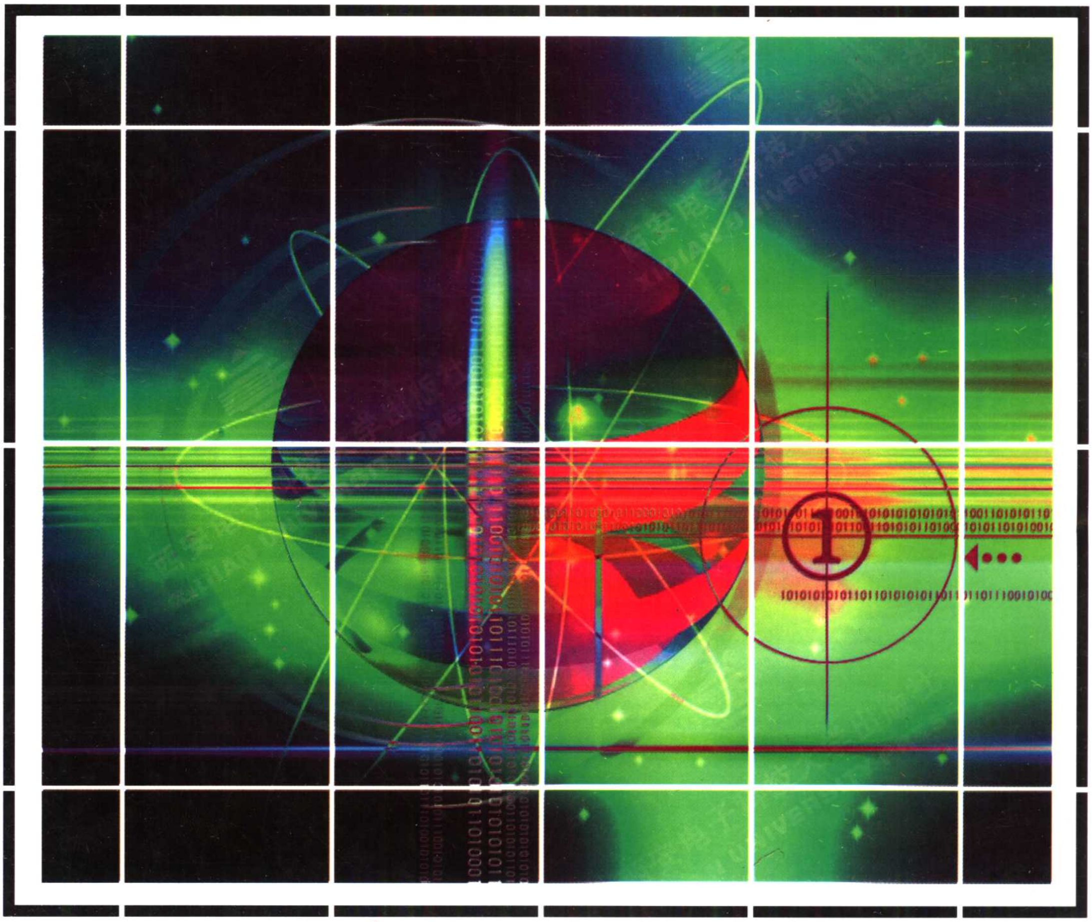
新世纪计算机类本科规划教材
COMPUTER

离散数学

武 波 黄健斌
尹忠海 毛立强 编著



西安电子科技大学出版社
http://www.xduph.com

新世纪计算机类本科规划教材

离散数学

武波　黄健斌　尹忠海　毛立强　编著

西安电子科技大学出版社

2007

内容简介

本书系统地介绍了离散数学的理论和方法。全书共7章，内容包括数理逻辑、集合与关系、代数系统和图论四部分。书中除对概念、性质及方法进行了严密的论述外，还精选了大量例题，便于读者理解书中理论的内涵及其应用。每一节最后精选了与本节重点内容相关的典型习题，并且配有部分英文习题，以便读者通过练习巩固已学的知识。

本书可作为高等院校计算机科学与技术、软件工程以及相关本科专业的离散数学教材，也可以作为其他需要学习离散数学的工作人员的参考读物。

图书在版编目(CIP)数据

离散数学/武波等编著．—西安：西安电子科技大学出版社，2007.10
新世纪计算机类本科规划教材
ISBN 978-7-5606-1920-0
Ⅰ.离… Ⅱ.武… Ⅲ.离散数学—高等学校—教材 Ⅳ.O158

中国版本图书馆 CIP 数据核字(2007)第 142119 号

策　　划　臧延新　陈宇光
责任编辑　许青青　臧延新　陈宇光
出版发行　西安电子科技大学出版社(西安市太白南路2号)
电　　话　(029)88242885　88201467　　邮　编　710071
http://www.xduph.com　　E-mail：xdupfxb@pub.xaonline.com
经　　销　新华书店
印刷单位　陕西华沐印刷科技有限责任公司
版　　次　2007年10月第1版　2007年10月第1次印刷
开　　本　787毫米×1092毫米　1/16　印张 18.125
字　　数　429千字
印　　数　1～4000册
定　　价　24.00元
ISBN 978-7-5606-1920-0/O·0087
XDUP 2212001-1

前　言

离散数学是随着计算机科学的发展和计算机应用的日趋广泛而建立起来的一个数学分支，它为计算机科学技术和工程应用提供了有力的理论工具，其中涉及的概念、原理和方法在计算机及相关学科领域都有着重要应用。

离散数学作为计算机专业的一门核心数学课程，它为学习高级语言程序设计、数据结构、数字逻辑设计、操作系统、数据库原理、编译原理、计算机网络、人工智能、信息安全等专业课程提供了必要的数学基础。同时，离散数学对于培养学生的抽象思维、逻辑推理能力、用数学模型分析和解决问题的能力等均具有十分重要的作用。本课程不但要培养学生的抽象思维能力，而且更要培养学生运用数学方法解决实际问题的能力。然而，从作者多年讲授离散数学的效果来看，有不少学生在学习完抽象的数学理论后，对于其中相关理论和方法的学习目标和作用并不明确，也不知道这些理论的真正作用。许多专业课教师也反映学生在学习专业课程中不能把离散数学的理论知识与实际应用对应起来，学生应用数学工具的能力较差，这是离散数学教学的一个亟待改进的问题。

为此，我们在编写本教材时，在内容取材和写作风格上作了相应整合与尝试。

(1) 对于学生必须掌握的重要内容，如数理逻辑、关系、函数、图论、组合计数、布尔代数，给予强化和突出，精炼了群环域方面的内容；

(2) 通过精选大量的实例来引导学生对基本理论和方法的实际用途的理解；

(3) 为了配合软件工程专业的双语教学，我们对书中的基本概念都给以英文注释，而且在每节后面增加了英文习题，促使学生掌握离散数学中的基本英文术语并提高英文阅读能力；

(4) 增加了一些扩展性内容引导学生阅读自学。

全书共分为 7 章，每一章根据内容又分为若干节，各节后均配备了大量相关习题。其中，第 1、2 章数理逻辑部分由毛立强编写，第 3、4 章集合论部分由黄健斌编写，第 5、6 章代数系统部分由尹忠海编写，第 7 章图论部分由武波编写。全书内容讲授约需 70 学时，其中一些较深奥的内容在目录中以“ * ”标注，以供授课教师取舍。

本书的出版得到了西安电子科技大学教材基金的资助。感谢西安电子科技大学出版社的大力支持，感谢臧延新和许青青编辑为本书的出版所做的大量工作！

由于作者理论实践水平有限，加之时间仓促，书中的不足之处在所难免，敬请读者不吝指正。

编　者

2007 年 7 月

目　　录

第1章 命题逻辑

逻辑学是一门研究思维的形式及推理的科学，传统的逻辑学是哲学家们的研究课题。1666 年，德国数学家莱布尼兹(Gottfried Leibniz)在《论组合的艺术》一书中首先提出了数理逻辑的思想，即通过引入一套符号体系来研究思维的形式及推理，所以数理逻辑又称为符号逻辑。19 世纪中期，英国数学家乔治 · 布尔(George Boole)创立了逻辑代数，从而使得数理逻辑的研究得到了很大发展。1893 年德国数学家弗雷格(Friedrich Ludwig Gottlob Frege)出版了《算术基本规律》一书，标志着一套完善的符号逻辑系统——命题逻辑的诞生。现代数理逻辑的分支很多，包括证明论、公理集合论、递归论、模型论、多值逻辑、模态逻辑、时态逻辑等。数理逻辑学与计算机科学密切相关，它在关系数据库、程序设计的自动化和验证、人工智能、有限自动机理论等领域都有重要应用。

本章主要讨论命题逻辑的基本概念和理论，重点讨论如何利用数学的方法来研究推理(reasoning)的形式和方法。本章的主要内容包括命题、命题公式、重言式和矛盾式、逻辑等价与永真蕴涵式、主范式和命题逻辑的推理理论。

1.1 命题和联结词

1.1.1 命题

命题逻辑研究以命题(proposition)为基本单位构成的前提(premises)和结论(conclusion)之间的逻辑关系。例如，"如果我努力学习，那么我就不会离散数学考试不及格。如果我不上网吧，那么我将努力学习。期末成绩出来，我的离散数学成绩不及格，因此，我上网吧。"显然，该逻辑推理语句中的前提和结论都是表达判断且具有确定真假值的陈述句，它们构成了推理的基本单位。为进一步讨论由前提到结论的论证(argument)过程及所用的相关规则，我们在本节首先抽象出命题的概念并进一步探讨命题的复合。

定义 1.1.1 一个或真或假但不能两者都是的断言称为命题。

命题的定义表明一个命题必须满足以下两个条件：

(1) 命题是表达判断的陈述句；

(2) 命题有确定的真假值，它的真值或者为真，或者为假，两者必居其一。

因此，疑问句、祈使句和感叹句等都不是命题，一个陈述句如果不能明确判断其真假，则它也不是命题。

如果一个命题为真，则称其"真值(truth value)"为真，用大写字母 T 或数字 1 表示；

如果一个命题为假，则称其“真值”为假，用大写字母 F 或数字 0 表示。

例 1 判断下列句子哪些是命题，给出真值；哪些不是命题，说明原因。

(a) 能整除 7 的正整数只有 1 和 7 本身。

(b) 小明出生那天，北京下雨。

(c) 雪是黑的。

(d) $x+y=3$。

(e) 2 是偶数，而 1 是奇数。

(f) $1+101=110$。

(g) 明天是否去春游啊?

(h) 全体起立!

(i) 我在说谎。

解 (a) 是命题，真值为真，表明 7 是素数。

(b) 是命题，真值由小明出生那天北京是否下雨而唯一确定。

(c) 是命题，真值为假，因为雪是白色的。

(d) 不是命题，因为 x 与 y 为变量，无法确定真假。

(e) 是命题，真值为真。

(f) 不是命题，因为在二进制中为真，在十进制中为假，需要根据上下文才能确定真值。

(g) 不是命题，该句不是陈述句。

(h) 不是命题，该句不是陈述句。

(i) 不是命题，他是在说谎还是在说真话呢? 如果他在说谎，那么他说的是假话，因为他承认他是在说谎，所以他实际上说的是真话，即结论是如果他是在说谎，那么他说的是真话；另一方面，如果他说真话，那么他说的是真话，也就是他在说谎，所以结论是如果他说真话，那么他在说谎。因此，没有办法确定该句子的真值，像这样的陈述句称为“悖论(paradox)”。

命题有两种类型：一种是不能分解成更简单的命题，称为原子命题(simple proposition)，如例 1 中的(a)、(b)和(c)；另一种是由原子命题、命题联结词和圆括号复合构成的命题，称为复合命题(compound proposition)，如例 1 中的(e)。

就像代数中用字母表示数字一样，在数理逻辑中一般使用大写字母 A，B，…，P，Q，…表示命题。例如，

P：2 是偶数。

这里用 P 来表示“2 是偶数”这个命题。

1.1.2 联结词

在一般的口语和书面语言中，常使用“和”、“或”、“如果……，那么……”等一些联结词来构成更复杂的复合命题。例如，

P：天正在下雨，Q：天很冷。

这是两个命题，利用联结词“不”、“并且”和“或”可以分别构成新命题：

“天不是很冷”。

"天正在下雨并且天很冷"。

"天正在下雨或天很冷"。

即 "非 Q"。

"P 并且 Q"。

"P 或 Q"。

在代数中，用"+"、"×"等运算符连接数字得到代数表达式，如 3+2。同样，在数理逻辑中，也存在这样的运算符，称为逻辑联结词(logic connective)，简称联结词(connective)。常用的联结词有 5 个：否定、合取、析取、条件和双条件。

1. 否定

定义 1.1.2 设 P 为一命题，P 的否定(negation)是一个新的命题，记做 $\neg P$，称为"非 P"。$\neg P$ 是真，当且仅当 P 为假；$\neg P$ 是假，当且仅当 P 为真。

这里引入真值表(truth table)来描述应用联结词的命题的真值结果。真值表的左边列出参与运算的命题真值的所有可能组合，所得命题的真值结果列在最右边的一列。于是，否定联结词的定义如表 1.1.1 所示。

表 1.1.1

P	$\neg P$
0	1
1	0

否定联结词也称为"非"运算，将每个命题 P 指派成 $\neg P$，它是对单个命题进行操作，是一个一元运算。

例 2 (a) P：2 是偶数。

$\neg P$：2 不是偶数。或者 2 是偶数，并非如此。

(b) Q：这些书都是刚刚出版的。

$\neg Q$：这些书不都是刚刚出版的。

注意：表示成"这些书都不是刚刚出版的"是错误的。

例 3 在大多数编程语言中，"非"的定义和定义 1.1.1 相同。例如在 Java 语言中，(逻辑)"非"记做!，表达式!($x<100$)为真当且仅当变量 x 不小于 100，也就是大于等于 100。

2. 合取

定义 1.1.3 如果 P 和 Q 是命题，那么"P 并且 Q"是一个复合命题，记做 $P\wedge Q$，称为 P 和 Q 的合取(conjunction)。当且仅当 P、Q 同时为 T 时，$P\wedge Q$ 为 T，否则，$P\wedge Q$ 为 F。

合取联结词的定义如表 1.1.2 所示。

表 1.1.2

P	Q	$P\wedge Q$
0	0	0
0	1	0
1	0	0
1	1	1

"合取"是一个二元运算。

例 4 如果 P：天正在下雨，Q：天很冷，那么 P 和 Q 的合取为

$P \wedge Q$：天正在下雨并且天很冷。

只有当"天正在下雨"和"天很冷"都为真时，$P \wedge Q$ 才是真。

由此可见，合取的概念和自然语言中的"与"意义相似，但也并不完全相同。例如，P：地球是圆的，Q：我去看电影，则 $P \wedge Q$ 表示"地球是圆的与我去看电影"。在自然语言中，这个命题是没有意义的，因为 P 和 Q 没有内在联系。但在数理逻辑中，P 和 Q 没有内在联系也是可以的，只要按照定义，P、Q 的真值确定后，$P \wedge Q$ 的真值就随之确定了。

例 5 在大多数编程语言中，"与"的定义与合取定义相同。例如在 Java 语言中，(逻辑)"与"记做 &&，表达式 $x<10$ && $y>1$ 为真当且仅当变量 x 小于 10 并且变量 y 大于 1。

3. 析取

定义 1.1.4 如果 P 和 Q 是命题，那么"P 或 Q"是一个复合命题，记做 $P \vee Q$，称为 P 和 Q 的析取(disjunction)。当且仅当 P、Q 至少有一个为 T 时，$P \vee Q$ 为 T，否则，$P \vee Q$ 为 F。

析取联结词的定义如表 1.1.3 所示。

表 1.1.3

P	Q	$P \vee Q$
0	0	0
0	1	1
1	0	1
1	1	1

"析取"是一个二元运算。

例 6 如果 P：天正在下雨，Q：天很冷，那么 P 和 Q 的析取为

$P \vee Q$：天正在下雨或天很冷。

只要"天正在下雨"为真，或"天很冷"为真，或两者同时为真，$P \vee Q$ 就为真。

由此可见，析取的概念和自然语言中的"或"意义相似，但也并不完全相同。自然语言中"或"常见的含义有两种：一种是"可兼或(inclusive-or)"，如例 6，存在天正在下雨同时天很冷这种情况；另一种是"不可兼或(exclusive-or)"，例如"人固有一死，或重于泰山，或轻于鸿毛"中的"或"，它表示非此即彼，不可兼得。析取表示可兼或，不可兼或用另外的符号表示(将在 1.4 节给出)。与合取类似，析取也可以联结两个没有内在联系的命题。

例 7 在大多数编程语言中，"兼或"与析取的定义相同。例如在 Java 语言中，(逻辑)"或"记做 ||，表达式 $x<10$ || $y>1$ 为真当且仅当变量 x 小于 10 或者变量 y 大于 1 或者两者同时成立。

例 8 Web 搜索。许多 Web 搜索引擎(如 Google、Yahoo)都允许用户输入关键词，然后由搜索引擎与网页进行匹配。例如，输入 mathematics 会产生一个包含 mathematics 的列表。有些搜索引擎允许用户使用操作符 AND、OR 和 NOT 以及括号进行关键词的组合，

这样可以实现更复杂的搜索。例如，为了搜索包含关键词“discrete”和“mathematics”的网页，用户应该输入 discrete AND mathematics。如果搜索包含关键词“discrete”和“mathematics”或关键词“finite”和“mathematics”的网页，则用户应该输入(discrete OR finite) AND mathematics。

4. 条件

定义 1.1.5 如果 P 和 Q 是命题，那么“如果 P，则 Q”是一个复合命题，记做 $P\rightarrow Q$，称为 P 和 Q 的条件命题(conditional proposition)。当且仅当 P 为 T，Q 为 F 时，$P\rightarrow Q$ 为 F，否则，$P\rightarrow Q$ 为 T。这里，称 P 为假设(hypothesis)或前件(antecedent)，称 Q 为结论(conclusion)或后件(consequent)。

条件联结词的定义如表 1.1.4 所示。

表 1.1.4

P	Q	$P\rightarrow Q$
0	0	1
0	1	1
1	0	0
1	1	1

“条件”是一个二元运算。

例 9 (a) P：天不下雨，Q：草木枯黄。

$P\rightarrow Q$：如果天不下雨，那么草木枯黄。

只有当天不下雨(P 为 T)，草木不枯黄(Q 为 F)时，$P\rightarrow Q$ 才为 F，其他情况下都为 T。

(b) P：地球是宇宙的中心，Q：我不去上课。

$P\rightarrow Q$：如果地球是宇宙的中心，那么我不去上课。

因为 P 为 F，如果我不去上课(Q 为 T)，那么 $P\rightarrow Q$ 为 T；如果我去上课(Q 为 F)，那么 $P\rightarrow Q$ 仍为 T。

在自然语言中，“如果……”与“那么……”之间是有因果关系的，否则就没有意义，如例 9 中的(a)。而在(b)中，“地球是宇宙的中心”与“我不去上课”之间没有必然联系，当前件 P 为 F 时，不管后件 Q 为 T 还是为 F，“如果 P，那么 Q”这种自然语言描述往往无法判断。在条件命题中，规定为“善意的推定”，即前件为 F 时，条件命题都为 T。这样定义在讨论逻辑和数学问题中不仅是正确的，而且应用起来十分方便。

条件命题 $P\rightarrow Q$ 可以有多种方式描述，例如：

“若 P，则 Q”。

“P 仅当 Q”。

“Q 每当 P”。

“P 是 Q 的充分条件(sufficient condition)”。

“Q 是 P 的必要条件(necessary condition)”。

给定条件命题 $P\rightarrow Q$，把 $Q\rightarrow P$，$\neg P\rightarrow\neg Q$，$\neg Q\rightarrow\neg P$ 分别称为它的逆命题(converse)、否命题(inverse)和逆否命题(contrapositive or transposition)。

5. 双条件

定义 1.1.6 如果 P 和 Q 是命题，那么"P 当且仅当 Q"是一个复合命题，记做 $P \leftrightarrow Q$，称为 P 和 Q 的双条件命题(biconditional proposition)。当且仅当 P 和 Q 的真值相同时，$P \leftrightarrow Q$ 为 T，否则，$P \leftrightarrow Q$ 为 F。

双条件联结词的定义如表 1.1.5 所示。

表 1.1.5

P	Q	$P \leftrightarrow Q$
0	0	1
0	1	0
1	0	0
1	1	1

由表 1.1.5 可见，如果 $P \leftrightarrow Q$ 为 T，那么 $P \rightarrow Q$ 和 $Q \rightarrow P$ 都为 T；反之亦然。所以$P \leftrightarrow Q$ 也可以称为"P 和 Q 互为充要条件"。

"双条件"也可记做"iff"，是一个二元运算。

例 10 (a) P：两个三角形全等，Q：两个三角形的三组对应边相等。

$P \leftrightarrow Q$：两个三角形全等，当且仅当两个三角形的三组对应边相等。

(b) P：地球是圆的，Q：雪是白的。

$P \leftrightarrow Q$：地球是圆的，当且仅当雪是白的。

由此可见，双条件命题与条件命题一样，也可以不管其因果关系。

由以上 5 个联结词的定义可以看出，联结词的意义由其真值表唯一决定，而不由命题的具体含义决定。

习　题

1.1-1 指出下列语句哪些是命题，哪些不是命题，如果是命题，指出它的真值。

(a) 纽约是中国的一座城市。

(b) 明天我们去郊游。

(c) 你今天有空吗?

(d) 上课时请不要大声喧哗!

(e) 月球上有人。

(f) 如果我学习了 C 语言，那么我就可以编程了。

(g) $10+5 \leqslant 12$。

(h) $6x+3=5-7x$。

(i) 我们要学好离散数学。

(j) 小明和小强是兄弟。

1.1-2 写出"如果天不下雨且我有时间，那么我去郊游"的逆命题、否命题和逆否命题。

1.1-3 Which of sentences (a)～(e) are either true or false (but not both)?

(a) The only positive integers that divide 7 are 1 and 7 itself.

(b) Edsgar Wybe Dijkstra won an ACM Turing Award in 1974.

(c) For every positive integer n, there is a prime number larger than n.

(d) Earth is the only planet in the universe that has life.

(e) Buy two tickets to U2 rock concert for Friday.

1.1－4 Write a command to search the Web for minor league base-ball teams in Illinois that are not in the Midwest League.

1.2 命题公式

1.2.1 命题公式的定义

在命题逻辑中，一个确定的具体的命题，其真值不是F就是T，我们将真值F和T称为命题常元(propositional constant)；一个不确定的泛指的任意命题，称为命题变元(propositional variable)。

由于在命题逻辑中并不关心具体命题的涵义，只关心其真值，为清楚起见，我们约定命题变元用字母表示，命题常元用T或1、F或0表示，因此可以形式地定义它们如下：

定义 1.2.1 以{T, F}或{1, 0}为其变域的变元，称为命题变元；T或1、F或0称为命题常元。

前面已经提到，命题可以分为原子命题和复合命题。设 P 和 Q 是任意两个命题，应用前面5个联结词，就可以得到新的复合命题，如$(\neg P)$、$(P \wedge Q)$、$((P \wedge Q) \rightarrow Q)$、$(Q \leftrightarrow (P \vee Q))$等。如果 P 和 Q 是命题变元，则前述各式均称为命题公式，P 和 Q 称为命题公式的分量。

并不是所有由命题变元、联结词和一些括号组成的字符串都能称为命题公式，为此，计算机科学中常用以下归纳方法进行定义。

定义 1.2.2 命题公式：

(1)(基础)单个命题变元或命题常元是命题公式。

(2)(归纳)如果 A 和 B 是命题公式，则$(\neg A)$、$(A \wedge B)$、$(A \vee B)$、$(A \rightarrow B)$、$(A \leftrightarrow B)$是命题公式。

(3)(极小性)只有有限次应用条款(1)和(2)生成的公式才是命题公式。

这种定义称为递归定义，也称为归纳定义，由这种定义得到的公式称为合式公式(Well-Formed Formula, WFF)。

例 1 (a) 验证$((P \wedge Q) \rightarrow ((\neg P) \vee (P \leftrightarrow Q)))$是命题公式。

(i) P 是命题公式，根据条款(1)；

(ii) Q 是命题公式，根据条款(1)；

(iii)$(P \wedge Q)$是命题公式，根据(i)、(ii)和条款(2)；

(iv)$(\neg P)$是命题公式，根据(i)和条款(2)；

(v) $(P \leftrightarrow Q)$是命题公式，根据(i)、(ii)和条款(2)；

(vi) $((\neg P) \vee (P \leftrightarrow Q))$是命题公式，根据(iv)、(v)和条款(2)；

(vii) $((P \wedge Q) \rightarrow ((\neg P) \vee (P \leftrightarrow Q)))$是命题公式，根据(iii)、(vi)和条款(2)。

其构造过程如图 1.2.1 所示。

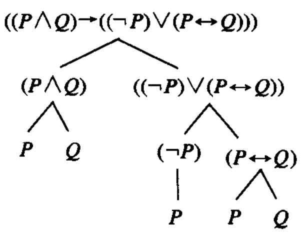


图 1.2.1

(b) 以下都不是命题公式，因为都不能通过定义1.2.2 的条款生成。

$\wedge B$、$(A\wedge B$、$(AB\rightarrow A)$、$A\rightarrow B\rightarrow C$。

为了减少括号的使用，可以做以下约定：

- 联结词运算的优先次序为：$\neg$、$\wedge$、$\vee$、$\rightarrow$、$\leftrightarrow$，符合此顺序的，括号可省去；
- 相同的联结词，按从左至右顺序计算时，括号可省去；
- 最外层的括号可省去。

有了联结词和命题公式的概念，我们还可以把自然语言描述的命题翻译成数理逻辑中的符号形式，这也是以后进行逻辑推理的基础。

定义 1.2.3 把一个用文字叙述的命题相应地写成由命题标识符、联结词和圆括号表示的合式公式，称为命题的符号化。

例 2 (a) 他虽然不聪明，但很用功。

这句话虽然没有出现前面几个联结词的表达形式，但实际意思是：他不聪明，并且他用功。于是提取出原子命题，设 P：他聪明，Q：他用功，则该句可翻译为：$\neg P\wedge Q$。

(b) 除非你努力，否则你这次考试将不及格。

这句话可以理解为：如果你不努力，那么你这次考试将不及格。设 P：你努力，Q：你这次考试将不及格，则该句翻译为：$\neg P\rightarrow Q$。

(c) 如果明天不下雨并且明天不下雪，则我去上课。

设 P：明天下雨，Q：明天下雪，R：我去上课，则翻译为$(\neg P\wedge\neg Q)\rightarrow R$。

(d) 如果明天不是雨夹雪，则我去上课。

设原子命题同(c)，则翻译为：$\neg(P\wedge Q)\rightarrow R$。

(e) 如果明天下雨或者下雪，则我不去上课。

设原子命题同(c)，则翻译为：$(P\vee Q)\rightarrow\neg R$。

(f) 当且仅当明天不下雪并且不下雨时，我才去上课。

设原子命题同(c)，则翻译为：$(\neg P\wedge\neg Q)\leftrightarrow R$。

(g) 明天，我将风雪无阻一定去上课。

设原子命题同(c)，则翻译为：$(P\wedge Q\wedge R)\vee(\neg P\wedge Q\wedge R)\vee(P\wedge\neg Q\wedge R)\vee(\neg P\wedge\neg Q\wedge R)$。

从这些例子可以看出，翻译时要按逻辑关系翻译，分析自然语言中各种联结词的具体含义，不能只凭字面翻译。如，设 P：小明去上课，Q：小强去上课，则“小明和小强都去上课”可翻译为 $P\wedge Q$，但“小明和小强是同学”就不能翻译成两个命题的合取，它是一个原子命题。

1.2.2 真值表

应该注意的是，命题公式没有确定的真值，所以命题公式不是命题，命题公式的真值取决于其所含命题变元的真值。为此，我们需要首先引入指派(assign)的概念。

定义 1.2.4 公式中所有命题变元的一组确定的真值称为该公式的一组真值指派(也可称为解释)。

用归纳法不难证明，对于含有 n 个命题变元的公式，由于每个命题变元可以有 F、T 两个不同指派，n 个命题变元的命题公式有 2^n 个不同指派，因此该公式的真值表中有 2^n 行。

为方便构造命题公式的真值表，对含 n 个命题变元的命题公式可以作如下约定：

(1) 将公式中出现的 n 个命题变元按字母升序或降序排列。

(2) 对 2^n 个不同指派，按其对应的二进制比特串以二进制数从小到大或从大到小的顺序排列。

(3) 若公式较复杂，则可先列出各子公式的真值(若有括号，则应从里层向外层展开)，最后列出所求公式的真值。

例 3 构造公式 $(P \wedge \neg P) \leftrightarrow (Q \wedge \neg Q)$ 的真值表。

解 公式 $(P \wedge \neg P) \leftrightarrow (Q \wedge \neg Q)$ 的真值表如表 1.2.1 所示。

表 1.2.1

P	Q	$(P \wedge \neg P)$	$(Q \wedge \neg Q)$	$(P \wedge \neg P) \leftrightarrow (Q \wedge \neg Q)$
0	0	0	0	1
0	1	0	0	1
1	0	0	0	1
1	1	0	0	1

例 4 构造公式 $(P \wedge Q) \wedge \neg Q$ 的真值表。

解 公式 $(P \wedge Q) \wedge \neg Q$ 的真值表如表 1.2.2 所示。

表 1.2.2

P	Q	$(P \wedge Q)$	$\neg Q$	$(P \wedge Q) \wedge \neg Q$
0	0	0	1	0
0	1	0	0	0
1	0	0	1	0
1	1	1	0	0

例 5 构造公式 $(P \rightarrow R) \vee (Q \rightarrow R)$ 的真值表。

解 公式 $(P \rightarrow R) \vee (Q \rightarrow R)$ 的真值表如表 1.2.3 所示。

表 1.2.3

P	Q	R	$P \rightarrow R$	$Q \rightarrow R$	$(P \rightarrow R) \vee (Q \rightarrow R)$
0	0	0	1	1	1
0	0	1	1	1	1
0	1	0	1	0	1
0	1	1	1	1	1
1	0	0	0	1	1
1	0	1	1	1	1
1	1	0	0	0	0
1	1	1	1	1	1