。实际上，任何一个带有$x_1, x_2, \cdots, x_n$等自由变量的公式可看作一个n元谓词。也就是说，在某种意义下，谓词作用于个体形成的原子公式和一般公式是相对的，正如在程序设计中，可将函数（子程序）等模块看作将一些语句组合起来命一个函数名以供使用相同的，基本语句与程序模块也是相对的。

练习 5.3.17 *对于任意的公式A，可使用归纳法定义它的所有约束变量构成的集合$bv(A)$。请读者根据归纳定义的思想完成$bv(A)$的定义。*

这一节的最后我们定义闭公式的概念：

定义 5.3.18 *如果一个公式不含有任何自由变量，则称为**闭公式**(closed formula)。*

例如，公式$x + y > 10$就不是闭公式（这里+是函数符号，>是谓词符号，只不过采用了中缀形式而已），而$\forall x \exists x(x + y > 10)$就是闭公式。读者不难看到，只有闭公式才对应命题逻辑中的命题，例如上一节对涉及量词的自然语言命题进行符号化得到的一阶公式，如$\forall x(S(x) \to J(x)), \forall x \forall y(T(x) \wedge W(y) \to K(x, y))$等都是闭公式；而含有自由变量的公式在命题逻辑中认为其含有公认是变量的符号，不能确定其真值则不认为是命题。在下一节公式的语义解释（即公式的真值）中我们也可看到闭公式与不是闭公式的公式在真值确定方面的不同之处。

5.4 一阶公式的真值

5.4.1 非逻辑符号的解释

与命题逻辑公式不同，一阶公式含有非逻辑符号，即从应用领域中抽象出来的符号，包括个体常量符号、函数符号以及谓词符号等，因此要确定一阶公式的真值，首先要给出这些非逻辑符号的解释。给出非逻辑符号解释的思路是确定一个应用领域，或更准确地说，是确定应用领域的所有个体组成的集合，这个集合称为解释一阶公式的**论域**，然后将个体常量符号解释为论域中的元素，函数符号解释为论域上的函数，谓词符号解释为论域上的关系。

定义 5.4.1 设一阶公式的非逻辑符号集是$\mathcal{C} \cup \mathcal{F} \cup \mathcal{P}$。一阶公式的**解释**$\mathcal{I}$包括：

1. 一个**非空**集合$\mathbf{D}$，称为该解释的**论域**(domain)；
2. 对每个个体常量符号$c \in \mathcal{C}$，有惟一的元素$[\![c]\!] \in \mathbf{D}$与之对应，称为个体常量符号c的解释；
3. 对每个n元函数符号$f \in \mathcal{F}$，有惟一的函数$[\![f]\!] : \mathbf{D}^n \to \mathbf{D}$与之对应，称为函数符号$f$的解释，注意，这里$\mathbf{D}^n$是$n$个集合$\mathbf{D}$的笛卡尔积；
4. 对每个n元谓词符号$F \in \mathcal{P}$，有惟一的关系$[\![F]\!] \subseteq \mathbf{D}^n$与之对应，称为谓词符号$F$的解释。

虽然我们在定义一阶公式时并没有明确指定一阶公式的非逻辑符号集，但在考虑公式的真值时，只要给出当前研究公式中出现的非逻辑符号的解释即可。

例子 5.4.2 假设我们要研究句子“我的矛能刺穿天下所有的盾，而我的盾天下所有的矛都不能刺穿”符号化得到的公式

$$\forall x(D(x) \to C(a, x)) \wedge \forall x(M(x) \to \neg C(x, b))$$

的真值，那么我们需要先给出其中个体常量a, b的解释，以及谓词符号$D(x), M(x), C(x, y)$的解释，也即我们需要在一个解释$\mathcal{I}$下来研究该公式的真值。例如，给定一个解释$\mathcal{I}$，定义为：

1. 论域是：

$$\mathbf{D} = \{\text{红樱长矛}, \text{红樱短矛}, \text{金盾}, \text{钢盾}\}$$

2. 个体常量符号的解释是：

$$[\![a]\!] = \text{红樱长矛} \qquad [\![b]\!] = \text{金盾}$$

3. 谓词符号的解释是：

$$\llbracket D \rrbracket = \{\text{金盾}, \text{钢盾}\} \subseteq \mathbf{D}$$
$$\llbracket M \rrbracket = \{\text{红樱长矛}, \text{红樱短矛}\} \subseteq \mathbf{D}$$
$$\llbracket C \rrbracket = \{\langle \text{红樱长矛}, \text{金盾}\rangle, \langle \text{红樱长矛}, \text{钢盾}\rangle, \langle \text{红樱短矛}, \text{钢盾}\rangle\} \subseteq \mathbf{D}\times\mathbf{D}$$

注意，一元谓词的解释是论域$\mathbf{D}$上一元关系，也即$\mathbf{D}$的子集，而二元谓词的解释是论域$\mathbf{D}$上二元关系，也即$\mathbf{D}\times\mathbf{D}$的子集。

上述解释的直观含义是十分明了的：我们假定“天下所有的盾”和“天下所有的矛”构成的集合是论域$\mathbf{D}$，而“我的矛”是“红樱长矛”，“我的盾”是“金盾”。矛与盾之间（已知）的刺穿关系用关系$\llbracket C \rrbracket$描述。

读者不难想到，只有给出这些非逻辑符号的解释之后，才能确定上述一阶公式的真值。不确定一个论域（也即应用领域），给出这些非逻辑符号的解释，是无法确定公式的真值的。当然上述解释用的是一些中文词语，这是为了让这个解释更接近现实世界，下面我们所讨论的解释则大多数是使用数学符号。

为了进一步讨论的方便，我们再考虑一个带有函数符号的例子：

例子 5.4.3 假定我们要考虑下列公式的真值：

1. $F(f(x,y),g(x,y))$
2. $\forall x F(g(x,y),z)$
3. $\forall x\forall y(F(f(x,a),y)\to F(f(y,a),x))$
4. $\exists x F(f(x,x),g(x,x))$

那么上述公式中出现的非逻辑符号包括：个体常量符号a，函数符号f,g，它们都是二元函数，以及谓词符号F，它是二元谓词。因此，如果我们要确定上述公式的真值，就必须确定论域以及这些非逻辑符号的解释。例如，给定解释：

1. 论域：$\mathbf{D}=\mathbb{N}$，即论域是自然数集合；
2. 个体常量符号a的解释是自然数零，也即：$\llbracket a \rrbracket = 0$；
3. 函数符号f的解释是自然数加法，而g的解释是自然数乘法，也即：

$$\llbracket f \rrbracket = + : \mathbb{N}\times\mathbb{N}\to\mathbb{N} \qquad \llbracket g \rrbracket = \cdot : \mathbb{N}\times\mathbb{N}\to\mathbb{N}$$

或更直接地：

$$\forall x,y\in\mathbb{N},\quad \llbracket f \rrbracket(x,y) = x+y \qquad \llbracket g \rrbracket(x,y) = x\cdot y$$

4. 谓词符号F的解释是恒等关系，也即：

$$\llbracket F \rrbracket = \{\langle x,x\rangle \mid x\in\mathbb{N}\} \subseteq \mathbb{N}\times\mathbb{N}$$

或更直接地：

$$\llbracket F \rrbracket(x,y) \text{ 为真} \quad \text{当且仅当} \quad x=y$$

后面我们将看到如何利用这个解释确定上述公式的真值。

5.4.2 个体变量指派函数

读者不难想到，给定一阶公式非逻辑符号的解释，就是给定了其中的个体常量、函数符号和谓词符号的解释，但是要确定公式的真值，还必须首先确定个体变量的值。下面先引入个体变量指派函数的概念：

定义 5.4.4 给定一阶公式非逻辑符号的解释$\mathcal{I}$，并设其中的论域是$\mathbf{D}$。**个体变量指派函数**是形如$\sigma:\mathbf{Var}\to\mathbf{D}$的函数，即对一阶公式中出现的任意的个体变量符号$x\in\mathbf{Var}$，个体变量指派函数$\sigma$都确定一个论域中的元素$\sigma(x)$与之对应。

上述定义是很自然的，因为个体变量的取值当然是个体，而解释的论域代表了应用领域所有可能个体的集合，因此为个体变量确定值，也就是为个体变量指派论域中的元素。同样，我们也不用真正对所有个体变量符号指派值，**只需要给当前要研究的公式中出现的变量符号指派值即可**。例如，我们要考虑例子5.4.2中的公式的真值，只需要考虑个体变量符号x即可，而要考虑例子5.4.3中四个公式的真值，至需要考虑个体变量符号x,y,z即可。

为了严格给出量词的语义，也即为了确定带有量词的公式的真值，还需要引入如下的定义：

定义 5.4.5 给定一阶公式非逻辑符号的解释$\mathcal{I}$，并设其中的论域是$\mathbf{D}$。对于任意的个体变量指派函数$\sigma:\mathbf{Var}\to\mathbf{D}$，以及任意的个体变量符号$x\in\mathbf{Var}$，和论域$\mathbf{D}$的任意元素$a\in\mathbf{D}$，**定义一个新的个体变量指派函数**，记为$\sigma[a/x]:\mathbf{Var}\to\mathbf{D}$：

$$\forall y\in\mathbf{Var},\quad \sigma[a/x](y)=\begin{cases}a & \text{如果 } y=x\\ \sigma(y) & \text{否则}\end{cases}$$

上述$\sigma[a/x]$也是个体变量指派函数，而且是在给定一个个体变量指派函数σ，以及个体变量符号x和论域元素a的基础上进行定义的。直观地说，$\sigma[a/x]$的函数定义，也即$\sigma[a/x]$指派个体变量符号的值的方式是：

1. 如果要确定个体变量符号x的值，则$\sigma[a/x](x)=a$；
2. 如果要确定其他个体变量符号（也即不等于x的）y的值，则与σ的指派方式相同，也即$\sigma[a/x](y)=\sigma(y)$。

下面我们马上就会看到上述个体变量指派函数σ，以及记号$\sigma[a/x]$的用途。

5.4.3 一阶公式真值的确定

在给定非逻辑符号集的解释，以及给定个体变量指派函数之后，我们可确定一阶公式的真值。首先我们用归纳法给出项的语义：

定义 5.4.6 给定一阶公式非逻辑符号的解释$\mathcal{I}$，以及个体变量指派函数$\sigma:\mathbf{Var}\to\mathbf{D}$，这里$\mathbf{Var}$是所有个体变量符号构成的集合，而$\mathbf{D}$是解释$\mathcal{I}$的论域。根据项$t$的语法结构，可归纳定义$t$**在指派$\sigma$下的语义**$\sigma(t)$为：

1. **归纳基**：如果项t是个体常量符号c，则$\sigma(t)=[\![c]\!]$；如果t是个体变量符号x，则$\sigma(t)=\sigma(x)$；
2. **归纳步**：若项t是由n元函数符号f作用于项$t_1,t_2,\cdots,t_n$得到，也即若$t=f(t_1,t_2,\cdots,t_n)$，则

$$\sigma(t)=[\![f]\!](\sigma(t_1),\sigma(t_2),\cdots,\sigma(t_n))$$

这里$[\![f]\!]$是n元函数符号f的解释，也即$\sigma(t)$等于f的解释（即论域上的n元函数）作用于项$t_1, t_2, \cdots, t_n$上的语义解释得到的函数值。

由上述定义，读者很容易看到，对于任意的项t，t的语义解释$\sigma(t)$是论域$\mathbf{D}$的元素。上述定义符号我们对项的直观解释，即个体常量和个体变量的语义解释是论域中的元素（即应用领域的个体），而项是函数作用于基本个体得到的复杂个体，它的解释仍然应该是论域中的元素。

例子 5.4.7 对于例子5.4.3给出的解释$\mathcal{I}$：

1. 论域：$\mathbf{D} = \mathbb{N}$，即论域是自然数集合；
2. 个体常量符号a的解释是自然数零，也即：$[\![a]\!] = 0$；
3. 函数符号f的解释是自然数加法，而g的解释是自然数乘法，也即：

$$[\![f]\!] = + : \mathbb{N} \times \mathbb{N} \to \mathbb{N} \qquad [\![g]\!] = \cdot : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$$

或更直接地：

$$\forall x, y \in \mathbb{N}, \quad [\![f]\!](x, y) = x + y \qquad [\![g]\!](x, y) = x \cdot y$$

4. 谓词符号F的解释是恒等关系，也即：

$$[\![F]\!] = \{\langle x, x\rangle \mid x \in \mathbb{N}\} \subseteq \mathbb{N} \times \mathbb{N}$$

或更直接地：

$$[\![F]\!](x, y) \text{ 为真} \quad \text{当且仅当} \quad x = y$$

在个体变量指派函数$\sigma : \mathbf{Var} \to \mathbb{N}$：

$$\sigma(x) = 0 \qquad \sigma(y) = 1 \qquad \sigma(z) = 2$$

的指派下，项$f(x, a)$的语义（即$f(x, a)$对应的论域元素）是：

$$\sigma(f(x, a)) = [\![f]\!](\sigma(x), \sigma(a) = [\![f]\!](0, [\![a]\!]) = [\![f]\!](0, 0) = 0 + 0 = 0$$

而项$\sigma(g(y, z))$的语义（即$g(y, z)$对应的论域元素）是：

$$\sigma(g(y, z)) = [\![g]\!](\sigma(y), \sigma(z)) = [\![g]\!](1, 2) = 1 \cdot 2 = 2$$

确定项的解释之后，我们可以给出一阶公式的语义，也即一阶公式真值的归纳定义：

定义 5.4.8 给定一阶公式非逻辑符号的解释$\mathcal{I}$，以及个体变量指派函数$\sigma : \mathbf{Var} \to \mathbf{D}$。根据一阶公式$A$的语法结构，可归纳定义**公式$A$在指派$\sigma$下的真值**$\sigma(A) \in \{0, 1\}$为：

1. **归纳基**：若公式是原子公式$F(t_1, t_2, \cdots, t_n)$，这里F是n元谓词符号，而$t_1, t_2, \cdots, t_n$是项，则

$$\sigma(A) = \sigma(F(t_1, t_2, \cdots, t_n)) = 1 \quad \text{当且仅当} \quad \langle \sigma(t_1), \sigma(t_2), \cdots, \sigma(t_n)\rangle \in [\![F]\!]$$

注意，n元谓词符号F的解释$[\![F]\!]$是论域$\mathbf{D}$上的n元关系，因此$\langle \sigma(t_1), \sigma(t_2), \cdots, \sigma(t_n)\rangle \in [\![F]\!]$意味着$t_1, t_2, \cdots, t_n$具有关系$[\![F]\!]$。

2. **归纳步**：

(1). 若公式A（最后一步）是由逻辑联结词联结得到的公式，则其真值的确定与命题逻辑相同：
  (a). 若公式A的形式是$(\neg B)$，则$\sigma(A)=\sigma(\neg B)=1$当且仅当$\sigma(B)=0$；
  (b). 若公式A的形式是$(B\wedge C)$，则$\sigma(A)=\sigma(B\wedge C)=1$当且仅当$\sigma(B)=\sigma(C)=1$；
  (c). 若公式A的形式是$(B\vee C)$，则$\sigma(A)=\sigma(B\vee C)=0$当且仅当$\sigma(B)=\sigma(C)=0$；
  (d). 若公式A的形式是$(B\to C)$，则$\sigma(A)=\sigma(B\to C)=0$当且仅当$\sigma(B)=1$而$\sigma(C)=0$；
  (e). 若公式A的形式是$(B\leftrightarrow C)$，则$\sigma(A)=\sigma(B\leftrightarrow C)=1$当且仅当$\sigma(B)=\sigma(C)$。

(2). 若公式A（最后一步）是由量词作用而得到的公式，则：
  (a). 若公式A的形式是$\forall xB$，则$\sigma(A)=\sigma(\forall xB)=1$当且仅当：

$$\text{对论域}\mathbf{D}\text{的任意元素}a\in\mathbf{D}\text{，都有}\sigma[a/x](B)=1$$

  (b). 若公式A的形式是$\exists xB$，则：$\sigma(A)=\sigma(\exists xB)=1$当且仅当：

$$\text{存在论域}\mathbf{D}\text{的某个元素}a\in\mathbf{D}\text{，使得}\sigma[a/x](B)=1$$

上述定义对于原子公式$F(t_1,t_2,\cdots,t_n)$的真值定义，直观地说就是，当$t_1,t_2,\cdots,t_n$这n个项具有关系$[\![F]\!]$时该原子公式为真。上述定义对于逻辑联结词的语义解释也与命题逻辑相同，无须过多讨论。因此下面只就带量词公式的语义解释再直观地讨论一下。

根据上述定义，公式$\forall xB$在指派σ下的真值为真当且仅当对论域$\mathbf{D}$的任意元素a都有公式B在指派$\sigma[a/x]$下的真值为真。而指派$\sigma[a/x]$的含义是，除了个体变量符号x指派为论域的元素a之外，其他个体变量符号的指派与σ相同。因此直观地说，**要计算$\forall xB$在指派σ下的真值是否为真，要对论域的每个元素a，检查在将指导变元符号x指派为a，而其他变量符号的指派不变的情况下，B的真值是否都为真**。因此，实际上，**$\forall xB$的真值与σ原先对x的指派无关**。

类似地，公式$\exists xB$在指派σ下的真值为真当且仅当对论域$\mathbf{D}$的某个元素a使得公式B在指派$\sigma[a/x]$下的真值为真。直观地说，**要计算$\exists xB$在指派σ下的真值是否为真，要对论域的每个元素a，检查在将指导变元符号x指派为a，而其他变量符号的指派不变的情况下，B的真值是否有为真的情况**。同样，**$\exists xB$的真值与σ原先对x的指派无关**。

例子 5.4.9 考虑公式

$$A=\forall x(D(x)\to C(a,x))\wedge\forall x(M(x)\to\neg C(x,b))$$

在下面的解释$\mathcal{I}$：

1. 论域是：

$$\mathbf{D}=\{\text{红樱长矛},\ \text{红樱短矛},\ \text{金盾},\ \text{钢盾}\}$$

2. 个体常量符号的解释是：

$$[\![a]\!]=\text{红樱长矛}\qquad\qquad[\![b]\!]=\text{金盾}$$

3. 谓词符号的解释是：

$$[\![D]\!]=\{\text{金盾},\ \text{钢盾}\}\subseteq\mathbf{D}$$
$$[\![M]\!]=\{\text{红樱长矛},\ \text{红樱短矛}\}\subseteq\mathbf{D}$$
$$[\![C]\!]=\{\langle\text{红樱长矛},\text{金盾}\rangle,\ \langle\text{红樱长矛},\text{钢盾}\rangle,\ \langle\text{红樱短矛},\text{钢盾}\rangle\}\subseteq\mathbf{D}\times\mathbf{D}$$

以及指派σ : **Var**→**D**下的真值，这里σ定义为：$\sigma(x)$ =金盾。注意，由于上述公式只出现一个个体变量符号x，因此个体变量指派函数只要给出对x的指派即可。

(1). 首先公式A在解释$\mathcal{I}$及指派σ下的真值为真，当且仅当公式$B_1 = \forall x(D(x) \to C(a,x))$以及公式$B_2 = \forall x(M(x) \to \neg C(x,b))$在解释$\mathcal{I}$及指派$\sigma$下真值都为真。

(2). 公式B_1在解释$\mathcal{I}$及指派σ下的真值为真，当且仅当对论域**D**的任意元素d∈**D**都有公式$B_{11} = D(x) \to C(a,x)$在解释$\mathcal{I}$及指派$\sigma[d/x]$下的真值为真，也即：

(a) 对于 红樱长矛∈ **D**，公式B_{11}在解释$\mathcal{I}$及指派σ[红樱长矛/x]下的真值是否为真；以及

(b) 对于 红樱短矛∈ **D**，公式B_{11}在解释$\mathcal{I}$及指派σ[红樱短矛/x]下的真值是否为真；以及

(c) 对于 金盾∈ **D**，公式B_{11}在解释$\mathcal{I}$及指派σ[金盾/x]下的真值是否为真；以及

(d) 对于 钢盾∈ **D**，公式B_{11}在解释$\mathcal{I}$及指派σ[钢盾/x]下的真值是否为真。

注意，指派σ[红樱长矛/x]将x指派为论域元素红樱长矛，即

$$\sigma[\text{红樱长矛}/x](x) = \text{红樱长矛}$$

公式B_{11}在此指派下的真值为假当且仅当$D(x)$在此指派下为真，而$C(a,x)$在此指派下为假。而$D(x)$在指派下的真值为真当且仅当

$$\sigma[\text{红樱长矛}/x](x) \in [\![D]\!]$$

根据$[\![D]\!]$的定义，σ[红樱长矛/x](x) = 红樱长矛 $\notin [\![D]\!]$，因此$D(x)$在指派σ[红樱长矛/x]下的真值为假，从而B_{11}在此指派下的真值为真。

直观地说，由于指派σ[红樱长矛/x]将x指派为论域元素红樱长矛，因此公式$D(x)$在该指派下是否为真，取决于红樱长矛是否属于$[\![D]\!]$。我们采用更方便的记号：**用公式$D($ 红樱长矛$)$的真值表示红樱长矛是否属于$[\![D]\!]$**，也即：

$$D(\text{ 红樱长矛}) \text{ 为真 当且仅当 } \text{红樱长矛} \in [\![D]\!]$$

从而公式$D(x)$在指派σ[红樱长矛/x]下的真值与公式$D($ 红樱长矛$)$的真值相同，实际上，后一个公式可看作$D(x)$中的个体变量符号x用指派σ[红樱长矛/x]为x所指派的论域元素代入而得到。

类似地，公式$C(a,x)$在指派σ[红樱长矛/x]下的真值与公式

$$C(\text{红樱长矛}, \text{红樱长矛})$$

的真值相同，此公式实际上是将公式$C(a,x)$中的个体常量符号a用a的解释$[\![a]\!]$代入，而其中的个体变量符号x用指派σ[红樱长矛/x]为x所指派的论域元素代入而得到，而：

$$C(\text{红樱长矛}, \text{红樱长矛}) \text{ 为真 当且仅当 } \langle \text{红樱长矛}, \text{红樱长矛} \rangle \in [\![C]\!]$$

也即公式$B_{11} = D(x) \to C(a,x)$在解释$\mathcal{I}$及指派σ[红樱长矛/x]下的真值与下面公式的真值相同：

$$D(\text{红樱长矛}) \to C(\text{红樱长矛}, \text{红樱长矛})$$

类似地，我们可得到$B_{11} = D(x) \to C(a,x)$在解释$\mathcal{I}$及指派σ[红樱短矛/x]下的真值与下面公式的真值相同：

$$D(\text{红樱短矛}) \to C(\text{红樱长矛}, \text{红樱短矛})$$

注意，上述公式是将B_{11}中的x用红樱短矛代入，个体常量符号a用$[\![a]\!]$ = 红樱长矛代入而得到。类似地，B_{11}在指派σ[金盾/x]下的真值与下面公式的真值相同：

$$D(\text{金盾}) \to C(\text{红樱长矛}, \text{金盾})$$

而B_{11}在指派σ[钢盾/x]下的真值与下面公式的真值相同：

$$D(\text{钢盾}) \to C(\text{红樱长矛}, \text{钢盾})$$

从而公式$B_1 = \forall x B_{11}$的真值与下面公式的真值相同：

$$(D(\text{红樱长矛}) \to C(\text{红樱长矛}, \text{红樱长矛})) \wedge (D(\text{红樱短矛}) \to C(\text{红樱长矛}, \text{红樱短矛})) \wedge$$
$$(D(\text{金盾}) \to C(\text{红樱长矛}, \text{金盾})) \wedge (D(\text{钢盾}) \to C(\text{红樱长矛}, \text{钢盾}))$$

注意上述公式中，D(红樱短矛)为真当且仅当红樱短矛$\in[\![D]\!]$，C(红樱长矛, 金盾)为真当且仅当$\langle$红樱长矛, 金盾$\rangle\in[\![C]\!]$，其他公式的真值含义类似。根据$[\![D]\!]$和$[\![D]\!]$的定义，以及逻辑联结词所确定的真值关系，不难得到上述公式的真值为真，从而公式B_1的真值为真。

类似地，不难得到，公式$B_2 = \forall x(M(x) \to \neg C(x, b))$在解释$\mathcal{I}$及指派$\sigma$下真值与下面公式的真值相同：

$$(M(\text{红樱长矛}) \to \neg C(\text{红樱长矛}, \text{金盾})) \wedge (M(\text{红樱短矛}) \to \neg C(\text{红樱短矛}, \text{金盾})) \wedge$$
$$(M(\text{金盾}) \to \neg C(\text{金盾}, \text{金盾})) \wedge (M(\text{钢盾}) \to \neg C(\text{钢盾}, \text{金盾}))$$

上述公式中，M(红樱长矛)为真当且仅当红樱长矛$\in[\![M]\!]$，其中公式的真值含义类似。根据$[\![M]\!]$和$[\![C]\!]$的定义，有：

$$\text{红樱长矛}\in[\![M]\!] \qquad \text{且} \qquad \langle\text{红樱长矛}, \text{金盾}\rangle\in[\![C]\!]$$

因此公式

$$M(\text{红樱长矛}) \to \neg C(\text{红樱长矛}, \text{金盾})$$

的真值为假，从而公式B_2的真值为假。最终得到公式$A = B_1 \wedge B_2$在解释$\mathcal{I}$及指派σ下的真值为假。

上面（也许有些过于冗长的）例子给出了如何按照定义确定一阶公式在某解释及某指派下的真值。为了后面讨论的方便，上述例子引入了简单方法描述原子公式的真值，例如对于原子公式$C(x, b)$，它在解释$\mathcal{I}$及指派σ下的真值，我们直接用公式C(红樱短矛, 金盾)的真值表示，其中$C(x, b)$中的x用σ对x的指派代入，而b用b的解释$[\![b]\!]$ = 金盾代入。这种方法简化了我们对原子公式的真值计算，后面也将采用这个方法，并将进一步说明当项中含有函数符号时应该如何处理。

上述例子主要是详细介绍了如何根据定义确定带量词公式的真值。假定公式B含有自由变量x，下面更明确地用$B(x)$表示，那么公式$A = \forall x B(x)$在解释$\mathcal{I}$及指派σ下的真值实际上与将$B(x)$中的x用论域的每一个元素代入，而其他个体变量符号（或更准确地说，其他自由变量）用σ指派的论域元素代入，个体常量符号用它的解释代入而得到的公式的真值相同。特别地，如果解释$\mathcal{I}$的论域$\mathbf{D} = \{d_1, d_2, \cdots, d_n\}$只有有限个元素，那么可认为公式$\forall x B(x)$的真值与公式

$$B(d_1) \wedge B(d_2) \wedge \cdots \wedge B(d_n)$$

相同，这里$B(d_i)$就是将B中的x用d_i代入，其他自由变量用σ指派的论域元素，个体常量用其解释代入得到的公式[1]。简单地说，**对于有限论域，全称量词可展开为适当公式的合取**。

对于存在量词，通过类似的分析可以得到，**对于有限论域，存在量词可展开为适当公式的析取**：设解释$\mathcal{I}$的论域$\mathbf{D}=\{d_1,d_2,\cdots,d_n\}$，公式$\exists xB(x)$的真值与公式

$$B(d_1)\vee B(d_2)\vee\cdots\vee B(d_n)$$

相同。

通过上述例子，读者还可以看到，**个体变量指派函数对于约束变量的指派不会影响公式的真值**。例如公式$\forall x(D(x)\to C(a,x))$的真值与$\sigma$将$x$指派什么值无关。进一步有：

引理 5.4.10 闭公式（也即不含有自由变量的公式）在任意个体变量指派函数下的真值都相同，或者说，**闭公式的真值与个体变量指派函数无关**。 □

例子 5.4.11 我们继续通过例子进一步说明如何确定带量词公式，特别是带多个量词公式的真值。考虑下列公式：

(1) $\exists xF(f(x,y),a)$

(2) $\exists x\forall yF(x,y)$

(3) $\forall y\exists xF(x,y)$

(4) $\forall xF(y,a)$

在下述解释$\mathcal{I}$：

(1) 论域$\mathbf{D}=\{1,2\}$；

(2) 个体常量符号a的解释是：$[\![a]\!]=1$；

(3) 二元函数符号f的解释是：$[\![f]\!]:\mathbf{D}\times\mathbf{D}\to\mathbf{D}$，定义为：

$$[\![f]\!](1,1)=1\qquad[\![f]\!](1,2)=2\qquad[\![f]\!](2,1)=2\qquad[\![f]\!](2,2)=1$$

(4) 二元谓词符号F的解释是：$[\![F]\!]=\{\langle1,2\rangle,\langle2,1\rangle\}\subseteq\mathbf{D}\times\mathbf{D}$。

以及指派$\sigma:\{x,y\}\to\mathbf{D}$下的真值，这里$\sigma$定义为：

$$\sigma(x)=1\qquad\sigma(y)=2$$

解答：

(1). 对于公式$A=\exists xF(f(x,y),a)$，按照定义5.4.6，它的真值为真当且仅当存在论域$\mathbf{D}$的**某个元素**d使得公式$B=F(f(x,y),a)$在指派$\sigma[d/x]$下的真值为真，也即：如果公式B在指派$\sigma[1/x]$下的真值为真，**或者**公式B在指派$\sigma[2/x]$下的真值为真，则公式A的真值为真。

根据上一个例子的分析，考虑公式$B=F(f(x,y),a)$在指派$\sigma[1/x]$下的真值时，我们可将其中个体变量符号用指派$\sigma[1/x]$所指定的论域元素代入，个体常量用它的解释代入。但是这里还出现了函数符号，该如何处理呢？

实际上，根据定义5.4.6，项$f(x,y)$在指派$\sigma[1/x]$下的值是

$$[\![f]\!](\sigma[1/x](x),\sigma[1/x](y))=[\![f]\!](1,2)=2$$

[1] 或更准确地说公式$B(d_i)$的真值与B在指派$\sigma[d_i/x]$下的真值相同。

注意，我们有$\sigma[1/x](x)=1,\sigma[1/x](y)=\sigma(y)=2$。因此$B=F(f(x,y),a)$在指派$\sigma[1/x]$下的真值与公式$F(2,1)$的真值相同，也即与公式$F([\![f]\!](1,2),1)$相同。由于$[\![f]\!]$本身不是一阶公式的符号，为简便记，我们仍用$f$代替$[\![f]\!]$写在公式中，从而我们说公式$B=F(f(x,y),a)$在指派$\sigma[1/x]$下的真值与公式

$$F(f(1,2),1)$$

的真值相同。这个公式是将$B=F(f(x,y),a)$中的变量符号用指派$\sigma[1/x]$为该变量所指派的论域元素代入，个体常量符号用该常量解释代入，函数符号保持不变而得到的公式。但注意，这里$f(1,2)$的值准确的说是$[\![f]\!](1,2)$，而$F(f(1,2),1)$的真值为真当且仅当$\langle[\![f]\!](1,2),1\rangle\in[\![F]\!]$。

类似地，$B=F(f(x,y),a)$在指派$\sigma[2/x]$下的真值就与公式

$$F(f(2,2),1)$$

相同，这个公式是将$F(f(x,y),a)$中的x用$\sigma[2/x]=2$代入，y用$\sigma[2/x](y)=\sigma(y)=2$代入，$a$用$[\![a]\!]=1$代入而得到，该公式为真当且仅当$\langle[\![f]\!](2,2),1\rangle\in[\![F]\!]$为真。

从而公式$A=\exists xF(f(x,y),a)$的真值就与公式

$$F(f(1,2),1)\vee F(f(2,2),1)$$

的真值相同，根据$[\![F]\!]$和$[\![f]\!]$的定义：

$$\langle[\![f]\!](1,2),1\rangle=\langle 2,1\rangle\in[\![F]\!]\qquad\langle[\![f]\!](2,2),1\rangle=\langle 1,1\rangle\notin[\![F]\!]$$

从而$F(f(1,2),1)\vee F(f(2,2),1)$的真值为1。借用命题逻辑的等值演算中所采用的等值符号（实际上，后面一阶公式的等值演算将采用相同的符号），我们将上述确定公式A在指派σ下的真值过程写为：

$$\sigma(A)\Leftrightarrow F(f(1,2),1)\vee F(f(2,2),1)\Leftrightarrow F(2,1)\vee F(1,1)\Leftrightarrow 1\vee 0\Leftrightarrow 1$$

(2). 对于公式$A=\exists x\forall yF(x,y)$，按照定义，$A$在上述解释和指派下的真值为真当且仅当存在论域的某个元素$d\in\mathbf{D}$使得$\forall yF(x,y)$在指派$\sigma[d/x]$下为真。也就是说，当论域$\mathbf{D}=\{1,2\}$时有：

$$\sigma(A)=\sigma[1/x](\forall yF(x,y))\vee\sigma[2/x](\forall yF(x,y))$$

这里$\sigma[1/x](\forall yF(x,y))$表示公式$\forall yF(x,y)$在指派$\sigma[1/x]$下的真值，类似地，$\sigma[2/x](\forall yF(x,y))$表示公式$\forall yF(x,y)$在指派$\sigma[2/x]$下的真值。

对于公式$\forall yF(x,y)$在指派$\sigma[1/x]$下的真值，由于$\sigma[1/x]$把变量符号x指派为论域元素1，因此我们可先将公式$\forall yF(x,y)$中的x用1代入，对于变量符号y，由于y在公式$\forall yF(x,y)$中是约束变量，**该公式的真值与个体变量指派函数对约束变量指派的值无关**，因此**不能将$\forall yF(x,y)$中的y用$\sigma[1/x]$为y所指派的值代入**。

总之，公式$\forall yF(x,y)$在**指派$\sigma[1/x]$**下的真值与公式$\forall yF(1,y)$在**指派σ**下的真值相同，对于x以外的个体变量符号，$\sigma[1/x]$所指派的论域元素与σ指派的论域元素完全相同。类似地，公式$\forall yF(x,y)$在指派$\sigma[2/x]$下的真值与公式$\forall yF(2,y)$在指派σ下的真值相同，因此公式$A=\exists x\forall yF(x,y)$在指派$\sigma$下的真值与公式

$$\forall yF(1,y)\vee\forall yF(2,y)$$

在指派σ下的真值相同。进一步，公式$\forall yF(1,y)$的真值与下述公式

$$F(1,1)\wedge F(1,2)$$

的真值相同，这个公式是用论域的每个元素代入公式$F(1,y)$中的y而得到的，由于是全称量词，所以将代入得到的公式用**合取**联结。类似地，公式$\forall yF(2,y)$的真值与下述公式

$$F(2,1)\wedge F(2,2)$$

的真值相同。最终有：

$$\begin{aligned}\sigma(A)&\Leftrightarrow \forall yF(1,y)\vee\forall yF(2,y)\\&\Leftrightarrow (F(1,1)\wedge F(1,2))\vee(F(2,1)\wedge F(2,2))\\&\Leftrightarrow (0\wedge 1)\vee(1\wedge 0)\Leftrightarrow 0\end{aligned}$$

(3). 对于公式$A=\forall y\exists xF(x,y)$在上述解释和指派下的真值，根据上面类似的分析有；

$$\begin{aligned}\sigma(A)&\Leftrightarrow \exists xF(x,1)\wedge\exists xF(x,2)\\&\Leftrightarrow (F(1,1)\vee F(2,1))\wedge(F(1,2)\vee F(2,2))\\&\Leftrightarrow (0\vee 1)\wedge(1\vee 0)\Leftrightarrow 1\end{aligned}$$

(4). 对于公式$\forall xF(y,a)$在上述解释和指派σ下的真值，根据定义5.4.6，如果对于论域$\mathbf{D}$的每个元素d都有$F(y,a)$在$\sigma[d/x]$下的真值为1，则公式$\forall xF(y,a)$的真值为1。而公式$F(y,a)$在$\sigma[d/x]$下的真值与公式$F(2,1)$的真值相同，这里$F(2,1)$是用$\sigma[d/x]$为y指派的值$\sigma[d/x](y)=\sigma(y)=2$代入公式$F(y,a)$出现的$y$，而且用$a$的解释$[\![a]\!]=1$代入公式$F(y,a)$中的个体常量符号$a$而得到的。由于$\langle 2,1\rangle\in[\![F]\!]$，因此$F(2,1)$的真值为1，因此公式$\forall xF(y,a)$在上述解释和指派下的真值为1。

由于指导变元x没有在$F(y,a)$出现，因此这里实际上无需用论域的每个元素d代入$F(y,a)$中出现的x，再根据是全称量词将代入的结果合取起来，因为无$F(y,a)$中无x可代入。也就是说，**公式$\forall xF(y,a)$的真值与$F(y,a)$的真值相同**。

通过上述例子，我们进一步看到：

1. 当项中含有函数符号时，一样可使用简便的代入方法确定原子公式的真值，但要注意代入的方法，以及代入后公式的确切含义。

2. 当公式含有多个量词时，可以**从外层量词开始逐一展开**，特别是**当论域是有限集时，可将全**

称量词展开为公式的合取，存在量词展开为公式的析取。例如，设论域$\mathbf{D}=\{a,b,c\}$，则：

$$\begin{aligned}
\forall x\forall yA(x,y) &\Leftrightarrow \forall yA(a,y)\wedge\forall yA(b,y)\wedge\forall yA(c,y)\\
&\Leftrightarrow (A(a,a)\wedge A(a,b)\wedge A(a,c))\ \wedge\ (A(b,a)\wedge A(b,b)\wedge A(b,c))\wedge\\
&\quad (A(c,a)\wedge A(c,b)\wedge A(c,c))\\
\forall x\exists yA(x,y) &\Leftrightarrow \exists yA(a,y)\wedge\exists yA(b,y)\wedge\exists yA(c,y)\\
&\Leftrightarrow (A(a,a)\vee A(a,b)\vee A(a,c))\ \wedge\ (A(b,a)\vee A(b,b)\vee A(b,c))\wedge\\
&\quad (A(c,a)\vee A(c,b)\vee A(c,c))\\
\exists x\forall yA(x,y) &\Leftrightarrow \forall yA(a,y)\vee\forall yA(b,y)\vee\forall yA(c,y)\\
&\Leftrightarrow (A(a,a)\wedge A(a,b)\wedge A(a,c))\ \vee\ (A(b,a)\wedge A(b,b)\wedge A(b,c))\vee\\
&\quad (A(c,a)\wedge A(c,b)\wedge A(c,c))\vee\\
\exists x\exists yA(x,y) &\Leftrightarrow \exists yA(a,y)\vee\exists yA(b,y)\vee\exists yA(c,y)\\
&\Leftrightarrow (A(a,a)\vee A(a,b)\vee A(a,c))\ \vee\ (A(b,a)\vee A(b,b)\vee A(b,c))\vee\\
&\quad (A(c,a)\vee A(c,b)\vee A(c,c))
\end{aligned}$$

3. 由上面的展开可看到，通常$\forall x\exists yA(x,y)$的真值与$\exists x\forall yA(x,y)$的真值并不相同，上面例子也表明$\exists x\forall yF(x,y)$与$\forall y\exists xF(x,y)$的真值不相同，因此当有多个量词时，要注意**不能随意交换量词的顺序**，因而也就要注意**展开量词的顺序**，以及**展开后各公式之间的逻辑关系**。

4. 最后，当公式A中**不含有（或更准确地，不自由出现）**变量x时，公式$\forall xA$的真值与公式A的真值相同。在这种意义下，也可以说教材[4]在定义一阶公式$\forall xA$时要求x在A中自由出现也是有一定道理的。

上面讨论的例子中给出的论域都是有限的，当论域是无限集时，带量词公式的真值就不能像上面的例子一样机械地计算，这时只有运用量词的直观含义确定公式的真值。

例子 5.4.12 考虑下列公式：

(1) $F(f(x,y),g(x,y))$

(2) $\forall xF(g(x,y),z)$

(3) $\forall x\forall y(F(f(x,a),y)\to F(f(y,a),x))$

(4) $\exists xF(f(x,x),g(x,x))$

在解释$\mathcal{I}$：

(1) 论域：$\mathbf{D}=\mathbb{N}$，即论域是自然数集合；

(2) 个体常量符号a的解释是自然数零，也即：$[\![a]\!]=0$；

(3) 函数符号f的解释是自然数加法，而g的解释是自然数乘法，也即：

$$\forall x,y\in\mathbb{N},\quad [\![f]\!](x,y)=x+y\qquad\qquad [\![g]\!](x,y)=x\cdot y$$

(4) 谓词符号F的解释是恒等关系，也即：

$$[\![F]\!](x,y)\text{ 为真}\quad\text{当且仅当}\quad x=y$$

以及指派$\sigma:\{x,y,z\}\to\mathbb{N}$下的真值，这里$\sigma$定义为$\sigma(x)=1,\sigma(y)=2,\sigma(z)=3$。

解答：

(1). 对于公式$F(f(x,y),g(x,y))$使用上面介绍的代入法，用$\sigma(x)=1$代入x，用$\sigma(y)=2$代入y，得到它与公式

$$F(f(1,2),g(1,2))\Leftrightarrow F(1+2,1\cdot 2)$$

的真值相同，也即当$1+2$与$1\cdot 2$相等时，该公式的真值为1，显然$1+2$不等于$1\cdot 2$，因此公式$F(f(x,y),g(x,y)$在上述解释和指派σ下的真值为0。

(2). 对于公式$\forall xF(g(x,y),z)$，根据一阶公式的真值定义，公式$\forall xF(g(x,y),z)$在指派σ下的真值为真当且仅当对论域$\mathbb{N}$的任意元素d都有$F(g(x,y),z)$在指派$\sigma[d/x]$下的真值为真。根据前面的讨论，我们知道，要计算$F(g(x,y),z)$在$\sigma[d/x]$下的真值，要将$F(g(x,y),z)$中的x用d代入，y用$\sigma[d/x](y)=\sigma(y)=2$代入，$z$用$\sigma[d/x](z)=\sigma(z)=3$代入，也即$F(g(x,y),z)$在$\sigma[d/x]$下的真值与公式$F(g(d,2),3)$的真值相同，从而公式$\forall xF(g(x,y),z)$在指派$\sigma$下的真值为真，当且仅当对任意自然数$d\in\mathbb{N}$，公式$F(g(d,2),3)$为真，即：

$$\forall xF(g(x,y),z)\text{在指派}\sigma\text{下的真值为真 当且仅当 对任意的自然数}d\text{有}d\cdot 2=3$$

因此我们对自然数性质的直观理解，得到公式$\forall xF(g(x,y),z)$在上述解释和指派σ下的真值为0。直观地说，公式$forallxF(g(x,y),z)$在上述解释和指派σ的含义是：对任意的自然数x，$x\cdot 2=3$，因而显然是假命题（即真值为假）。

(3). 对于公式$\forall x\forall y(F(f(x,a),y)\to F(f(y,a),x))$，不难得到它在上述解释和指派下的直观含义是：对任意的自然数x，和任意的自然数y，如果$x+0=y$，则$y+0=x$，因而是真命题，则该公式的真值为1。实际上，由于该公式是闭公式，因此**该公式的真值与个体变量指派函数σ无关**。

(4). 对于公式$\exists xF(f(x,x),g(x,x))$，不难得到它在上述解释和指派下的直观含义是：存在自然数x，使得$x+x=x\cdot x$，为真命题，因为当$x=2$时，有$2+2=2\cdot 2=4$。因此该公式在上述解释和指派下的真值为1。同样，这个公式是闭公式，其真值与指派函数σ无关。

小结

通过上面的例子我们可以看出，一阶公式的真值计算显然比命题逻辑公式的真值计算要复杂得多。命题逻辑公式可根据公式的语法结构，在使用真值赋值函数指定公式中命题变量的真值之后，可以一步一步机械地根据公式真值的归纳定义计算其真值，进而能计算公式在任意真值赋值函数下的真值（直观地看，就是列公式的真值表）。

但是对于一阶公式，只有当解释的论域是有限集时，才能计算公式在某个个体变量指派函数下的真值，这时由于个体变量指派函数的个数也是有限的，我们也能计算一阶公式在任意个体变量指派函数下的真值。而当解释的论域是无穷集时，就没有机械可行的办法计算一阶公式的真值。

通过上面的讨论，对于一阶公式的真值，我们看到：

1. 一阶公式的真值通常**需要在给定解释，以及某个个体变量指派函数后才能确定**。但是，对于闭公式而言，其真值与个体变量指派函数无关，因而**只要给定解释就可确定闭公式的真值**。有的教材没有引入个体变量指派函数，因而只讨论闭公式的真值。注意，只有在给定解释之后，才能给出个体变量指派函数，因为指派函数是将个体变量指派为论域解释的元素。

2. 对于某个一阶公式，**个体变量指派函数只对该公式中自由出现的个体变量起作用**，在公式真值计算过程中，公式中约束出现的个体变量不是由个体变量指派函数指派值，而是根据量词的真值含义用论域中的元素代入。

3. **当解释的论域是有限集时，全称量词可展开为适当公式的合取，而存在量词可展开为适当公式的析取**。但当公式带有多个量词（特别是多个不同量词）时，量词之间的顺序不能任意交换，这时要注意展开的顺序以及展开公式之间的逻辑关系。

4. 当解释的论域是无穷集时，我们需要根据一阶公式的直观含义确定公式的真值，因为这时没有机械办法能一步一步根据公式的语法结构递归地计算公式的真值。

5.4.4 一阶公式的类型

根据公式的真值，我们也能对一阶公式进行分类：

定义 5.4.13

1. 如果一阶公式A在**任意解释**的**任意个体变量指派函数**下的**真值都为真**，则称A为**普遍有效式**，有的教材也称为**永真式**；

2. 如果一阶公式A在**某个解释**的**某个个体变量指派函数**下的真值为真，则称A为**可满足式**；

3. 如果一阶公式A在**任意解释**的**任意个体变量指派函数**下的**真值都为假**，则称A为**矛盾式**。

不难看到，一个一阶公式不是矛盾式就是可满足式，普遍有效式也是可满足式，因此可满足式可分为普遍有效式（永真式）和非普遍有效式的可满足式。一个一阶公式是可满足式，只要存在一个解释，以及该解释下的一个个体变量指派函数使得公式的真值为真，就是可满足式，而永真式，则是在任意解释和任意指派函数下的真值都为真。

显然要判断一个一阶公式是永真式、非永真式的可满足式还是矛盾式是一件十分困难的事情，因为公式的解释是无穷多的，解释的论域也可能是无穷集，因而不存在一个通用的方法判断一阶公式的类型，也即，从计算机的角度看，**一阶公式的类型判定是不可判定问题**。注意，命题逻辑公式的类型判定是可判定问题，可以根据公式的语法结构列真值表而确定公式的类型。

当然不存在通用方法判定一阶公式的类型，并不意味我们不能判定一些一阶公式的是否是永真式、矛盾式或可满足式。实际上，可以利用命题逻辑中的永真式和矛盾式去判断某些一阶公式的类型，特别是命题公式的替换实例的类型。

定义 5.4.14 设公式A_0是含有命题变量$p_1, p_2, \cdots, p_n$的命题公式，$A_1, A_2, \cdots, A_n$是n个一阶公式，对所有的$1 \leq i \leq n$，用A_i替换A_0中命题变量p_i的所有出现，得到的一阶公式A称为命题公式A_0的**替换实例**。

例子 5.4.15

1. 对于命题逻辑公式$p \to (q \to p)$，用$\forall x F(x, y)$替换其中的p，用$\exists y F(x, y)$替换其中的q，得到该公式的一个替换实例：

$$\forall x F(x, y) \to (\exists y F(x, y) \to \forall x F(x, y))$$

用$\forall x \forall y G(x, y)$替换其中的$p$，用$\exists x \exists y G(x, y)$替换其中的$q$，我们得到该公式的另一个替换实例：

$$\forall x \forall y G(x, y) \to (\exists x \exists y G(x, y) \to \forall x \forall y G(x, y))$$

2. 在多数情况下，我们通常**要针对一个一阶公式考察它是否是某个命题公式的替换实例**。例如对于公式：

$$(\forall xF(x,y) \vee \exists xG(x,y)) \wedge \neg\exists xG(x,y) \rightarrow \forall xF(x,y)$$

读者要能看出它实际上是命题公式$(p \vee q) \wedge \neg q \rightarrow p$的替换实例，其中$p$使用$\forall xF(x,y)$替换，而$q$使用$\exists xG(x,y)$替换。

对于命题公式的替换实例，我们有：

定理 5.4.16 **命题逻辑中的永真式的替换实例是一阶逻辑中的永真式，而命题逻辑中的矛盾式的替换实例是一阶逻辑中的矛盾式。** □

由于$p \rightarrow (q \rightarrow p)$和$(p \vee q) \wedge \neg q \rightarrow p$都是命题逻辑中的永真式，前者是命题逻辑公理化演算系统的公理，而后者实际上是析取三段论，因此上面给出的替换实例：

$$\forall xF(x,y) \rightarrow (\exists yF(x,y) \rightarrow \forall xF(x,y))$$
$$\forall x\forall yG(x,y) \rightarrow (\exists x\exists yG(x,y) \rightarrow \forall x\forall yG(x,y))$$
$$(\forall xF(x,y) \vee \exists xG(x,y)) \wedge \neg\exists xG(x,y) \rightarrow \forall xF(x,y)$$

都是一阶逻辑的永真式。

上述定理是一个十分重要的定理，其证明比较复杂，这里不作进一步的讨论。由于命题逻辑的永真式的替换实例在一阶逻辑中也是永真式，这使得我们在一阶公式的等值演算和自然推理中可使用命题逻辑的基本等值式和自然推理规则。也就是说，上述定理实际上在命题逻辑和一阶逻辑之间架起了一道桥梁，使得我们在一阶逻辑中可很好地运用命题逻辑的成果。

除了命题逻辑公式的替换实例之外，我们还可使用分析证明的方法判断一些简单公式的类型：

例子 5.4.17 判断下列公式的类型：

(1) $\forall x(F(x) \rightarrow G(x))$

(2) $\forall xF(x) \rightarrow \exists xF(x)$

(3) $\forall x\exists yH(x,y) \rightarrow \exists x\forall yH(x,y)$

分析：根据永真式、矛盾式和可满足式的定义，**要证明一个公式是可满足式，只要找到一个解释及一个个体变量指派函数使得该公式的真值为真即可**；而**要证明一个公式不是永真式，只要找到一个解释及一个个体变量指派函数使得该公式的真值为假即可**。也就是说，我们可以通过例举解释和个体变量指派函数的方法证明一个公式是非永真式的可满足式。另外，我们也可根据定义，证明简单公式是永真式或矛盾式。

解答：

(1). 对于公式$\forall x(F(x) \rightarrow G(x))$，首先我们猜想它是非永真式的可满足式，因为按照公式的直观含义：对任意的x，$F(x)$成立意味着$G(x)$成立，由于$F(x)$和$G(x)$的任意性，不难想到它不会是永真式，也不会是矛盾式。另外，这个公式是闭公式，其真值与个体变量指派函数无关。因此我们只要列举一个解释，使得该公式在此解释下为真，再列举一个解释，使得该公式在该解释下为假即可。

(a). 给定解释$\mathcal{I}_1$的论域是$\mathbf{D} = \{0,1\}$，一元谓词符号F和G的解释是：

$$[\![F]\!] = \{0\} \qquad [\![G]\!] = \{1\}$$

也即$F(0)$的真值为真，而$F(1)$为假，$G(0)$为假而$G(1)$为真。公式$\forall x(F(x) \to G(x))$在此解释下的真值是：

$$\forall x(F(x) \to G(x)) \Leftrightarrow (F(0) \to G(0)) \wedge (F(1) \to G(1)) \Leftrightarrow (1 \to 0) \wedge (0 \to 1) \Leftrightarrow 0$$

也即，公式$\forall x(F(x) \to G(x))$在解释$\mathcal{I}_1$下的真值为假。

(b). 给定解释$\mathcal{I}_2$的论域仍是$\mathbf{D} = \{0, 1\}$，一元谓词符号F和G的解释是：

$$[\![F]\!] = \{1\} \qquad [\![G]\!] = \{1\}$$

公式$\forall x(F(x) \to G(x))$在此解释下的真值是：

$$\forall x(F(x) \to G(x)) \Leftrightarrow (F(0) \to G(0)) \wedge (F(1) \to G(1)) \Leftrightarrow (0 \to 0) \wedge (1 \to 1) \Leftrightarrow 1$$

也即，公式$\forall x(F(x) \to G(x))$在解释$\mathcal{I}_2$下的真值为真。从而表明该公式是非永真式的可满足式。

(2). 对于公式$\forall x F(x) \to \exists x F(x)$，其直观含义是，对任意的$x$，如果$F(x)$成立，则存在$x$使得$F(x)$成立，因此我们可猜想此公式是永真式。对此，我们根据永真式的定义证明。同样，由于该公式是闭公式，因此无需考虑个体变量指派函数。对于任意的解释$\mathcal{I}$，如果其论域$\mathbf{D}$存在元素d使得$F(d)$为假，那么公式$\forall x F(x) \to \exists x F(x)$的前件假，从而整个公式的真值为真。若对其论域的任意元素d，都有$F(d)$为真，则由于论域必须是非空集，因此存在论域元素d使得$F(d)$为真，从而公式$\forall x F(x) \to \exists x F(x)$的前件和后件都为真，整个公式也为真。因此对任意的解释$\mathcal{I}$，该公式的真值都为真，从而该公式是永真式。

(3). 对于公式$\forall x \exists y H(x, y) \to \exists x \forall y H(x, y)$，我们同样猜想其是非永真式的可满足式。这个公式也是闭公式，我们只要列举一个解释使得该公式为真，再列举一个解释使得该公式为假即可。

(a). 给定解释$\mathcal{I}_1$的论域为自然数集$\mathbb{N}$，谓词$H(x, y)$的解释为：

$$[\![H]\!] = \{\langle n, m\rangle \mid n \leq m\} \subseteq \mathbb{N} \times \mathbb{N}$$

也即，对任意的自然数x, y，$H(x, y)$为真当且仅当$x \leq y$。在此解释下，公式$\forall x \exists y H(x, y) \to \exists x \forall y H(x, y)$的直观含义是：

若对任意自然数x，存在自然数y使得$x \leq y$，则存在自然数x满足对任意的自然数y都有$x \leq y$

因为确实对任意自然数x都存在比它大的自然数，而且也存在自然数0使得任意的自然数都比它大，因此上述命题是真命题，也即公式$\forall x \exists y H(x, y) \to \exists x \forall y H(x, y)$在此解释下的真值为真。

(b). 给定解释$\mathcal{I}_1$的论域为自然数集$\mathbb{N}$，谓词$H(x, y)$的解释为自然数上的恒等关系：

$$[\![H]\!] = \{\langle n, n\rangle \mid n \in \mathbb{N}\}$$

也即，对任意的自然数x, y，$H(x, y)$为真当且仅当$x = y$。在此解释下，公式$\forall x \exists y H(x, y) \to \exists x \forall y H(x, y)$的直观含义是：

若对任意自然数x，存在自然数y使得$x = y$，则存在自然数x满足对任意的自然数y都有$x = y$

显然此命题是假命题，也即公式$\forall x \exists y H(x, y) \to \exists x \forall y H(x, y)$在此解释下的真值为假。综上有公式$\forall x \exists y H(x, y) \to \exists x \forall y H(x, y)$是非永真式的可满足式。

作 业

作业 5.1 （此作业是教材[4]习题4的第5题）将下列命题在一阶逻辑中符号化：

(1) 一切事物都是发展的。

(2) 凡有理数都可写成分数。

(3) 所有的油脂都不溶于水。

(4) 存在着会说话的机器人。

(5) 过平面上两个点，有且仅有一条直线通过。

(6) 凡实数都能比较大小。

(7) 在北京工作的人未必都是北京人。

(8) 只有一个北京。

(9) 任何金属都可溶解在某种溶液里。

(10) 如果明天天气好，有些学生将去香山。

作业 5.2 （此作业是教材[4]习题4的第5题，略有修改）设：

$P(x)$：表示x是有理数	$Q(x)$：表示x是实数	$R(x)$：表示x是无理数
$L(x)$：表示x是整数数	$S(x)$：表示x是偶数	$W(x)$:表示x是奇数

试将下列公式翻译成自然语句：

(1) $\forall x(P(x) \to Q(x))$

(2) $\exists x(P(x) \wedge Q(x))$

(3) $\neg\forall x(Q(x) \to P(x))$

(4) $\forall x(Q(x) \to ((P(x) \wedge \neg R(x)) \vee (\neg P(x) \wedge R(x))))$

(5) $\neg\exists x(L(x) \wedge S(x) \wedge W(x))$

(6) $\forall x(L(x) \to P(x)) \wedge \neg\forall x(P(x) \to L(x))$

作业 5.3 指出下列公式中量词的辖域，说明每个变量符号的出现是指导变元、约束出现还是自由出现，并给出整个公式的自由变量和约束变量（下列公式来自[4]习题4的第3题）：

(1) $\forall x(P(x) \wedge Q(x)) \to (\forall x R(x) \wedge Q(z))$

(2) $\forall x(P(x) \wedge \exists y Q(y)) \ \vee \ (\forall x P(x) \to Q(z))$

(3) $\forall x(P(x) \leftrightarrow Q(x)) \ \wedge \ \exists y R(y) \wedge S(z)$

作业 5.4 给定解释$\mathcal{I}$如下：

(1) 论域$\mathbf{D} = \mathbb{R}$是实数集合；

(2) 个体常量符号a的解释是$[\![a]\!] = 0$；

(3) 二元函数符号f的解释是$[\![f]\!]$是通常的减法运算，即：$\forall x, y{\in}\mathbb{R}, \quad [\![f]\!](x, y) = x - y$

(4) 二元谓词符号F的解释是$[\![F]\!]$是相等关系，即$F(x, y)$为真当且仅当$x = y$，而二元谓词符号G的解释是$[\![R]\!]$是小于关系，即$G(x, y)$为真当且仅当$x < y$。

试说明下列公式在上述解释下的直观含义，并指出各公式的真值（注意下列公式都是闭公式所以无需个体变量指派函数）：

(1) $\forall x\forall y(G(x, y) \to \neg F(x, y))$

(2) $\forall x\forall y(F(f(x,y),a)\to G(x,y))$
(3) $\forall x\forall y(G(x,y)\to\neg F(f(x,y),a))$
(4) $\forall x\forall y(G(f(x,y),a)\to F(x,y))$

作业 5.5 给定解释$\mathcal{I}$如下：
(1) 论域$\mathbf{D}=\{3,4\}$；
(2) 一元函数符号f的解释$[\![f]\!]$定义为$[\![f]\!](3)=4,[\![f]\!](4)=3$；
(3) 二元谓词符号F的解释为$[\![F]\!]=\{\langle 3,4\rangle,\langle 4,3\rangle\}$，即：

$$F(3,4)=F(4,3)=1 \qquad F(3,3)=F(4,4)=0$$

试给出下列公式在解释$\mathcal{I}$下的真值（同样下述公式是闭公式，因此无需个体变量指派函数）：
(1) $\forall x\exists yF(x,y)$
(2) $\exists x\forall yF(x,y)$
(3) $\forall x\forall y(F(x,y)\to F(f(x),f(y)))$

作业 5.6 设解释的论域$\mathbf{D}=\{a,b,c\}$，请展开下列公式中的量词：
(1) $\forall x\exists y(F(x)\wedge G(y))$
(2) $\forall x\forall y(F(x)\vee G(y))$
(3) $\forall xF(x)\to\forall yG(y)$
(4) $\forall x(F(x,y)\to\exists yG(y))$

作业 5.7 试判断下列公式的类型（永真式、非永真式的可满足式亦或是矛盾式）：
(1) $\forall x(F(x)\to F(x))\to\exists y(G(y)\wedge\neg G(y))$
(2) $\neg(\forall xF(x)\to\exists yG(y))\wedge\exists yG(y)$
(3) $\forall x\forall y(F(x,y)\to F(y,x))$
(4) $\forall x(F(x)\to\exists y(G(y)\wedge H(x,y))$

第六章　一阶逻辑的等值演算

与命题逻辑类似，这一章我们讨论一阶公式等值的含义，并且给出一些基本等值式，探讨如何使用等值演算法证明两个一阶公式等值，进一步讨论一阶公式在某种意义下的标准形式，即一阶公式的前束范式。

6.1　一阶逻辑的基本等值式

两个一阶公式等值的含义与命题逻辑公式等值的含义类似：

定义 6.1.1 设A和B是两个一阶公式，如果对任意的解释及任意的个体变量指派函数下，A和B都具有相同的真值，则称**A与B等值**，记为$A \Leftrightarrow B$。同样，根据等价联结词的含义，A和B等值当且仅当$A \leftrightarrow B$是永真式。

由前面对一阶公式永真式的定义，我们知道，不像命题逻辑公式可使用列真值表的方法判断命题逻辑公式是否是永真式，一阶公式在很多时候很难判断是否是永真式。因此，判断两个一阶公式是否等值只有使用等值演算的方法，即**从一些基本的等值式出发，利用等值置换规则，对两个公式进行变形，如果它们能变形为相同的公式则等值**。

由于**命题逻辑中永真式的替换实例也是一阶逻辑公式的永真式**，因此命题逻辑的所有基本等值式（模式）在一阶逻辑的代入实例（即用一阶公式代入这些基本等值式中的代表任意公式的符号A, B, C等得到的公式）都是一阶公式的等值式。例如，对于蕴涵等值式$(A \to B) \Leftrightarrow (\neg A \vee B)$，用一阶公式$F(x, y)$代入其中的$A$，用$\forall x F(x, y)$代入其中的$B$，得到一阶公式的等值式：

$$(F(x, y) \to \forall x F(x, y)) \Leftrightarrow (\neg F(x, y) \vee \forall x F(x, y))$$

因此，通过替换实例这个桥梁，在一阶逻辑的等值演算中可使用前面所学的命题逻辑中的基本等值式。

为方便起见，我们先复习一下命题逻辑中的基本等值式，这些等值式实际上体现了各个命题逻辑联结词的基本性质。例如，否定联结词有双重否定律，合取和析取联结词有幂等律、交换律和结合律，合取对析取，以及析取对合取有分配律等。在这些基本等值式中，其中最常用而又比较难记的有：

德摩根律	$\neg(A \wedge B) \Leftrightarrow \neg A \vee \neg B$	$\neg(A \vee B) \Leftrightarrow \neg A \wedge \neg B$
吸收律	$A \wedge (A \vee B) \Leftrightarrow A$	$A \vee (A \wedge B) \Leftrightarrow A$

另外初学者容易记错的还有：

零律	$A \wedge 0 \Leftrightarrow 0$	$A \vee 1 \Leftrightarrow 1$
同一律	$A \wedge 1 \Leftrightarrow A$	$A \vee 0 \Leftrightarrow A$
矛盾律	$A \wedge \neg A \Leftrightarrow 0$	
排中律	$A \vee \neg A \Leftrightarrow 1$	

最后，在命题逻辑中，关于蕴涵和等价联结词的基本等值式是：

蕴涵等值式	$A \to B \Leftrightarrow \neg A \vee B$
等价等值式	$A \leftrightarrow B \Leftrightarrow (A \to B) \wedge (B \to A)$

除了以上由命题逻辑的基本等值式的替换实例所得到的等值式外，一阶逻辑还有许多特有的等值式，这些等值式都与量词密切相关。首先是当个体域是有限集时，可以将量词展开：

定理 6.1.2 **消去量词等值式**：设个体域是有限集$D = \{a_1, a_2, \cdots, a_n\}$，则有：

$$\forall x A(x) \Leftrightarrow A(a_1) \wedge A(a_2) \wedge \cdots \wedge A(a_n)$$
$$\exists x A(x) \Leftrightarrow A(a_1) \wedge A(a_2) \wedge \cdots \wedge A(a_n)$$

注意，上面用记号$A(x)$表示量词$\forall x(\exists x)$的辖域对应的公式，强调该公式含有自由变量x，而对任意的$1 \le i \le n$，$A(a_i)$是用个体域中的元素a_i替换公式$A(x)$中出现的所有x后得到的公式。

在上一章已经提到，当个体域有限时，全称量词可展开为适当公式的合取，而存在量词可展开为适当公式的析取。所谓的适当公式，就是**用个体域的每一个元素代入量词指导变元在该量词辖域对应公式的所有出现后得到的公式**。注意，在上述消去量词等值式中，公式$A(x)$还可能含有其他个体变量，甚至是自由出现的个体变量。例如，设个体域是$D = \{a, b\}$，公式$\forall x \exists y(F(x, y) \to G(y, z))$展开为：

$$\begin{aligned}\forall x \exists y(F(x, y) \to G(y, z)) &\Leftrightarrow \exists y(F(a, y) \to G(y, z)) \wedge \exists y(F(b, y) \to G(y, z)) \\ &\Leftrightarrow ((F(a, a) \to G(a, z)) \vee (F(a, b) \to G(b, z))) \wedge \\ &\quad ((F(b, a) \to G(a, z)) \vee (F(b, b) \to G(b, z)))\end{aligned}$$

注意，公式的最后展开可含有自由变量。

在一阶逻辑中，最常用的与否定联结词及量词有关的等值式是：

定理 6.1.3 **量词否定等值式**：设$A(x)$是任意的含有自由变量x的公式，则：

$$\neg \forall x A(x) \Leftrightarrow \exists \neg A(x) \qquad \neg \exists x A(x) \Leftrightarrow \forall x \neg A(x)$$

量词否定等值式的直观含义很容易理解：“不是所有的”那么意味着“存在不是的”，“不存在”则意味着“所有的都不”。例如，“不是所有的实数都是无理数”等价于“存在不是无理数的实数”，如果设$R(x)$表示“x是实数”，$W(x)$表示“x是无理数”，那么有等值式：

$$\neg \forall x(R(x) \to W(x)) \Leftrightarrow \exists x \neg(R(x) \to W(x)) \Leftrightarrow \exists x(R(x) \wedge \neg W(x))$$

类似地，“不存在能表示成分数的无理数”等价于“所有无理数都不能表示成分数”，进一步设$F(x)$表示“x能表示成分数”，则有等值式：

$$\neg\exists x(W(x)\wedge F(x)) \Leftrightarrow \forall x\neg(W(x)\wedge F(x)) \Leftrightarrow \forall x(W(x)\rightarrow\neg F(x))$$

我们可根据永真式的定义，证明下面的两个公式：

$$(\neg\forall xA(x))\leftrightarrow(\exists\neg A(x)) \qquad (\neg\exists xA(x))\leftrightarrow(\forall x\neg A(x))$$

都是永真式，但为简单起见，此处不再对一阶逻辑的基本等值式加以证明，读者只要能理解和运用它们即可。

对于量词和析取、合取以及蕴涵等联结词之间的关系，一阶逻辑常用的等值式有量词辖域扩张和收缩等值式：

定理 6.1.4 **量词辖域扩张和收缩等值式**：设$A(x)$是任意的含有自由变量x的公式，且x**不在**B**中出现**，则：

$$\begin{aligned}
\forall x(A(x)\vee B) &\Leftrightarrow \forall xA(x)\vee B\\
\forall x(A(x)\wedge B) &\Leftrightarrow \forall xA(x)\wedge B\\
\forall x(A(x)\rightarrow B) &\Leftrightarrow \exists xA(x)\rightarrow B\\
\forall x(B\rightarrow A(x)) &\Leftrightarrow B\rightarrow\forall xA(x)\\
\exists x(A(x)\vee B) &\Leftrightarrow \exists xA(x)\vee B\\
\exists x(A(x)\wedge B) &\Leftrightarrow \exists xA(x)\wedge B\\
\exists x(A(x)\rightarrow B) &\Leftrightarrow \forall xA(x)\rightarrow B\\
\exists x(B\rightarrow A(x)) &\Leftrightarrow B\rightarrow\exists xA(x)
\end{aligned}$$

对于量词辖域扩张和收缩等值式，读者需要注意两点：

1. 上述等值式要求量词的指导变元符号x不在公式B中出现，或更准确地说，**要求**x**不在公式**B**中自由出现**，因为当x在公式B中约束出现时（这时实际上可能出现量词辖域嵌套），我们可以使用约束变量改名规则，将B中约束出现的x改名为其他变量符号。例如：

$$\forall x(A(x)\vee\forall xB(x)) \Leftrightarrow \forall x(A(x)\vee\forall yB(y)) \Leftrightarrow \forall xA(x)\vee\forall yB(y)$$

特别地，从另一个方向使用上述等值式，我们可使用约束变量改名规则或自由变量替换规则使得B中不出现x，例如：

$$\forall xA(x)\vee B(x) \Leftrightarrow \forall xA(x)\vee B(y) \Leftrightarrow \forall x(A(x)\vee B(y))$$

注意上面第一步使用自由变量替换规则将B中自由出现的x替换为y，这里假定y不在原来的整个公式$\forall xA(x)\vee B(x)$中出现。类似地，我们也有：

$$\forall xA(x)\vee\forall xB(x) \Leftrightarrow \forall xA(x)\vee\forall yB(y) \Leftrightarrow \forall x(A(x)\vee\forall yB(y)) \Leftrightarrow \forall x\forall y(A(x)\vee B(y))$$

注意上面第一步使用约束变量改名规则将$\forall xB(x)$中的量词指导变元符号x改名为y，然后使用量词辖域扩张等值式，将公式$\forall yB(y)$放到量词$\forall x$的辖域中。上面最后一步，由于$A(x)$中不出现y（因为

在使用约束变量改名规则时，我们选择的是新的变量符号y)，因此可再一次使用量词辖域扩张等值式，扩张量词$\forall y$的辖域。

2. 上面与蕴涵联结词有关的等值式中，注意当量词约束的是蕴涵前件时，**扩张和收缩量词时要改变量词**，全称量词改为存在量词，存在量词改为全称量词：

$$\forall x(B \to A(x)) \Leftrightarrow B \to \forall x A(x)$$
$$\exists x(A(x) \to B) \Leftrightarrow \forall x A(x) \to B$$

但当量词约束的是蕴涵后件时，则量词无需改变：

$$\forall x(A(x) \to B) \Leftrightarrow \exists x A(x) \to B$$
$$\exists x(B \to A(x)) \Leftrightarrow B \to \exists x A(x)$$

实际上，与蕴涵联结词有关的量词辖域扩张收缩等值式可由与析取有关的量词辖域扩张等值式和量词否定等值式推出：

$$\begin{aligned}
\forall x(A(x) \to B) &\Leftrightarrow \forall x(\neg A(x) \vee B) && // \text{蕴涵等值式}\\
&\Leftrightarrow \forall x(\neg A(x)) \vee B && // \text{量词辖域收缩}\\
&\Leftrightarrow \neg\exists x A(x) \vee B && // \text{量词否定等值式}\\
&\Leftrightarrow \exists x A(x) \to B && // \text{蕴涵等值式}
\end{aligned}$$

其他等值式也可类似推出。同样地，如果量词中的辖域含有等价联结词，那么**要先将等价联结词化成蕴涵等价联结词**之后，再将量词辖域的扩张与收缩，例如，假设x不在B中出现：

$$\begin{aligned}
\forall x(A(x) \leftrightarrow B) &\Leftrightarrow \forall x((A(x) \to B) \wedge (B \to A(x))) && // \text{等价等值式}\\
&\Leftrightarrow \forall x((A(x) \to B)) \wedge \forall x(B \to A(x)) && // \text{量词分配等值式}\\
&\Leftrightarrow (\exists x A(x) \to B) \wedge (B \to \forall x A(x)) && // \text{量词辖域收缩}
\end{aligned}$$

上面使用的量词分配等值式我们将在下面讨论。因此：

$$\forall x(A(x) \leftrightarrow B) \not\Leftrightarrow \forall x A(x) \leftrightarrow B \qquad \forall x(B \leftrightarrow A(x)) \not\Leftrightarrow B \leftrightarrow \forall x A(x)$$

最后，在一阶逻辑中，与量词及析取和合取联结词有关的常用等值式还有：

定理 6.1.5 **量词分配等值式**：设$A(x), B(x)$是任意的含有自由变量x的公式，则：

$$\forall x(A(x) \wedge B(x)) \Leftrightarrow \forall x A(x) \wedge \forall x B(x)$$
$$\exists x(A(x) \vee B(x)) \Leftrightarrow \exists x A(x) \vee \exists x B(x)$$

上述等值式表明，**全称量词对合取有分配律，存在量词对析取有分配律**，这不难理解，因为全称量词在某种意义下相当于合取（例如在有限个体域中展开为适当公式的合取），而存在量词在某种意义下相当于析取。不过，虽然在命题逻辑中，合取对析取有分配律，在一阶逻辑中，**全称量词对析取没有分配律，存在量词对合取也没有分配律**：

$$\forall x(A(x) \vee B(x)) \not\Leftrightarrow \forall x A(x) \vee \forall x B(x)$$
$$\exists x(A(x) \wedge B(x)) \not\Leftrightarrow \exists x A(x) \wedge \exists x B(x)$$

小 结

我们将一阶逻辑中常用的基本等值式总结如下。首先，下面的表给出了命题逻辑中的基本等值式模式，这些等值式模式中的A, B, C等用一阶公式代入，则得到一阶逻辑的等值式：

名称	基本等值式模式
双重否定律	$A \Leftrightarrow \neg\neg A$
幂等律	$A \Leftrightarrow A \wedge A$　　$A \Leftrightarrow A \vee A$
交换律	$A \wedge B \Leftrightarrow B \wedge A$　　$A \vee B \Leftrightarrow B \vee A$
结合律	$(A \wedge B) \wedge C \Leftrightarrow A \wedge (B \wedge C)$　　$(A \vee B) \vee C \Leftrightarrow A \vee (B \vee C)$
分配律	$A \vee (B \wedge C) \Leftrightarrow (A \vee B) \wedge (A \vee C)$　　$A \wedge (B \vee C) \Leftrightarrow (A \wedge B) \vee (A \wedge C)$
吸收律	$A \wedge (A \vee B) \Leftrightarrow A$　　$A \vee (A \wedge B) \Leftrightarrow A$
德摩根律	$\neg(A \wedge B) \Leftrightarrow (\neg A \vee \neg B)$　　$\neg(A \vee B) \Leftrightarrow (\neg A \wedge \neg B)$
零律	$A \wedge 0 \Leftrightarrow 0$　　$A \vee 1 \Leftrightarrow 1$
同一律	$A \wedge 1 \Leftrightarrow A$　　$A \vee 0 \Leftrightarrow A$
矛盾律	$A \wedge \neg\, A \Leftrightarrow 0$
排中律	$A \vee \neg\, A \Leftrightarrow 1$
蕴涵等值式	$A \to B \Leftrightarrow \neg\, A \vee B$
等价等值式	$A \leftrightarrow B \Leftrightarrow (A \to B) \wedge (B \to A)$

而下面的表则给出了一阶逻辑中特有的等值式，对于这些等值式，读者还需注意这些等值式成立的条件：

名称	基本等值式模式	成立条件
消除量词等值式	$\forall x A(x) \Leftrightarrow A(a_1) \wedge A(a_2) \wedge \cdots \wedge A(a_n)$ $\exists x A(x) \Leftrightarrow A(a_1) \vee A(a_2) \vee \cdots \vee A(a_n)$	个体域是有限集： $D = \{a_1, a_2, \cdots, a_n\}$
量词否定等值式	$\neg\forall x A(x) \Leftrightarrow \exists \neg A(x)$ $\neg\exists x A(x) \Leftrightarrow \forall x \neg A(x)$	
量词辖域扩张/收缩	$\forall x(A(x) \vee B) \Leftrightarrow \forall x A(x) \vee B$ $\forall x(A(x) \wedge B) \Leftrightarrow \forall x A(x) \wedge B$ $\forall x(A(x) \to B) \Leftrightarrow \exists x A(x) \to B$ $\forall x(B \to A(x)) \Leftrightarrow B \to \forall x A(x)$ $\exists x(A(x) \vee B) \Leftrightarrow \exists x A(x) \vee B$ $\exists x(A(x) \wedge B) \Leftrightarrow \exists x A(x) \wedge B$ $\exists x(A(x) \to B) \Leftrightarrow \forall x A(x) \to B$ $\exists x(B \to A(x)) \Leftrightarrow B \to \exists x A(x)$	$A(x)$是含自由变量x的公式，而且x不在公式B中出现
量词分配等值式	$\forall x(A(x) \wedge B(x)) \Leftrightarrow \forall x A(x) \wedge \forall x B(x)$ $\exists x(A(x) \vee B(x)) \Leftrightarrow \exists x A(x) \vee \exists x B(x)$	$A(x), B(x)$是含自由变量x的公式

读者在下面进行等值演算时，请使用上述基本等值式，并在注释中写明所用的基本等值式名称，其中量词辖域扩张和收缩等值式可根据使用的方向分别使用名称“量词辖域扩张”或“量词辖域收缩”。

6.2 一阶逻辑的等值演算举例

与命题逻辑类似，证明两个一阶公式等值的演算过程主要使用等值替换规则：

定理 6.2.1 **等值置换规则**：设Φ是任意命题逻辑公式，A是Φ的子公式，B是与A等值的公式，则将Φ中的子公式A**置换成**B得到的公式Φ'与Φ等值。

一阶公式的子公式定义与命题逻辑公式的子公式定义类似，例如，对于公式$(A \wedge B)$，那么A和B就是它的子公式。对于带量词的公式，我们定义公式A是$\forall xA$和$\exists xB$的子公式。例如，公式$\exists y(G(x,y) \to F(x,y))$就是公式$\forall x\exists y(G(x,y) \to F(x,y))$的子公式，而$(G(x,y) \to F(x,y))$又是公式$\exists y(G(x,y) \to F(x,y))$的子公式。我们定义的子公式概念具有传递性，即$A$是$B$的子公式，$B$是$C$的子公式，则$A$也是$C$的子公式。

与命题逻辑的等值演算相同，等值置换规则中的将Φ中的子公式A**置换成**与A等值的公式B是将Φ中的子公式A整个换成公式B，例如$\neg\forall xF(x,y)$是公式$(\neg\forall xF(x,y) \to G(x,y))$的子公式，根据量词否定等值式，$\neg\forall xF(x,y)$与$\exists x\neg F(x,y)$等值，用$\exists x\neg F(x,y)$置换$(\neg\forall xF(x,y) \to G(x,y))$中的子公式$\neg\forall xF(x,y)$得到$(\exists x\neg F(x,y) \to G(x,y))$。等值置换规则断定

$$(\neg\forall xF(x,y) \to G(x,y)) \Leftrightarrow (\exists x\neg F(x,y) \to G(x,y))$$

备注 6.2.2 在一阶逻辑中，等值置换规则的正确性建立在以下事实的基础上：

1. 若$A \Leftrightarrow A'$，则$(\neg A) \Leftrightarrow (\neg A')$；

2. 若$A \Leftrightarrow A'$且$B \Leftrightarrow B'$，则：$(A \wedge B) \Leftrightarrow (A' \wedge B')$、$(A \vee B) \Leftrightarrow (A' \vee B')$、$(A \to B) \Leftrightarrow (A' \to B')$以及$(A \leftrightarrow B) \Leftrightarrow (A' \leftrightarrow B')$；

3. 若$A \Leftrightarrow A'$，则对任意的个体变量符号x，有$\forall xA \Leftrightarrow \forall xA'$及$\exists xA \Leftrightarrow \exists xA'$。

根据公式等值的定义，任意证明上述事实，然后根据这些事实，以及子公式的定义可证明等值置换规则的正确性，此处不作更深入的讨论。

下面我们以一些例子说明如何利用等值演算方法证明一阶公式的等值。

例子 6.2.3 证明下列各等值式：

(1) $\forall x(T(x) \to \forall y(W(y) \to K(x,y))) \Leftrightarrow \forall x\forall y(T(x) \wedge W(y) \to K(x,y))$

(2) $\exists x(T(x) \wedge \exists y(W(y) \wedge K(x,y))) \Leftrightarrow \exists x\exists y(T(x) \wedge W(y) \wedge K(x,y))$

(3) $\neg\forall x\forall y(T(x) \wedge W(y) \to K(x,y)) \Leftrightarrow \exists x\exists y(T(x) \wedge W(y) \wedge \neg K(x,y))$

(4) $\neg\exists x\exists y(T(x) \wedge W(y) \wedge K(x,y)) \Leftrightarrow \forall x\forall y(T(x) \to W(y) \to \neg K(x,y))$

(5) $\neg\exists x(T(x) \wedge \forall y(W(y) \to K(x,y))) \Leftrightarrow \forall x(T(x) \to \exists y(W(y) \wedge \neg K(x,y)))$

(6) $\neg\forall x(T(x) \to \exists y(W(y) \wedge K(x,y))) \Leftrightarrow \exists x(T(x) \wedge \forall y(W(y) \to \neg K(x,y))$

解答：这里给出的等值式大部分都是在前面符号化时碰到的等值式。设$T(x)$表示x是兔子，$W(y)$表示y是乌龟，$K(x,y)$表示x比y快，则上述公式都可翻译成自然语言命题，从而也可看出上述等值式的某种直观含义。

1. 实际上，公式$\forall x(T(x) \to \forall y(W(y) \to K(x,y)))$的直观含义是，对任意的$x$，如果$x$是兔子，那么对任意的$y$，若$y$是乌龟，则$x$比$y$快。而公式$\forall x\forall y(T(x) \wedge W(y) \to K(x,y))$的含义是，对任意

的x和y，如果x是兔子，而且y是乌龟，则x比y快。这两个公式都可看作是句子“兔子比乌龟跑得快”的符号化，它们是等值的：

$$\begin{aligned}
&\forall x(T(x)\rightarrow\forall y(W(y)\rightarrow K(x,y)))\\
&\quad\Leftrightarrow\forall x\forall y(T(x)\rightarrow(W(y)\rightarrow K(x,y))) && // \text{ 量词辖域扩张}\\
&\quad\Leftrightarrow\forall x\forall y(\neg T(x)\vee\neg W(y)\vee K(x,y)) && // \text{ 蕴涵等值式}\\
&\quad\Leftrightarrow\forall x\forall y(\neg(T(x)\wedge W(y))\vee K(x,y)) && // \text{ 德摩根律}\\
&\quad\Leftrightarrow\forall x\forall y(T(x)\wedge W(y)\rightarrow K(x,y)) && // \text{ 蕴涵等值式}
\end{aligned}$$

上面第一步可使用量词辖域扩张，是因为对于公式$T(x)\rightarrow\forall y(W(y)\rightarrow K(x,y))$而言，$y$没有在$T(x)$中出现，因此根据量词辖域扩张等值式有：

$$T(x)\rightarrow\forall y(W(y)\rightarrow K(x,y))\ \Leftrightarrow\ \forall y(T(x)\rightarrow(W(y)\rightarrow K(x,y)))$$

2. 公式$\exists x(T(x)\wedge\exists y(W(y)\wedge K(x,y)))$的直观含义是，存在$x$，$x$是兔子，而且存在$y$，$y$是乌龟，且$x$比$y$快。而公式$\exists x\exists y(T(x)\wedge W(y)\wedge K(x,y))$的直观含义是，存在$x$，存在$y$，$x$是兔子而且$y$是乌龟，而且$x$比$y$快。简短而言，它们都是句子“有的兔子比有的乌龟跑得快”的符号化，它们是等值的：

$$\begin{aligned}
&\exists x(T(x)\wedge\exists y(W(y)\wedge K(x,y)))\\
&\quad\Leftrightarrow\exists x\exists y(T(x)\wedge W(y)\wedge K(x,y))) && // \text{ 量词辖域扩张}
\end{aligned}$$

3. 公式$\neg\forall x\forall y(T(x)\wedge W(y)\rightarrow K(x,y))$的直观含义是，并非对任意的$x$和任意的$y$，如果$x$是兔子而且$y$是乌龟，则$x$比$y$快，简短而言是“并非所有兔子都比所有乌龟跑得快”。而公式$\exists x\exists y(T(x)\wedge W(y)\wedge\neg K(x,y))$的直观含义是，存在$x$且存在$y$，$x$是兔子，$y$是乌龟，且$x$不比$y$快，简短而言是“有的兔子不比有的乌龟跑得快”。从而自然语言角度看，这两个说法是相当的，从一阶公式角度看它们也是等值的：

$$\begin{aligned}
&\neg\forall x\forall y(T(x)\wedge W(y)\rightarrow K(x,y))\\
&\quad\Leftrightarrow\exists x(\neg\forall y(T(x)\wedge W(y)\rightarrow K(x,y))) && // \text{ 量词否定等值式}\\
&\quad\Leftrightarrow\exists x\exists y(\neg(T(x)\wedge W(y)\rightarrow K(x,y))) && // \text{ 量词否定等值式}\\
&\quad\Leftrightarrow\exists x\exists y(\neg(\neg(T(x)\wedge W(y))\vee K(x,y))) && // \text{ 蕴涵等值式}\\
&\quad\Leftrightarrow\exists x\exists y(T(x)\wedge W(y)\wedge\neg K(x,y))) && // \text{ 德摩根律}
\end{aligned}$$

4. 公式$\neg\exists x\exists y(T(x)\wedge W(y)\wedge K(x,y))$的直观含义是，并非存在$x$和存在$y$，$x$是兔子，$y$是乌龟，且$x$比$y$快，简单地说，就是“并非有的兔子比有的乌龟跑得快”。而公式$\forall x\forall y(T(x)\rightarrow W(y)\rightarrow\neg K(x,y))$的直观含义是，对任意的$x$，如果$x$是兔子，那么对任意的$y$，如果$y$是乌龟，则$x$不比$y$跑得快，简单地说，就是“对任意的兔子和任意的乌龟，兔子不一定比乌龟跑得快”，从自然语言的角度

看，后者比较拗口，通常不这么说，但从一阶公式看它们是等值的：

$$
\begin{aligned}
&\neg\exists x\exists y(T(x)\wedge W(y)\wedge K(x,y))\\
&\quad\Leftrightarrow \forall x(\neg\exists y(T(x)\wedge W(y)\wedge K(x,y))) && \text{// 量词否定等值式}\\
&\quad\Leftrightarrow \forall x\forall y(\neg(T(x)\wedge W(y)\wedge K(x,y))) && \text{// 量词否定等值式}\\
&\quad\Leftrightarrow \forall x\forall y(\neg T(x)\vee\neg W(y)\vee K(x,y))) && \text{// 德摩根律}\\
&\quad\Leftrightarrow \forall x\forall y(T(x)\rightarrow(W(y)\rightarrow\neg K(x,y))) && \text{// 蕴涵等值式}
\end{aligned}
$$

5. 公式$\neg\exists x(T(x)\wedge\forall y(W(y)\rightarrow K(x,y)))$的直观含义是，并非存在$x$，$x$是兔子，而且对任意的$y$，若$y$是乌龟，则$x$比$y$快，简单地说，就是“不存在比所有乌龟跑得快的兔子”。而公式$\forall x(T(x)\rightarrow\exists y(W(y)\wedge\neg K(x,y)))$的直观含义是，对所有的$x$，如果$x$是兔子，则存在乌龟$y$，$x$不比$y$快，简单地说，就是“所有的兔子都存在比它跑得快的乌龟”。从自然语言角度看，这两种说法相当，从一阶公式它们等值：

$$
\begin{aligned}
&\neg\exists x(T(x)\wedge\forall y(W(y)\rightarrow K(x,y)))\\
&\quad\Leftrightarrow \forall x(\neg(T(x)\wedge\forall y(W(y)\rightarrow K(x,y)))) && \text{// 量词否定等值式}\\
&\quad\Leftrightarrow \forall x(\neg T(x)\vee\neg\forall y(W(y)\rightarrow K(x,y))) && \text{// 德摩根律}\\
&\quad\Leftrightarrow \forall x(\neg T(x)\vee\exists y(\neg(W(y)\rightarrow K(x,y)))) && \text{// 量词否定等值式}\\
&\quad\Leftrightarrow \forall x(\neg T(x)\vee\exists y(\neg(\neg W(y)\vee K(x,y)))) && \text{// 蕴涵等值式}\\
&\quad\Leftrightarrow \forall x(\neg T(x)\vee\exists y(W(y)\wedge\neg K(x,y)))) && \text{// 德摩根律}\\
&\quad\Leftrightarrow \forall x(T(x)\rightarrow\exists y(W(y)\wedge\neg K(x,y)))) && \text{// 蕴涵等值式}
\end{aligned}
$$

6. 公式$\neg\forall x(T(x)\rightarrow\exists y(W(y)\wedge K(x,y)))$的直观含义是，并非对任意的$x$，如果$x$是兔子，则存在$y$，$y$是乌龟，而且$x$比$y$快，简单地说，就是“并非所有的兔子都比某个乌龟跑得快”。而公式$\exists x(T(x)\wedge\forall y(W(y)\rightarrow\neg K(x,y)))$的含义是，存在$x$，$x$是兔子，而且对任意的$y$，如果$y$是乌龟，则$x$不比$y$快，简单地说，就是“存在比所有乌龟都跑得慢的兔子”。从自然语言角度看，这两种说法是否相当需要斟酌，但从一阶公式角度看，它们是等值的：

$$
\begin{aligned}
&\neg\forall x(T(x)\rightarrow\exists y(W(y)\wedge K(x,y)))\\
&\quad\Leftrightarrow \exists x(\neg(T(x)\rightarrow\exists y(W(y)\wedge K(x,y)))) && \text{// 量词否定等值式}\\
&\quad\Leftrightarrow \exists x(\neg(\neg T(x)\vee\exists y(W(y)\wedge K(x,y)))) && \text{// 蕴涵等值式}\\
&\quad\Leftrightarrow \exists x(T(x)\wedge\neg\exists y(W(y)\wedge K(x,y))) && \text{// 德摩根律}\\
&\quad\Leftrightarrow \exists x(T(x)\wedge\forall y(\neg(W(y)\wedge K(x,y)))) && \text{// 量词否定等值式}\\
&\quad\Leftrightarrow \exists x(T(x)\wedge\forall y(\neg W(y)\vee\neg K(x,y))) && \text{// 德摩根律}\\
&\quad\Leftrightarrow \exists x(T(x)\wedge\forall y(W(y)\rightarrow\neg K(x,y))) && \text{// 蕴涵等值式}
\end{aligned}
$$

从这个例子，读者可进一步看到自然语言与一阶公式之间的差别：一阶逻辑中等值的公式，在自然语言中对应说法大致相当的公式，但自然语言有自然语言的习惯，有些说法可能拗口，有些说法可能难以理解而不常用，例如，我们看到，对于涉及量词的句子而言，否定在原子公式（即谓词作

用到项得到的公式）上并不比否定放在最前面更容易理解，这与命题逻辑公式（即不涉及量词句子的符号化）不同，这种公式否定放在原子上更容易理解。

另外，读者通过这个例子也应进一步加深自然语言命题在一阶逻辑中的符号化，特别地，应进一步理解，在全称量词的辖域中，特征谓词与其他谓词最自然的是蕴涵关系，而在存在量词的辖域中，特征谓词与其他谓词最自然的是合取关系。

下面的例子进一步说明全称量词对析取没有分配律，以及存在量词对合取没有分配律的原因。

例子 6.2.4 试证明：

(1) $\forall x(A(x) \vee B(x)) \not\Leftrightarrow \forall xA(x) \vee \forall xB(x)$

(2) $\exists x(A(x) \wedge B(x)) \not\Leftrightarrow \exists xA(x) \wedge \exists xB(x)$

证明 这里根据公式等值以及永真式的定义进行证明。

(1) 根据公式等值的定义，只要证明$\forall x(A(x) \vee B(x)) \not\leftrightarrow \forall xA(x) \vee \forall xB(x)$不是永真式即可，由于此公式是闭公式，只要找到一个解释使得公式$\forall x(A(x) \vee B(x))$的真值与公式$\forall xA(x) \vee \forall xB(x)$的真值不同即可。定义解释$\mathcal{I}$，其论域是自然数集$\mathbb{N}$，$A(x)$表示$x$是奇数，$B(x)$表示$x$是偶数，则公式$\forall x(A(x) \vee B(x)$在此解释下的含义是，对任意的自然数$x$，$x$要么是奇数，要么是偶数，因此其真值为真，而公式$\forall xA(x) \vee \forall xB(x)$的含义是，要么对任意的自然数$x$，$x$都是奇数，要么对任意的自然数$x$，$x$都是偶数，这显然为假，因此这两个公式不等值。

(2) 使用与(1)相同的解释，即论域是自然数集，$A(x)$表示x是奇数，$B(x)$表示x是偶数，则公式$\exists x(A(x) \wedge B(x))$的含义是存在自然数$x$，$x$既是奇数又是偶数，这显然为假。而公式$\exists xA(x) \wedge \exists xB(x)$的含义是存在自然数$x$，$x$是奇数，也存在自然数$x$，$x$是偶数，这显然为真，因此这两个公式不等值。

□

6.3 一阶公式的前束范式

范式就是在某种意义下的标准形式，命题逻辑公式的范式能够很容易地判断公式的类型，而主范式则与公式的真值表有着明显的对应关系。由于一阶逻辑的复杂性，一阶逻辑公式的范式也比较复杂，我们这一节介绍有关一阶逻辑公式范式的最基本知识，讨论一阶逻辑公式的前束范式，其目的主要是为了下一章的自然推理，在那里所讨论的量词引入和消除规则只能针对前束范式应用。

定义 6.3.1 设A是一个一阶公式，如果A具有如下形式：

$$Q_1x_1Q_2x_2\cdots Q_kx_kB$$

则称A是**前束范式**(prenex normal form)，其中$Q_i(1 \leq i \leq k)$是量词符号$\forall$或$\exists$，而B是**不含量词**的公式。

例子 6.3.2 例如，下面的一阶公式是前束范式：

$$\forall x\forall y(T(x) \wedge W(y) \rightarrow K(x,y))$$

$$\forall x\forall y\exists z(F(x) \wedge G(y) \wedge H(z) \rightarrow L(x,y,z,u))$$

而下面的公式都不是前束范式：

$$\forall x(T(x) \to \forall y(W(y) \to K(x,y)))$$
$$\exists x(T(x) \wedge \forall y(W(x) \to K(x,y)))$$

在一阶逻辑中，可使用等值演算的方法求出与一个一阶公式等值的前束范式。

定理 6.3.3 **前束范式存在定理**：一阶逻辑中的任何公式都存在与之等值的前束范式。

上述定理的严格证明比较复杂，此处不作详细介绍，下面给出求与一阶公式等值的前束范式的一个启发式步骤：

1. 使用**约束变量改名规则**或**自由变量替换规则**使得一阶公式满足：(i) 每个变量符号要么约束出现，要么自由出现；(ii) 每个量词的指导变元互不相同。应用这一步的目的是使得在后面可以方便地使用量词辖域扩张等值式。

2. 使用**量词否定等值式**将否定联结词放到量词的后面；

3. 使用**量词辖域扩张等值式**将量词放到所有命题逻辑联结词的前面。

例子 6.3.4 求与下面各公式等值的前束范式：

(1) $\forall xF(x) \wedge \neg\exists xG(x)$

(2) $\forall xF(x) \vee \neg\exists xG(x)$

解答：

(1).

$$\begin{aligned}
\forall xF(x) \wedge \neg\exists xG(x) &\Leftrightarrow \forall xF(x) \wedge \neg\exists yG(y) && // \text{约束变量改名}\\
&\Leftrightarrow \forall xF(x) \wedge \forall y(\neg G(y)) && // \text{量词否定等值式}\\
&\Leftrightarrow \forall x(F(x) \wedge \forall y(\neg G(y))) && // \text{量词辖域扩张}\\
&\Leftrightarrow \forall x\forall y(F(x) \wedge \neg G(y)) && // \text{量词辖域扩张}
\end{aligned}$$

(2).

$$\begin{aligned}
\forall xF(x) \vee \neg\exists xG(x) &\Leftrightarrow \forall xF(x) \vee \neg\exists yG(y) && // \text{约束变量改名}\\
&\Leftrightarrow \forall xF(x) \vee \forall y(\neg G(y)) && // \text{量词否定等值式}\\
&\Leftrightarrow \forall x(F(x) \vee \forall y(\neg G(y))) && // \text{量词辖域扩张}\\
&\Leftrightarrow \forall x\forall y(F(x) \vee \neg G(y)) && // \text{量词辖域扩张}
\end{aligned}$$

上述的例题充分展示了上面所说的求前束范式的步骤，但实际上，由于全称量词对合取分配，上面第一个公式还可使用如下更为简单的方法得到与它等值的前束范式：

$$\begin{aligned}
\forall xF(x) \wedge \neg\exists xG(x) &\Leftrightarrow \forall xF(x) \wedge \forall x(\neg G(x)) && // \text{量词否定等值式}\\
&\Leftrightarrow \forall x(F(x) \wedge \neg G(x)) && // \text{量词分配等值式}
\end{aligned}$$

这说明**与某个一阶公式等值的前束范式不惟一**。实际上，上面例子中求前束范式的最后两步，如果选择不同的量词辖域扩张方式，也能得到不同的前束范式。例如对于第二个公式，我们可以这样求

与它等值的前束范式：

$$\begin{aligned}
\forall x F(x) \vee \neg \exists x G(x) &\Leftrightarrow \forall x F(x) \vee \neg \exists y G(y) && \text{// 约束变量改名} \\
&\Leftrightarrow \forall x F(x) \vee \forall y(\neg G(y)) && \text{// 量词否定等值式} \\
&\Leftrightarrow \forall y(\forall x F(x) \vee (\neg G(y))) && \text{// 量词辖域扩张} \\
&\Leftrightarrow \forall y \forall x(F(x) \vee \neg G(y)) && \text{// 量词辖域扩张}
\end{aligned}$$

注意，由于全称量词对析取没有分配律，因此第二个公式没有像第一个公式一样的更简单步骤求其前束范式。

例子 6.3.5 求与下列各公式等值的前束范式：

(1) $\exists x F(x) \wedge \forall x G(x)$

(2) $\forall x F(x) \rightarrow \exists x G(x)$

(3) $\exists x F(x) \rightarrow \forall x G(x)$

(4) $\forall x F(x) \rightarrow \forall x G(x)$

(5) $\exists x F(x) \rightarrow \exists x G(x)$

解答：

(1).

$$\begin{aligned}
\exists x F(x) \wedge \forall x G(x) &\Leftrightarrow \exists x F(x) \wedge \forall y G(y) && \text{// 约束变量改名} \\
&\Leftrightarrow \exists x(F(x) \wedge \forall y G(y)) && \text{// 量词辖域扩张} \\
&\Leftrightarrow \exists x \forall y(F(x) \wedge G(y)) && \text{// 量词辖域扩张}
\end{aligned}$$

(2).

$$\begin{aligned}
\forall x F(x) \rightarrow \exists x G(x) &\Leftrightarrow \forall x F(x) \rightarrow \exists y G(y) && \text{// 约束变量改名} \\
&\Leftrightarrow \exists x(F(x) \rightarrow \exists y G(y)) && \text{// 量词辖域扩张} \\
&\Leftrightarrow \exists x \exists y(F(x) \rightarrow G(y)) && \text{// 量词辖域扩张}
\end{aligned}$$

(3).

$$\begin{aligned}
\exists x F(x) \rightarrow \forall x G(x) &\Leftrightarrow \exists x F(x) \rightarrow \forall y G(y) && \text{// 约束变量改名} \\
&\Leftrightarrow \forall x(F(x) \rightarrow \forall y G(y)) && \text{// 量词辖域扩张} \\
&\Leftrightarrow \forall x \forall y(F(x) \rightarrow G(y)) && \text{// 量词辖域扩张}
\end{aligned}$$

(4).

$$\begin{aligned}
\forall x F(x) \rightarrow \forall x G(x) &\Leftrightarrow \forall x F(x) \rightarrow \forall y G(y) && \text{// 约束变量改名} \\
&\Leftrightarrow \exists x(F(x) \rightarrow \forall y G(y)) && \text{// 量词辖域扩张} \\
&\Leftrightarrow \exists x \forall y(F(x) \rightarrow G(y)) && \text{// 量词辖域扩张}
\end{aligned}$$

(5).

$$\begin{aligned}\exists xF(x)\to\exists xG(x) &\Leftrightarrow \exists xF(x)\to\exists yG(y) && // \text{约束变量改名}\\ &\Leftrightarrow \forall x(F(x)\to\exists yG(y)) && // \text{量词辖域扩张}\\ &\Leftrightarrow \forall x\exists y(F(x)\to G(y)) && // \text{量词辖域扩张}\end{aligned}$$

对于上述例子有几点值得作进一步的说明：

1. 首先，第一公式在作量词辖域扩张时可选择不同的顺序，从而得到不同的前束范式：

$$\begin{aligned}\exists xF(x)\wedge\forall xG(x) &\Leftrightarrow \exists xF(x)\wedge\forall yG(y) && // \text{约束变量改名}\\ &\Leftrightarrow \forall y(\exists xF(x)\wedge G(y)) && // \text{量词辖域扩张}\\ &\Leftrightarrow \forall y\exists x(F(x)\wedge G(y)) && // \text{量词辖域扩张}\end{aligned}$$

由此我们可得到$\exists x(F(x)\wedge G(y))$与$\forall y\exists x(F(x)\wedge G(y))$，细心的读者也许记得我们在前面说过，量词的顺序不能随意交换，但这里我们看到，量词的顺序在某些特定的情况也可以交换（即交换前后的公式也可能等值）。读者可通过查找资料或自己思考，研究一下在什么情况下量词顺序能够交换，什么时候不能交换。

2. 其次，读者要注意与蕴涵联结词有关的量词辖域扩张等值式中，**当扩张蕴涵前件量词的辖域时，要注意改变量词符号**。读者应该特别注意下面的量词辖域扩张等值式：

$$\begin{aligned}\forall xA(x)\to B &\Leftrightarrow \exists x(A(x)\to B)\\ \exists xA(x)\to B &\Leftrightarrow \forall x(A(x)\to B)\end{aligned}$$

如果读者记得不是很清楚，就最好利用蕴涵等值式将蕴涵联结词化成析取来进行等值演算。

3. 最后，对于上面后四个公式，我们可将蕴涵化成析取联结词来求与其等值的前束范式。例如，对公式(3)：

$$\begin{aligned}\exists xF(x)\to\forall xG(x) &\Leftrightarrow \exists xF(x)\to\forall yG(y) && // \text{约束变量改名}\\ &\Leftrightarrow \neg\exists xF(x)\vee\forall yG(y)) && // \text{蕴涵等值式}\\ &\Leftrightarrow \forall x\neg F(x)\vee\forall yG(y)) && // \text{量词否定等值式}\\ &\Leftrightarrow \forall x\forall y(\neg F(x)\vee G(y)) && // \text{两次量词辖域扩张}\end{aligned}$$

特别地，由于存在量词对析取有分配律，对于公式(2)，我们可采用更为简单的步骤求与其等值的前束范式：

$$\begin{aligned}\forall xF(x)\to\exists xG(x) &\Leftrightarrow \neg\forall xF(x)\vee\exists xG(x) && // \text{蕴涵等值式}\\ &\Leftrightarrow \exists x\neg F(x)\vee\exists xG(x)) && // \text{量词否定等值式}\\ &\Leftrightarrow \exists x(\neg F(x)\vee G(x)) && // \text{量词分配等值式}\\ &\Leftrightarrow \exists x(F(x)\to G(x)) && // \text{蕴涵等值式}\end{aligned}$$

注意，这说明在一阶逻辑中有如下等值式：

$$\exists x\exists y(F(x)\to G(y))\ \Leftrightarrow\ \forall xF(x)\to\exists xG(x)\ \Leftrightarrow\ \exists x(F(x)\to G(x))$$

但**要注意上述等值式成立的条件，因为量词辖域扩张等值式的成立本身是有条件的**。请读者自行思考上述等值式成立的条件。

例子 6.3.6 求与下列各公式等值的前束范式：

(1) $\forall x F(x,y) \to \exists y G(x,y)$

(2) $(\forall x_1 F(x_1,x_2) \to \exists x_2 G(x_2)) \to \forall x_1 H(x_1,x_2,x_3)$

分析：这里给出的公式稍微有一点复杂，在求与其等值的前束范式之前，读者必须弄清除哪些个体变量是自由出现，哪些个体变量是约束出现，**使用约束变量改名规则和自由变量替换规则将公式变换成每个量词后的指导变元都不同，每个变量符号的出现要么是约束出现，要么是自由出现**，再使用量词否定等值式和量词辖域扩张等值式就比较方便了。

解答：

(1).

$$\begin{aligned}
\forall x F(x,y) \to \exists y G(x,y) &\Leftrightarrow \forall x F(x,y) \to \exists z G(x,z) && // \text{约束变量改名}\\
&\Leftrightarrow \forall x F(x,y) \to \exists z G(u,z) && // \text{自由变量替换}\\
&\Leftrightarrow \exists x (F(x,y) \to \exists z G(u,z)) && // \text{量词辖域扩张}\\
&\Leftrightarrow \exists x \exists z (F(x,y) \to G(u,z)) && // \text{量词辖域扩张}
\end{aligned}$$

(2).

$$\begin{aligned}
&(\forall x_1 F(x_1,x_2) \to \exists x_2 G(x_2)) \to \forall x_1 H(x_1,x_2,x_3)\\
&\quad\Leftrightarrow (\forall x_1 F(x_1,x_2) \to \exists x_4 G(x_4)) \to \forall x_5 H(x_5,x_2,x_3) && // \text{约束变量改名}\\
&\quad\Leftrightarrow (\exists x_1 (F(x_1,x_2) \to \exists x_4 G(x_4))) \to \forall x_5 H(x_5,x_2,x_3) && // \text{量词辖域扩张}\\
&\quad\Leftrightarrow \exists x_1 \exists x_4 (F(x_1,x_2) \to G(x_4)) \to \forall x_5 H(x_5,x_2,x_3) && // \text{量词辖域扩张}\\
&\quad\Leftrightarrow \forall x_1 \, (\exists x_4 (F(x_1,x_2) \to G(x_4)) \to \forall x_5 H(x_5,x_2,x_3)) && // \text{量词辖域扩张}\\
&\quad\Leftrightarrow \forall x_1 \forall x_4 \, ((F(x_1,x_2) \to G(x_4)) \to \forall x_5 H(x_5,x_2,x_3)) && // \text{量词辖域扩张}\\
&\quad\Leftrightarrow \forall x_1 \forall x_4 \forall x_5 \, ((F(x_1,x_2) \to G(x_4)) \to H(x_5,x_2,x_3)) && // \text{量词辖域扩张}
\end{aligned}$$

对于像上面公式(2)出现许多变量的公式，读者必须仔细区分清楚每个变量符号的出现是自由出现还是约束出现。例如上面公式(2)中的x_2在$\forall x_1 F(x_1,x_2)$和$\forall x_1 H(x_1,x_2,x_3)$中都是自由出现，而在$\exists x_2 G(x_2)$中是约束出现。注意，在进行约束变量改名时要将指导变元符号**在辖域内的全部出现**改成新的变量符号，在进行自由变量替换时要要将该变量符号**在整个公式的所有自由出现**替换成新的变量符号。

注意，首先使用约束变量改名和自由变量替换规则使得整个公式满足：(i) **每个量词的指导变元符号不同**；(ii) **每个个体变量符号的出现要么是约束出现，要么是自由出现**。这样就自然满足使用量词辖域扩张等值式的条件，从而不容易出错。为使得公式满足上述条件，我们建议：

(1) 首先使用约束变量改名规则使得每个量词的指导变元符号不同：从左至右扫描公式中出现的指导变元符号，如果某个指导变元符号已经在前面出现（**自由出现或约束出现**），则使用约束变量改名规则将该指导变元及其在辖域内的所有出现改名为新的变量符号；

(2) 经过上一步之后能保证每个量词的指导变元符号不同，但可能还有一些变量符号既是自由出现又是约束出现，因此再从左至右扫描公式，检查公式中自由出现的变量符号，如果它在前面有约束出现（或说已经作为指导变元），则使用自由变量替换规则将该变量符号**在整个公式中所有自由出现的地方**替换为新的变量符号。注意根据上一步，当前处理的**这个自由出现的变量符号只可能在此之前是既约束出现又自由出现，在此之后一定是自由出现**（请读者思考为什么？），因此此时只要将该当前的这个变量符号及此处之后所有的出现替换即可；

(3) 所谓选择新的变量符号都是指在整个公式中没有出现的变量符号。

例如，对如下公式：

$$(\forall xF(x,y) \to \exists yG(x,y)) \wedge (\forall x\forall yH(x,y,z) \to \forall zF(x,z))$$

使用约束变量改名和自由变量替换使得它满足指导变元互不相同，变量出现要么是自由出现要么是约束出现：

$$\begin{aligned}
&(\forall xF(x,y) \to \exists yG(x,y)) \wedge (\forall x\forall yH(x,y,z) \to \forall zF(x,z)) \\
&\quad \Leftrightarrow (\forall xF(x,y) \to uG(x,u)) \wedge (\forall v\forall w(v,w,z) \to \forall rF(x,r)) && \text{// 约束变量改名} \\
&\quad \Leftrightarrow (\forall xF(x,y) \to uG(s,u)) \wedge (\forall v\forall w(v,w,z) \to \forall rF(s,r)) && \text{// 自由变量替换}
\end{aligned}$$

第一步使用约束变量改名规则改变指导变元符号，首先保留$\forall xF(x,y)$中的x不变，然后看到$\exists yG(x,y)$中的y已经在前面出现，将其改名为u，得到$\exists uG(x,u)$。然后再看到$\forall x\forall yH(x,y,z)$，其中的$x,y$都已经在前面出现，将其分别改名为$u,w$，而得到$\forall v\forall wH(v,w,z)$，再看到$\forall zF(x,z)$，其中的$z$也已经在前面出现，将其改名为$r$得到$\forall rF(x,r)$。经过这一步之后，再看整个公式，还有$\exists uG(x,u)$是自由出现的，而在其前面$x$已经约束出现，因此再使用自由变量替换规则将$\exists uG(x,u)$中的$x$以及**在此之后所有出现**的$x$（这之后出现的$x$一定是自由出现的）替换为$s$，上述公式中除了$\exists uG(x,u)$中的$x$之外还有$\forall rF(x,r)$中的$x$，因此要将这两处的$x$都替换为$s$。

因此在变量比较多，特别是在许多变量都既有自由出现又有约束出现时，容易出错。但读者应该注意到，不管怎样使用约束变量改名规则和自由变量替换规则，甚至不管怎样做等值演算，等值变换前后的公式有如下的关系：

(1) 除非使用了量词分配规则，否则等值变换前后的公式中的量词个数不变；

(2) 更重要的，不管是否使用量词分配规则，除指导变元之外，**变量符号在每个位置出现的身份不变**，即原先在某个位置是约束出现的，变换后在这个位置出现的变量符号仍是约束出现，原先是自由出现，变换在这个位置出现的变量符号仍是自由出现的。这一点不应有改变，**如果改变了就表明变换前后的公式不等值**!

例如，上面例子中的公式(2)中各变量符号（除指导变元之外）的出现情况如下：

$$\begin{array}{cccccccccc}
\text{约束出现} & & \text{自由出现} & & \text{约束出现} & & \text{约束出现} & \text{自由出现} & \text{自由出现} \\
\downarrow & & \downarrow & & \downarrow & & \downarrow & \downarrow & \downarrow \\
(\forall x_1 F(x_1, & & x_2) \to \exists x_2 G(x_2)) & & \to & & \forall x_1 H(x_1, & x_2, & x_3)
\end{array}$$

在使用约束变量改名规则之后得到的公式各变量符号（除指导变元之外）的出现情况如下：

$$\begin{array}{c} \text{约束出现}\ \ \text{自由出现}\quad\ \ \text{约束出现}\qquad\ \ \text{约束出现}\ \ \text{自由出现}\ \ \text{自由出现} \\ \downarrow\qquad\ \ \downarrow\qquad\quad\ \ \downarrow\qquad\qquad\ \ \downarrow\qquad\quad\ \ \downarrow\qquad\quad\ \ \downarrow \\ (\forall x_1 F(x_1,\qquad x_2) \to \exists x_4 G(x_4)) \to \forall x_5 H(x_5,\qquad x_2,\qquad x_3) \end{array}$$

最后得到的与它等值的前束范式的各变量符号（除指导变元之外）的出现情况如下：

$$\begin{array}{c} \text{约束出现}\ \ \text{自由出现}\ \ \text{约束出现}\ \ \text{约束出现}\ \ \text{自由出现}\ \ \text{自由出现} \\ \downarrow\qquad\ \ \downarrow\qquad\ \ \downarrow\qquad\ \ \downarrow\qquad\ \ \downarrow\qquad\ \ \downarrow \\ \forall x_1 \forall x_4 \forall x_5\ ((F(x_1,\qquad x_2) \to G(x_4)) \to H(x_5,\qquad x_2,\qquad x_3)) \end{array}$$

读者通过上面的对比可以看出，**各个位置上出现的变量符号虽然由于约束变量改名以及自由变量替换规则而有所改变，但该位置上的变量符号出现的身份不变**。根据一阶公式的定义，个体变量符号除作为指导变元符号之外，只能出现在谓词作用的原子公式里，或函数作用的项里，因此这里所说的位置就是是指某个变量符号出现在哪个谓词或函数中，作为该谓词或函数作用的第几个变量。

小 结

一阶逻辑的前束范式，直观地说就是所有量词约束都放在前面的公式。后面将看到，在一阶逻辑的自然推理中，只能针对前束范式使用与量词有关的推理规则。通常来说，求与一个公式等值的前束范式大致可分为如下几步：

1. 使用**约束变量改名**和**自由变量替换**规则使得公式满足：各量词的指导变元不同，各变量符号要么是约束出现要么自由出现。这一步不仅方便后面使用量词辖域扩张等值式，而且由于前束范式中所有量词的辖域（本质上）都是整个公式，因而也是必需的步骤；

2. 使用**量词否定等值式**将否定移到量词的后面；

3. 使用**量词辖域扩张等值式**将量词移到整个公式的前面，这时应该注意量词辖域扩张等值式的使用条件。不过在第一步的基础上，这时公式总是应该满足量词辖域扩张等值式的使用条件的。

在求与一个公式等值的前束范式，或者更一般地在对公式使用等值演算进行变换时，读者应该注意：

1. 在进行变换之前应该分清楚公式中每个变量符号除作为指导变元之外的出现身份，即是自由出现还是约束出现；

2. 每次对公式进行等值变换的前后，各个位置上出现的变量符号的身份应该保持不变，否则表明变换有错！

3. 注意在将位于蕴涵前件的量词辖域扩张时，要改变量词符号，即全称量词改为存在量词，存在量词改为全称量词；

4. 在确认自己完全理解只有全称量词对合取分配、存在量词对析取分配的基础上，才使用量词分配等值式求出与公式等值的形式更为简单的前束范式。

作 业

作业 6.1 设论域（个体域）$\mathbf{D}=\{a,b,c\}$，请通过展开量词验证下面的等值式：

$$\neg\forall xA(x)\Leftrightarrow\exists x\neg A(x)$$
$$\neg\forall x\exists yA(x,y)\Leftrightarrow\exists x\forall y\neg A(x,y)$$

作业 6.2 在一阶逻辑中证明下面的等值式：

(1) $\exists x(A(x)\rightarrow B(x))\ \Leftrightarrow\ \forall xA(x)\rightarrow\exists xB(x)$

(2) $\forall x\forall y(P(x)\rightarrow Q(y))\ \Leftrightarrow\ \exists xP(x)\rightarrow\forall yQ(y)$

作业 6.3 求与下列各公式等值的前束范式：

(1) $\forall x(P(x)\rightarrow\exists yQ(x,y))$

(2) $\exists x(\neg\exists yP(x,y)\rightarrow(\exists zQ(z,y)\rightarrow R(x,y)))$

(3) $\forall x\forall y((\exists zP(x,y,z)\wedge\exists uQ(x,u,v))\rightarrow\exists vQ(y,v,u))$

第七章　一阶逻辑的自然推理

这一章讨论利用一阶逻辑进行的自然推理。一阶逻辑的推理是命题逻辑的推理的深化，能够对自然语言表达的推理进行更为精细的符号化，从而能描述更多的在数学和生活中遇到的推理问题。一阶逻辑的推理仍是从几个前提推出一个结论，并且也是从前提引入规则出发，根据一些推理规则逐步构造验证推理正确性的证明序列。

同样，由于命题逻辑的永真式的替换实例是一阶逻辑的永真式，以此为桥梁可在一阶逻辑的推理中使用命题逻辑的推理规则，因此首先我们在定义一阶逻辑推理的有效性之后，简要复习命题逻辑中一些常用的推理规则，然后重点讨论一阶逻辑中与量词有关的推理规则，最后举例说明一阶逻辑的自然推理及应用。

7.1　一阶逻辑推理的有效性

一阶逻辑推理也是从几个前提公式推出一个结论，其有效性的定义与命题逻辑相同：

定义 7.1.1 **一阶逻辑的推理**是从前提$A_1, A_2, \cdots, A_n$推出结论A的过程，记为$A_1, A_2, \cdots, A_n \vdash A$。说**推理**$A_1, A_2, \cdots, A_n \vdash A$**是有效的**，如果公式$(A_1 \wedge A_2 \wedge \cdots \wedge A_n) \rightarrow A$是**永真式**。通常，我们将推理形式结构记为：$\Gamma \vdash A$，这里$\Gamma = \{A_1, A_2, \cdots, A_n\}$是推理的前提集。

这是从一阶公式真值语义的角度定义推理的有效性，但与我们日常所理解的推理差别很大，而构造证明序列验证一个推理的有效性，不仅是我们日常推理的模拟与抽象，而且是一种形式化的方法，向推理的自动化和机械化迈进了一大步。所以，无论是在命题逻辑，还是一阶逻辑，本课程学习的重点都是根据形式化方法的思想，在自然推理系统中，构造证明序列验证推理的有效性。从形式化的角度，我们用如下方式定义一阶逻辑推理的有效性：

定义 7.1.2 说推理$\Gamma \vdash A$在自然推理系统$\mathcal{N}$中是**有效的**，如果存在如下的**证明序列**：

$$
\begin{aligned}
&(1) \quad \Gamma_1 \vdash A_1 \\
&(2) \quad \Gamma_2 \vdash A_2 \\
&\qquad \vdots \\
&(n-1) \quad \Gamma_{n-1} \vdash A_{n-1} \\
&(n) \quad \Gamma_n \vdash A_n
\end{aligned}
$$

其中$\Gamma_i \vdash A_i (1 \leq i \leq n)$都是推理的形式结构，$\Gamma_n = \Gamma, A_n = A$，且满足：对任意的$i, 1 \leq i \leq n$，都存

在$j_1, j_2, \cdots, j_m (1 \leq j_k < i, k = 1, \cdots, m)$，使得

$$\frac{\Gamma_{j_1} \vdash A_{j_1} \qquad \Gamma_{j_2} \vdash A_{j_2} \qquad \cdots \qquad \Gamma_{j_m} \vdash A_{j_m}}{\Gamma_i \vdash A_i}$$

是自然推理系统$\mathcal{N}$中某个推理规则的实例。

通俗地说，要形式地验证推理$\Gamma \vdash A$是有效的，就需要构造一个推理形式结构的序列，此序列称为该推理的证明序列，且**该序列中的每个推理形式结构都是通过排在它前面的一些推理形式结构根据自然推理系统的某个规则得到的**。所谓的“得到”，就是将该推理形式结构放在横线下面，而将排在它前面的一些推理形式结构放在横线上面，就形成了自然推理系统规则的实例。

自然推理系统的规则都具有如下形式：

$$\frac{\Gamma_1 \vdash A_1 \qquad \Gamma_2 \vdash A_2 \qquad \cdots \qquad \Gamma_n \vdash A_n}{\Gamma \vdash A}$$

实际上，这是一个规则模式，也即上面的Γ_i, Γ可代表任意的公式集，而$A_i (1 \leq i \leq n), A$则可代表任意的公式，或者说，**用具体的公式集和公式代入上面的Γ_i, Γ, A_i, A就得到该规则的一个实例**。需要注意的是，在代入时，**一个符号的所有出现必须用相同的公式集（或公式）代入**。

从真值语义的角度看，上述规则的直观含义是，如果$\Gamma_i \vdash A_i (1 \leq i \leq n)$都是有效的推理，则$\Gamma \vdash A$也是有效的推理。从真值语义的角度说，符合这个含义的规则才是有效的推理，不过对于自然推理系统而言，所有给出的规则都是预先确定为有效的规则，我们只是关心如何利用这些规则形式化地构造证明序列，而不关心规则本身的有效性。正如我们在第四章所看到的，规则本身的有效性是在研究形式系统与它的语义模型之间关系时所考虑的问题。

7.2 一阶逻辑自然推理系统的推理规则

这一节详细讨论一阶逻辑自然推理系统的推理规则。由于命题逻辑永真式的替换实例在一阶逻辑中仍然是永真式，因此在命题逻辑中有效的推理规则在一阶逻辑中仍然有效，从而一阶逻辑的自然推理系统包含命题逻辑自然推理系统的所有推理规则，这些规则是针对构造命题逻辑公式的每个逻辑联结词的消除和引入规则，再加上最基本的前提引入规则。由于在一阶逻辑中，还可通过全称量词和存在量词构造公式，因此一阶逻辑自然推理系统特有的规则是量词的引入和消除规则。

下面先用一小节给出一阶逻辑自然推理系统中与量词无关的推理规则，这也是对前面命题逻辑自然推理系统的规则的复习和总结，然后再用一小节讨论一阶逻辑自然推理系统的量词引入和消除规则。

7.2.1 与量词无关的推理规则

在命题逻辑自然推理系统中，针对每个命题逻辑联结词都有引入和消除规则（否定联结词除外），再加上最基本的前提引入规则，构成了推理系统最基本的规则，为了推理的方便，我们也使用证明序列验证了一些派生规则，这些派生规则通常都涉及到两个联结词（例如，否定和蕴含、否定和析取等）。与在程序中调用子程序类似，在证明序列中使用派生规则相当于引用该验证该派生规则的证明序列。

基本规则

最基本的规则是前提引入规则：

$$\text{前提引入}\quad \frac{}{\Gamma, A, \Delta \vdash A}$$

前提引入规则的直观含义是，对于推理形式结构$\Gamma \vdash A$，如果要推出的结论A已经在前提集Γ中，则它肯定是有效的推理。上面规则中，Γ, A, Δ相当于$\Gamma \cup \{A\} \cup \Delta$，即逗号表示这些前提公式一起来推出结论，我们用这种形式明确表明结论A在前提集中。注意到前提实际上是多个公式的集合，所以这里的逗号相当于集合并，虽然它在逻辑上表示的是这些前提公式的合取。

对于否定联结词，我们只需要否定消除规则：

$$\text{否定消除}\quad \frac{\Gamma, \neg A \vdash B \qquad \Gamma, \neg A \vdash \neg B}{\Gamma \vdash A}$$

该规则的直观含义是，如果前提集Γ加上A的否定能够推出矛盾（即既能推出B，又能推出B的否定），则Γ能推出A，这实际上是我们常用的“反证法”推理方法的抽象。使用这个规则的关键在于找到合适的公式B，即找到前提集Γ加上结论的否定之后到底能导出怎样的矛盾。

关于合取联结词的规则非常简单：

$$\text{合取引入}\quad \frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B}$$

$$\text{合取消除}\quad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A} \qquad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}$$

这几个规则的含义也很简单：如果前提集Γ能推出A，又能推出B，则它能推出$A \wedge B$，反之亦然。

析取的引入规则比较简单，但析取的消除规则则比较复杂：

$$\text{析取引入}\quad \frac{\Gamma \vdash A}{\Gamma \vdash A \vee B} \qquad \frac{\Gamma \vdash B}{\Gamma \vdash A \vee B}$$

$$\text{析取消除}\quad \frac{\Gamma \vdash A \vee B \qquad \Gamma, A \vdash C \qquad \Gamma, B \vdash C}{\Gamma \vdash C}$$

析取消除规则的直观含义是，如果Γ能推出$A \vee B$，且Γ加上A能推出C，加上B也能推出C，则Γ直接可推出C。使用析取消除规则需要一定的技巧，它通常是在证明一些常用的派生规则（如析取三段论），或证明一些与析取有关的基本等值式（如析取的交换律、分配律、德摩根律）时才用，大多数推理的验证是使用析取三段论规则。

蕴涵的引入和消除规则也比较简单：

$$\text{蕴含引入}\quad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B} \qquad \text{蕴含消除}\quad \frac{\Gamma \vdash A \to B \qquad \Gamma \vdash A}{\Gamma \vdash B}$$

蕴含引入规则给出了我们证明具有蕴含形式的结论的一种基本思路，即将蕴含的前件作为附加前提进行证明。蕴含消除规则则就是常见的假言推理。

等价的引入和消除规则则给出了等价联结词与蕴含联结词的基本关系：

$$\text{等价引入}\quad \frac{\Gamma \vdash A \to B \qquad \Gamma \vdash B \to A}{\Gamma \vdash A \leftrightarrow B}$$

$$\text{等价消除}\quad \frac{\Gamma \vdash A \leftrightarrow B}{\Gamma \vdash A \to B} \qquad \frac{\Gamma \vdash A \leftrightarrow B}{\Gamma \vdash B \to A}$$

上述规则提示我们在碰到等价联结词的时候，总是利用上述规则化成蕴含联结词来处理。

重要的派生规则

正如在命题逻辑自然推理系统一章所看到的，任何一个有效的推理都可看成一个派生规则，这里只是给出一些常用的和重要的派生规则。首先是双重否定律：

$$\text{双重否定律}\quad \frac{\Gamma \vdash A}{\Gamma \vdash \neg\neg A} \qquad \frac{\Gamma \vdash \neg\neg A}{\Gamma \vdash A}$$

这两个规则都可使用否定消除规则证明。与否定联结词相关的另一个重要派生规则是归谬律：

$$\text{归谬律}\quad \frac{\Gamma, A \vdash B \qquad \Gamma, A \vdash \neg B}{\Gamma \vdash \neg A}$$

归谬律与否定消除规则的基本含义相同，因为$\neg A$与A互为否定。

析取消除规则的使用需要一定的技巧，在大多数推理的验证中使用的是析取三段论规则：

$$\text{析取三段论}\quad \frac{\Gamma \vdash A \vee B \qquad \Gamma \vdash \neg A}{\Gamma \vdash B} \qquad \frac{\Gamma \vdash A \vee B \qquad \Gamma \vdash \neg B}{\Gamma \vdash A}$$

$$\frac{\Gamma \vdash \neg A \vee B \qquad \Gamma \vdash A}{\Gamma \vdash B} \qquad \frac{\Gamma \vdash A \vee \neg B \qquad \Gamma \vdash B}{\Gamma \vdash A}$$

析取三段论规则是我们常用的推理方法——排除法的抽象：如果Γ推出A或B，而又已知Γ推出A的否定，即不能推出A，那么Γ就只能推出B。上面列出了四种形式的析取三段论，其基本思想都一样，在应用中灵活选用可避免析取交换律以及双重否定律的多余使用。

假言推理的基本含义是，如果从Γ推出A蕴含B，又可从Γ推出A，则可从Γ推出B，即通常所说的：对于蕴含式，肯定前件，就可肯定后件。在日常推理中，对于蕴含式，我们可从否定后件得到前件的否定，这就是所谓的拒取式规则，或称为假言易位规则：

$$\text{假言易位}\quad \frac{\Gamma \vdash A \to B \qquad \Gamma \vdash \neg B}{\Gamma \vdash \neg A} \qquad \frac{\Gamma \vdash \neg A \to B \qquad \Gamma \vdash \neg B}{\Gamma \vdash A}$$

$$\frac{\Gamma \vdash A \to \neg B \qquad \Gamma \vdash B}{\Gamma \vdash \neg A} \qquad \frac{\Gamma \vdash \neg A \to \neg B \qquad \Gamma \vdash B}{\Gamma \vdash A}$$

同样这里列出了各种形式的假言易位规则，使得在验证推理中可避免双重否定律的多余使用。还有一条与蕴含有关的派生规则是传递规则：

$$\text{传递规则}\quad \frac{\Gamma \vdash A \to B \qquad \Gamma \vdash B \to C}{\Gamma \vdash A \to C}$$

最后一个派生规则是弱化规则：

$$\text{弱化规则}\quad \frac{\Gamma \vdash A}{\Gamma, B \vdash A}$$

弱化规则的基本思想是，如果Γ能推出A，那么它再加上任意的公式B还是能推出A。这说明经典逻辑推理的“单调性”，即增加新的前提不会推翻已经证明的结论。更一般地有，可将公式B换成任意的公式集Δ：

$$\text{弱化规则}\quad \frac{\Gamma \vdash A}{\Gamma, \Delta \vdash A}$$

弱化规则与联结词无关，它也不能利用基本规则通过构造证明序列证明，而需要根据自然推理系统推理有效性的定义，使用归纳法进行证明，具体的证明超出了本教程的范围，此处不讨论。

7.2.2 与量词有关的推理规则

对于一阶逻辑自然推理系统，我们重点讨论的是与量词有关的推理规则。下面也是先给出基本规则，然后再讨论一些重要的派生规则。

基本规则

正如与量词无关的基本规则是联结词的引入和消除规则，与量词有关的规则则分别是全称量词和存在量词的引入和消除规则。由于量词的使用于变量符号的使用紧密相关，因此一阶逻辑自然推理系统中与量词有关的推理规则会有一些附加条件，这些附加条件用来从形式上约束推理形式结构的公式中某些变量符号的出现。在使用这些规则时，必须注意相应的附加条件必须满足。

我们首先给出全称量词消除规则：

$$\forall\text{消除I}\quad \frac{\Gamma\vdash \forall xA}{\Gamma\vdash A[y/x]} \qquad\qquad \forall\text{消除II}\quad \frac{\Gamma\vdash \forall xA}{\Gamma\vdash A[a/x]}$$

我们使用两种形式的全称量词消除规则，在第一种形式的规则中，$A[y/x]$是用变量符号y替换A中所有自由出现的x，这里变量符号y必须满足：**A中每一处自由出现的x，都不在量词$\forall y$或$\exists y$的辖域中**。这个条件是为了保证用y替换A中自由出现的x之后，y在这个位置的出现仍然是自由出现。例如，若公式A是$\exists yA(x,y)$，则不能选择变量符号y去替换x，因为这时A中自由出现的x在量词$\exists y$的辖域中。违反这个条件我们就会得到不正确的推理。例如，$\forall x\exists yA(x,y)\vdash \exists yA(y,y)$就不是有效的推理，因为很容易找一个解释表明$\forall x\exists yA(x,y)\to \exists yA(y,y)$不是永真式。

练习 7.2.1 请找一个解释使得公式$\forall x\exists yA(x,y)\to \exists yA(y,y)$的真值为假。

在使用$\forall$消除I规则时，要满足上述条件很容易，我们可选取y是A中不出现的变量，也可选择x本身，即令$y=x$，这时$A[y/x]=A$，因为A中**自由出现**的x显然不在$\forall x$或$\exists x$的辖域中。注意，这里考虑的是公式A，而不是公式$\forall xA$。

上述第二形式的全称量词消除规则中，$A[a/x]$是使用个体常量符号a替换A中所有自由出现的x，这里对个体常量符号a没有任何约束条件。

下面是全称量词引入规则：

$$\forall\text{引入}\quad \frac{\Gamma\vdash A}{\Gamma\vdash \forall xA}$$

这个规则中，用于作为全称量词指导变元的x必须满足：**x不在Γ中的任何公式中自由出现**。这个条件使得我们不能对前提集中已经出现的自由变量作全称量词的约束。例如，$A(x)\vdash \forall xA(x)$不是有效的推理，因为很容易找一个解释表明$A(x)\to \forall xA(x)$不是永真式。

练习 7.2.2 请找一个解释（及对自由变量的指派）使得公式$A(x)\to \forall xA(x)$的真值为假。

备注 7.2.3 注意这里A与$A(x)$本质上是没有差别的，$A(x)$只是为了说明x是$A(x)$的自由变量而已，而A则没有强调这一点。上面在给出规则时都没有强调x是A的自由变量，这也意味着在这些规则中，变量符号x可以不是公式A的自由变量，不过若x不是公式A的自由变量时，$\forall xA$和$\exists xA$的真值与A完全相同，那么使用这些规则就变得没有意义。

下面是存在量词消除规则：

$$\exists\text{消除}\quad \frac{\Delta\vdash\exists xA \qquad \Gamma,A\vdash B}{\Delta,\Gamma\vdash B}$$

这个规则要求：**变量符号x不能在Γ中的任何公式以及B中自由出现**（不过，可以在Δ的公式以及A中自由出现）。下面在证明这些规则的有效性时会看到这个条件不可少。当取$\Delta=\{\exists xA\}$时，我们得到该规则最常用的一种形式：

$$\exists\text{消除}\quad \frac{\Gamma,A\vdash B}{\Gamma,\exists xA\vdash B}$$

这里同样要求x不在Γ的任何公式以及B中自由出现。违反这个条件同样可能会得到不正确的推理，例如$\exists xA(x)\vdash A(x)$就不是有效的推理。

练习 7.2.4 请找一个解释（及对自由变量的指派）使得公式$\exists xA(x)\to A(x)$的真值为假。

最后是存在量词引入规则：

$$\exists\text{引入I}\quad \frac{\Gamma\vdash A[y/x]}{\Gamma\vdash\exists xA} \qquad\qquad \exists\text{引入II}\quad \frac{\Gamma\vdash A[a/x]}{\Gamma\vdash\exists xA}$$

存在量词引入规则与全称量词引入规则相对应，也有两种形式的规则，第一种形式中$A[y/x]$是使用y替换A中所有自由出现的x得到的公式，这里同样要求：**A中每一处自由出现的x，都不在量词$\forall y$或$\exists y$的辖域中**。而第二种形式中的$A[a/x]$是使用个体常量符号a替换A中所有自由出现的x得到的公式。下面是错误使用$\exists$引入I规则的一个例子：

(1) $\forall yA(y,y)\vdash\forall yA(y,y)$ // 前提引入

(2) $\forall yA(y,y)\vdash\exists x\forall yA(x,y)$ // (1)$\exists$引入I

注意，对于上面的$\exists$引入I规则，这里取公式A为$\forall yA(x,y)$，从而$A[y/x]$是$\forall yA(y,y)$，不过这里变量符号x（也可以说是y）的选取是错误的，因为$\forall yA(x,y)$中自由出现的x在$\forall y$的辖域中，从而上述推理是错误的，也即$\forall yA(y,y)\vdash\exists x\forall yA(x,y)$不是有效的推理。

练习 7.2.5 请找一个解释使得公式$\forall yA(y,y)\to\exists x\forall yA(x,y)$的真值为假。

下表总结了自然推理系统中与量词有关的基本规则：

规则名称	规则模式	附加条件
$\forall$消除I	$\dfrac{\Gamma\vdash\forall xA}{\Gamma\vdash A[y/x]}$	A中每一处自由出现的x，都不在量词$\forall y$或$\exists y$的辖域中
$\forall$消除II	$\dfrac{\Gamma\vdash\forall xA}{\Gamma\vdash A[a/x]}$	
$\forall$引入	$\dfrac{\Gamma\vdash A}{\Gamma\vdash\forall xA}$	x不在Γ中的任何公式中自由出现
$\exists$消除	$\dfrac{\Delta\vdash\exists xA \qquad \Gamma,A\vdash B}{\Delta,\Gamma\vdash B}$	变量符号x不能在Γ中的任何公式以及B中自由出现
$\exists$引入I	$\dfrac{\Gamma\vdash A[y/x]}{\Gamma\vdash\exists xA}$	A中每一处自由出现的x，都不在量词$\forall y$或$\exists y$的辖域中
$\exists$引入II	$\dfrac{\Gamma\vdash A[a/x]}{\Gamma\vdash\exists xA}$	

我们可以从真值语义的角度证明这些规则的正确性。对于规则：

$$\frac{\Gamma \vdash A}{\Delta \vdash B}$$

为了证明它的正确性，设$\Gamma = \{A_1, A_2, \cdots, A_n\}, \Delta = \{B_1, B_2, \cdots, B_m\}$，则要证明当$A_1 \wedge A_2 \wedge \cdots \wedge A_n \to A$是永真式时，$B_1 \wedge B_2 \wedge \cdots \wedge B_m \to B$也是永真式。下面按照这个思路证明上述规则的正确性。

定理 7.2.6 上面给出的∀消除I规则、∀消除II规则、∀引入规则、∃消除规则、∃引入I规则、∃引入II规则是正确的。

证明 不妨设$\Gamma = \{A_1, A_2, \cdots, A_n\}$，而且为方便起见，我们将$A_1 \wedge A_2 \wedge \cdots \wedge A_n$也记为$\Gamma$。对任意的解释$I$及其指派$\sigma$，我们记公式$\Gamma = A_1 \wedge A_2 \wedge \cdots \wedge A_n$在该解释和指派下的真值为$\sigma(\Gamma)$。

(1) 对于∀消除I规则：

$$\frac{\Gamma \vdash \forall x A}{\Gamma \vdash A[y/x]}$$

若$\Gamma \to \forall x A$是永真式，则对任意的解释I及指派σ，我们需要证明$A_1 \wedge A_2 \wedge \cdots \wedge A_n \to A[y/x]$的真值为真。若$\sigma(\Gamma) = 0$，则显然该公式的真值为真，而若$\sigma(\Gamma) = 1$，则由$\Gamma \to \forall x A$是永真式，则$\sigma(\forall x A) = 1$。注意到，**当$A$中自由出现的$x$不在量词$\forall y$或$\exists y$的辖域时**，$\sigma(A[y/x]) = \sigma[\sigma(y)/x](A)$，而由$\sigma(\forall x A) = 1$，根据全称量词的语义，则有

$$\sigma(A[y/x]) = \sigma[\sigma(y)/x](A) = 1$$

因此当$\Gamma \to \forall x A$是永真式时，$\Gamma \to A[y/x]$也是永真式。同理，对于∀消除II规则，对任意解释I及指派σ，由

$$\sigma(A[a/x]) = \sigma[\sigma(a)/x](A) = \sigma[[\![a]\!]/x](A)$$

可得所要证明的结论。

(2) 对于∀引入规则：

$$\frac{\Gamma \vdash A}{\Gamma \vdash \forall x A}$$

若$\Gamma \to A$是永真式，则对任意的解释I及指派σ，若$\sigma(\Gamma) = 1$，则对解释I的论域中的任意元素d，由$\Gamma \to A$是永真式，则公式$A_1 \wedge A_2 \wedge \cdots \wedge A_n \to A$在指派$\sigma[d/x]$下的真值也为真，而且**由于$x$不在$\Gamma$的任意公式中自由出现**，因此

$$\sigma[d/x](\Gamma) = \sigma(\Gamma) = 1$$

从而$\sigma[d/x](A) = 1$，从而根据全称量词的语义由$\sigma(\forall x A) = 1$。

(3) 对于∃消除规则：

$$\frac{\Delta \vdash \exists x A \qquad \Gamma, A \vdash B}{\Delta, \Gamma \vdash B}$$

设$\Delta = \{B_1, B_2, \cdots, B_m\}$，同样记公式$B_1 \wedge \cdots \wedge B_m$为$\Delta$。若公式$\Delta \to \exists x A$和$\Gamma \wedge A \to B$是永真式，我们需要证明$\Delta \wedge \Gamma \to B$也是永真式。对任意解释$I$及指派$\sigma$，若$\sigma(\Gamma) = \sigma(\Delta) = 1$，我们需要证明$\sigma(B) = 1$。由于$\Delta \to \exists x A$是永真式，所以$\sigma(\exists x A) = 1$，也即存在解释$I$的论域中的元素$d$使

得$\sigma[d/x](A)=1$，由于公式$\Gamma\wedge A\to B$是永真式，所以它在指派$\sigma[d/x]$下的真值也为1，又注意到**x不在Γ的任意公式中自由出现**，所以

$$\sigma[d/x](\Gamma)=\sigma(\Gamma)=1$$

所以$\sigma[d/x](\Gamma\wedge A)=1$，所以$\sigma[d/x](B)=1$，而**$x$也不在$B$中自由出现**，因此

$$\sigma(B)=\sigma[d/x](B)=1$$

(4) 对于∃引入I规则：

$$\frac{\Gamma\vdash A[y/x]}{\Gamma\vdash\exists xA}$$

若$\Gamma\to A[y/x]$是永真式，则对任意解释I及指派σ，若$\sigma(\Gamma)=1$，则$\sigma(A[y/x])=1$，而**当A中自由出现的x不在量词$\forall y$或$\exists y$的辖域时**，

$$\sigma(A[y/x])=\sigma[\sigma(y)/x](A)$$

也即存在解释I的论域中的元素$d=\sigma(y)$，使得$\sigma[d/x](A)=1$，从而根据存在量词的语义有$\sigma(\exists xA)=1$。同理，对于∃引入II规则，由

$$\sigma(A[a/x])=\sigma[[\![a]\!]/x](A)$$

可证明所需要的结论。 □

备注 7.2.7 注意，只有**当A中自由出现的x不在量词$\forall y$或$\exists y$的辖域时**，才有下式成立：

$$\sigma(A[y/x])=\sigma[\sigma(y)/x](A)$$

若A中自由出现的x在量词$\forall y$或$\exists y$的辖域时，那么用y替换A中自由出现的x后得到的公式$A[y/x]$中y是约束出现的，从而$A[y/x]$的真值就与$\sigma(y)$无关，但显然当x在A自由出现时，$\sigma[\sigma(y)/x](A)$的值与$\sigma(y)$有关，因此**若A中自由出现的x在量词$\forall y$或$\exists y$的辖域时，上式就有可能不成立**。例如，若A是$\exists yA(x,y)$时，$A[y/x]$是$\exists yA(y,y)$，显然A在指派$\sigma[\sigma(y)/x]$下的真值与$\sigma(y)$的值有关，而$A[y/x]$在指派σ下的真值与$\sigma(y)$无关。

需要指出的是，对于全称量词消除I规则，最常用的一种形式是选取x本身作为规则中的y，这样得到的全称量词消除I规则的形式如下：

$$\forall\text{消除I特例}\qquad\frac{\Gamma\vdash\forall xA(x)}{\Gamma\vdash A(x)}$$

而对于存在量词消除规则，最常用的一种形式是取其中的$\Delta=\{\exists xA(x)\}$，从而得到如下的存在量词消除规则：

$$\exists\text{消除特例}\qquad\frac{\Gamma,A(x)\vdash B}{\Gamma,\exists xA(x)\vdash B}$$

最后对于存在量词引入I规则，同样，最常用的一种形式是选取x本身作为规则中的y，这样得到的存在量词引入I规则的形式如下：

$$\exists\text{引入I特例}\qquad\frac{\Gamma\vdash A(x)}{\Gamma\vdash\exists xA(x)}$$

下表总结了这些与量词有关的特殊形式的规则：

规则名称	规则模式	附加条件
∀消除I特例	$\dfrac{\Gamma \vdash \forall x A(x)}{\Gamma \vdash A(x)}$	
∃消除特例	$\dfrac{\Gamma, A(x) \vdash B}{\Gamma, \exists x A(x) \vdash B}$	变量符号x不能在Γ中的任何公式以及B中自由出现
∃引入I特例	$\dfrac{\Gamma \vdash A(x)}{\Gamma \vdash \exists x A(x)}$	

派生规则

下面利用基本推理规则证明一些一阶逻辑等值式，并相应地导出一些派生规则。根据第三章有关命题逻辑自然推理系统的派生规则的讨论，我们知道，只要构造证明序列证明了$A \vdash B$是有效的推理，就可以得到一个派生规则：对任意的前提集Γ有：

$$\frac{\Gamma \vdash A}{\Gamma \vdash B}$$

所以下面为简单起见，我们都是通过验证推理的形式导出派生规则。

下面一些推理的验证有一定的难度，所以**这一小节可作为选学的内容，只需要在构造证明序列时能够应用下面的约束变量改名规则以及量词否定等值式所对应的规则即可**。基础好的读者则可通过学习验证这些推理的证明序列领悟一阶逻辑自然推理系统中与量词有关的规则的使用。

首先考虑与约束变量改名规则对应的派生规则：

引理 7.2.8 设x在A的自由出现都不出现在量词$\forall y$或$\exists y$的辖域中，而且y不在A中自由出现，则：

(1) $\forall x A \vdash \forall y A[y/x]$且$\forall y A[y/x] \vdash \forall x A$，以及

(2) $\exists x A \vdash \exists y A[y/x]$且$\exists y A[y/x] \vdash \exists x A$

都是有效的推理。注意，这里$A[y/x]$是用变量符号y替换A中所有自由出现的x后得到的公式。

证明 (1) 验证推理$\forall x A \vdash \forall y A[y/x]$的证明序列如下：

(1)	$\forall x A \vdash \forall x A$	// 前提引入
(2)	$\forall x A \vdash A[y/x]$	// (2)∀消除I
(3)	$\forall x A \vdash \forall y A[y/x]$	// (3)∀引入

这里第(2)步的全称量词消除需要条件“x在A的自由出现都不出现在量词$\forall y$或$\exists y$的辖域中”，而第(3)步的全称量词引入需要条件“y不在A中自由出现”。

验证推理$\forall y A[y/x] \vdash \forall x A$的证明序列如下：

(1)	$\forall y A[y/x] \vdash \forall y A[y/x]$	// 前提引入
(2)	$\forall y A[y/x] \vdash (A[y/x])[x/y]$	// (2)∀消除I
(3)	$\forall y A[y/x] \vdash \forall x A$	// (3)∀引入

这里第(2)步使用全称量词消除时，y在$A[y/x]$中的自由出现就是原来x的自由出现，所以y在$A[y/x]$中的自由出现不会出现在$\forall x$或$\exists x$中，所以满足使用全称量词消除的条件。对于$A[y/x]$，再使用x替换其中自由出现的y得到的公式$A[y/x][x/y]$显然就是A，而且由于x不在$\forall y A[y/x]$中自由出现，所以第(3)步可使用全称量词引入规则。 □

(2) 验证推理$\exists xA \vdash \exists yA[y/x]$的证明序列如下：注意$A = A[y/x][x/y]$，

(1)	$A \vdash A[y/x][x/y]$	// 前提引入
(2)	$A \vdash \exists yA[y/x]$	// (2)∃引入I
(3)	$\exists xA \vdash \exists yA[y/x]$	// (3)∃消除特例

这里第(2)步中，y在$A[y/x]$中的自由出现不会出现在$\forall x$或$\exists$中，所以可使用存在量词引入规则，而第(3)步x不会在$\exists yA[y/x]$中自由出现，因此也可使用存在量词消除规则。

验证推理$\exists yA[y/x] \vdash \exists xA$的证明序列如下：

(1)	$\exists yA[y/x] \vdash \exists x(A[y/x])[x/y]$	// 上一个推理
(2)	$\exists yA[y/x] \vdash \exists xA$	// $A[y/x])[x/y] = A$

由上面的引理，可得到两个派生规则：设x在A的自由出现都不出现在量词$\forall y$或$\exists y$的辖域中，而且y不在A中自由出现，

$$\text{约束变量改名}\ \frac{\Gamma \vdash \forall xA}{\Gamma \vdash \forall yA[y/x]} \qquad \text{约束变量改名}\ \frac{\Gamma \vdash \exists xA}{\Gamma \vdash \exists yA[y/x]}$$

注意，为了满足使用派生规则的条件，我们**可选择y为不在A中出现的新的变量符号**。

备注 7.2.9 上面推理的验证比较简单，但这里不仅给出了每个推理的证明序列，而且详细地说明了其中使用量词规则时所需要满足的条件，希望读者从上述证明序列中进一步熟悉这些条件。

备注 7.2.10 对于推理，如果需要替换自由变量，则需要某个变量符号**在前提集以及结论公式**中的所有自由出现都替换成另外一个变量符号，而**不能仅仅替换结论公式中的自由出现的变量**。例如，很显然，$A(x) \vdash A(y)$不是有效的推理。而且，通常来说，有约束变量改名规则就可满足实际应用中的需求。

其次考虑量词否定等值式，为此先给出两个由假言易位规则变形而得到的派生规则：

引理 7.2.11 有如下的假言易位规则：

$$\text{假言易位}\ \frac{\Gamma,\ \neg A \vdash B}{\Gamma, \neg B \vdash A} \qquad \text{假言易位}\ \frac{\Gamma,\ A \vdash \neg B}{\Gamma,\ B \vdash \neg A}$$

$$\text{假言易位}\ \frac{\Gamma,\ \neg A \vdash \neg B}{\Gamma, B \vdash A} \qquad \text{假言易位}\ \frac{\Gamma,\ A \vdash B}{\Gamma,\ \neg B \vdash \neg A}$$

证明 对于第一个规则，可使用如下的证明序列：

(1)	$\Gamma, \neg A \vdash B$	// 派生规则的前提
(2)	$\Gamma \vdash \neg A \to B$	// (1)蕴含引入
(3)	$\Gamma, \neg B \vdash \neg A \to B$	// (2)弱化规则
(4)	$\Gamma, \neg B \vdash \neg B$	// 前提引入
(5)	$\Gamma, \neg B \vdash A$	// (3)(4)原来的假言易位规则

对于其他规则，不难用类似的证明序列证明。 □

现在证明量词否定等值式并给出相应的派生规则：

引理 7.2.12 下面的推理是有效的推理：

1. $\neg\forall xA \vdash \exists x\neg A$
2. $\exists x\neg A \vdash \neg\forall xA$
3. $\neg\exists xA \vdash \forall x\neg A$
4. $\forall x\neg A \vdash \neg\exists xA$

证明 1. 可使用如下证明序列验证：

(1) $\neg A \vdash \neg A$ // 前提引入
(2) $\neg A \vdash \exists x\neg A$ // (1)∃引入I特例
(3) $\neg\exists x\neg A \vdash A$ // (2)假言易位
(4) $\neg\exists x\neg A \vdash \forall xA$ // (3)∀引入
(5) $\neg\forall xA \vdash \exists x\neg A$ // (4)假言易位

注意，这里第(4)步的全称量词引入是可以的，因为在$\neg\exists x\neg A$中没有自由出现的x。

2. 可使用如下证明序列验证：

(1) $\neg A, \forall xA \vdash \forall xA$ // 前提引入
(2) $\neg A, \forall xA \vdash A$ // (1)∀消除I特例
(3) $\neg A, \forall xA \vdash \neg A$ // 前提引入
(4) $\neg A \vdash \neg\forall xA$ // (2),(3)归谬律
(5) $\exists x\neg A \vdash \neg\forall xA$ // (4)∃消除特例

3. 可使用如下证明序列验证：

(1) $\neg\exists xA,\ A \vdash A$ // 前提引入
(2) $\neg\exists xA,\ A \vdash \exists xA$ // (1)∃引入I特例
(3) $\neg\exists xA,\ A \vdash \neg\exists xA$ // 前提引入
(4) $\neg\exists xA \vdash \neg A$ // (2),(3)归谬律
(5) $\neg\exists xA \vdash \forall x\neg A$ // (4)∀引入

4. 可使用如下证明序列验证：

(1) $\forall x\neg A \vdash \forall x\neg A$ // 前提引入
(2) $\forall x\neg A \vdash \neg A$ // (1)∀消除I特例
(3) $A \vdash \neg\forall x\neg A$ // (2)假言易位
(4) $\exists xA \vdash \neg\forall x\neg A$ // (3)∃消除特例
(5) $\forall x\neg A \vdash \neg\exists xA$ // (4)假言易位

□

备注 7.2.13 上面引理的证明中，验证推理2和3的基本思路是利用归谬律:

为了验证推理$\exists x\neg A\vdash\neg\forall xA$，将$\forall xA$作为附加前提来推出矛盾，即希望由$\exists x\neg A$和$\forall xA$导出矛盾，显然$\forall xA$可推出$A$，但是不能直接由$\exists x\neg A$得到$\neg A$，而根据存在量词消除规则，应该从前提$\neg A$开始，也即我们需要从$\neg A$和$\forall xA$导出矛盾，从而得到了上面的证明序列。

同样地，为了验证$\neg\exists xA\vdash\forall x\neg A$，根据全称量词引入规则，我们验证$\neg\exists xA\vdash\neg A$，这时再将$A$引入作为附加前提，从$\neg\exists xA$和$A$导出矛盾，注意到从$A$可直接得到$\exists A$，于是得到上面的证明序列。

上面引理的证明中，验证推理1和4则需要两次使用假言易位规则，以间接地构造证明序列，这需要一定的技巧和经验。作业中给出了验证这两个推理的另外一些尝试，请读者判断其正确性。

根据上述引理，就得到如下否定全称量词的派生规则:

$$\text{否定}\forall\ \frac{\Gamma\vdash\neg\forall xA}{\Gamma\vdash\exists x\neg A}\qquad \text{否定}\forall\ \frac{\Gamma\vdash\exists x\neg A}{\Gamma\vdash\neg\forall xA}\qquad \text{否定}\forall\ \frac{\Gamma\vdash\neg\forall x\neg A}{\Gamma\vdash\exists xA}\qquad \text{否定}\forall\ \frac{\Gamma\vdash\exists xA}{\Gamma\vdash\neg\forall x\neg A}$$

以及否定存在量词的派生规则:

$$\text{否定}\exists\ \frac{\Gamma\vdash\neg\exists xA}{\Gamma\vdash\forall x\neg A}\qquad \text{否定}\exists\ \frac{\Gamma\vdash\forall x\neg A}{\Gamma\vdash\neg\exists xA}\qquad \text{否定}\exists\ \frac{\Gamma\vdash\neg\exists x\neg A}{\Gamma\vdash\forall xA}\qquad \text{否定}\exists\ \frac{\Gamma\vdash\forall xA}{\Gamma\vdash\neg\exists x\neg A}$$

下面考虑量词辖域收缩和扩张等值式。

引理 7.2.14 假定x不在B中自由出现，则下面的推理都是有效的推理:

1. $\exists x(A(x)\vee B)\vdash\exists xA(x)\vee B$
2. $\exists xA(x)\vee B\vdash\exists x(A(x)\vee B)$
3. $\forall x(A(x)\vee B)\vdash\forall xA(x)\vee B$
4. $\forall xA(x)\vee B\vdash\forall x(A(x)\vee B)$

证明 1. 可使用如下证明序列验证:

(1) $A(x)\vee B,A(x)\vdash A(x)$ // 前提引入
(2) $A(x)\vee B,A(x)\vdash\exists xA(x)$ // (1)∃引入I特例
(3) $A(x)\vee B,A(x)\vdash\exists xA(x)\vee B$ // (2)析取引入
(4) $A(x)\vee B,B\vdash B$ // 前提引入
(5) $A(x)\vee B,B\vdash\exists xA(x)\vee B$ // (4)析取引入
(6) $A(x)\vee B,B\vdash A(x)\vee B$ // 前提引入
(7) $A(x)\vee B\vdash\exists xA(x)\vee B$ // (3),(5),(6)析取消除
(8) $\exists x(A(x)\vee B)\vdash\exists xA(x)\vee B$ // (7)∃消除特例

在这个证明序列中，第(8)步使用存在量词消除时，需要x在B中无自由出现。

2. 可使用类似(1)的证明序列进行验证:

(1) $\exists xA(x)\vee B,A(x)\vdash A(x)$ // 前提引入
(2) $\exists xA(x)\vee B,A(x)\vdash A(x)\vee B$ // (1)析取引入
(3) $\exists xA(x)\vee B,A(x)\vdash\exists x(A(x)\vee B)$ // (2)∃引入I特例
(4) $\exists xA(x)\vee B,\exists xA(x)\vdash\exists x(A(x)\vee B)$ // (3)∃消除特例

(5) $\exists x A(x) \vee B, B \vdash B$ // 前提引入

(6) $\exists x A(x) \vee B, B \vdash A(x) \vee B$ // (5)析取引入

(7) $\exists x A(x) \vee B, B \vdash \exists x(A(x) \vee B)$ // (6)∃引入I特例

(8) $\exists x A(x) \vee B \vdash \exists x A(x) \vee B$ // 前提引入

(9) $\exists x A(x) \vee B \vdash \exists x(A(x) \vee B)$ // (4),(7),(8)析取消除

在这个证明序列中，第(4)步使用存在量词消除时，需要x在B中无自由出现。

3. 可使用如下证明序列验证：

(1) $\neg(\forall x A(x) \vee B),\ \forall x A(x) \vdash \forall x A(x)$ // 前提引入

(2) $\neg(\forall x A(x) \vee B),\ \forall x A(x) \vdash \forall x A(x) \vee B$ // (1)析取引入

(3) $\neg(\forall x A(x) \vee B),\ \forall x A(x) \vdash \neg(\forall x A(x) \vee B)$ // 前提引入

(4) $\neg(\forall x A(x) \vee B) \vdash \neg\forall x A(x)$ // (2),(3)归谬律

(5) $\neg(\forall x A(x) \vee B) \vdash \exists x \neg A(x)$ // (4)否定∀

(6) $\forall x(A(x) \vee B),\ \neg A(x) \vdash \forall x(A(x) \vee B)$ // 前提引入

(7) $\forall x(A(x) \vee B),\ \neg A(x) \vdash A(x) \vee B$ // (6)∀消除I特例

(8) $\forall x(A(x) \vee B),\ \neg A(x) \vdash \neg A(x)$ // 前提引入

(9) $\forall x(A(x) \vee B),\ \neg A(x) \vdash B$ // (7),(8)析取三段论

(10) $\forall x(A(x) \vee B),\ \neg(\forall x A(x) \vee B) \vdash B$ // (5),(9)∃消除

(11) $\forall x(A(x) \vee B),\ \neg(\forall x A(x) \vee B),\ B \vdash B$ // 前提引入

(12) $\forall x(A(x) \vee B),\ \neg(\forall x A(x) \vee B),\ B \vdash \forall x A(x) \vee B$ // (11)析取引入

(13) $\forall x(A(x) \vee B),\ \neg(\forall x A(x) \vee B),\ B \vdash \neg(\forall x A(x) \vee B)$ // 前提引入

(14) $\forall x(A(x) \vee B),\ \neg(\forall x A(x) \vee B) \vdash \neg B$ // (12),(13)归谬律

(15) $\forall x(A(x) \vee B) \vdash \forall x A(x) \vee B$ // (10),(14)否定消除

在这个证明序列的第(10)步中使用存在量词消除规则时要求x不在B中自由出现。

4. 可使用如下证明序列验证：

(1) $\forall x A(x) \vee B, \forall x A(x) \vdash \forall x A(x)$ // 前提引入

(2) $\forall x A(x) \vee B, \forall x A(x) \vdash A(x)$ // (1)∀消除I特例

(3) $\forall x A(x) \vee B, \forall x A(x) \vdash A(x) \vee B$ // (2)析取引入

(4) $\forall x A(x) \vee B, \forall x A(x) \vdash \forall x(A(x) \vee B)$ // (3)∀引入

(5) $\forall x A(x) \vee B, B \vdash B$ // 前提引入

(6) $\forall x A(x) \vee B, B \vdash A(x) \vee B$ // (5)析取引入

(7) $\forall x A(x) \vee B, B \vdash \forall x(A(x) \vee B)$ // (6)∀引入

(8) $\forall x A(x) \vee B \vdash \forall x A(x) \vee B$ // 前提引入

(9) $\forall x A(x) \vee B \vdash \forall x(A(x) \vee B)$ // (4),(7),(8)析取消除

在这个序列的第(4)步以及第(7)步实用全称量词引入规则时要求x不在B中自由出现。 □

备注 7.2.15 在这个引理的证明中，推理1, 2和4的验证都是利用析取消除规则，而推理$\forall x(A(x)\vee B)\vdash \forall xA(x)\vee B$的验证是最难的。实际上，上述证明序列的思路是利用否定消除规则，通过验证如下两个推理来验证该推理:

$$\forall x(A(x)\vee B),\ \neg(\forall xA(x)\vee B)\vdash B \qquad \forall x(A(x)\vee B),\ \neg(\forall xA(x)\vee B)\vdash \neg B$$

由于$\neg(\forall xA(x)\vee B)$与$\exists x\neg A(x)\wedge\neg B$等值，因此要验证后一个推理比较容易（这也从上面的证明序列可以看到），而对于前一个推理，则希望从$\forall x(A(x)\vee B)$以及$\exists x\neg A(x)$得到B，即希望验证下面的推理:

$$\forall x(A(x)\vee B),\ \exists x\neg A(x)\vdash B$$

而这根据存在量词消除规则，当x不在B中自由出现时，则只需验证如下的推理:

$$\forall x(A(x)\vee B),\ \neg A(x)\vdash B$$

总的来说，验证上述推理的基本思路来自命题逻辑自然推理系统中验证推理$A\to B\vdash \neg A\vee B$的证明序列。

引理 7.2.16 假定x不在B中自由出现，则下面的推理都是有效的推理:

1. $\exists x(A(x)\wedge B)\vdash \exists xA(x)\wedge B$
2. $\exists xA(x)\wedge B\vdash \exists x(A(x)\wedge B)$
3. $\forall x(A(x)\wedge B)\vdash \forall xA(x)\wedge B$
4. $\forall xA(x)\wedge B\vdash \forall x(A(x)\wedge B)$

证明 相对于析取，有关合取的辖域扩张推理式的证明容易很多，留作练习。 □

引理 7.2.17 假定x不在B中自由出现，则下面的推理都是有效的推理:

1. $\exists x(A(x)\to B)\vdash \forall xA(x)\to B$
2. $\exists xA(x)\to B\vdash \forall x(A(x)\to B)$
3. $\forall x(A(x)\to B)\vdash \exists xA(x)\to B$
4. $\forall xA(x)\to B\vdash \exists x(A(x)\to B)$

证明 1, 3的证明比较简单，留作练习，下面验证2和4。

2. 可使用如下的证明序列验证:

(1)	$\exists xA(x)\to B, A(x)\vdash A(x)$	// 前提引入
(2)	$\exists xA(x)\to B, A(x)\vdash \exists xA(x)$	// (1)∃引入I特例
(3)	$\exists xA(x)\to B, A(x)\vdash \exists xA(x)\to B$	// 前提引入
(4)	$\exists xA(x)\to B, A(x)\vdash B$	// (2),(3)蕴涵消除
(5)	$\exists xA(x)\to B\vdash A(x)\to B$	// (4)蕴涵引入
(6)	$\exists xA(x)\to B\vdash \forall x(A(x)\to B)$	// (5)∀引入

在这个证明序列中，第(6)步使用全称量词引入规则时要求x不在B中自由出现。

4. 可使用如下的证明序列验证：

(1) $\forall xA(x)\to B,\forall x\neg(A(x)\to B),\neg A(x),A(x),\neg B\vdash A(x)$ // 前提引入

(2) $\forall xA(x)\to B,\forall x\neg(A(x)\to B),\neg A(x),A(x),\neg B\vdash \neg A(x)$ // 前提引入

(3) $\forall xA(x)\to B,\forall x\neg(A(x)\to B),\neg A(x),A(x)\vdash B$ // (1),(2)否定消除

(4) $\forall xA(x)\to B,\forall x\neg(A(x)\to B),\neg A(x)\vdash A(x)\to B$ // (3)蕴含引入

(5) $\forall xA(x)\to B,\forall x\neg(A(x)\to B),\neg(A(x)\to B)\vdash A(x)$ // (4)假言易位

(6) $\forall xA(x)\to B,\forall x\neg(A(x)\to B)\vdash \neg(A(x)\to B)\to A(x)$ // (5)蕴涵引入

(7) $\forall xA(x)\to B,\forall x\neg(A(x)\to B)\vdash \forall x\neg(A(x)\to B)$ // 前提引入

(8) $\forall xA(x)\to B,\forall x\neg(A(x)\to B)\vdash \neg(A(x)\to B)$ // (7)∀消除I特例

(9) $\forall xA(x)\to B,\forall x\neg(A(x)\to B)\vdash A(x)$ // (6),(8)蕴含消除

(10) $\forall xA(x)\to B,\forall x\neg(A(x)\to B)\vdash \forall xA(x)$ // (9)∀引入

(11) $\forall xA(x)\to B,\forall x\neg(A(x)\to B)\vdash \forall xA(x)\to B$ // 前提引入

(12) $\forall xA(x)\to B,\forall x\neg(A(x)\to B)\vdash B$ // (10),(11)蕴涵消除

(13) $\forall xA(x)\to B,\forall x\neg(A(x)\to B),B,A(x)\vdash B$ // 前提引入

(14) $\forall xA(x)\to B,\forall x\neg(A(x)\to B),B\vdash A(x)\to B$ // (13)蕴含引入

(15) $\forall xA(x)\to B,\forall x\neg(A(x)\to B),\neg(A(x)\to B)\vdash \neg B$ // (14)假言易位

(16) $\forall xA(x)\to B,\forall x\neg(A(x)\to B)\vdash (\neg(A(x)\to B))\to\neg B$ // (15)蕴含引入

(17) $\forall xA(x)\to B,\forall x\neg(A(x)\to B)\vdash \neg B$ // (8), (16)蕴含消除

(18) $\forall xA(x)\to B\vdash \neg\forall x\neg(A(x)\to B)$ // (12), (17)归谬律

(19) $\forall xA(x)\to B\vdash \exists x(A(x)\to B)$ // (18)否定∀

这个证明序列的第(10)步的蕴含引入要求x不在B中自由出现。 □

备注 7.2.18 实际上，构造$\forall xA(x)\to B\vdash\exists x(A(x)\to B)$的证明序列的基本思路是利用否定消除规则，将$\neg\exists x(A(x)\to B)$作为附加前提推出矛盾。由于$\neg\exists x(A(x)\to B)$与$\forall x\neg(A(x)\to B)$等值，而$\neg(A(x)\to B)$又与$(A(x)\wedge\neg B)$，所以它应该可以得到$\forall xA(x)$及$\neg B$。因此实际上我们是希望验证如下两个推理：

$$\forall xA(x)\to B,\forall x\neg(A(x)\to B)\vdash B \qquad \forall xA(x)\to B,\forall x\neg(A(x)\to B)\vdash\neg B$$

根据上述思路，读者不难发现，上述证明序列中，第(1)至(6)步是验证$\neg(A\to B)\vdash A$的思路，而相应地，第(13)至(16)步是验证$\neg(A\to B)\vdash\neg B$的思路。

引理 7.2.19 假定x不在B中自由出现，则下面的推理都是有效的推理：

1. $\exists x(B\to A(x))\vdash B\to\exists xA(x)$
2. $B\to\exists xA(x)\vdash\exists x(B\to A(x))$
3. $\forall x(B\to A(x))\vdash B\to\forall xA(x)$

4. $B \to \forall x A(x) \vdash \forall x(B \to A(x))$

证明 1, 3, 4的证明都比较简单，留作练习。下面的证明序列可用于验证2：

(1) $B \to \exists x A(x),\ \neg B, B, \neg A(x) \vdash B$ // 前提引入

(2) $B \to \exists x A(x),\ \neg B, B, \neg A(x) \vdash \neg B$ // 前提引入

(3) $B \to \exists x A(x),\ \neg B, B \vdash A(x)$ // (1),(2)否定消除

(4) $B \to \exists x A(x),\ \neg B \vdash B \to A(x)$ // (3)蕴含引入

(5) $B \to \exists x A(x),\ \neg B \vdash \exists x(B \to A(x))$ // (4)∃引入I特例

(6) $B \to \exists x A(x),\ \neg\exists x(B \to A(x)) \vdash B$ // (5)假言易位

(7) $B \to \exists x A(x),\ \neg\exists x(B \to A(x)) \vdash B \to \exists x A(x)$ // 前提引入

(8) $B \to \exists x A(x),\ \neg\exists x(B \to A(x)) \vdash \exists x A(x)$ // (6),(7)蕴涵消除

(9) $B \to \exists x A(x),\ A(x), B \vdash A(x)$ // 前提引入

(10) $B \to \exists x A(x),\ A(x) \vdash B \to A(x)$ // (9)蕴涵引入

(11) $B \to \exists x A(x),\ A(x) \vdash \exists x(B \to A(x))$ // (10)∃引入I特例

(12) $B \to \exists x A(x),\ \exists x A(x) \vdash \exists x(B \to A(x))$ // (11)∃消除特例

(13) $B \to \exists x A(x),\ \neg\exists x(B \to A(x)) \vdash \neg\exists x A(x)$ // (13)假言易位

(14) $B \to \exists x A(x) \vdash \exists x(B \to A(x))$ // (8),(13)否定消除

□

备注 7.2.20 构造上述证明序列的基本思路是利用否定消除规则，将$\neg\exists x(B \to A(x))$作为附加前提，由于它与$\forall x\neg(B \to A(x))$等值，而后者又与$\forall x(B \wedge \neg A(x))$等值，因此由$\neg\exists x(B \to A(x))$应该可以得到$B$及$\forall x\neg A(x)$，而$\forall x\neg A(x)$与$\neg\exists x A(x)$等值，所以实际上我们是希望验证如下两个推理：

$$B \to \exists x A(x),\ \neg\exists x(B \to A(x)) \vdash \exists x A(x) \qquad B \to \exists x A(x),\ \neg\exists x(B \to A(x)) \vdash \neg\exists x A(x)$$

从上面的例子可以看出，当待验证的推理的结论是存在量词公式时，构造证明序列往往比较难，其原因往往是因为不能直接消除前提集中的存在量词公式来得到使得结论成立的那个不确定个体，因为$\exists x A(x) \vdash A(x)$本身不是有效的推理。所以对于这一类的推理，我们往往需要使用反证法。

最后考虑与量词分配等值式相对应的推理：

引理 7.2.21 下面的推理都是有效的推理；

1. $\forall x(A(x) \wedge B(x)) \vdash \forall x A(x) \wedge \forall x B(x)$
2. $\forall x A(x) \wedge \forall B(x) \vdash \forall x(A(x) \wedge B(x))$
3. $\forall x A(x) \vee \forall x B(x) \vdash \forall x(A(x) \vee B(x))$
4. $\exists x(A(x) \vee B(x)) \vdash \exists x A(x) \vee \exists x B(x)$
5. $\exists x A(x) \vee \exists x B(x) \vdash \exists x(A(x) \vee B(x))$
6. $\exists x(A(x) \wedge B(x)) \vdash \exists x A(x) \wedge \exists x B(x)$

证明 1, 2的验证非常简单，下面验证其他的推理式。

3. 可使用如下证明序列验证：

(1) $\forall xA(x) \vee \forall xB(x), \forall xA(x) \vdash \forall xA(x)$ // 前提引入

(2) $\forall xA(x) \vee \forall xB(x), \forall xA(x) \vdash A(x)$ // (1)∀消除I特例

(3) $\forall xA(x) \vee \forall xB(x), \forall xA(x) \vdash A(x) \vee B(x)$ // (2)析取引入

(4) $\forall xA(x) \vee \forall xB(x), \forall xA(x) \vdash \forall x(A(x) \vee B(x))$ // (3)∀引入

(5) $\forall xA(x) \vee \forall xB(x), \forall xB(x) \vdash \forall xB(x)$ // 前提引入

(6) $\forall xA(x) \vee \forall xB(x), \forall xB(x) \vdash B(x)$ // (5)∀消除I特例

(7) $\forall xA(x) \vee \forall xB(x), \forall xB(x) \vdash A(x) \vee B(x)$ // (6)析取引入

(8) $\forall xA(x) \vee \forall xB(x), \forall xB(x) \vdash \forall x(A(x) \vee B(x))$ // (7)∀引入

(9) $\forall xA(x) \vee \forall xB(x) \vdash \forall xA(x) \vee \forall xB(x)$ // 前提引入

(10) $\forall xA(x) \vee \forall xB(x) \vdash \forall x(A(x) \vee B(x))$ // (4),(8),(9)析取消除

4. 可使用如下证明序列验证：

(1) $A(x) \vee B(x),\ A(x) \vdash A(x)$ // 前提引入

(2) $A(x) \vee B(x),\ A(x) \vdash \exists xA(x)$ // (1)∃引入I特例

(3) $A(x) \vee B(x),\ A(x) \vdash \exists xA(x) \vee \exists xB(x)$ // (2)析取引入

(4) $A(x) \vee B(x),\ B(x) \vdash B(x)$ // 前提引入

(5) $A(x) \vee B(x),\ B(x) \vdash \exists xB(x)$ // (4)∃引入I特例

(6) $A(x) \vee B(x),\ B(x) \vdash \exists xA(x) \vee \exists xB(x)$ // (5)析取引入

(7) $A(x) \vee B(x) \vdash A(x) \vee B(x)$ // 前提引入

(8) $A(x) \vee B(x) \vdash \exists xA(x) \vee \exists xB(x)$ // (3),(6),(7)析取消除

(9) $\exists x(A(x) \vee B(x)) \vdash \exists xA(x) \vee \exists xB(x)$ // (8)∃消除特例

5. 可使用如下证明序列验证：

(1) $\exists xA(x) \vee \exists xB(x),\ A(x) \vdash A(x)$ // 前提引入

(2) $\exists xA(x) \vee \exists xB(x),\ A(x) \vdash A(x) \vee B(x)$ // (1)析取引入

(3) $\exists xA(x) \vee \exists xB(x),\ A(x) \vdash \exists x(A(x) \vee B(x))$ // (2)∃引入I特例

(4) $\exists xA(x) \vee \exists xB(x),\ \exists xA(x) \vdash \exists x(A(x) \vee B(x))$ // (3)∃消除特例

(5) $\exists xA(x) \vee \exists xB(x),\ B(x) \vdash B(x)$ // 前提引入

(6) $\exists xA(x) \vee \exists xB(x),\ B(x) \vdash A(x) \vee B(x)$ // (5)析取引入

(7) $\exists xA(x) \vee \exists xB(x),\ B(x) \vdash \exists x(A(x) \vee B(x))$ // (6)∃引入I特例

(8) $\exists xA(x) \vee \exists xB(x),\ \exists xB(x) \vdash \exists x(A(x) \vee B(x))$ // (7)∃消除特例

(9) $\exists xA(x) \vee \exists xB(x) \vdash \exists xA(x) \vee \exists xB(x)$ // 前提引入

(10) $\exists xA(x) \vee \exists xB(x) \vdash \exists x(A(x) \vee B(x))$ // (4),(8),(9)析取消除

6. 可使用如下证明序列验证：

(1) $A(x) \wedge B(x) \vdash A(x)$ // 前提引入
(2) $A(x) \wedge B(x) \vdash \exists x A(x)$ // (1)∃引入I特例
(3) $A(x) \wedge B(x) \vdash B(x)$ // 前提引入
(4) $A(x) \wedge B(x) \vdash \exists x B(x)$ // (3)∃引入I特例
(5) $A(x) \wedge B(x) \vdash \exists x A(x) \wedge \exists x B(x)$ // (2),(4)合取引入
(6) $\exists x(A(x) \wedge B(x)) \vdash \exists x A(x) \wedge \exists x B(x)$ // (5)∃消除特例

□

备注 7.2.22 实际上，上述引理中推理3,4和5的验证都主要是利用析取消除规则，这一点不难想到，因为各推理的前提公式都是含有析取的公式。希望读者能够从上述证明序列进一步熟悉析取消除规则的使用。

读者可以看到，这一小节给出的一些证明序列的构造需要一定的技巧，因此并不要求初学者都能掌握。初学者只要能正确使用约束变量改名规则以及量词否定规则即可。对于量词辖域扩张以及量词分配等值，上面没有给出相应的派生规则，因为这些等值式在推理的验证中并不常用。

7.3 一阶逻辑的自然推理举例

这一节通过一些例子进一步说明如何利用一阶逻辑进行自然推理。

例子 7.3.1 验证下面推理的正确性（教材[4]p80例1）：

$$\forall x(P(x) \to Q(x)),\ \forall x(Q(x) \to R(x)) \ \vdash\ \forall x(P(x) \to R(x))$$

分析：实际上，很容易构造上面推理的证明序列，因为前提和结论中的公式都是全称量词约束的闭公式，将全称量词消除之后，很明显得到前提是$(P(x) \to Q(x))$和$(Q(x) \to R(x))$，而结论是$(P(x) \to R(x))$，显然由命题逻辑中的传递规则很容易得到。

解答：记$\Gamma = \{\forall x(P(x) \to Q(x)),\ \forall x(Q(x) \to R(x))\}$，验证上面推理正确性的证明序列如下：

(1) $\Gamma \vdash \forall x(P(x) \to Q(x))$ // 前提引入
(2) $\Gamma \vdash P(x) \to Q(x)$ // (1)∀消除I特例
(3) $\Gamma \vdash \forall x(Q(x) \to R(x))$ // 前提引入
(4) $\Gamma \vdash Q(x) \to R(x)$ // (3)∀消除I特例
(5) $\Gamma \vdash P(x) \to R(x)$ // (2),(4)传递规则
(6) $\Gamma \vdash \forall x(P(x) \to R(x))$ // (5)∀引入, x在Γ中无自由出现

例子 7.3.2 验证下面推理的正确性：

$$\forall x(F(x) \to G(x)),\ \exists x(F(x) \wedge H(x)) \ \vdash\ \exists x(G(x) \wedge H(x))$$

分析：由于前提中有存在量词，因此很容易想到需要使用存在量词消除，注意，根据存在量词消除规则的特例形式，我们需先在前提中消除存在量词再推出结论，也即我们需要考虑如下推理：

$$\forall x(F(x) \to G(x)),\ F(x) \wedge H(x) \ \vdash\ \exists x(G(x) \wedge H(x))$$

实际上，这里前提集中的$F(x) \wedge H(x)$可认为是**假定有一个不确定的个体（用x表示）使得$F(x) \wedge H(x)$成立**，再来试图推出结论。而对于上述推理，根据存在量词引入规则，我们只需考虑如下推理即可：

$$\forall x(F(x) \to G(x)),\ F(x) \wedge H(x) \ \vdash\ G(x) \wedge H(x)$$

这个推理的证明序列就容易构造了。

解答：记$\Gamma = \{\forall x(F(x) \to G(x))\}$，验证上述推理的证明序列如下：

(1) $\Gamma, F(x) \wedge H(x) \vdash \forall x(F(x) \to G(x))$ // 前提引入

(2) $\Gamma, F(x) \wedge H(x) \vdash F(x) \to G(x)$ // (1)$\forall$消除I特例

(3) $\Gamma, F(x) \wedge H(x) \vdash F(x) \wedge H(x)$ // 前提引入

(4) $\Gamma, F(x) \wedge H(x) \vdash F(x)$ // (3)合取消除

(5) $\Gamma, F(x) \wedge H(x) \vdash G(x)$ // (2),(4)蕴含消除

(6) $\Gamma, F(x) \wedge H(x) \vdash H(x)$ // (3)合取消除

(7) $\Gamma, F(x) \wedge H(x) \vdash G(x) \wedge H(x)$ // (5),(6)合取引入

(8) $\Gamma, F(x) \wedge H(x) \vdash \exists x(G(x) \wedge H(x))$ // (7)$\exists$引入I特例

(9) $\Gamma, \exists x(F(x) \wedge H(x)) \vdash \exists x(G(x) \wedge H(x))$ // (8)$\exists$消除特例

注意，在这里，由于存在量词消除规则的特殊性，我们是令$\Gamma = \{\forall x(F(x) \to G(x))\}$，而不是令$\Gamma = \{\forall x(F(x) \to G(x)),\ \exists x(F(x) \wedge H(x))\}$。另外，在第(9)步使用存在量词消除规则时，$x$没有在$\Gamma$中的公式以及$\exists x(G(x) \wedge H(x))$中自由出现，因此符合存在量词消除规则的使用条件。

例子 7.3.3 验证下面推理的正确性：

$$\exists x F(x) \to \forall x G(x) \ \vdash\ \forall x(F(x) \to G(x))$$

分析：对于这个推理，显然最后一步需使用全称量词引入规则，也即我们实际上可考虑如下推理：

$$\exists x F(x) \to \forall x G(x) \ \vdash\ F(x) \to G(x)$$

而对于这个推理，由于结论是蕴含式，我们将其前件作为附加前提，也即可考虑如下推理：

$$\exists x F(x) \to \forall x G(x),\ F(x) \ \vdash\ G(x)$$

对于这个推理，为了利用到第一个前提，我们希望得到结论$\exists x F(x)$，而这由前提$F(x)$可以得到，也即我们有如下推理：

$$\exists x F(x) \to \forall x G(x),\ F(x) \ \vdash\ \exists x F(x)$$

到这一步，就容易得到整个证明序列了。

解答：验证上述推理的证明序列如下：

(1) $\exists xF(x) \to \forall xG(x),\ F(x) \vdash F(x)$ // 前提引入

(2) $\exists xF(x) \to \forall xG(x),\ F(x) \vdash \exists xF(x)$ // (1)∃引入I

(3) $\exists xF(x) \to \forall xG(x),\ F(x) \vdash \exists xF(x) \to \forall xG(x)$ // 前提引入

(4) $\exists xF(x) \to \forall xG(x),\ F(x) \vdash \forall xG(x)$ // (2),(3)蕴涵消除

(5) $\exists xF(x) \to \forall xG(x),\ F(x) \vdash G(x)$ // (4)∀消除I特例

(6) $\exists xF(x) \to \forall xG(x) \vdash F(x) \to G(x)$ // (5)蕴含引入

(7) $\exists xF(x) \to \forall xG(x) \vdash \forall x(F(x) \to G(x))$ // (6)∀引入

例子 7.3.4 验证下面推理的正确性：

$$\forall x(F(x) \to G(x)) \vdash \forall xF(x) \to \forall xG(x)$$

分析：这个推理的结论是一个蕴涵式，将其前件作为附加前提来进行推理，也即只要考虑如下推理：

$$\forall x(F(x) \to G(x)),\ \forall xF(x) \vdash \forall xG(x)$$

这个推理的证明序列很容易构造。

解答：验证上述推理的证明序列如下：

(1) $\forall x(F(x) \to G(x)),\ \forall xF(x) \vdash \forall x(F(x) \to G(x))$ // 前提引入

(2) $\forall x(F(x) \to G(x)),\ \forall xF(x) \vdash F(x) \to G(x)$ // (1)∀消除I特例

(3) $\forall x(F(x) \to G(x)),\ \forall xF(x) \vdash \forall xF(x)$ // 前提引入

(4) $\forall x(F(x) \to G(x)),\ \forall xF(x) \vdash F(x)$ // (3)∀消除I特例

(5) $\forall x(F(x) \to G(x)),\ \forall xF(x) \vdash G(x)$ // (2),(4)蕴涵消除

(6) $\forall x(F(x) \to G(x)),\ \forall xF(x) \vdash \forall xG(x)$ // (5)∀引入

(7) $\forall x(F(x) \to G(x)) \vdash \forall xF(x) \to \forall xG(x)$ // (6)蕴涵引入

例子 7.3.5 验证下面推理的正确性：

$$\forall x(F(x) \vee G(x)) \vdash \neg\forall xF(x) \to \exists xG(x)$$

分析：这个推理的结论也是一个蕴涵式，将其前件作为附加前提来进行推理，也即只要考虑如下推理：

$$\forall x(F(x) \vee G(x)),\ \neg\forall xF(x) \vdash \exists xG(x)$$

根据存在量词引入规则，我们需要考虑如下推理：

$$\forall x(F(x) \vee G(x)),\ \neg\forall xF(x) \vdash G(x)$$

根据全称量词消除规则，由第一个前提不难得到$F(x) \vee G(x)$，而由这个公式再要得到$G(x)$，则需要利用析取三段论由$\neg F(x)$来得到，现在的问题变成需要考虑如下推理：

$$\forall x(F(x) \vee G(x)),\ \neg\forall xF(x) \vdash \neg F(x)$$

但很可惜，这不是一个有效的推理，因为$\neg\forall xF(x) \to \neg F(x)$不是永真式。考虑到$\neg\forall xF(x)$与$\exists x\neg F(x)$等值，所以我们可利用存在量词消除规则：

$$\frac{\Delta \vdash \exists xA \qquad \Gamma, A \vdash B}{\Delta, \Gamma \vdash B}$$

的如下实例：

$$\frac{\neg\forall xF(x) \vdash \exists x\neg F(x) \qquad \forall x(F(x)\vee G(x)), \neg F(x) \vdash \exists xG(x)}{\forall x(F(x)\vee G(x)),\ \neg\forall xF(x)\ \vdash\ \exists xG(x)}$$

也即我们通过验证推理$\neg\forall xF(x) \vdash \exists x\neg F(x)$以及$\forall x(F(x)\vee G(x), \neg F(x) \vdash \exists xG(x)$来得到原来的推理$\forall x(F(x)\vee G(x)),\ \neg\forall xF(x)\ \vdash\ \exists xG(x)$。

不难构造推理$\forall x(F(x)\vee G(x), \neg F(x) \vdash \exists xG(x)$的证明序列，而$\neg\forall xF(x) \vdash \exists x\neg F(x)$的验证则可利用量词否定规则。

解答：验证上述推理的证明序列如下；

(1) $\forall x(F(x)\vee G(x)),\ \neg F(x)\ \vdash\ \forall x(F(x)\vee G(x))$ // 前提引入

(2) $\forall x(F(x)\vee G(x)),\ \neg F(x)\ \vdash\ F(x)\vee G(x)$ // (1)∀消除I特例

(3) $\forall x(F(x)\vee G(x)),\ \neg F(x)\ \vdash\ \neg F(x)$ // 前提引入

(4) $\forall x(F(x)\vee G(x)),\ \neg F(x)\ \vdash\ G(x)$ // (2),(3)析取三段论

(5) $\forall x(F(x)\vee G(x)),\ \neg F(x)\ \vdash\ \exists xG(x)$ // (4)∃引入I特例

(6) $\neg\forall xF(x)\ \vdash\ \neg\forall xF(x)$ // 前提引入

(7) $\neg\forall xF(x)\ \vdash\ \exists x(\neg F(x))$ // (7)否定∀

(8) $\forall x(F(x)\vee G(x)),\ \neg\forall xF(x)\ \vdash\ \exists xG(x)$ // (5),(7)∃消除

(9) $\forall x(F(x)\vee G(x))\ \vdash\ \neg\forall xF(x)\to\exists xG(x)$ // (8)蕴含引入

回头再看这个推理的验证思路。因为我们知道$\neg\forall xF(x)$与$\exists x\neg F(x)$等值，所以将不妨将要验证的推理看作：

$$\forall x(F(x)\vee G(x)),\ \exists x\neg F(x)\ \vdash\ \exists xG(x)$$

如果像前面希望最后一步使用存在量词引入规则，而考虑如下推理：

$$\forall x(F(x)\vee G(x)),\ \exists x\neg F(x)\ \vdash\ G(x)$$

则会走入歧路，因为这已经不是一个有效的推理！如何少走这种弯路呢？关键是要记住，在这种推理形式结构中，**不管是前提集还是结论公式中出现的自由变量是代表一个不确定的个体**。

这意味着什么呢？这意味着，当我们要从$\Gamma \vdash A(x)$使用存在量词引入规则得到$\Gamma \vdash \exists xA(x)$时，一般需要$x$在$\Gamma$的某个公式中自由出现，也即是这个公式中自由出现的$x$所代表的那个不确定个体也会使得$A(x)$也成立，才让我们可以从$\Gamma \vdash A(x)$得到$\Gamma \vdash \exists xA(x)$。

从另一个角度说，如果x不在Γ中的任意公式中自由出现，那么根据全称量词引入规则，我们甚至可以从$\Gamma \vdash A(x)$得到$\Gamma \vdash \forall xA(x)$，而不仅仅是$\Gamma \vdash \exists xA(x)$。这也反证了，**当我们需要从$\Gamma \vdash A(x)$得到$\Gamma \vdash \exists xA(x)$时，$x$往往需要在$\Gamma$中的某个公式中自由出现**。

进一步，若待验证的推理的前提集没有自由出现的x，那么上面所需要的自由出现的变量x就只能由存在量词消除规则在前提集中引入，所以为了验证推理$\forall x(F(x)\vee G(x)),\ \exists x\neg F(x)\ \vdash\ \exists xG(x)$，

就需要利用存在量词消除规则而考虑验证如下的推理：

$$\forall x(F(x)\vee G(x)),\ \neg F(x)\ \vdash\ \exists xG(x)$$

从这个例子再次可以看出，在与量词有关的规则中，存在量词消除规则的使用有一定的难度，需要使用者有一定的预见性，读者可多练习以掌握其中的技巧。

最后，值得指出的，正如前面所说，下面的推理不是一个有效的推理：

$$\forall x(F(x)\vee G(x)),\ \neg\forall xF(x)\ \vdash\ \neg F(x)$$

但有些读者觉得也许可以构造验证这个推理的证明序列：

(1) $\forall x(F(x)\vee G(x)),\ \neg\forall xF(x),\ F(x)\ \vdash\ F(x)$ // 前提引入

(2) $\forall x(F(x)\vee G(x)),\ \neg\forall xF(x),\ F(x)\ \vdash\ \forall xF(x)$ // (1)∀引入

(3) $\forall x(F(x)\vee G(x)),\ \neg\forall xF(x),\ F(x)\ \vdash\ \neg\forall xF(x)$ // 前提引入

(4) $\forall x(F(x)\vee G(x)),\ \neg\forall xF(x)\ \vdash\ \neg F(x)$ // (2),(3)否定消除

但这是一个错误的证明序列，其错误在于**第(2)步不能由(1)通过全称量词引入规则得到，因为在前提集中存在自由出现的x**。

下面对日常生活中遇到的一些推理进行验证。首先是在引入谓词逻辑时所讨论的一个例子：

例子 7.3.6 验证下面推理的正确性：

所有人都要呼吸，张三是人，所以张三要呼吸

分析：与在命题逻辑中一样，我们先将上述推理符号化，然后再构造证明序列验证它的正确性。当然将用自然语言表示的推理进行符号化的过程，与前面讨论的在一阶逻辑中符号化自然语言命题的过程一样：**先抽取出谓词，特别是要注意特征谓词，并将句子划分成几个子句，分析子句的精确含义，使用正确的量词将其符号化**。

解答：上面推理的符号化比较简单：令$M(x)$表示x是人，$F(x)$表示x要呼吸，个体常量符号a代表张三，则符号化为：$\forall x(M(x)\to F(x)),\ M(a)\ \vdash F(a)$，验证上面推理的证明序列也比较简单：

(1) $\forall x(M(x)\to F(x)),\ M(a)\ \vdash\forall x(M(x)\to F(x))$ // 前提引入

(2) $\forall x(M(x)\to F(x)),\ M(a)\ \vdash M(a)\to F(a)$ // (1)∀消除II

(3) $\forall x(M(x)\to F(x)),\ M(a)\ \vdash M(a)$ // 前提引入

(4) $\forall x(M(x)\to F(x)),\ M(a)\ \vdash F(a)$ // (2),(3)蕴涵消除

例子 7.3.7 验证下面推理的正确性：

每个喜欢步行的人都不喜欢骑自行车。每个人或者喜欢骑自行车或者喜欢乘汽车。有的人不喜欢乘汽车。所以，有的人不喜欢步行

解答：句子“每个喜欢步行的人都不喜欢骑自行车”出现的谓词有：

$F(x):x$ 是喜欢步行的人　　$G(x):x$ 喜欢骑自行车

句子“每个人或者喜欢骑自行车或者喜欢乘汽车”中好像需要提炼出谓词$M(x)$表示x是人，但通读整个推理，我们可以发现，整个推理都是描述人怎样，因此可以简化我们的符号化，将论域（变量的个体域）限定在人，从而无需考虑特征谓词$M(x)$，那么这个句子中出现的新的谓词只有

$$Q(x): x \text{ 喜欢步行乘汽车}$$

总结一下，当**假定论域是人**时，符号化上述推理需要下面三个谓词：

$$F(x): x \text{ 喜欢步行} \qquad G(x): x \text{ 喜欢骑自行车} \qquad Q(x): x \text{ 喜欢步行乘汽车}$$

使用这些谓词，可将上面的推理符号化为：

(1) 句子“每个喜欢步行的人都不喜欢骑自行车”符号化为：$\forall x(F(x) \to \neg G(x))$；

(2) 句子“每个人或者喜欢骑自行车或者喜欢乘汽车”符号化为：$\forall x(G(x) \vee Q(x))$，注意，这里不需要将“或者…或者…”理解为排斥或，因为句子没有强调这种意思；

(3) 句子“有的人不喜欢乘汽车”符号化为：$\exists x(\neg Q(x))$；

(4) 最后，结论“有的人不喜欢步行”符号化为：$\exists x(\neg F(x))$。

因此要验证的推理是：

$$\forall x(F(x) \to \neg G(x)),\ \forall x(G(x) \vee Q(x)),\ \exists x(\neg Q(x)) \vdash \exists x(\neg F(x))$$

分析：我们来思考一下如何验证上述推理。首先，根据存在量词引入规则，我们可将要验证推理的结论看作是$\neg F(x)$，因为从这个结论再用一次存在量词引入规则就可得到所需要的结论。而对于前提中的$\exists x(\neg Q(x))$，根据存在量词消除规则，我们可其看作$\neg Q(x)$，而前提中其他公式都是全称量词公式，根据全称量词消除规则，可将其中的全称量词消除，因此最基本地，我们要验证的推理如下：

$$F(x) \to \neg G(x),\ G(x) \vee Q(x),\ \neg Q(x) \vdash \neg F(x)$$

这个推理的验证不难构造。

解答（续）：记$\Gamma = \{\forall x(F(x) \to \neg G(x)),\ \forall x(G(x) \vee Q(x))\}$，验证该推理的证明序列如下：

(1)	$\Gamma, \neg Q(x) \vdash \forall x(G(x) \vee Q(x))$	// 前提引入
(2)	$\Gamma, \neg Q(x) \vdash G(x) \vee Q(x)$	// (1)$\forall$消除I特例
(3)	$\Gamma, \neg Q(x) \vdash \neg Q(x)$	// 前提引入
(4)	$\Gamma, \neg Q(x) \vdash G(x)$	// (2),(3)析取三段论
(5)	$\Gamma, \neg Q(x) \vdash \forall x(F(x) \to \neg G(x))$	// 前提引入
(6)	$\Gamma, \neg Q(x) \vdash F(x) \to \neg G(x))$	// (5)$\forall$消除I特例
(7)	$\Gamma, \neg Q(x) \vdash \neg F(x)$	// (4),(6)假言易位
(8)	$\Gamma, \neg Q(x) \vdash \exists x \neg F(x)$	// (7)$\exists$引入I特例
(9)	$\Gamma, \exists x \neg Q(x) \vdash \exists x \neg F(x)$	// (8)$\exists$消除特例

注意，由于使用存在量词消除规则的缘故，我们并不是将待验证推理中的所有前提构成的集合记为Γ，而是单独列出了$\exists x \neg Q(x)$。另外，还要注意存在量词引入和消除规则使用的顺序，不能先使

用存在量词消除规则，再使用存在量词引入规则，例如下面的证明序列是错误的：

$$\begin{array}{lll} & \vdots & // \text{(1)至(6)与上面相同} \\ (7) & \Gamma, \neg Q(x) \vdash \neg F(x) & // \text{(4),(6)假言易位} \\ (8) & \Gamma, \exists x \neg Q(x) \vdash \neg F(x) & // \text{(7)}\exists\text{消除特例} \\ (9) & \Gamma, \exists x \neg Q(x) \vdash \exists x \neg F(x) & // \text{(8)}\exists\text{引入I特例} \end{array}$$

因为(8)中使用存在量词消除规则时，$\neg F(x)$含有自由出现的x，这违反存在量词规则使用的条件。

例子 7.3.8 验证下面推理的正确性：

有的病人喜欢所有的医生，但没有病人会喜欢某一个庸医。所以，没有医生是庸医。

解答：句子“有的病人喜欢所有的医生”中有谓词：

$$P(x): x \text{ 是病人} \qquad D(x): x \text{ 是医生} \qquad L(x,y): x \text{ 喜欢 } y$$

句子“没有病人会喜欢某一个庸医”有一个新的谓词：

$$Q(x): x \text{ 是庸医}$$

使用上述谓词，可将上面的推理符号化为：

(1) 句子“有的病人喜欢所有的医生”符号化为：$\exists x(P(x) \wedge \forall y(D(y) \to L(x,y)))$；

(2) 句子“没有病人会喜欢某一个庸医”符号化为：$\neg\exists x(P(x) \wedge \exists y(Q(y) \wedge L(x,y)))$；

(3) 句子“没有医生是庸医”符号化为：$\neg\exists y(D(y) \wedge Q(y))$。

因此要验证的推理是：

$$\exists x(P(x) \wedge \forall y(D(y) \to L(x,y))),\ \neg\exists x(P(x) \wedge \exists y(Q(y) \wedge L(x,y))) \ \vdash\ \neg\exists y(D(y) \wedge Q(y))$$

分析：这个推理看起来好像挺复杂的，但读者看到其结论是一个否定式，第一反应就应该使用归谬律，也即将$\exists y(D(y) \wedge Q(y))$引入作为附加前提来导出矛盾。导出怎样的矛盾呢？前提中也有一个否定式$\neg\exists x(P(x) \wedge \exists y(Q(y) \wedge L(x,y)))$，因此最自然的想法就是利用其他两个前提来推出$\exists x(P(x) \wedge \exists y(Q(y) \wedge L(x,y)))$，即考虑如何验证如下的推理：

$$\exists x(P(x) \wedge \forall y(D(y) \to L(x,y))),\ \exists y(D(y) \wedge Q(y)) \vdash\ \exists x(P(x) \wedge \exists y(Q(y) \wedge L(x,y)))$$

如果读者熟悉下面的假言易位规则，

$$\frac{\Gamma,\ \neg A \vdash \neg B}{\Gamma, B \vdash A}$$

那么可看出从上面的推理应用一次该规则可得到需要验证的推理。而对于上面的推理，结论是存在量词公式，根据存在量词引入规则，不妨将其看作$P(x) \wedge \exists y(Q(y) \wedge L(x,y))$，这是一个合取式，其中$P(x)$应该从何处来？观察前提集，只能通过第一个前提公式得到，而这个前提公式也是存在量词公式，根据存在量词消除规则，我们只需要验证如下推理：

$$P(x) \wedge \forall y(D(y) \to L(x,y)),\ \exists y(D(y) \wedge Q(y)) \vdash\ P(x) \wedge \exists y(Q(y) \wedge L(x,y))$$

对于结论中的$\exists y(Q(y) \wedge L(x,y))$，再根据存在量词引入规则，我们可将其看作$Q(y) \wedge L(x,y)$，而前提中第二个公式$\exists x(D(y) \wedge Q(y))$，利用存在量词消除规则，可用公式$D(y) \wedge Q(y)$来进行推导，它可得到$Q(y)$，并且再利用第一个前提中的全称量词公式就可得到$L(x,y)$。通过上述分析，我们可得到验证题目中推理的证明序列。

解答（续）：为书写简便，我们记：

$$A(x) = P(x) \wedge \forall y(D(y) \to L(x,y)) \qquad B(x) = P(x) \wedge \exists x(Q(y) \wedge L(x,y))$$

注意，这两个公式确实只含自由变量x，y是它们的约束变量。验证此推理的证明序列如下：

(1)	$A(x),\ D(y) \wedge Q(y) \vdash\ A(x)$	// 前提引入
(2)	$A(x),\ D(y) \wedge Q(y) \vdash\ \forall y(D(y) \to L(x,y))$	// (1)合取消除
(3)	$A(x),\ D(y) \wedge Q(y) \vdash\ D(y) \to L(x,y)$	// (2)$\forall$消除I特例
(4)	$A(x),\ D(y) \wedge Q(y) \vdash\ D(y) \wedge Q(y)$	// 前提引入
(5)	$A(x),\ D(y) \wedge Q(y) \vdash\ D(y)$	// (4)合取消除
(6)	$A(x),\ D(y) \wedge Q(y) \vdash\ L(x,y)$	// (3),(5)蕴含消除
(7)	$A(x),\ D(y) \wedge Q(y) \vdash\ Q(y)$	// (4)合取消除
(8)	$A(x),\ D(y) \wedge Q(y) \vdash\ Q(y) \wedge L(x,y)$	// (6),(7)合取引入
(9)	$A(x),\ D(y) \wedge Q(y) \vdash\ \exists y(Q(y) \wedge L(x,y))$	// (8)$\exists$引入I特例
(10)	$A(x),\ \exists y(D(y) \wedge Q(y)) \vdash\ \exists y(Q(y) \wedge L(x,y))$	// (9)$\exists$消除特例
(11)	$A(x),\ \exists y(D(y) \wedge Q(y)) \vdash\ A(x)$	// 前提引入
(12)	$A(x),\ \exists y(D(y) \wedge Q(y)) \vdash\ P(x)$	// (11)合取消除
(13)	$A(x),\ \exists y(D(y) \wedge Q(y)) \vdash\ B(x)$	// (10),(12)合取引入
(14)	$A(x),\ \exists y(D(y) \wedge Q(y)) \vdash\ \exists x B(x)$	// (13)$\exists$引入I特例
(15)	$\exists x A(x),\ \exists y(D(y) \wedge Q(y)) \vdash\ \exists x B(x)$	// (14)$\exists$消除特例
(16)	$\exists x A(x),\ \neg\exists x B(x)\ \vdash\ \neg\exists y(D(y) \wedge Q(y))$	// (15)假言易位

注意，上述证明序列中，(9)和(10)之间的顺序不能调换，必须先使用存在量词引入规则，在使用存在量词消除规则。(14)和(15)也类似，请读者理解其中的原因，并由此掌握存在量词推理规则的使用技巧。

另外，上面在符号化推理的结论时，我们“聪明地”选择了y作为指导变元，如果我们“不幸”选择了x（因为单从该句子来说，按照我们的习惯可能选择x），那么在验证推理时可能会引起一些混淆，这时仍需要将其看作是y作指导变元，然后在最后一步再使用约束变量改名规则即可。

例子 7.3.9 验证下面的推理（此句子来自[2]p117例7）：

如果所有的思想都是清楚的，那么没有思想需要解释；如果所有的思想都不是清楚的，那么没有思想能够解释清楚。因此，如果有的思想既需要解释又能解释清楚，那么有的思想清楚，有的思想不清楚

分析：初看起来这个推理的符号化会非常复杂，但稍加分析，就可发现整个推理都是在对“思想”加以判断，因此我们可将论域限定在“思想”。

解答：假定论域是“思想”，分析句子，可发现其中出现的谓词包括：

$C(x)$：x 是清楚的　　$M(x)$：x 需要解释　　$N(x)$ x 能够解释清楚

而上述推理符号化为：

(1) 句子“如果所有的思想都是清楚的，那么没有思想需要解释”符号化为：

$$\forall x C(x) \to \neg\exists x M(x)$$

(2) 句子“如果所有的思想都不是清楚的，那么没有思想能够解释清楚”符号化为：

$$\forall x \neg C(x) \to \neg\exists x N(x)$$

(3) 句子“如果有的思想既需要解释又能解释清楚，那么有的思想清楚，有的思想不清楚”符号化为：

$$\exists x(M(x) \wedge N(x)) \to (\exists x C(x) \wedge \exists x \neg C(x))$$

因此要验证的推理是：

$$\forall x C(x) \to \neg\exists x M(x),\ \forall x \neg C(x) \to \neg\exists x N(x) \vdash \exists x(M(x) \wedge N(x)) \to (\exists x C(x) \wedge \exists x \neg C(x))$$

为书写方便，我们记

$$\begin{aligned} \Gamma &= \{\forall x C(x) \to \neg\exists x M(x),\ \forall x \neg C(x) \to \neg\exists x N(x)\} \\ \Gamma_1 &= \{\forall x C(x) \to \neg\exists x M(x),\ \forall x \neg C(x) \to \neg\exists x N(x),\ M(x) \wedge N(x)\} \end{aligned}$$

验证此推理的证明序列如下：

(1)	$\Gamma_1 \vdash M(x) \wedge N(x)$	// 前提引入
(2)	$\Gamma_1 \vdash M(x)$	// (1)合取消除
(3)	$\Gamma_1 \vdash \exists x M(x)$	// (2)∃引入I特例
(4)	$\Gamma_1 \vdash \forall x C(x) \to \neg\exists x M(x)$	// 前提引入
(5)	$\Gamma_1 \vdash \neg\forall x C(x)$	// (3),(4)假言易位
(6)	$\Gamma_1 \vdash \exists x \neg C(x)$	// (5)否定∀
(7)	$\Gamma_1 \vdash N(x)$	// (1)合取消除
(8)	$\Gamma_1 \vdash \exists x N(x)$	// (7)∃引入I特例
(9)	$\Gamma_1 \vdash \forall x \neg C(x) \to \neg\exists x N(x)$	// 前提引入
(10)	$\Gamma_1 \vdash \neg\forall x \neg C(x)$	// (8),(9)假言易位
(11)	$\Gamma_1 \vdash \exists x C(x)$	// (10)否定∀
(12)	$\Gamma_1 \vdash \exists x C(x) \wedge \exists x \neg C(x)$	// (6),(11)合取引入
(13)	$\Gamma, \exists x(M(x) \wedge N(x)) \vdash \exists x C(x) \wedge \exists x \neg C(x)$	// (12)∃消除特例
(14)	$\Gamma \vdash \exists x(M(x) \wedge N(x)) \to \exists x C(x) \wedge \exists x \neg C(x)$	// (13)蕴含引入

实际上，验证上述推理的基本思路是：由于推理的结论是蕴含式，因此我们将其前件作为附加前提进行验证，也即我们验证如下推理；

$$\forall x C(x) \to \neg\exists x M(x),\ \forall x \neg C(x) \to \neg\exists x N(x),\ \exists x(M(x) \wedge N(x)) \vdash\ \exists x C(x) \wedge \exists x \neg C(x)$$

此推理的结论是一个合取式，我们分别从前提导出$\exists x C(x)$和$\exists x \neg C(x)$即可。而对于前提中的存在量词公式$\exists x(M(x) \wedge N(x))$，根据存在量词消除规则，可看作$M(x) \wedge N(x)$，也即我们需要验证如下推理：

$$\forall x C(x) \to \neg\exists x M(x),\ \forall x \neg C(x) \to \neg\exists x N(x),\ M(x) \wedge N(x) \vdash\ \exists x C(x)$$
$$\forall x C(x) \to \neg\exists x M(x),\ \forall x \neg C(x) \to \neg\exists x N(x),\ M(x) \wedge N(x) \vdash\ \exists x \neg C(x)$$

而由$M(x) \wedge N(x)$我们可得到$M(x)$，再由存在量词引入规则可得到$\exists x M(x)$，再使用假言易位，由$\forall x C(x) \to \neg\exists x M(x)$就可得到$\neg\forall x C(x)$，而这由量词否定等值式就可得到$\exists x \neg C(x)$，另外一个推理可类似验证，于是得到了上面的证明序列。

从上面的例子可以看到：

(1) 使用自然推理系统中的规则验证一阶逻辑中的推理的关键在于考虑如何消除前提中的量词，以及如何引入结论中的量词。使用合适的规则消除前提中的量词，以及预见到如何引入结论中的量词之后，剩下的不带量词的公式之间的推理就与命题逻辑中的推理相同；

(2) 在消除和引入量词时，特别要注意引入全称量词时，要求指导变元符号不在前提集任意公式中自由出现，而消除存在量词时，要求指导变元符号不在某些公式中自由出现。实际上，简单地看，需要这些条件的关键在于$A(x) \to \forall x A(x)$**和**$\exists x A(x) \to A(x)$**都不是永真式**。消除全称量词和引入存在量词的附加条件看起来好像很复杂，但我们最常用的是其中的特例情况，这时并不需要任何附加条件，这里的关键则在于$\forall x A(x) \to A(x)$**以及**$A(x) \to \exists x A(x)$**是永真式**。

作 业

作业 7.1 对于一阶逻辑的自然推理中，判断下面各证明序列的正确性，如果错误请说明哪些步骤是错误的：

1.	(1)	$\neg\forall x A(x) \vdash \neg\forall x A(x)$	// 前提引入
	(2)	$\neg\forall x A(x) \vdash \neg A(x)$	// (1)∀消除I特例
	(3)	$\neg\forall x A(x) \vdash \exists x \neg A(x)$	// (2)∃引入I特例
2.	(1)	$\neg\forall x A(x),\ A(x) \vdash A(x)$	// 前提引入
	(2)	$\neg\forall x A(x),\ A(x) \vdash \forall x A(x)$	// (1)∀引入
	(3)	$\neg\forall x A(x),\ A(x) \vdash \neg\forall x A(x)$	// 前提引入
	(4)	$\neg\forall x A(x),\ A(x) \vdash \neg A(x)$	// (2),(3)归谬律
	(5)	$\neg\forall x A(x) \vdash \exists x \neg A(x)$	// (4)∃引入I特例

3. (1) $\forall x\neg A(x),\ A(x)\vdash A(x)$ // 前提引入
(2) $\forall x\neg A(x),\ \exists xA(x)\vdash A(x)$ // (1)∃消除特例
(3) $\forall x\neg A(x),\ \exists xA(x)\vdash \forall x\neg A(x)$ // 前提引入
(4) $\forall x\neg A(x),\ \exists xA(x)\vdash \neg A(x)$ // (3)∀消除I特例
(5) $\forall x\neg A(x)\vdash \neg\exists xA(x)$ // (2),(4)归谬律

作业 7.2 在一阶逻辑中，使用自然推理规则验证下列推理的正确性：

(1) $\exists xF(x)\to\forall y((F(y)\vee G(y))\to R(y)),\ \exists xF(x)\ \vdash\ \exists xR(x)$
(2) $\forall x(F(x)\to(G(a)\wedge R(x))),\ \exists xF(x)\ \vdash\ \exists x(F(x)\wedge R(x))$
(3) $\forall x(F(x)\vee G(x)),\ \neg\exists xG(x)\ \vdash\ \exists xF(x)$
(4) $\forall x(F(x)\vee G(x)),\ \forall x(\neg G(x)\vee\neg R(x)),\ \forall xR(x)\ \vdash\ \exists xF(x)$

作业 7.3 在一阶逻辑中，符号化下列推理，并使用自然推理规则验证其正确性：

(1) 有理数都是实数。无理数也都是实数。虚数不是实数。因此，虚数既不是有理数，也不是无理数。

(2) 大学里的学生不是本科生就是研究生。有的大学里的学生是高材生。张三不是研究生但是高材生。因此，如果张三是大学里的学生就一定是本科生。

(3) 每个科学工作者都是刻苦钻研的。每个刻苦钻研而由聪明的人在他的事业中都将获得成功。王大海是科学工作者，并且是聪明的。所以，王大海在他的事业中将获得成功。

(4) 演绎系统是完全的当且仅当他能表达的任一真公式在该演绎系统中都可证。演绎系统是一致的，当且仅当存在公式在该系统中可表达但不可证。因此，任何不一致的演绎系统一定是完全的。(**提示**: 令$D(x)$表示x是演绎系统；$A(x)$表示X是完全的；$T(x)$表示x是真的；$F(x)$表示x是公式；$R(x,y)$表示x在y中可表达；$P(x,y)$表示x在y中可证；$B(x)$表示x是一致的)

第八章　非经典逻辑简介

8.1　非经典逻辑概述

逻辑学是研究思维形式结构及其规律的科学，其主要研究对象是有效推理形式和有效证明形式。现代逻辑是指经典数理逻辑和以它为基础产生发展起来的各种非经典逻辑系统。现代逻辑的基本特点是使用形式化方法建立逻辑对象的形式系统，并且对于形式系统的性质进行元逻辑研究[9]。

经典数理逻辑是指为适合经典数学建立可靠的基础的需要而建立的现代形式逻辑系统，可以以1910-1913年出版的怀特海和罗素所著的《数学原理》给出的PM系统为代表。1930年发表的哥德尔完全性定理，证明了该系统是具有完全性的经典逻辑演算系统；1931年发表的哥德尔不完全性定理证明了PM系统对于论证形式算术系统的算术定理具有不完全性。这两个元定理的发表标志着对经典逻辑演算PM系统的研究已臻于成熟[9]。简单地说，经典数理逻辑包括命题逻辑及其演算系统，谓词逻辑及其演算系统，以及基于谓词逻辑的算术演算系统，和研究摹状词等与谓词逻辑相关问题的理论。

经典数理逻辑有若干固有的逻辑特征，决定了其一定的适用领域和非使用领域。经典数理逻辑固有的逻辑特征包括（根据文献[9]的总结，但根据我自己的理解略有修改）：

1. 它是外延逻辑，也即，其个体变量是以经典集合论为论域；
2. 它是真假二值逻辑，研究的命题具有确定的真值；
3. 它是承认排中律、矛盾律和反证法的逻辑；
4. 它的推理具有保真性和单调性，保真性是指从真的命题可推出真的结论，单调性是指前提的增加不会否定已证的结论；
5. 它以**实质蕴涵**作为有效推理的基础，实质蕴涵是指仅考虑蕴涵式前后件的真值关系，认为蕴涵式$A \to B$为真当且仅当并非A真而B假，不考虑前后件之间的其它联系，例如是否意义相关等；
6. 它是不含模态词的逻辑，简单地说，模态词也是从原子命题构造复合命题的方法，但这时复合命题的真值并不完全由其支命题的真值确定，而是具有更为复杂的关系。

经典数理逻辑自产生以来，逐步暴露其固有的局限性，为克服其局限性，自20世纪30年代以来，人们建立了许多非经典逻辑系统。所谓**非经典逻辑是指这样的逻辑系统，它至少具有一个不同于经典逻辑的逻辑特征**。人们从哲学角度将各种各样的非经典逻辑都纳入所谓**哲学逻辑**的范畴，所谓哲学逻辑，是这样的一个哲学分支，研究在经典数理逻辑基础上发展起来的，以传统的哲学概念、范畴以及逻辑在各门具体学科的应用为对象，构造出来的各种具有直接哲学意义的逻辑系统[10]。这些逻辑系统大致可分为两大类：**变异逻辑**和**应用逻辑**[10]。

变异逻辑是通过否定或修改经典数理逻辑的某些基本假定而形成的逻辑分支。经典数理逻辑的

基本假定，按照文献[10]中的观点，包括外延原则、实质蕴涵、二值原则以及解释个体变量的论域非空原则。从某种意义上可以说是这些基本假定导出了经典数理逻辑的上述逻辑特征。重要的变异逻辑包括：

(1) **直觉逻辑**：修改了经典数理逻辑中对命题真值的解释，而不仅仅是简单的二值原则。直觉逻辑认为，**一个命题为真当且仅当有一个可行的方法证明它是真的**。因此命题真和证明它是真的可行办法密不可分，这一点使得直觉逻辑与计算机科学的关系十分密切，计算机科学的特点之一就是可构造性，因为在计算机科学中对任何问题的研究都希望最后能编写计算机程序来求解。在直觉逻辑中，命题逻辑联结词赋予了不同的含义，排中律和反证法也不成立。我们在后面还将对直觉逻辑作更加详细一些的介绍。

(2) **相干逻辑**：简单地说，相干逻辑试图研究蕴涵式前件和后件之间的内容联系，相干逻辑认为，$A \to B$为真，当且仅当，A和B之间具有某种共同的意义内容，使得A可逻辑地推出B。形式地，著名的“相干原理”要求，A相干蕴涵B，则A与B至少有一个共同的命题变量。

(3) **多值逻辑**：简单地说，多值逻辑认为命题的真值除真和假之外还可能取其他的值，例如“不确定”等。多值逻辑的进一步延伸是**模糊逻辑**，模糊逻辑认为命题的真值可能是模糊的，可以取$[0,1]$区间的任何值。鉴于模糊逻辑在自动控制领域的广泛应用，例如目前有许多家电都宣称使用了模糊控制技术，我们在后面也对多值逻辑和模糊逻辑作更为详细一些的介绍。

(4) **内涵逻辑**：任何语言成分都有内涵和所指，例如，就“语言”这个词汇，其内涵简单地可以说是符号串的集合，其外延则是指像汉语、英语等这些具体的语言。从一阶逻辑公式的解释可以看出，经典数理逻辑是研究外延的逻辑，个体变量指称到论域的元素，公式$\forall x A(x)$为真，是指对论域的任意元素d，$A(d)$为真。如果使用数学化的方法同时研究处理内涵和所指，得到的逻辑理论就是内涵逻辑。

应用逻辑是在经典数理逻辑的基础上，增加一些具有明显哲学意味的初始符号及相关的公理和规则，以用于分析某些具体学科特别是哲学中的概念和范畴而建立起来的逻辑系统。应用逻辑中最著名的是**模态逻辑**，而对传统归纳逻辑的发展，即现代归纳和概率逻辑，以及对传统辩证逻辑的发展，即现代辩证逻辑，特别是数理辩证逻辑也都属于应用逻辑的范畴。

广义模态逻辑包括**狭义模态逻辑**、**道义逻辑**、**时态逻辑**、**认知逻辑**等。狭义模态逻辑又称真值模态逻辑，是关于事物存在的必然性和可能性的逻辑，或者说是含有模态词“必然”、“可能“的命题及其推理的逻辑。必然性与偶然性、可能性与现实性都是哲学中最重要的范畴，模态逻辑对它们进行系统的研究，加深了人们对这些范畴的认识。

道义逻辑、时态逻辑和认知逻辑是广义上的模态逻辑，它们所研究的广义模态词与狭义模态词（即“必然”与“可能”）十分相似。例如，道义逻辑研究模态词“应该”和“允许”与“必然”和“可能”之间的关系类似：一个命题是必然的，当且仅当它的否定是不可能的，而一个行为是必须的即应该的，当且仅当不做该行为是不允许的。时态逻辑研究与“过去”、“现在”、“将来”等与时间有关的模态词，而认知逻辑研究与“知道”、“相信”、“怀疑”等与信念有关的模态词。

近年来，广义模态逻辑，特别是时态逻辑在计算机科学中的应用也日益广泛，特别是在描述程序，尤其是并发程序的动态特性方面有十分重要的作用，因此我们在后面也将以狭义模态逻辑为核心进一步介绍广义模态逻辑的基本概念。

由于二十世纪三十年代以来出现的逻辑非常多，例如除上面提到的逻辑系统之外，还有弗协调

逻辑、非单调逻辑、量子逻辑、偏逻辑、线性逻辑、自由逻辑、问题逻辑、命令逻辑、无穷逻辑等等，要对这些逻辑做一个好的分类几乎是不可能的，不同的学者对逻辑的分类也有不同的观点，同一个逻辑系统也可能属于不同的逻辑，上面的介绍仅仅是希望能提供一个大致的框架，让读者能够对非经典逻辑有一个基本的了解。

下面几节将对与计算机科学密切相关的**直觉逻辑**、**多值逻辑与模糊逻辑**，以及**模态逻辑**再作稍微详细一些的介绍，使读者对这些逻辑系统有一些基本的了解，为读者在后续的学习和研究中，特别是在研究生学习阶段打下一定的基础。

8.2 直觉逻辑简介

本节是作者在阅读[11, 6, 9, 10, 12]等文献的基础上，根据自己对直觉逻辑的理解综合而成。

8.2.1 直觉主义与直觉逻辑

直觉逻辑(intuitive logic)是是数学哲学中**直觉主义**流派构造数学时所用的逻辑，属于构造性逻辑，反映了建立直觉主义数学时所采用的证明方法。直觉主义是在二十世纪初对于数学基础的探讨而形成的一种学派，其创始人是荷兰数学家布劳维尔(L. E. J. Brouwer, 1881-1966)，他创造性地继承了康德的先验直观理论，把对时间的先验直觉作为数学的基础，因而被称为直觉主义。

直觉主义在数学基础及数学的构造方面有以下基本观点：

1. **坚持潜无穷，否认实无穷**。所谓实无穷就是将无穷视为现实的、完成了的总体，可以将无穷的对象作为一个整体进行研究，例如自然数集、实数集等。所谓潜无穷就是认为无穷只能看作是一种无休止扩展或延伸的可能性或过程，而不是一种实际得到的总体，例如自然数就是通过0和后继操作不断构造出来。

2. **存在等于被构造**。数学对象的存在以可构造为前提，即要能够具体给出该对象，或者至少给出能找到该对象的方法。因此说**命题为真**，意味着**找到一个证明命题为真的方法**，说**命题为假**，意味着有一个方法，在假设命题为真时能导出荒谬。

3. **排中律不普遍有效**。这有两个方面的原因：一是对于无穷域不可能对每个元素检查其是满足性质A还是不满足性质A，因此$A \vee \neg A$在无穷论域内无效；二是由于对命题真值的理解使得命题$A \vee \neg A$理解为：命题A是可证明为真的，或者有一个方法在假设命题A为假时会导出荒谬，显然这二者可能都不成立。

潜无穷是直觉主义的哲学基础，存在等于构造以及否认排中律是直觉主义在构造直觉逻辑的出发点，所以我们这里再讨论一下什么是可构造，然后在下一小节对于逻辑联结词在直觉逻辑中的解释作进一步的介绍，这一节的最后一小节则简单介绍直觉逻辑演算系统及其内定理的证明。

要说清楚什么是可构造实际上是一件很难的事情，下面只能通过一些例子让读者对可构造有一些基本的认识。首先，自然数集是可构造的，从0开始，不断地使用后继操作可得到所有的自然数，或者说可得到你想要的任何自然数。大家在学习可计算理论时，也将知道，递归函数（也即计算机能够计算的函数）就是从常函数0，以及后继函数，再加上一些基本函数，通过函数复合、递归以及最小化得到的。或者简单地，对于讨论从自然数到自然数的递归函数，要想产生任何自然数，都是从0和后继操作得到的。

自然数集是可构造的，整数集从本质上来说可看作是自然数对（整数可看作是两个自然数做减法的结果），有理数可看作是整数对（有理数可看作是两个整数做除法的结果），因此整数集和有理数集都是可构造的，但对于实数集，其情况则复杂得多，直觉主义最初的努力就是如何探讨如何构造出实数集，或者实数集的某个合适的子集。从计算机的角度直观看，我们也知道，自然数在计算机是精确处理的，因为自然数可被简单地构造出来，而实数在计算机只能近似处理，因为实数不能被简单地构造。

简单地，可构造性的数学对象就是可以编写一个计算机程序得到，例如，$1+2+\cdots+100$是可构造性的数学对象，可容易地编写一个程序计算它的值。而下面的定义就不是可构造性的：

$$P=\begin{cases}1 & \text{如果哥德巴赫猜想成立}\\ 0 & \text{否则}\end{cases}$$

因为哥德巴赫猜想是否成立目前还不知道，所以P的值到底是1还是0就也不知道，没有办法编写一个程序计算P的值。

存在就是被构造这个观点看起来是很强的，因为它排除了数学中经常使用的一些存在性证明，例如：证明存在无理数a和b，使得a^b是有理数，通常的存在性证明是：$\sqrt{2}^{\sqrt{2}}$要么是有理数，要么是无理数，如果它是有理数，则命题已证，因为可取$a=b=\sqrt{2}$，若它是无理数，则取$a=\sqrt{2}^{\sqrt{2}}$，而$b=\sqrt{2}$，则$a^b=2$也是有理数命题得证。这个证明不是可构造性的，因为没有构造出具体的a和b使得a^b是有理数。

8.2.2 命题真值的直觉逻辑解释

直觉主义的存在就是被构造的观点体现在直觉逻辑中就是对命题真值的解释：**命题为真意味着命题可以构造性地证明为真**，或者说有一个能行的办法证明该命题为真，或者更进一步地说，**有一个证明该命题为真的程序**。这一点使得直觉逻辑与计算机科学的关系十分密切，Martin-Löf的**直觉类型论**使用直觉逻辑为程序的规范说明、程序的构造以及程序正确性的验证提供了一整套完整的理论，而在程序设计理论中研究的各种类型理论，包括描述面向对象程序设计语言的多态性、子类型等各种高阶类型理论都是在直觉类型论的基础上发展起来的。

直觉逻辑中上述对命题真值的解释扩充对各种逻辑联结词的解释如下：

1. 公式$A\wedge B$为真，是指既存在证明公式A真的方法（程序）a，又存在证明公式B真的方法b，因此证明公式$A\wedge B$为真的方法是有序对$\langle a,b\rangle$；

2. 公式$A\vee B$为真，是指存在方法p，首先它能选择是证明A是证明B，如果选择证明A，则存在证明A的方法a，如果选择证明B，则存在证明B的方法b，或者说p是一个有序对$\langle m,q\rangle$，其中$m=0$或1，$m=0$表示q是证明A的方法，$m=1$表示q是证明B的方法；

3. 公式$A\to B$为真，是指存在方法p，它能将任意证明A的方法a，转换为证明公式B的方法$p(a)$，也即p是能转换证明方法的函数；

4. 公式$\bot$，其含义是永假，也即不存在任何证明$\bot$的方法；

5. 公式$\neg A$定义为$A\to\bot$，也即$\neg A$为真，表示存在一个方法p，它能将证明A为真的方法a，转换为证明$\bot$的方法（从而产生矛盾，因为定义公式$\bot$不存在任何证明方法）。

上面对命题真值的解释最先由布罗维尔(L. E. J. Brouwer)、海廷(A. Heyting)和科尔莫哥洛夫(A. N. Kolmogorov)等人给出，因此又称为BHK-解释，可用下表将上述解释总结如下：

公式构造	BHK-解释（直觉逻辑解释）
A	证明A的可行方法a
$A \wedge B$	证明A的可行方法a和证明B的可行方法构成的有序对$\langle a, b\rangle$
$A \vee B$	选择标记0或1，和证明A的可行方法a或证明B的可行方法组成的有序对$\langle 0, a\rangle$或$\langle 1, b\rangle$
$A \to B$	把任意证明A的可行方法转换为证明B的可行方法的函数（方法）
$\bot$	不存在任何可行证明方法
$\neg A$	定义为$A \to \bot$，能把任意证明A的可行方法转换为证明$\bot$的方法的函数（方法）

进一步，我们可将公式A和证明公式A为真的方法a从不同的角度作不同的解读，从这些解读中，我们可看到直觉逻辑与程序类型构造、程序规范及程序构造之间的关系：

(1) 公式A是表示一个集合，a是该集合的元素，公式的构造对应直觉主义中集合的构造，$A \wedge B$对应集合的笛卡尔积，$A \vee B$对应集合的不相交并$\{\langle 0, a\rangle \mid a{\in}A\} \cup \{\langle 1, b\rangle \mid b{\in}B\}$，$A \to B$对应集合$A$到集合$B$的所有函数构成的集合等等；

(2) 公式A代表一个类型，a是该类型中的数据，公式的构造对应类型的构造，$A \wedge B$对应积类型，相当于C++语言中的结构(struct)类型，$A \vee B$对应共积类型，相当于程序设计语言中的联合(union)类型，$A \to B$相当于函数类型等等；

(3) 公式A是一个目标（规范），a是达到该目标（满足该规范）的一个可实现的方法（程序），$A \wedge B$表示要同时实现两个目标，$A \vee B$表示要选择地实现一个目标，$A \to B$是这样的目标，要将实现目标A方法转换为实现目标B的方法等等，这是克林(Kleene)解释。

公式如同类型(formula-as-types)、集合以及目标（规范），以及证明如同程序(proofs-as-program)的这种对应关系被称为Curry-Howard同构（对应），是克雷(H. B. Curry)、霍华德(W. Howard)等许多学者在研究逻辑、类型理论中发现并提出的，揭示了类型化λ–演算系统与直觉逻辑系统之间的对应，例如，直觉逻辑的命题逻辑部分对应简单类型化λ–演算系统(simply typed λ calculi)，一阶逻辑部分对应依赖类型系统(dependent type systems)，二阶逻辑部分对应多态类型系统(polymorphic type systems)等[12]。λ–演算最初由邱奇提出，与递归函数、波斯特系统、图灵机等都是现代计算机的理论模型，λ–演算后来发展为各种类型系统，是程序设计理论，特别是函数式程序设计的基础理论。有关类型理论与λ–演算的简单介绍可参考作者2000年撰写的文章[13]。

8.2.3 直觉逻辑的演算系统

这里仅仅介绍直觉逻辑的命题逻辑部分的演算系统。实际上，直觉逻辑的命题演算系统是经典命题逻辑演算系统的一部分，在直觉逻辑中认为真必然在经典命题逻辑中为真，反之则不然，这很显然，因为直觉逻辑对命题的真值作出了比经典命题逻辑更强的解释。

直觉逻辑的命题演算系统采用与命题演算系统类似的形式语言，只是在直觉逻辑的自然推理系统中，把$\bot$作为初始符号，而将$\neg A$定义成$A \to \bot$更为方便。这里不打算严格定义直觉逻辑的命题演算系统，实际上，只要将前面所讨论的经典命题逻辑的自然推理系统中有关否定联结词的两条规则，

即否定引入和否定消除规则去掉，代之以如下的规则：

$$\frac{\Gamma \vdash \bot}{\Gamma \vdash A}$$

而其他有关合取、析取、蕴涵、等价等联结词的引入和消除规则不变，则得到直觉逻辑（的命题逻辑部分）的自然推理系统。

上述规则的直观含义也是说能推出矛盾（用$\bot$表示）的前提能推出任何命题，在直觉逻辑中，对任意的公式A，公式$\bot \to A$都是永真式，因为按照直觉逻辑的解释，公式$\bot \to A$为真意味着存在一个方法p，它能将任意的证明$\bot$的方法转换为证明A的方法，由于$\bot$没有任何的证明方法，因此可认为这样的方法p是存在的，正如，我们可认为空集到任何集合都存在惟一的函数（即空函数）一样。

在直觉逻辑的自然推理系统中，能证明推理$\vdash A \to \neg\neg A$的有效性，但我们要把公式$A \to \neg A\neg A$看作$A \to ((A \to \bot) \to \bot)$，其证明序列如下：

(1) $A, A \to \bot \vdash A$ // 前提引入

(2) $A, A \to \bot \vdash A \to \bot$ // 前提引入

(3) $A, A \to \bot \vdash \bot$ // (1),(2)蕴含消除

(4) $A \vdash (A \to \bot) \to \bot$ // (3)蕴涵引入

(5) $\vdash A \to ((A \to \bot) \to \bot)$ // (4)蕴涵引入

虽然我们没有给出直觉逻辑的自然推理系统中证明序列的定义，但是读者可参照前面所讨论的命题逻辑的自然推理系统，只要记住除否定引入和否定消除两个规则不能用之外，其他规则都可用即可。实际上，公式$A \to ((A \to \bot) \to \bot)$的直觉逻辑解释为：

$A \to ((A \to \bot) \to \bot)$为真要求存在一种方法$p$，它将任意证明$A$为真的方法$a$，转换为证明$(A \to \bot) \to \bot$的方法$p(a)$。注意证明$(A \to \bot) \to \bot$为真的方法$q$本身也是一种转换方法，它将证明$A \to \bot$的方法$b$转换为证明$\bot$的方法$q(b)$，而证明$A \to \bot$的方法$b$本身就是一种将证明$A$的方法转换为$\bot$的方法，因此前述的方法$p$就存在了：对任意证明$A$的方法$a$，$p(a)$利用证明$A \to \bot$的方法$b$本身将$a$转换为证明$\bot$的方法。

因此公式$A \to \neg\neg A$，也即$A \to ((A \to \bot) \to \bot)$在直觉逻辑的解释下也为永真式，但是公式$((A \to \bot) \to \bot) \to A$，也即$\neg\neg A \to A$则在直觉逻辑的解释下不是永真式，推理$\vdash \neg\neg A \to A$在直觉逻辑的自然推理系统中也不是有效的推理。读者可自行思考，在直觉逻辑的解释下，要求公式$\neg\neg A \to A$为真，意味着存在什么样的能行方法？这种方法为什么不是一定存在的？

类似的，不矛盾律，即公式$\neg(A \wedge \neg A)$在直觉逻辑的解释下是永真式，而排中律$(A \vee \neg A)$则不是永真式。相应地，在直觉逻辑的自然推理系统中，推理$\vdash \neg(A \wedge \neg A)$是有效的，但$\vdash A \vee \neg A$却不是有效的推理。有关直觉逻辑的更多内定理（有效的推理），读者可参考[6]的第五章5.2节有关直觉逻辑形式演算系统的介绍。

8.3 多值逻辑简介

经典数理逻辑的主要特点之一是命题的真值只有两个：真或假。这种二值性一直以来是逻辑研究的主流，但是它并不能解决全部的逻辑问题。例如，命题“地球之外存在生物”在现今的科学

技术水平下还无法确定它到底是真还是假，简单地认为它是真的，或是假都不确切。**多值逻辑**正是基于这种需要而产生的：如果一个逻辑系统**允许命题可以取多于两个不同的真值**，就称为**多值逻辑**(many valued logic)。

多值逻辑的历史可追溯到亚里士多德，但最早系统地研究多值逻辑则是20世纪20年代的波兰逻辑科学家卢卡西维茨(J. Lukasiewicz)以及美国逻辑科学家波斯特(E. L. Post)。最基本的多值逻辑是三值逻辑，其中有名的三值逻辑系统包括卢卡西维茨为处理描述未来可能发生事件的命题而提出的三值逻辑系统L_3，美国数学家克林(S. C. Kleene)由于研究不可判定问题而构造的三值逻辑系统K_3，以及前苏联逻辑学家布奇瓦尔(D. A. Bochvar)为克服语义悖论而提出的三值逻辑系统B_3。

下面对上述三个三值逻辑的基本思想作简单的介绍，这些介绍是作者在参考[11, 14, 15]等文献有关章节的基础上撰写而成。下面的介绍主要讨论在各种三值逻辑系统中如何确定命题的真值，以及如何计算由传统逻辑联结词联结起来的公式的真值，至于它们的推理演算系统则会比较复杂，这里不打算介绍，有兴趣的读者可参考[11, 14, 15]等文献。

8.3.1 卢卡西维茨的三值逻辑系统

卢卡西维茨认为像命题“我明年12月21日下午我将在华沙”在讲这句话时既不真也假，而只是一种可能，因此一个命题可以有三个真值：真(T)、假(F)和中间值(I)。也就是说，卢卡西维茨所建立的三值逻辑系统L_3的第三个真值I用于描述命题所处的一种中间状态，可能真也可能假。从这个观点出发，L_3定义了各种联结词的真值表：

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \to q$	$p \leftrightarrow q$
F	F	T	F	F	T	T
F	T		F	T	T	F
T	F	F	F	T	F	F
T	T		T	T	T	T
I	F	I	F	I	I	I
F	I		F	I	T	I
I	T		I	T	T	I
T	I		I	T	I	I
I	I		I	I	T	T

上述真值表的规律实际上很简单，首先该真值表是经典逻辑真值表的扩充，也即，当p和q的真值限于T和F时，各联结词的真值计算与经典逻辑完全相同。当涉及到值I时，联结词对真值表的计算规律是：用$t(A)\in\{T, I, F\}$表示A的真值，

(1) 三个真值按真值性排列为$T > I > F$；

(2) 否定联结词的计算有以I为镜像的特点，即$\neg T = F, \neg I = I, \neg F = T$；

(3) 合取联结词取两者中真值性弱的真值：$t(A \wedge B) = \min(t(A), t(B))$；

(4) 析取联结词取两者中真值性强的真值：$t(A \vee B) = \max(t(A), t(B))$；

(5) 蕴涵联结词的真值规律可总结如下：

$$t(A \to B) = \begin{cases} T & \text{如果 } t(A) \le t(B) \\ t(B) & \text{否则} \end{cases}$$

(6) 等价联结词的真值可由蕴涵和合取得到：$t(A \to B) = t((A \to B) \wedge (B \to A))$。

我们注意到，如何将真值取值排列为$T > I > F$，那么L_3的蕴涵联结词的真值计算规则是，对于公式$A \to B$，当$t(A) \leq t(B)$时为真，否则就取$t(B)$的真值，这种定义应该经典逻辑中蕴涵联结词真值计算的简单扩充，但是：

$$A \to B \not\Leftrightarrow \neg A \vee B$$

因为当$t(A) = t(B) = I$时，$t(A \to B) = T$，而$t(\neg A \vee B) = I$。

容易根据上述真值表证明，在L_3中德摩根律仍然成立：

$$A \wedge B \Leftrightarrow \neg(\neg A \vee \neg B) \qquad A \vee B \Leftrightarrow \neg(\neg A \wedge \neg B)$$

也不难证明：$(A \vee B) \Leftrightarrow (A \to B) \to B$，因此联结词集合$\{\neg, \to\}$是$L_3$中的完备集：

$$A \vee B \Leftrightarrow (A \to B) \to B$$
$$A \wedge B \Leftrightarrow \neg(\neg A \vee \neg B) \Leftrightarrow \neg((\neg A \to \neg B) \to \neg B)$$
$$A \leftrightarrow B \Leftrightarrow (A \to B) \wedge (B \to A)$$

卢卡西维茨将L_3中的永真式（或重言式）定义为：在任意的真值赋值函数下真值都为T的公式。按这种定义，所以L_3中的永真式都是经典命题逻辑中的永真式，例如下面四个公式既是L_3的永真式，也是经典命题逻辑的永真式：

$$A \to (B \to A)$$
$$(A \to B) \to ((B \to C) \to (A \to C))$$
$$(\neg A \to \neg B) \to (B \to A)$$
$$((A \to \neg A) \to A) \to A$$

细心的读者可能注意到上面前三条公式与命题逻辑公理化演算系统的公理十分类似，实际上有的学者就是利用上述四条公式作为公理建立L_3的公理化演算系统，规则仍然是代入规则和分离规则。

不过在L_3中，排中律$\neg A \vee A$，以及无矛盾律$\neg(A \wedge \neg A)$都不是永真式，因为当A的真值为I时，这两个公式的真值都为I，而不为T。

如果将永真式定义为在任意的真值赋值函数下的真值为T或I的公式，则L_3中的永真式与经典命题逻辑中的永真式完全等同。

8.3.2 克林的三值逻辑系统

克林是为了描述未确定的数学命题而提出三值逻辑系统K_3，它的第三个真值的直观含义是“未定义”，把它赋值到某个公式时并不是想表明该公式既不真也不假，而是想要表明一种未知的状态。这与程序的运行有某种相似之处，程序的运行可能有三种结果：一程序运行终止，得到正确的结果，一是运行终止但得到错误的结果，一是程序运行不终止，但计算机专业的学生都知道，一旦程序陷入长时间不出结果，实际上很难判断它是否真的死机，因此这是一种未知的状态。

在克林的三值逻辑系统中，命题可取的真值包括三个：真(T)、假(F)以及未定义(U)，各种联结词的真值表如下：

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \rightarrow q$	$p \leftrightarrow q$
F	F	T	F	F	T	T
F	T		F	T	T	F
T	F	F	F	T	F	F
T	T		T	T	T	T
U	F	U	F	U	U	U
F	U		F	U	T	U
U	T		U	T	T	U
T	U		U	T	U	U
U	U		U	U	U	U

上述真值表与卢卡西维茨的三值逻辑系统L_3极为相似，只是在蕴涵和等价联结词方面稍有不同。该真值表也是经典逻辑真值表的扩充，当涉及到值U时，联结词对真值表的计算规律是：用$t(A)\in\{T,U,F\}$表示A的真值，

(1) 三个真值按真值性排列为$T > U > F$；

(2) 否定联结词的计算有以I为镜像的特点，即$\neg T = F, \neg U = U, \neg F = T$；

(3) 合取联结词取两者中真值性弱的真值：$t(A \wedge B) = \min(t(A), t(B))$；

(4) 析取联结词取两者中真值性强的真值：$t(A \vee B) = \max(t(A), t(B))$；

(5) 蕴涵联结词的真值规律可总结如下：当$t(A)$和$t(B)$的真值只取T或F时，$t(A \rightarrow B)$的真值与经典逻辑的计算相同，否则（也即$t(A)$和$t(B)$中至少有一个取U值）：

$$t(A \rightarrow B) = \begin{cases} T & \text{如果 } t(A) < t(B) \\ U & \text{否则} \end{cases}$$

(6) 等价联结词的真值可由蕴涵和合取得到：$t(A \rightarrow B) = t((A \rightarrow B) \wedge (B \rightarrow A))$。

实际上，蕴涵联结词的真值可由否定和析取计算，因为在K_3中有$(A \rightarrow B) \Leftrightarrow (\neg A \vee B)$成立，这也是$K_3$与$L_3$的最大区别。同样，在$K_3$中德摩根律成立，因此$\neg, \wedge$可看作最基本的联结词，其他联结词都可使用这两个联结词定义，而且与经典逻辑的定义在形式上完全相同，即：

$$A \vee B \Leftrightarrow \neg(\neg A \wedge \neg B)$$
$$A \rightarrow B \Leftrightarrow \neg A \wedge B$$
$$A \leftrightarrow B \Leftrightarrow (A \rightarrow B) \wedge (B \rightarrow A)$$

如果定义在任意的真值赋值函数下都取值为T的公式才是永真式，那么在K_3中没有一个公式是永真式，因为当公式中所有的命题变元都取U值时，公式的真值肯定是U，而不可能得到T。而如果定义在任意的真值赋值函数下只要取值为T或U就是永真式，则K_3的永真式与经典命题逻辑中的永真式完全等同。

8.3.3 布奇瓦尔的三值逻辑

布奇瓦尔提出的三值逻辑是为了试图解决一些语义悖论，他认为像“我正在说假话”这种悖论

是无意义的，因此不能以真假论之，而将这种悖论句子的真值定义为第三个值M，其直观含义是“无意义”，各种联结词的真值表如下；

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \to q$	$p \leftrightarrow q$
F	F	T	F	F	T	T
F	T		F	T	T	F
T	F	F	F	T	F	F
T	T		T	T	T	T
M	F	M	M	M	M	M
F	M		M	M	M	M
M	T		M	M	M	M
T	M		M	M	M	M
M	M		M	M	M	M

上述真值表的规律十分简单，当命题变量的真值都只限于T或F时与经典命题逻辑完全相同，而如果有一个命题变量的真值是U，则由任何联结词联结而得到的复合命题的真值都是U，因为布奇瓦尔认为，只要复合命题中含有无意义的成分，那么整个命题也就无意义了。实际上，克林在上述L_3的基础上还曾经定义一组弱的三值联结词，其定义方式与上述真值表完全相同，相对而言，上面所给出的L_3的三值联结词的真值表称为“强的”三值联结词。

很容易证明，在布奇瓦尔的三值逻辑系统B_3中，仍然可通过联结词$\neg$和$\wedge$定义其他的联结词，即有如下等值式：

$$A \vee B \Leftrightarrow \neg(\neg A \wedge \neg B)$$
$$A \to B \Leftrightarrow \neg A \wedge B$$
$$A \leftrightarrow B \Leftrightarrow (A \to B) \wedge (B \to A)$$

同样地，如果定义在任意的真值赋值函数下都取值为T的公式才是永真式，那么在B_3中也没有一个公式是永真式。而如果定义在任意的真值赋值函数下只要取值为T或U就是永真式，则K_3的永真式与经典命题逻辑中的永真式完全等同。布奇瓦尔为加强与经典命题逻辑的联系，还定义了所谓的外部联结词（相对而言，上面定义的是内部联结词），外部联结词的命题变元可取值T, F和M，但复合命题对应的公式的真值就只有T和F了，具体定义可参见[14]和[11]。

如果在B_3中限于使用内部联结词，那么大致可避开罗素悖论，因为当定义$R = \{x \mid \neg(x \in x)\}$时，$R \in R$和$R \notin R$并不矛盾，它们可都取$M$值，即都属于无意义的句子。

8.4 模糊逻辑简介

多值逻辑和模糊逻辑都是在命题允许取的真值方面对经典数理逻辑进行扩充。在多值逻辑中，人们往往还是用确定的符号表示命题的真值，例如，三值逻辑，用T表示真，F表示假，U表示真值不确定等，而**模糊逻辑**则进一步使用$[0,1]$区间内的**任意数值**描述命题的真值，0代表完全假，1代表完全真，其他的值则表示命题真假的程度。更进一步，在模糊逻辑中还用区间本身（通常是$[0,1]$区间的子区间）表示命题的真值，也即，这时命题的真值不再是一个确定的值，而是一个区间。

模糊逻辑(fuzzy logic)是模糊理论应用于逻辑而产生的一个逻辑学分支，也可以说是模糊数学的一部分。模糊数学是在1965年美国控制论专家扎德(L. A. Zadeh)提出模糊集(fuzzy set)这一概念后逐步发展起来的，目前已经成为一个十分重要的数学分支，其应用几乎遍及自然科学、社会科学以及工程技术的各个领域，许多模糊技术成果和模糊技术产品已经从实验室走向社会并取得了显著的社会和经济效益[16]。

这一小节对于模糊逻辑的介绍只限于在模糊逻辑中如何描述模糊命题，模糊命题真值的确定，包括由传统联结词联结起来的复合模糊命题的真值确定。要对模糊逻辑以及模糊推理有更深的了解，还需要许多有关模糊数学的基础知识，特别是有关模糊集合的知识，这里限于篇幅而不予介绍，有兴趣的读者可参考[16, 15, 17]等文献。

8.4.1 模糊命题

日常生活中有许多词汇的含义都是模糊的，例如，“高”和“矮”、“胖”和“瘦”、“热”和“冷”，到底到少米才算高，多少米才算矮？日常生活中并没有严格的定义，这些都是日常生活中的模糊现象。所谓的模糊现象就是指没有严格的界限划分，使得很难用精确的尺度来刻划的现象。模糊概念则是模糊现象的反映。像高矮、胖瘦、冷热等就都是模糊概念。

同样地，有些命题涉及模糊概念，其真值也比较难确定。我们将涉及模糊概念，或真值是模糊值的命题称为**模糊命题**。例如下面的命题都是模糊命题：

- 今天天气热；
- 张三比李四聪明多了，但也比李四懒多了。
- 你八成是感冒了，所以你应多喝水。

很显然，模糊命题是普通命题的推广，具有下列几个特点[16]：

1. 模糊命题也是可以判别其真伪的陈述句，但通常不再是绝对的真或假，而只能以多大的程度$\alpha\in[0,1]$属于真或假；

2. 和通常的命题一样，模糊命题同样可进行各种逻辑运算，例如上面的第二个命题用到了逻辑与，而第三个命题用到了逻辑蕴含。

根据所研究问题的需要，模糊命题的真值有不同的表示方法，其中主要有四种方法[16]：

1. **数值表示法**：这种方法是二值命题真值表示的自然推广。设模糊命题p为真的程度是$\alpha\in[0,1]$，则称α为p的真值。例如，对模糊命题“今天天气热”，抽样100人进行调查，有70人认为热，那么可令该命题的真值为0.7。对一个真值为α的模糊命题，若$\alpha=0$或$\alpha=1$，则该命题就是普通命题。

2. **区间值表示法**：用一个$[0,1]$中的数值来刻划某些模糊命题的真值有时还不准确。例如，对模糊命题“今天天气很热”这个命题，不同的人会用$[0,1]$中不同的数值α来表示热的程度，取α_m,α_l分别为这些值中的最大值和最小值，从而可用$[0,1]$的子区间$[\alpha_l,\alpha_m]$来表示该命题的真值，这种方法称为用区间值表示模糊命题的真值法。

3. **模糊值表示法**：将区间值表示法推广，用一个模糊数甚至一个模糊集表示模糊命题真值的方法称为模糊值表示法。该方法虽然复杂，但却可能是最为贴切的一种方法，这种方法在模糊推理的研究中应用非常普遍。

4. **语言真值表示法**：扎德等用语言刻画一个模糊命题的真值[17]，成为“语言真值”的词汇是一

个可数的语言集TV：

$$TV = \{\text{真}, \text{假}, \text{极真}, \text{极假}, \text{很真}, \text{很假}, \text{相当真}, \text{相当假}, \text{不太真}, \text{不太假}, \text{有点真}, \text{有点假}, \cdots\}$$

模糊命题的真值用语言表达的方法使得模糊逻辑系统具有处理自然语言的能力，这在实现机器智能化，模糊专家系统等领域具有广阔的应用前景。

由于我们不打算介绍有关模糊集的知识，所以下面两小节就模糊命题真值的数值表示法和区间表示法做简单的介绍。

8.4.2 狭义模糊命题逻辑

使用$[0,1]$区间内的数值表示命题的真值实际上是多值逻辑思想的推广，被称为**狭义模糊逻辑**。命题之间的逻辑运算基本遵从大多数多值逻辑所采用的逻辑运算规则：即认为“逻辑或”为真值的“取大”原则，“逻辑与”为真值“取小”原则，而“逻辑非”则取其原命题真值的补：

$$\begin{aligned} t(A \vee B) &= \max(t(A), t(B)) \\ t(A \wedge B) &= \min(t(A), t(B)) \\ t(\neg A) &= 1 - t(A) \end{aligned}$$

对于蕴涵联结词，在现有的狭义模糊逻辑中，为了适应在不同情况下的“模糊推理”，其“蕴涵”运算规则（以及由其引伸而来的“等价”运算规则）选用了多种不同的形式，不过其中绝大多数也是由多值逻辑中“借用”而来的。一些主要的运算规则如下所示[15]：

$$R_a : t(A \to B) = \min(1, 1 - (t(A) - T(B)))$$

$$R_b : t(A \to B) = \min(1 - t(A), t(B))$$

$$R_g : t(A \to B) = \begin{cases} 1 & \text{如果 } t(A) \le t(B) \\ t(B) & \text{否则} \end{cases}$$

$$R_m : t(A \to B) = \max(\min(t(A), t(B)), 1 - t(A))$$

$$R_* : t(A \to B) = 1 - t(A) \cdot (1 - t(B))$$

$$R_\blacktriangle : t(A \to B) = \begin{cases} 1 & \text{若} t(A) = 0 \text{或} t(B) = 1 \\ \min(1, t(B)/t(A), (1 - t(A))/(1 - t(B))) & \text{若} t(A) > 0 \text{且} t(B) < 1 \end{cases}$$

$$R_{bp} : t(A \to B) = \max(0, (t(A) + t(B) - 1))$$

文献[15]还列出了许多“蕴涵”运算规则，但该文献没有说明规则的直观含义，因此全部罗列在这里意义不大。实际上，上述R_a和R_g都可说是从卢卡西维茨的三值逻辑系统L_3采用的运算规则转化而来，R_b是经典命题逻辑所采用的运算规则等。

在狭义模糊逻辑中也可定义两个等值：$A \Leftrightarrow B$，如果对任意的真值赋值函数$t : \mathbf{Var} \to [0,1]$，都有$t(A) = t(B)$，只要将经典命题逻辑中的真值赋值函数的值域从$\{0,1\}$集合扩充到$[0,1]$区间即可，而在某个真值赋值函数下公式$A$的真值$t(A)$可根据上述运算规则进行递归运算，其中“蕴涵”运算规则只要选取上述之一即可。

可以证明，大多数经典命题逻辑中的基本等值式也在狭义模糊逻辑中成立，例如有德摩根律$\neg(A \wedge B) \Leftrightarrow (\neg A \vee \neg B)$等成立[15]，但没有排中律和无矛盾律成立，即：

$$A \vee \neg A \not\Leftrightarrow 1 \qquad A \wedge \neg A \not\Leftrightarrow 0$$

在狭义模糊逻辑中可定义绝对恒真式，即在任意真值赋值函数下的真值都为1，和绝对恒假式，及在任意真值赋值函数下的真值都为0。进一步，对任意的实数$0 < \lambda \leq 1$，若公式在任意真值赋值函数下的真值都大于等于λ，则称该公式为λ–恒真公式，特别地当$\lambda = 0.5$时，称为模糊恒真公式；若公式在任意真值赋值函数下的真值都小于等于λ，则称该公式为λ–恒假公式，特别地当$\lambda = 0.5$时，称为模糊恒假公式。

有了模糊命题真值的定义，以及等值的定义就可进行等值演算，并且可进一步讨论与公式等值的范式等[15]。同样地，也可建立狭义模糊逻辑的命题演算系统，这时需要预先（根据客观实事）指定前提公式的真值取值，并在推理的过程中计算每一步得到的公式的真值，而每个规则也要给出规则前提公式和规则结论公式之间的真值关系，最后可推出结论的真值，从而在某种意义给出结论成立的程度。

例如，文献[15]在其2.3节中给出的一个狭义模糊逻辑的命题演算系统，采用与经典命题逻辑的自然推理系统大致相同的规则，只是额外地给出了每个规则所确定的前提公式和结论公式之间的真值关系。例如，对于假言推理，从$A \to B$和A可得到B，这时B的真值$t(B) = t((P \to Q) \wedge P)$。其中，大多数规则结论公式的真值等于前提各公式合取的真值，但对于化简律：由$A \wedge B$得A，规定$t(A) \not< t(A \wedge B)$，而对于附加律：由A得$A \vee B$，规定$t(A \vee B) = t(A)$等。文献[15]还给出了例子说明如何进行推理演算，有关的细节请参考该文献，此处不再赘述。

8.4.3 区间值模糊命题逻辑

狭义模糊命题逻辑使用区间$[0,1]$上的一个实数表达命题的真值，直观而又简便，较易为人们所接受。但在实际运用过程中，由于某些前提（或证据）的真值不易以“点值”给出，故又进一步引入**区间值模糊命题**和**区间模糊命题逻辑**[15]。

称一个真值取区间值的模糊命题为区间值模糊命题。使用这类模糊命题，可以更方便地刻画那些具有某种不确定型的知识（包括实事和规则），并运用它们进行非确定型推理－－区间值模糊推理。这一节我们主要根据文献[15]的2.4节简单介绍区间值模糊命题逻辑的基本内容。

在区间值模糊命题逻辑中，命题的真值实际上是命题所描述实事的清晰度。命题A的真值$t(A)$取区间值$[n,p]$：

(1) n表示命题A为真的必要支持程度，也就是最低肯定程度；

(2) p表示命题A为真的可能支持程度，也就是最高肯定程度；

(3) $1-p$表示命题$\neg A$为真的必要支持程度；

(4) $1-n$表示命题$\neg A$为真的可能支持程度；

(5) $p-n=m$表示对命题A不知道（不了解）的程度，也可称为盲度。

其中，$[n,p]$的几个特殊值的含义是：

(1) $[1,1]$表示命题绝对真；

(2) $[0,0]$表示命题绝对假；

(3) $[0,1]$表示对命题一无所知；

对于n和p的获取，可根据命题所描述事实，通过某种函数分布，或通过领域专家统计而得，即让多个领域专家分别估计该事实为真的必要支持度和可能支持度，然后计算某种平均值而得到n和p。

在区间值模糊命题逻辑中，各个命题也可用传统的逻辑联结词联结，这时命题的真值运算规则可在上述狭义模糊命题逻辑的真值运算规则的基础上进行扩充。设$t(A)=[n_A,p_A], t(B)=[n_B,p_B]$，则

$$\begin{aligned} t(\neg A) &= [1-p_A,\ 1-n_A] \\ t(A\wedge B) &= [\min(n_A,n_B),\ \min(p_A,p_B)] \\ t(A\vee B) &= [\max(n_A,n_B),\ \max(p_A,p_B)] \\ t(A\to B) &= t(\neg A\vee B) = [\max(1-p_A,n_B),\ \max(1-n_A,p_B)] \\ t(A\leftrightarrow B) &= t((A\to B)\wedge(B\to A)) \end{aligned}$$

其中$t(A\to B)$的真值计算也可利用上一小节给出的各种蕴涵真值计算规则。

根据上述真值计算规则，也可建立起区间穿着入时模糊命题逻辑的等值演算和自然推理，这时推理是一种不确定性推理，实际上是公式真值的取值区间变换过程，其中可能更注重盲度的传播，进一步的讨论可参考文献[15, 16]等。

8.5 模态逻辑简介

模态逻辑(modal logic)在经典数理逻辑的基础上，增加一些模态词而形成的逻辑，狭义模态逻辑是指处理与模态词“必然”和“可能”有关的逻辑，广义模态逻辑则还包括道义逻辑、时态逻辑以及认知逻辑等，分别处理与“应该”和“允许”、“将来”和“过去”以及“相信”和“怀疑”等有关的模态词。

模态逻辑是逻辑学的一个古老分支，其研究适于亚里士多德，但现代模态逻辑则创立于十九世纪末至二十世纪三十年代，其创始人是美国逻辑学家刘易斯(C. I. Lewis)。刘易斯等从分析经典逻辑中的实质蕴涵所存在的问题出发，提出所谓的“严格蕴涵”试图避免实质蕴涵的问题，建立了一系列的模态逻辑系统**S1**, **S2**, **S3**, **S4**和**S5**，奠定了现代模态逻辑的基础。后来经过许多学者的研究，特别是克里普克(S. A. Kripke)的可能世界语义模型的建立，使得模态逻辑成为现代逻辑中最重要的分支之一，并在数学、计算机等学科有着十分重要的应用。

这一节主要介绍狭义模态逻辑，简单讨论模态命题在模态逻辑中的符号化，概述模态逻辑的一些演算系统，并简单介绍模态逻辑的可能世界语义。这一节的内容是作者在参考[9, 11, 18]等文献相关章节的基础上撰写而成，有关模态逻辑的参考文献十分多，读者还可参考[19, 20, 21, 22]等。

8.5.1 模态命题与模态算子

亚里士多德就曾经将命题划分为：**实然命题**、**必然命题**和**偶然命题**，经典逻辑中研究的都是实然命题，而模态逻辑则研究必然命题和偶然命题，因此将它们统称为**模态命题**。可以说，经典逻辑研究的实然命题是反映客观事物是否存在的状态的命题，而模态命题是反映事物存在的必然性和可能性的命题。例如下面是一些基本的模态命题：

- 生物体必然要进行新陈代谢。

- 长期大量吸烟可能致癌。
- 客观规律必然不以人的意志为转移。
- 强盗的儿子可能不是强盗。

其中，句子“生物体必然要进行新陈代谢”的逻辑形式是:“必然p”，这里p代表原子命题“生物体要进行新陈代谢”，通常用符号$\Box$表示“必然”，因此该句子可进一步符号化为：$\Box p$。而句子“长期大量吸烟可能致癌”的逻辑形式是:“可能p”，这里p代表原子命题“长期大量吸烟致癌”，通常用符号$\Diamond$表示“可能”，该句子可进一步符号化为：$\Diamond p$。句子“客观规律必然不以人的意志为转移”的逻辑形式是:“必然非p”，符号化为$\Box\neg p$，句子“强盗的儿子可能不是强盗”的逻辑形式是:“可能非p”，符号化为$\Diamond\neg p$。

模态命题也可用传统的逻辑联结词联结起来构成复合模态命题，例如：

(1) 命题“科学不可能是一个人的事业”的逻辑形式是:“非 可能 p”，因而应符号化为：$\neg\Diamond p$；

(2) 命题“必然物体受到摩擦就会生热”的逻辑形式是:“必然 p蕴涵q，符号化为：$\Box(p \to q)$；

(3) 命题“液体沸腾的原因可能是温度升高，也可能是压力下降”的逻辑形式是:“可能 p 或 可能 q”，符号化为：$\Diamond p \vee \Diamond q$。

与量词类似，模态命题前还可加上模态词，从而构成重叠模态命题。例如，命题“永动机不可能创造出来是必然的”的逻辑形式是:“必然 非 可能 p”，符号化为：$\Box\neg\Diamond p$，而命题“张三不可能必然是小偷”的逻辑形式是:“非 可能 必然 p”，符号化为：$\neg\Diamond\Box p$。

通过上述例子，读者不难看到，模态命题就是模态词“可能”和“必然”作用在原子命题或复合命题而得到的命题，从符号化的角度看，就是模态算子$\Box$和$\Diamond$作用到经典命题逻辑公式得到的公式，也就是说，在模态逻辑中，这两个模态算子像经典逻辑的联结词一样，也是构造公式的方法。

但是，模态算子与传统的逻辑联结词有着迥然不同的性质。**逻辑联结词实际是真值函数，由逻辑联结词联结而得到的复合命题的真值由其支命题惟一确定**。例如，当p和q都为真时，$p \to q$的真值就肯定为真，而不会是其他值。但**模态算子却不能看作真值函数，它具有自己的内涵，即含有模态词的模态命题的真值不是由该模态词所限定的命题本身的真值惟一确定**。例如，从“二加二等于四”这个实然命题必然为真，不能确定地推知“二加二必然等于四”，譬如，在三进制中该命题就不真。

在给出模态逻辑公式的定义之前，我们先讨论一下模态算子$\Box$和$\Diamond$的各种解释。在不同的（广义）模态逻辑中，公式$\Box A$有不同的解释[18]：

1. **狭义模态逻辑**：A**必然**为真(It is *necessarily* true that A)；
2. **时态逻辑**：A**永远**为真(It will *always* be true that A)；
3. **道义逻辑**：A**应该**为真(It *ought to* be true that A)；
4. **信念逻辑**：智能体Q**相信**A(Agent Q *believes* that A)；
5. **认知逻辑**：智能体Q**知道**A(Agent Q *knows* that A)；
6. **动态逻辑**：程序P**执行后**A为真(*After any execution* of program P, A holds)。

对应地，模态公式$\Diamond A$的解读是：

1. **狭义模态逻辑**：A**可能**为真(It is *possibly* true that A)；
2. **时态逻辑**：A**将来某些时候**为真(*Sometime in the future* A)；
3. **道义逻辑**：A**允许**为真(It is *permitted* to be that A)；
4. **信念逻辑**：A与智能体Q的信念**相容**(A is *consistent* with agent Q's beliefs)；
5. **认知逻辑**：A与智能体Q的知识**相容**(A is *consistent* with what agent Q knows)；

6. **动态逻辑**：程序P**某此执行后**A为真(*After some execution* of program P, A holds)。

总之，将模态算子$\Box$解释为“必然”，而$\Diamond$解释为“可能”的逻辑称为**狭义模态逻辑**，相应地，其他解释则称为时态逻辑、道义逻辑、信念逻辑、认知逻辑、动态逻辑等，这些都属于**广义模态逻辑**。

8.5.2 模态公式及其可能世界语义

给定一个命题变量符号集合**Var**，在此集合上构造模态命题逻辑公式（简称**模态公式**）的方式是经典命题逻辑公式构造的扩充：

(1) **归纳基**：命题变量$x\in$**Var**是模态公式；

(2) **归纳步**：若A,B是模态公式，则$(\neg A),(A\wedge B),(A\vee B),(A\to B),(A\leftrightarrow B)$是公式；

(3) **归纳步**：如果A是模态公式，则$\Box A$和$\Diamond A$是模态公式；

(4) **最小化**：所有模态公式都是通过上述几步得到的。

在实际使用时，也不需要引入所有的逻辑联结词，只要完备集$\{\neg,\wedge\}$、$\{\neg,\vee\}$或者$\{\neg,\to\}$即可，而对于模态算子$\Box,\Diamond$，也只需要其中一个算子即可，通常选用$\Box$，而将$\Diamond$定义为：$\neg\Box\neg$。该定义的直观含义十分简单：“可能”等同于“不会必然不”。类似地，“必然”等同于“不会可能不”，因此也可将$\Box$定义为：$\neg\Diamond\neg$。

下面是一些在模态逻辑中十分重要的公式（实际上是公式模式，因为下面公式中的A,B可代入任意模态公式），它们都被赋予了某个名称，如“K”、“T”、“D”等（后面我们将看到其中一些名称的含义）：

$$(K)\ \Box(A\to B)\to(\Box A\to\Box B)$$

$$(4)\ \Diamond\Diamond A\to\Diamond A \text{ 或 } \Box A\to\Box\Box A$$

$$(T)\ A\to\Diamond A \text{ 或 } \Box A\to A$$

$$(5\text{或}E)\ \Diamond A\to\Box\Diamond A$$

$$(.3\text{或}H)\ \Diamond A\wedge\Diamond B\to\Diamond(A\wedge\Diamond B)\vee\Diamond(A\wedge B)\vee\Diamond(\Diamond A\wedge B)$$

$$(Dual)\ \Diamond A\leftrightarrow\neg\Box\neg A$$

$$(B)\ A\to\Box\Diamond A$$

$$(D)\ \Box A\to\Diamond A$$

模态公式的经典语义是克里普克提出的可能世界语义。实际上，十七世纪的莱布尼茨就曾经使用“可能世界”这一概念解释模态词[9]。什么是“可能世界”呢？莱布尼茨认为，凡不违反逻辑，能够为人们所想象的情况或场合都是可能世界。我们生活在其中的现实世界，是众多可能世界中的一个。莱布尼茨利用“可能世界”这一概念解释模态词“必然”与“可能”：必然就是在所有的可能世界中真，而可能就是在有些可能世界中真。

美国逻辑学家克里普克从莱布尼茨的上述思想出发，为模态逻辑建立了一套严格的语义理论，叫做可能世界语义学，又叫克里普克语义学。可能世界语义学对经典逻辑语义理论做了三个重大推进[9]：

1. 使命题的真假相对化，即命题的真假只能相对一个可能世界而言，在不同的可能世界可以有不同的真值；

2. 进一步，使必然性和可能性概念相对化，我们必须说在某一世界是必然的，或可能的；

3. 最后，使必然性和可能性相对化是通过可能世界之间的可达关系来得到的：**说命题p在可能世界w必然成立，是指命题p对任意w可达的可能世界u都成立；说命题p在可能世界w可能成立，是指存在w可达的可能世界u，命题p在可能世界u成立**。

可能世界语义的核心概念是**框架**和**模型**。**框架**(frame)是二元组$\mathfrak{F}=(W,R)$，其中W是非空集，称为**可能世界集**，其中的元素成为**世界**(world)或**状态**(state)，而$R\subseteq W\times W$是W上的一个二元关系，称为**可达关系**(accessibility relation)。**模型**(model)是二元组$\mathfrak{M}=(\mathfrak{F},V)$，这里$\mathfrak{F}=(W,R)$是一个框架，而$V:W\to\mathcal{P}(\mathbf{Var})$将每个世界$w\in W$指派$\mathbf{Var}$的一个子集$V(w)$，其直观含义是世界$w$所满足的所有原子性质，$V$称为**赋值函数**(valuation function)。

给定基本模态语言的模型$\mathfrak{M}=(\mathfrak{F},V)$，对任意模态公式$A$和世界$w\in W$，**模型$\mathfrak{M}$在世界$w$满足$A$**，或**$A$在模型$\mathfrak{M}$的世界$w$中为真**，记为$\mathfrak{M},w\Vdash A$，对应地，$\mathfrak{M},w\nVdash\varphi$为**$\varphi$在模型$\mathfrak{M}$的世界$w$中为假**，递归定义为：

1. $\mathfrak{M},w\Vdash p$当且仅当$p\in V(w)$；
2. $\mathfrak{M},w\Vdash\neg A$当且仅当$\mathfrak{M},w\nVdash A$；
3. $\mathfrak{M},w\Vdash A\wedge A'$当且仅当$(\mathfrak{M},w\Vdash A\wedge\mathfrak{M},w\Vdash A')$，类似可给出其他逻辑联结词的语义；
4. $\mathfrak{M},w\Vdash\Diamond A$当且仅当存在$w'\in W$，使得$\langle w,w'\rangle\in R$且$\mathfrak{M},w'\Vdash A$；
5. $\mathfrak{M},w\Vdash\Box A$当且仅当对任意的$w'\in W$，如果$\langle w,w'\rangle\in R$，则$\mathfrak{M},w'\Vdash A$。

例如，设$W=\{a,b,c,d\}$，$R=\{\langle a,b\rangle,\langle c,b\rangle,\langle a,d\rangle,\langle c,d\rangle\}$，假定$\mathbf{Var}=\{p,q\}$，赋值函数$V:W\to\mathcal{P}(\mathbf{Var})$为$V(a)=\{p,q\},V(b)=\{q\},V(c)=\{p\},V(d)=\varnothing$，考虑公式：$K=\Box(p\to q)\to(\Box p\to\Box q)$：

$$\begin{aligned}\mathfrak{M},a\Vdash K&\iff\mathfrak{M},a\Vdash\Box(p\to q)\Longrightarrow\mathfrak{M},a\Vdash\Box p\to\Box q\\\mathfrak{M},a\Vdash\Box(p\to q)&\iff\mathfrak{M},b\Vdash p\to q\wedge\mathfrak{M},d\Vdash p\to q\\\mathfrak{M},b\Vdash(p\to q)&\iff\mathfrak{M},b\Vdash p\Longrightarrow\mathfrak{M},b\Vdash q\\\mathfrak{M},d\Vdash(p\to q)&\iff\mathfrak{M},d\Vdash p\Longrightarrow\mathfrak{M},d\Vdash q\end{aligned}$$

由于$p\notin V(b)$且$p\notin V(d)$，因此确实有$\mathfrak{M},a\Vdash\Box(p\to q)$，而：

$$\begin{aligned}\mathfrak{M},a\Vdash\Box p\to\Box q&\iff\mathfrak{M},a\Vdash\Box p\Longrightarrow\mathfrak{M},a\Vdash\Box q\\\mathfrak{M},a\Vdash\Box p&\iff\mathfrak{M},b\Vdash p\ \wedge\ \mathfrak{M},d\Vdash q\end{aligned}$$

同样由于$p\notin V(b)$且$p\notin V(d)$，因此有$\mathfrak{M},a\nVdash\Box p$，因此有$\mathfrak{M},a\Vdash\Box p\to\Box q$，因此总有$\mathfrak{M},a\Vdash K$。

为方便起见，在不会发生混淆时，我们在上面引入的记号中省略$\mathfrak{M}$，即$\mathfrak{M},w\Vdash\varphi$简记为$w\Vdash\varphi$，而$\mathfrak{M}\Vdash\varphi$简记为$\Vdash\varphi$，以及$\Gamma\Vdash_{\mathfrak{M}}\varphi$简记为$\Gamma\Vdash\varphi$等。

给定模型$\mathfrak{M}=(\mathfrak{F},V)$，对任意模态公式$A$，称**$A$在模型$\mathfrak{M}$中为普遍有效式**，记为$\mathfrak{M}\Vdash A$，如果对任意的$w\in W$有$\mathfrak{M},w\Vdash A$；称**$A$在模型中为可满足式**，如果存在$w\in W$使得$\mathfrak{M},w\Vdash A$。如果公式$A$在任意模型中都是普遍有效式，则称为**普遍有效式**，如果在任意模型中都是可满足式，则称为**可满足式**。不难证明，**命题逻辑中的任意永真式作为模态公式都是普遍有效式**。

设Γ是一组模态公式组成的集合，定义$\mathfrak{M}\Vdash\Gamma$为：$\forall A\in\Gamma$有$\mathfrak{M}\Vdash A$；定义$\Gamma\Vdash_{\mathfrak{M}}A$为：$\mathfrak{M}\Vdash\Gamma$蕴含$\mathfrak{M}\Vdash A$，特别地当$\Gamma=\{B\}$时，记为$B\Vdash A$。给定两个公式$A,B$，如果$A\Vdash B$且$B\Vdash A$，则称**$A$和$B$等值**，记为$A\dashv\vdash B$或$A\Leftrightarrow B$。不难证明，**命题逻辑公式的任意等值式的替换实例仍然是基本模态语言中的等值式**。再例如，设A,B是任意的模态公式，模态逻辑中有如下基本等值式：

$$\begin{aligned}&(1).\ \neg\Box A\Leftrightarrow\Diamond\neg A&&(2).\ \neg\Diamond A\Leftrightarrow\Box\neg A\\&(3).\ \Box(A\wedge B)\Leftrightarrow\Box A\wedge\Box B&&(4).\ \Diamond(A\vee B)\Leftrightarrow\Diamond A\vee\Diamond B\end{aligned}$$

特别地，对任意的模态公式A,B，模态公式$K=\Box(A\to B)\to(\Box A\to\Box B)$是普遍有效式。

8.5.3 模态逻辑的演算系统

上面提到，$K = \Box(A \to B) \to (\Box A \to \Box B)$在任意可能世界语义模型中都是普遍有效式，下面是一些有名的模态公式（模式）：

1. K：$\Box(A \to B) \to (\Box A \to \Box B)$，为纪念克里普克(Kripke)而命名，极小模态逻辑的公理；
2. D：$\Box A \to \Diamond A$，D表示道义的(deontic)，标准道义逻辑的特征公理；
3. T：$\Box A \to A$，R. Feys提出和命名，揭示必然的基本性质，称为必然性公理；
4. 4：$\Box A \to \Box\Box A$，模态系统S4的特征公理；
5. 5(或E)：$\Diamond A \to \Box\Diamond A$，E表示欧几里德，模态系统S5的特征公理；
6. B：$A \to \Box\Diamond A$，布罗维尔系统的特征公理，与直觉逻辑密切相关；

人们通过这些名字来命名常见的一些模态逻辑（及其演算系统），例如：

1. K**系统**：最小的模态逻辑系统，所有的模态系统都是这种模态系统的扩充。K系统由M.Lemmon最早发现，并用Kripke的第一个字母命名；
2. KD**系统**：简称D系统，是标准道义(Deontic)逻辑系统，D公式$\Box A \to \Diamond A$表示“必然的，就是可能的”，或者按照道义逻辑的解释是“应该的，就是允许的”，这描述了道义逻辑的特征；
3. KT**系统**：简称T系统，又称为歌德尔(Gödel)－费依斯(Feys)－冯·赖特系统(Von Wright)，是标准的模态逻辑系统，其特征公理T公式$\Box A \to A$被认为是描述了必然性的基本性质；
4. KT4**系统**：简称S4系统，是T系统的扩充，由C. Lewis等人研究的系统；
5. KTB**系统**：简称B系统，是T系统的扩充，这个名称来源于O. Becker，他发现了B系统余直觉逻辑之间的相似；
6. KT4B**系统**：简称S5系统，是S4系统的扩充。

下面我们只简单介绍模态系统K，对于其他的模态系统，有兴趣的读者可参考文献[19, 20, 21, 22]等。

简单地说，**模态系统K**就是在命题演算系统的基础上增加公理：

$$\mathbf{K}: \Box(A \to B) \to (\Box A \to \Box B)$$

以及推理规则：

$$\mathbf{R}\Box: A \Rightarrow \Box A$$

注意推理规则$\mathbf{R}\Box$与模态公式$A \to \Box A$的区别，前者说，如果A是普遍有效式，则$\Box A$也是普遍有效式，而模态公式$A \to \Box A$并不是普遍有效式（相当于命题逻辑中的永真式），其实这与一阶逻辑中$A(x) \Rightarrow \forall x A(x)$及公式$A(x) \to \forall x A(x)$之间的区别类似。实际上，容易证明，若$A$是普遍有效式，则$\Box A$也是普遍有效式。

下面讨论一些派生规则，以展示模态系统K的基本性质。例如，模态系统K有下述导出规则：

1. $A \to B \Rightarrow \Box A \to \Box B$;
2. $(A \wedge B) \to C \Rightarrow (\Box A \wedge \Box B) \to \Box C$;
3. $(A \leftrightarrow B) \Rightarrow (\Box A \leftrightarrow \Box B)$;
4. $A \to B \Rightarrow \Diamond A \to \Diamond B$;

对于第1个派生规则，可构造如下的证明序列证明：

(1)	$A \to B$	// 前提
(2)	$\Box(A \to B)$	// (1), **R**$\Box$
(3)	$\Box A \to \Box B$	// (2), **K**

而对于第2个派生规则，有如下证明序列：

(1)	$(A \wedge B) \to C$	// 前提
(2)	$\Box(A \wedge B) \to \Box C$	// (1), Dr1
(3)	$(\Box A \wedge \Box B) \to \Box C$	// $\Box(A \wedge B) \Leftrightarrow \Box A \wedge \Box B$

第3个派生规则可由第1个派生规则得到，对于第4个派生规则，有如下证明序列：

(1)	$\Box(\neg B \to \neg A) \to (\Box \neg B \to \Box \neg A)$	// **K**
(2)	$\Box(A \to B) \to (\neg \Box \neg A \to \neg \Box \neg B)$	// $A \to B \Leftrightarrow \neg B \to \neg A$
(3)	$\Box(A \to B) \to (\Diamond A \to \Diamond B)$	// 传递规则，及$\neg \Box \neg A \Leftrightarrow \Diamond A$
(4)	$A \to B$	// 前提
(5)	$\Box(A \to B)$	// **R**$\Box$
(6)	$\Diamond A \to \Diamond B$	// (3), 传递规则

进一步，读者可自行利用命题逻辑的推理规则，加上上述公理**K**及推理规则**R**$\Box$，可构造证明序列验证下面的公式都是系统K的定理：

(1). $\Box A \vee \Box B \to \Box(A \vee B)$

(2). $\Diamond(A \wedge B) \to \Diamond A \wedge \Diamond B$

(3). $\Box(\neg A \to A) \leftrightarrow \Box A$

(4). $\Box(A \to \neg A) \leftrightarrow \Box \neg A$

(5). $\Box(B \to A) \wedge \Box(\neg B \to A) \leftrightarrow \Box A$

(6). $\Box(A \to B) \wedge \Box(A \to \neg B) \leftrightarrow \Box \neg A$

(7). $\Box A \to \Box(B \to A)$

(8). $\Box \neg A \to \Box\Box(A \to B)$

实际上，**公式A是系统K的定理当且仅当它在任意模型中都是普遍有效式**，这是系统K的完备性和合理性。

8.6 小结

上面我们从每个**逻辑系统的提出**、**基本概念**、**命题公式及其真值**、**公式等值的基本含义**，以及**推理演算**等几个方面简单介绍了直觉逻辑、多值逻辑、模糊逻辑和模态逻辑，其中，由于多值逻辑和模糊逻辑演算系统的复杂性，我们主要是介绍了命题公式及联结词在这些逻辑系统中的真值运算规则，而没有涉及其推理演算系统。

直觉逻辑、多值逻辑、模糊逻辑以及模态逻辑等都是非经典逻辑，至少具有一个与经典逻辑不同的基本特性：

1. 直觉逻辑从直觉主义的基本观点出发，对命题真值给出了不同的解释，认为命题为真，当且仅当有可行的方法证明其为真；

2. 多值逻辑和模糊逻辑则扩充了命题的真值的取值范围：在多值逻辑中，命题的真值可取多个确定的值，而在模糊逻辑中，命题的真值可取$[0,1]$区间的任意值，甚至取$[0,1]$区间的子区间作为命题的真值。

3. 模态逻辑则是经典逻辑的扩充，增加了对模态算子的处理。模态算子也是从命题构造新命题的方法，但与逻辑联结词不同：逻辑联结词是真值函数，复合命题的真值由支命题惟一确定，而模态算子不是真值函数，具有自己的内涵，复合命题的真值不能由模态算子所作用的支命题惟一确定。

作业

作业 8.1 试谈谈你对直觉主义的基本观点的理解，并讨论直觉逻辑与计算机科学之间有何密切关系。

作业 8.2 试给出公式$\neg\neg A \to A$在直觉逻辑中的解释，并谈谈为什么在这种解释下它不是永真式？

作业 8.3 请讨论卢卡西维茨的三值逻辑系统、克林的三值逻辑系统，以及布奇瓦尔的三值逻辑系统之间的异同。

作业 8.4 请讨论模态算子$\Box$和$\Diamond$的不同解释。

作业 8.5 证明在模态逻辑中有如下等值式：

(1). $\neg\Box A \Leftrightarrow \Diamond\neg A$　　(2). $\neg\Diamond A \Leftrightarrow \Box\neg A$

(3). $\Box(A \wedge B) \Leftrightarrow \Box A \wedge \Box B$　　(4). $\Diamond(A \vee B) \Leftrightarrow \Diamond A \vee \Diamond B$

参考文献

[1] 吴家国. 普通逻辑原理. 高等教育出版社, 2000. 全国高等教育自学考试指定教材.

[2] 陈慕泽. 数理逻辑教程. 上海人民出版社, 2001.

[3] 耿素云、屈婉玲. 离散数学（修订版）. 高等教育出版社, 2004年1月.

[4] 石纯一、王家□. 数理逻辑与集合论（第二版）. 清华大学出版社, 2000.

[5] 陆钟万. 面向计算机科学的数理逻辑. 科学出版社, 1998.

[6] 王捍贫. 数理逻辑–离散数学第一分册. 北京大学出版社, 1997.

[7] 王宪钧. 数理逻辑引论. 北京大学出版社, 1998.

[8] 毕富生. 数理逻辑. 高等教育出版社, 2004.

[9] 赵总宽、陈慕泽、杨武金. 现代逻辑方法论. 中国人民大学出版社, 1998.

[10] 陈波. 逻辑哲学引论. 中国人民大学出版社, 2000.

[11] 张清宇、郭世铭、李小五. 哲学逻辑研究. 社会科学文献出版社, 1997.

[12] M. H. Sørensen and P. Urzyczyn. *Lectures on the Curry-Howard Isomorphism*. Number 149 in Sutdies in Logic and the Foundations of Mathematics. Elsevier, 2006. Draft is available in: http://citeseer.ist.psu.edu/519604.html.

[13] 周晓聪，李文军，李师贤. 类型系统的研究与进展. 计算机科学, 27(5):5–13, 2000.

[14] 马振华主编. 现代应用数学手册 · 离散数学卷. 清华大学出版社, 2002.

[15] 刘增良、刘有才. 模糊逻辑与神经网络–理论研究与探索. 北京航空航天大学出版社, 1996.

[16] 刘普寅、吴孟达. 模糊理论及其应用. 国防科技大学出版社, 1998.

[17] L. A. 扎德. 模糊集合、语言变量及模糊逻辑. 科学出版社, 1982.

[18] M. Huth and M. Ryan. *Logic in Computer Science: Modelling and Reasoning about System*. Cambridge University Press, second edition, 2004.

[19] 弓肇祥. 广义模态逻辑. 中国社会科学出版社, 1993.

[20] B. F. Chellas. *Modal Logic: An Introduction*. Cambridge University Press, 1980. 郑文辉、张宜生译. 林铭钧校. 模态逻辑导论. 中山大学出版社. 1989.

[21] 周礼全. 模态逻辑引论. 上海人民出版社, 1986.

[22] 周北海. 模态逻辑. 中国社会科学文献出版社, 1996.