人工智能丛书

Prolog高级程序设计技巧

裴 琪 主编

西北工业大学出版社

Prolog高级程序设计技巧

人工智能丛书

Prolog 高级程序设计技巧

裴 琨 主编

裴 琨 马 莎 李连生 编著

西北工业大学出版社

1991年8月 西安

内 容 简 介

本书全面地介绍了逻辑编程语言 Prolog 及其程序设计技巧．它阐明了逻辑程序的基础、逻辑程序和 Prolog 语言，并着重讨论了 Prolog 高级编程技术，最后给出了 Prolog 语言的应用实例．

全书共五篇(二十四章)．一、逻辑程序的基础——数理逻辑；二、逻辑程序；三、Prolog 语言；四、Prolog 高级程序设计技术；五、应用．

本书可作为高等学校计算机类、电子、自动化、管理、机械等有关专业研究生和高年级本科生的必修课或选修课教材，也可供从事人工智能和计算机应用的科技人员学习和参考使用．

人工智能丛书
Prolog 高级程序设计技巧
主　　编　　裴　琪
责任编辑　　柴文强
责任校对　　杨长照
*
西北工业大学出版社出版发行
(西安市友谊西路 127 号)
全国各地 新华书店 经销
陕西省富平县印刷厂印装
ISBN 7-5612-0366-7 / TP.54
*
开本 787×1092 毫米　1/16　20.75　印张　499 千字
1991 年 8 月第 1 版　　1991 年 8 月第 1 次印刷
印数：00001—2000 册　　定价：8.55 元

前　言

1982 年当我在北京航空学院计算机系任教的时候，有幸从国外留学归来人员手中拿到了介绍 Prolog 语言的一本经典著作《Programming in Prolog》（W.F.Clocksin，C.S.Mellish 著）。恰巧那时候我在讲授数理逻辑和编程科学（The Science of Programming Gries 著）中的形式化程序推导。逻辑语言可以成为实际的编程语言，把一种精确的数学语言稍加变动就可以在机器上运行，这确实是一件令人兴奋的事。逻辑编程的简洁、新颖和描述性等特点使我感到极大的兴趣。一种清晰的表达思维和推理的工具既可以用来作程序规范说明，又可以作为实际运行的程序，这大大地吸引了我。我开始学习和研究逻辑编程和 Prolog 语言。1983 年上半年我就为北航 79 届学生（毕业班）讲授"Prolog 语言"、"Prolog 系统实现"等两门课程。Prolog 语言课的教材就是取自《Programming in Prolog》一书，并用 Prolog 系统实现和 Prolog 语言为题，指导了 3 名毕业生的毕业设计。同时我和赵沁平老师和高峰同学在当时的 VT-60 机上用 LISP 语言实现了一个小型的 Prolog 解释系统——L-Prolog 系统。这个系统和现在商用的 Prolog 语言软件相比自然是不足为道的。但那时候全国尚无一个单位从事 Prolog 语言的研究，也没有 Prolog 系统的实现，因此，我们率先的工作意义还是很大的。它对我们学习和理解 Prolog 语言及其推理起了很大的作用。1983 年 7 月北航计算机系 603 教研室召开了 Prolog 语言全国讨论班，在会上我们介绍了 Prolog 语言及我们研制的小型 Prolog 解释系统。讨论班推动了 Prolog 语言在全国的研究和推广。在这以后，一些大学、研究所和公司才开始竞相开展这方面的研究，产生了许多很好的实现系统和有价值的成果。北京航空航天大学的孙怀民老师对 Prolog 语言在我国的开创研究和推广也作了许多有益的工作，起了重要的作用。后来在他的指导下不少同志在这方面继续作了许多有意义的工作，我很高兴地看到 Prolog 语言在我国生根、开花、结果。

1985 年我离开北航去自动化工程学院任教并担任教学副院长。由于要在从事教学行政工作的同时才能从事具体教学科研业务，又由于学校专业的方面，我只能着重于计算机和人工智能的应用研究，这样就不可能集中精力开展 Prolog 语言的研究，但我仍以极大的兴趣关注着逻辑编程研究的发展。1988 年当我读到了 Ehud Shapiro 编写的《The Art of Prolog: Advanced Programming Techniques》一书，使我产生编一本有关 Prolog 程序设计技术方面书的念头，这就是本书的来源。本书的许多材料取自上述那最后一本书，另外一些是我讲课和研究中积累的资料及所获得的经验和体会，它们体现在我 1984 年编写的教材《逻辑编程与 Prolog 语言》一书中。在繁忙的教学行政事务和教学科研任务中，要挤出时间来撰写这本书确实是一件辛苦的事。希望本书的出版有助于 Prolog 语言，逻辑编程的进一步推广、应用和发展。

与我合作编写本书的有原自动化工程学院马莎讲师、总参 61 所李连生高级工程师，感

谢他们真诚地合作．由于他们所做的大量的认真而耐心细致的工作，才使本书能早日与读者见面．

感谢研究生王军为本书提供了程序和资料，黄燕玲同志为本书的出版编排打印了全部书稿．

感谢西北工业大学何华灿同志为本书出版所作的努力，感谢西北工业大学出版社柴文强同志为支持本书的出版所做的工作．感谢他们对人工智能事业的支持．

裴　珉

1990 年 6 月

目　录

第三篇 Prolog 语言

第五篇 应 用

绪　论

自从电子计算机诞生以来，起初人们用它来进行科学计算，而后推广到进行各种非数值的数据处理，现在又发展到知识处理，用以从事各种问题求解。不论我们利用电子计算机做什么，我们都必须首先告诉计算机要解什么问题和如何去解这些问题，即给出各种各样解题的算法，它通常是非常具体的一步一步的解题方法。当我们把算法用计算机能理解的语言输给计算机时，这就是计算机程序。可见，要利用计算机来工作时，遇到的第一个问题就是人—机通讯问题和用什么语言通讯的问题。

几乎所有的现代计算机都是以40年代冯诺依曼和他的伙伴们的早期概念基础制造的。冯诺依曼机器是以拥有大量相同的存储单元和含有寄存器的处理器为特征的。处理器可以把数据从存储单元中取出传送到寄存器，在处理器中进行数学计算或逻辑计算，然后再把寄存器中的计算结果值送回存储器中。对于冯诺依曼机器来说，一个程序是由指令序列和控制命令集组成的，指令是用来执行各种操作的，而控制命令是以寄存器中的内容为依据来控制下一步要执行的指令。

人们开始研究编程语言。第一种能为机器理解用来和机器对话的语言是机器语言，即0，1序列的机器代码。这是完全面向机器的语言，用这种语言写的程序所表达的算法一步一步非常细，如同教一个婴儿学话、走路一样。它是纯指令性语言，它一点不漏地说明问题是如何求解的，而且每条指令都和具体的机器有关，要关心信息安放的单元地址和存储分配，但这种语言的程序很难被人们读懂，没有框图几乎无法理解程序解决什么问题。因此这类程序容易出错，难以修改和纠正。为了克服这些缺点，首先出现了使用和机器语言一一对应的助记的汇编语言，而后随着各种编译和解释程序的出现，形成了各种高级程序设计语言，例如：Fortran，Algol，PL/I，Pascal，Ada等。高级语言的算法表达虽然已接近问题的原始概念，但它仍然是指令性的，尽管抽象程度提高了，但语言发展的主流都带有冯诺依曼机器结构的基本特征。冯诺依曼机是顺序机，它按操作顺序来设计程序，描述求解的问题及其求解方法，这不符合人们的思维习惯。虽然结构程序设计方法的推广，改善了程序的可读性，但要读懂一个大程序，在大型软件设计中用程序在设计人员中交换信息，仍是十分费力的事。程序正确性验证或程序测试依然是一个难题。现在，某些高级语言的程序已经可能没有解释来了解程序要做什么，某些指令的细节例如存储分配和过程调用中指令已经不见了。即程序不仅有指令的效果，也还有些描述事物的能力。这种进展仍然不能满足要求。

随着计算机的应用日益广泛和深入，它不仅用于学校、工厂作科学研究和过程控制，而且深入办公室、家庭用作事务处理和电气设备自动化的工具。现代化在信息时代离不开计算机，不仅专业人员要用，而且普通人员也要用。作为人-机通讯工具的语言能否更易书写、更简单些，且被机器接受而运行呢？能否更接近于自然语言呢？这是人们亟待解决的问题。

逻辑是分析和论证的重要的思维工具，它研究与事物自身内容和它的真假值无关的前提与结论之间的关系，用符号语言来精确描述逻辑推理，就形成了逻辑语言和逻辑推理理论。

逻辑几乎一开始就作为计算机设计的工具以及计算机程序的原理，但将逻辑方法直接用于计算机编程，用逻辑语言作为程序语言则是近十余年的事。

一阶谓词逻辑语言体现了人们思维的主要特征，它接近于自然语言。它可以看作一个纯粹的描述性语言。用它来书写程序，可以仅仅只表达一个程序要做什么而不必说明如何做；当我们给定一个证明过程时，某种形式的逻辑规范说明（Horn 子句）就蕴涵了它要执行的指令。因此，逻辑语言就成为一种非常高级的语言，这种语言既可做规范说明语言又可以做实际运行的程序语言，它与机器无关。目前它已经过若干扩展成为可执行的编程语言。这就是最早于 1972 年实现的 Prolog 语言。它是面向用户的语言。这种崭新的以一阶谓词为基础的编程语言颇有一番兴味。受到许多人的欢迎，容易编程，编出来的程序清晰可读、简洁、错误较少，这些特色颇有吸引力。

逻辑程序的真正开始使用要归功于 Kowalski 和 Colmeraner.Kowalski 阐述了 Horn 子句逻辑的过程解释理论。他说明一条公理

$$A \text{ if } B_1 \text{ and } B_2 \text{ and } \cdots \text{ and } B_n$$

可作为递归编程语言的一个过程，既可以阅读又可以执行，其中 A 是过程的头部，B_i 是过程体。描述性地理解上述子句为：如果 B_i 都为真，则 A 就为真。它也可以理解为要求解 A 就要求解 B_1、B_2…和 B_n。按这种理解 Horn 子句逻辑的证明过程就是语言的解释器，而作为归结证明过程核心的合一算法，则执行变量赋值的基本数据操作，传递参数，数据选择和数据传递。

同时，70 年代初，Colimeraucr 和他在马赛尔大学的研究组用 Fortran 编写了用于实现自然语言处理系统的专用的定理证明器，这个定理证明器称为 Prolog（全称为用逻辑编程——for Programationet Logique（法语）），它体现了 Kowalski 的过程解释。后来 Van Emden 和 Kowalski 又研究了逻辑程序语言的形式语义，并说明它和操作语义、模型论语义和不动点语义是等价的。

从 Prolog 语言创建开始，它经历了从青年期到成熟期的急速转变。初期 Prolog 由于它的新颖和特色受到了许多人的欢迎，但也因为它的效率低和难以控制而受到反对和责难。70 年代中到 70 年代末期 David H.D.Warren 和他的同事研制了 Prolog-10 编译程序，这个有效的 Prolog 实现消除了所有关于逻辑编程不可实用的神话。这个编译程序在纯的表处理程序方面的性能可以和当时最好的 Lisp 系统相比拟，而且它到现在仍是 Prolog 语言最好的实现之一。另外，这个编译程序本身就几乎全部用 Prolog 编写的，它指出不仅高级的 AI 人工智能程序可以应用它，而且古典的编程任务也可以从逻辑编程的效力中得到好处。现在已出现众多的商业化的 Prolog 实现文本，可以广泛使用于多种计算机上和微机上。例如 Trubo-Prolog 1.0 和 2.0 已在我国广泛传播。同时，还出现了大量的 Prolog 编程书籍，着重介绍了语言的不同方面。另外，Prolog 语言自身或多或少地已经稳定下来，它有了实际的标准，即 Edinburgh Prolog 家族。Prolog 的程序设计逐渐积累了丰富的经验并在广泛的领域中获得应用。目前世界上许多国家都在研究和发展这种语言，有的还作为第五代计算机核心语言来开发。

自从 Prolog 语言成为一门程序设计语言的十余年中涌现出大量的编程技术。本书的目的是将那些使我们兴奋、激动、富有创造力的程序设计技巧进行广泛地传播。本书由五部分组成：逻辑程序的基础——数理逻辑，逻辑程序，Prolog 语言，Prolog 高级程序设计技

术，应用等五篇。

本书第一部分介绍逻辑程序的基础，即数理逻辑的基本知识。命题演算、谓词、逻辑的子句型，本篇给出了逻辑编程基本结构的来源。

第二部分介绍自成体系的逻辑程序设计。本书强调逻辑编程与 Prolog 编程的关系与区别。逻辑程序用真值和逻辑推理两个与机器无关的抽象概念来研究。Prolog 语言基本结构来自逻辑，但它既是计算机上执行的指令又是逻辑语句，由于它继承了逻辑程序的某些抽象特性，所以我们首先要对逻辑编程理论有一个清楚的认识。本篇由五章组成，前三章介绍逻辑程序的基本结构及两种基本的逻辑编程类型，即数据库编程和递归编程，后两章介绍了逻辑编程的计算模型和逻辑程序理论，最后一章提出的一些理论结论，按要求没有给出证明。本篇的叙述和其它书籍中的逻辑编程的介绍不同，我们根据逻辑演绎推理来介绍逻辑编程的基础而其它书是以逻辑编程起源的归结法作为背景来介绍基本知识的。前者可以使读者感到直观并且易于理解。

第三部分介绍 Prolog 语言，它由第九章到第十五章组成。第九章讨论了与逻辑编程不同的 Prolog 计算模型，并将 Prolog 语言和传统的编程语言作了比较；另外着重讨论了组合逻辑程序和 Prolog 程序的区别，给出了基本的编程方法的实例。第十章至十四章等五章介绍了 Prolog 系统提供的系统谓词，它是使 Prolog 语言成为实际编程语言所必不可少的。我们将 Prolog 系统谓词分成四类：即关于高效的算术谓词，结构检查谓词，说明计算状态的元逻辑谓词，超出逻辑编程计算模型之外产生副作用的外逻辑谓词。其中有一章专门讨论最有名的外逻辑谓词 cut。各章分别说明了使用上述系统谓词的方法。本篇的最后一章讨论了编程方面的各种各样语言学上的问题。

第四篇是本书的主要部分，它介绍了现已在 Prolog 编程界流传的高级程序设计技巧，阐述的每一个实例虽然很小但功能很强。它典型地说明了这些有用的技术的应用。它共有六章（16 至 21 章），各章分别讨论了不确定性编程，不完全的数据结构，用 DCG 确定性子句文法的语法分析，二阶编程，搜索技术，和元解释器的使用。

最后第五篇由三章组成，它说明了用以上各章内容如何构成实用程序的。程序员要在理解怎样设计一个精巧的短小程序的基础上来构造大型的程序软件。本篇的应用涉及逻辑智力游戏程序，原型的专家系统——有关信贷的评估，和符号方程求解等。

任何编程的进步，包括语言工具的改进，用更加高级的语言来书写程序，都可以看作是向自动化编程迈进，可以想象成这样，当我们用某些软件、硬件进一步武装机器时，机器就变得更"聪明"了，更有智能了。这时，我们就可以用描述性多一些，指令性少一些的语言去指挥计算机，去对计算机布置任务。就如同扮演司令员的角色，给军长、师长下达任务而不是班长给战士下达任务。这种区别可以精确地描述为将算法分成逻辑成分和控制成分两部分，它分别反映问题的描述部分——做什么，和问题的求解部分——如何做的两个方面。人给出问题的描述，当机器更加"聪明"时，它就能理解更高级的语言，能根据问题的逻辑描述自己去寻找如何求解，电子技术的飞跃发展，开辟了计算机作为智能工具的前景，人们充当高级指挥员用高级智能语言指挥计算机是可以期待的，也是令人神往的快事。

Prolog 语言作为初级的逻辑编程语言在 70 年代初诞生，现在正风靡于世，它已在许多机上实现和许多实际领域中应用，让我们为它的进一步发展改进和应用开辟道路吧!

第一篇　逻辑程序的基础
——数理逻辑

逻辑学是研究思维形式及思维规律的一门科学，它是分析和论证的重要工具，数理逻辑是用数学的方法研究形式逻辑中推理规律的一种理论．所谓数学方法主要指引进一套符号体系的方法，所以数理逻辑又称符号逻辑．数理逻辑研究推理，即研究推理中前提和结论之间的形式关系，这种形式关系是日常生活中思维和科学思维的抽象，它是由作为前提和结论的命题的逻辑形式决定的。

符号逻辑是自然语言和人类推理的形式化，它起源于自然语言，很久以来，它在计算机科学中作为计算机程序的规范说明语言，并且作为数据库查询语言的基础．随着自动推理的发展，计算机已经成为处理逻辑的有效工具，它导致了逻辑的进一步应用．逻辑语言可以直接成为计算机程序语言，本书的主要内容是论述逻辑语言如何演变为逻辑程序、Prolog 语言及如何运用逻辑程序语言有效地编制程序的方法，而它的全部内容的基础是数理逻辑。

本篇主要介绍数理逻辑的最基本的内容：命题演算和谓词演算．这两种逻辑演算都是一种形式系统，是在研究前提和结论之间的形式关系中发展起来的．逻辑演算反映自然语言的某些特征，其中的合式公式反映了命题的逻辑形式，形式推理反映了演绎推理．为了便于讨论逻辑语言是怎样成为计算机程序语言的，本书不仅对逻辑的标准型进行了介绍，而且还着重地介绍了逻辑的子句型，逻辑的子句型比标准型更接近于数据处理和计算机编程中使用的其它形式体系．同时我们还讨论了标准型和子句型之间的相互变换．最后引入子句型的一类重要子集 Horn 子句，这种形式的子句实际上就是下一篇要讨论的逻辑程序的基本语句，它的过程解释已经成为一种有效的计算机逻辑型描述性语言的基础。

第一章　命题演算

命题演算是数理逻辑中最初等的，然而也是最基本的组成部分，它是谓词演算的基础。

§1.1　命题与结构

1. 命题

凡是能分辨其真假的语句叫做命题。它是对事物的某种性质进行肯定或者否定的一种思

维形式，它代表人们进行思维活动时的一种判断．一般说，命题是一个有意义的陈述句，它总有一个"值"称为真值．真值只有"真"和"假"两种，记作 Truc(真)和 False(假)，分别用符号 T 和 F 表示．只有具有能确定真值的陈述句才是命题,一切没有判断内容的句子,无所谓是非或真假的句子,如感叹句、疑问句、祈使句等都不能作为命题．例如下述语句都是命题．

(1) 地球是圆的。

(2) 实践是检验真理的唯一标准。

(3) 雪是黑的。

(4) 2+3＝6。

(5) 王芳既会跳舞又会唱歌．

(6) 今晚我去看电影或去听音乐．

(7) 如果天下雨，则天上一定有云．

(8) 只有勤奋刻苦地学习，才能成为好学生．

下述语句都不是命题:

(1) 祝您生日愉快!

(2) 朋友，请问去中山公园怎么走?

(3) 天气多好啊!

在讨论命题时我们不仅关心其真值是真还是假，更关心的是，如果规定了某命题的真假值之后，它与其它命题之间的关系，在研究命题与命题之间的关系时，可以把命题分成两种类型：第一种类型是不能分解为更简单的陈述句的命题，称为原子命题；第二种类型是由原子命题与逻辑联结词复合而构成的新的命题，称为复合命题．在以上所举的命题实例中，(1)至(4)都是原子命题，(5)至(8)都是复合命题．

在数理逻辑中，我们用英文字母或词来表示一个命题，例如用 p，q，r 来表示命题（通常用大写字母表示命题，本书为了和逻辑程序的符号一致，采用小写字母)．

2. 逻辑联结词

在日常语言中，由一些简单的陈述句，通过一些"联结词"而构成的较为复杂的语句称为复合语句．命题演算中，同样由原子命题通过特定的"联结词"可以构成复合命题．但由于自然语言中的联结词有二义性，所以在命题演算中，必须对联结词作出明确而又严格的定义，并加以符号化．

命题演算中有五个常用的逻辑联结词，它们是"非"、"与"、"或"、"如果…则"和"当且仅当"．可用符号表示如下:

"非"(not)或称否定用～表示

"与"(and)或称合取用∧表示

"或"(or)或称析取用∨表示

"如果……则"(if…then)或称蕴含，用 →表示

"当且仅当"（if…and only if）或称等价用←→表示。

命题演算中，逻辑联结词～是一个一元运算，其它联结词∧，∨，→，←→都是二元运算．它们分别都可以用真值表给出定义，真值表用各个逻辑联结词构成的新的复合命题与构

成它的原子命题之间真值的关系来定义。

设有命题p、q逻辑联结词的真值表如下:

p	q	~p	p∧q	p∨q	p→q	p←→q
T	T	F	T	T	T	T
T	F	F	F	T	F	F
F	T	T	F	T	T	F
F	F	T	F	F	T	T

上述真值表的含义为: 设P为一命题, ~P为一新命题, 若P为T, 则~P为F, 若P为F, 则~P为T。两个命题p和q的合取是一个复合命题, p∧q, 当且仅当p、q同时为T时, p∧q才为T, 其它的情况下p∧q都为F。两个命题p, q, p蕴含q时为一复合命题P→q, 当且仅当P的真值为T, 而q的真值为F时, p→q的真值才为F, 其余情况都为T。两个命题p、q, p和q等价是一个复合命题p←→q, 当p和q真值相同时, p←→q的真值为T, 否则p←→q真值为F。

§1.2 命题公式及其解释

为了用数学的方法来研究逻辑, 就要把命题作为一个一般性的数学概念来处理, 撇开命题的内在涵义, 而把它看成一个抽象的形式化概念。所以, 在研究中我们并不关心具体命题的实际含义只注意其真假值。

一个特定的命题是一个常值命题, 它一定有值为真或假(二者必居其一, 也仅居其一)。一个任意的没有赋予具体内容的命题是一个变量命题, 在公式中的变量命题称为命题变元, 或原子公式, 简称原子, 它的变化域是集合 {T, F}。我们用英文字母或词既表示具体命题也表示命题变元。

定义 命题演算的合式公式是以下递归定义的符号串:

(1)命题变元是公式, 即原子公式。

(2)如果p、q是公式, 则~p, (p∧q), (p∨q), (p→q), (p←→q)也是公式。

(3)有限次应用上述规则生成的符号串都是公式。

按照定义, 下述公式都是命题演算的合式公式:

$$\sim(p\wedge q),\ \sim(p\rightarrow q),\ \sim(p\rightarrow q)\longleftrightarrow(p\vee q)$$

而下面所表示的符号串都不是合式公式

$$p\vee,\ P\rightarrow Q,\ (P\rightarrow Q)\rightarrow(\wedge Q)$$

为了减少使用圆括号的次数, 约定最外层圆括号可以省略, 另外我们规定联结词运算的优先次序为: ~, ∧, ∨, →, ←→, 则省去括号的下述符号串 p∧q→r 也是合式公式。

一个命题公式有真假, 它的真假由组成它的命题的真假值唯一地确定, 故命题公式可以看成是一个以真、假为定义域, 以真、假为值域的函数。可以用真值表方法来确定一个命题公式的真假, 此真值表即称为命题公式的真值表。

定义 命题公式的解释I 设G是n个命题变元 P_1, P_2, …, P_n 组成的命题公式, 对

命题变元 P_1，P_2，…，P_n 真值的一组指派，就称为公式 G 的一个解释 I。

含 n 个命题变元的公式就有 2 的 n 次方个解释。当命题公式有一个解释后，就能得到一个确定的真值。在真值表中对命题变元某一行上作的真值的指派就是命题公式的一个解释。

我们用一个例子说明构造命题公式真值表的方法。

例：构造命题公式$\sim(p\wedge q)\rightarrow(\sim P\wedge\sim q)$的真值表：

p	q	$p\wedge q$	$\sim(p\wedge q)$	$\sim p$	$\sim q$	$\sim p\wedge\sim q$	$\sim(p\wedge q)\rightarrow(\sim p\wedge\sim q)$
T	T	T	F	F	F	F	T
T	F	F	T	F	T	F	F
F	T	F	T	T	F	F	F
F	F	F	T	T	T	T	T

§1.3 命题公式的性质

本节讨论命题公式之间的等价关系及具有某些特性的命题公式。

定义 公式的等价关系

设 A 和 B 是含 n 个相同命题变元 P_1，P_2，…，P_n 的命题公式，如果在所有的解释 I 下(即对 2 的 n 次方个可能的真值指派)A 和 B 两公式的真值都相同，则称 A 和 B 是等价的或逻辑相等。记为 $A\Leftrightarrow B$。

上述定义中"$\Leftrightarrow$"并不是属于我们建立的逻辑体系的符号，它是描述逻辑体系的元语言，所以是一个元逻辑符号。

可以用真值表的方法证明公式的等价关系。

例 证明 $p\leftrightarrow q\Leftrightarrow(p\rightarrow q)\wedge(q\rightarrow p)$

p	q	$p\rightarrow q$	$q\rightarrow p$	$p\leftrightarrow q$	$(p\rightarrow q)\wedge(q\rightarrow p)$
T	T	T	T	T	T
T	F	F	T	F	F
F	T	T	F	F	F
F	F	T	T	T	T

由上述真值表可知 $p\leftrightarrow q$ 与$(p\rightarrow q)\wedge(q\rightarrow p)$真值相同，命题得证。

下表列出常用的命题定律，即等价的命题公式，它们都可以用真值表验证其等价关系。