



机器定理证明（自动定理证明）

- 机器定理证明是人工智能的重要研究领域，它的成果可应用于问题求解、自然语言理解、程序验证和自动程序设计等方面。
- 数学定理证明的过程尽管每一步都很严格有据，但决定采取什么样的证明步骤，却依赖于经验、直觉、想象力和洞察力，需要人的智能。因此，数学定理的机器证明和其他类型的问题求解,就成为人工智能研究的热点。

早在**19**世纪中叶，**莱布尼兹**就提出过用机器实现定理证明的思想。**19**世纪后期**G.弗雷格**的“思想语言”的形式系统，即后来的谓词演算，奠定了符号逻辑的基础，为自动演绎推理提供了必要的理论工具。

自动证明的发展



- 由于传统的兴趣和应用的價值，初等几何问题的自动求解成为数学机械化的研究焦点。
- 希尔伯特
 - 20世纪初，在他的名著《几何基础》中给出了一条可以对一类几何命题进行判定的定理。
 - 希尔伯特对命题的要求太高，当时仅能解决很少的一类几何定理的机器证明，却是历史上第一个关于某类几何命题的机械化检验方法的定理。



希尔伯特

自动证明的发展



■ 塔斯基 (波兰)

- 1950年, 证明了: “一切初等几何和初等代数范围的命题都可以用机械方法判定”
- 为几何定理的机器证明开拓了一条利用代数方法的途径
- 方法太复杂, 即使用高速计算机也证明不了稍难的几何定理



塔斯基

自动证明的发展

■ 纽厄尔,西蒙和肖

- 1956年,发表了论文《逻辑理论机》(LTM)
- 认为LTM不仅是计算机智力的有力证明,也是人类认知本质的证明
- 1957年开发了最早的AI程序设计语言IPL语言
- 1960年,成功地合作开发了“通用问题求解系统”GPS (General Problem Solver)



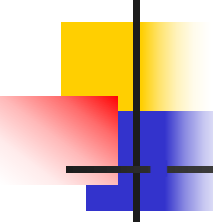
艾伦·纽厄尔



赫伯特·西蒙

智能科学基础

1956年, Newell, Shaw和Simon给出了一个称为“**逻辑机器**”的程序,证明了罗素、怀德海所著《数学原理》中的许多定理,这标志着自动定理证明的开端。

- 
- **1959年**, **Gelernter**给出了一个称为“**几何机器**”的程序,能够做一些中学的几何题,速度与学生相当。
 - **1960年**, **Davis、Putnan**等给出了**D—P**过程,大大简化了命题逻辑的处理。
 - **1965年**, **Robinson**提出**归结方法**,使得自动定理证明领域发生了质的变化。
 - **1968年**, 苏联**Maslov**提出了**逆向法**,是苏联人六、七十年代在该领域做的主要工作。
 - **归结方法有许多重要改进**,每种改进有各自的优点。语义归结由**Slagle**提出,它将超归结(**Robinson**)、换名归结(**Meltzer**)、支架集策略(**Wos, RobinSon**)等方法一体化。锁归结,由**Boyer**在**1971年**提出,是一个很高效的规则。线性归结由**Loveland**和**Luckham**在**1970年**独立提出。

自动证明的发展



- 1977年，美国年轻的数学家阿佩尔等在高速电子计算机上耗费 1200 小时的计算时间，证明了著名的“四色定理”，人类百年悬而未决的疑问最终被圆满解决了。这一成就轰动一时，成为机器定理证明的一个典范。



自动证明的发展—王浩



■ 美籍华裔王浩

- 1983年，获国际人工智能联合会“数学定理机械证明里程碑奖”，表彰他在数学定理机械证明研究领域的开创性贡献。
- 1972年以后，王浩数次回国讲学。1985年兼任北京大学教授；1986年兼任清华大学教授。
- 新中国成立之初，公开发表演说表示对新中国的支持。1972年回国时曾受周恩来总理的接见。1973年撰写了《访问中国的沉思》，赞美新中国，被报纸与杂志广泛刊载。为此，他受到了许多攻击。他热爱祖国和中华民族的精神值得人们学习与称道。

智能科学与技术系

1960年,美籍华裔王浩在**IBM704**上,编程实现了三个程序,第一个程序用于命题逻辑,第二个程序让机器从基本符号出发自动生成合适命题逻辑公式并选出其中定理,第三个程序用于判定一阶逻辑中的定理。他证明了罗素、怀德海《数学原理》中的几乎所有定理。他的方法人们称之为“王浩”算法。他的这项工作在**1984**年首届自动定理证明大会上获最高奖—“里程碑”奖。

吴文俊：数学机械化

吴方法：用计算机证明几何定理的方法，采用代数化的思想，与通常基于逻辑的方法根本不同，首次实现了高效的几何定理自动证明，不仅限于几何，还给出了部分物理、化学、机器人问题的自动证明。

属于中国的自动证明方法—吴方法



■ 吴文俊



- 1984年出版专著《几何定理机器证明的基本原理》，利用中国古典数学的成就，提出具有中国特色的定理自动证明方法，被国际上誉为“**吴氏方法**”。
- 1985年发表论文“关于代数方程组的零点”—吴文俊消元法，即“**吴氏公式**”。
- 2010年5月4日，国际小行星中心发布公报通知国际社会，将国际永久编号为第7683号的小行星永久命名为“吴文俊星”。

智能科学与技术系

属于中国的自动证明方法—吴方法



■ 著名数学家吴文俊



- 中国科学院数学与系统科学研究院研究员，中国科学院院士，第三世界科学院院士
- 1919年出生于上海
- 1940年毕业于交通大学数学系
- 1949年获法国国家博士学位
- 1951年回到祖国，任北京大学数学系的教授
- 1956年与华罗庚、钱学森同台领取国家自然科学奖一等奖；38岁时当选为中国科学院学部委员
- 1993年获得陈嘉庚数理科学奖
- 1994年获首届求是科技基金会杰出科学家奖
- 1997年获Herbrand自动推理杰出成就奖

智能科学与技术系



第3章 确定性推理

- 智能系统的推理过程实际上就是一种思维过程。
- 按照推理过程所用知识的确定性，推理可分为确定性推理和不确定性推理。
- 3.1 推理的基本概念
- 3.2 3.3 推理的逻辑基础
- 3.4 自然演绎推理
- 3.5 3.6 归结演绎推理



3.1 推理的基本概念

- 3.1.1 什么是推理
- 3.1.2 推理方法及其分类
- 3.1.3 推理的控制策略及其分类

3.1.1 什么是推理

推理的概念

是指运用已掌握的知识，从已知事实出发去推出结论的过程。

推理所用事实

- 初始证据：推理前用户提供的
- 中间结论：推理过程中所得到的
- 推理过程：由推理机来完成，所谓推理机就是智能系统中用来实现推理的那些程序。

推理的一般形式：

已知：事实**1**，事实**2**， ...

如果 事实**1** 那么 结论**1**

如果 事实**2** 那么 结论**2**

...

得到：结论**1**， 结论**2**， ...



■ 推理的两个基本问题

推理的方法：解决前提和结论的逻辑关系。

推理的控制策略：解决推理方向，冲突消解策略等。



3.1.2 推理方法及其分类

1. 按推理的逻辑基础分类(1/5)

- 可分为演绎推理、归纳推理

■ 演绎推理

■ 从已知的一般性知识出发，去推出蕴含在这些已知知识中的适合于某种个别情况的结论。是一种由一般到个别的推理方法。

■ 最常用的是三段论。

■ 假言三段论

■ 常用的三段论是由一个大前提、一个小前提和一个结论这三部分组成的。其中，**大前提**是已知的一般性知识或推理过程得到的判断；**小前提**是关于某种具体情况或某个具体实例的判断；**结论**是由大前提推出的，并且适合于小前提的判断。



3.1.2 推理方法及其分类

例：有如下三个判断：

① 计算机系的学生都会编程序；（一般性知识）

② 程强是计算机系的一位学生；（具体情况）

③ 程强会编程序。（结论）

这是一个三段论推理。其中，①是大前提，②是小前提；③是经演绎推出来的结论。

可见，其结论是蕴含在大前提中的。



3.1.2 推理方法及其分类

1. 按推理的逻辑基础分类(2/5)

- 归纳推理
- 是一种由个别到一般的推理方法。
- 归纳推理的类型
 - 按照所选事例的广泛性可分为完全归纳推理和不完全归纳推理
 - 按照推理所使用的方法可分为枚举、类比等



3.1.2 推理方法及其分类

完全归纳推理

是指在进行归纳时需要考察相应事物的全部对象，并根据这些对象是否都具有某种属性，推出该类事物是否具有此属性。例如：计算机质量检验。

不完全归纳推理

是指在进行归纳时只考察了相应事物的部分对象，就得出了关于该事物的结论。例如：计算机随机抽查。

3.1.2

1. 按推理

例：设有如下事例：

王强是计算机系学生，他会编程序；

高华是计算机系学生，她会编程序；

.....

.....

当这些具体事例足够多时，就可归纳出一个一般性的知识：
凡是计算机系的学生，就一定会编程序。

枚举归纳推理

- 是指在进行归纳时，如果已知某类事物的有限可数个体事物都具有某种属性，则可推出该类事物都具有此种属性。

类比归纳推理

- 是指在两个或两类事物有许多属性都相同或相似的基础上，推出它们在其他属性上也相同或相似的一种归纳推理。

类比归纳推理的基础是相似原理，其可靠程度取决于两个或两类事物的相似程度以及这两个或两类事物的相同属性与推出的那个属性之间的相关程度。

例：声和光有不少属性相同：直线传播、反射、折射、干扰。由此推出：既然声有波动性，光也有波动性。



3.1.2 推理方法及其分类

1. 按推理的逻辑基础分类 (5/5)

- 演绎推理与归纳推理的区别
- **演绎推理**是在已知领域内的一般性知识的前提下，通过演绎求解一个具体问题或者证明一个结论的正确性。它所得出的结论实际上早已蕴含在一般性知识的前提中，演绎推理只不过是已将已有事实揭露出来，因此它**不能增殖新知识**（此处不一定正确）。
- **归纳推理**所推出的结论是没有包含在前提内容中的。这种由个别事物或现象推出一般性知识的过程，**是增殖新知识**的过程。
- 例如，一位计算机维修员，通过大量实例积累经验，是一种**归纳推理**方式。运用书本上一般性知识去维修计算机的过程则是**演绎推理**。



3.1.2 推理方式及其分类

2、单调推理、非单调推理(1/2)

按推理过程中推出的结论是否单调地增加，或推出的结论是否越来越接近目标，可分为**单调推理**和**非单调推理**。

单调推理：在推理过程中随着推理的向前及新知识的加入，推出的结论是呈单调增加的趋势，并且越来越接近最终目标，在推理过程中不出现反复的情况，即不会由于新知识的加入否定了前面推出的结论，从而使推理又回到前面的某一步。



3.1.2 推理方式及其分类

2、单调推理、非单调推理(2/2)

非单调推理（包括缺省推理即默认推理和模态非单调推理）：

在推理过程中由于新知识的加入，不仅没有加强已推出的结论，反而要否定它，使得推理退回到前面的某一步，重新开始。

注意：非单调推理是在知识不完全的情况下发生的。由于知识不完全，为使推理进行下去，就要先做某些假设，并在此基础上进行推理，当以后由于新知识的加入发现原先的假设不正确的时候，就需要推翻该假设以及基于此假设而推出的一切结论，再用新的知识重新进行推理。



3.1.2 推理方式及其分类

3、确定性推理、不确定性推理

按推理时所用知识的确定性来划分，推理可分为确定性推理、不确定性推理。

确定性推理（精确推理）：如果在推理中所用的知识都是精确的，即可以把知识表示成必然的因果关系，然后进行逻辑推理，推理的结论或者为真，或者为假，这种推理就称为确定性推理。

不确定性推理（不精确推理）：在人类知识中，有相当一部分属于人们的主观判断，是不精确的和含糊的。由这些知识归纳出来的推理规则往往是不确定的。基于这种不确定的证据或推理规则进行推理，形成的结论也是不确定的，这种推理称为不确定性推理。



3.1.2 推理方式及其分类

3、确定性推理、不确定性推理

不确定性研究的重点可能会：

- 一是：解决现有处理不确定性的理论中存在的问题；
- 二是：大力研究人类高效、准确的识别能力和判断机智，开拓新的处理不确定性的理论和方法；
- 三是：探索可以综合处理多种不确定性的方法和技术。



3.1.2 推理方式及其分类

4、启发式推理、非启发式推理(1/2)

如果按推理过程中是否运用于问题有关的启发性知识，推理可分为**启发式推理**与**非启发式推理**。

启发式推理：如果在推理过程中，运用了**与问题有关的启发性知识**，即解决问题的策略、技巧及经验，以加快推理过程，提高搜索效率，这种推理过程就被称为启发式推理。

如第五章的A*、A0*等算法就属于此类推理。



3.1.2 推理方式及其分类

4、启发式推理、非启发式推理 (2/2)

非启发式推理：如果在推理过程中，不运用启发性知识，只按照一般的控制逻辑进行推理，这种推理过程就被称为非启发式推理。

注意：这种推理缺乏对待求解问题的针对性，所以推理效率较低，容易出现组合爆炸问题。例如图搜索策略中的宽度优先搜索法，虽然是完备的算法，但是对复杂问题的求解，将出现组合爆炸现象，其搜索效率较低。



3.1.3 推理的控制策略

- **推理的控制策略**主要包括推理方向、搜索策略、冲突消解策略、求解策略及限制策略等。

1、推理方向

- 推理方向用于确定推理的驱动方式，分为四种：

正向推理

逆向推理

混合推理

双向推理



3.1.3 推理的控制策略

- ① **正向推理** 从初始状态出发，使用规则，到达目标状态。
又称为数据驱动推理、前向链推理、模式制导推理及前件推理。

- ② **逆向推理** 以某个假设目标为出发点的一种推理。又称为
目标驱动推理、逆向链推理、目标制导推理及后件推理。



3.1.3 推理的控制策略

- **混合推理** 已知的事实不充分。推理的形式：
 - 先正向再逆向，通过正向推理，即从已知事实演绎出部分结果，然后再用逆向推理证实该目标或提高其可信度。
 - 先逆向再正向，先假设一个目标进行逆向推理，推出一些中间假设，然后再用正向推理对这些中间假设进行证实。



3.1.3 推理的控制策略

④

双向推理

- 双向推理是指正向推理与逆向推理同时进行，且在推理过程中的某一步骤上“碰头”的一种推理。
- 正向推理所得的中间结论恰好是逆向推理此时要求的假设



2、求解策略

推理是只求一个解还是求所有解以及最优解等

3、限制策略

对推理的深度、宽度、时间、空间等进行限制



3.1.3 推理的控制策略

4、冲突消解策略

- 在推理过程中，匹配会出现三种情况：
 - ✓ 已知事实（假设）不能与知识库中的任何知识匹配成功
 - ✓ 已知事实（假设）恰好只与知识库中的一个知识匹配成功
 - ✓ 已知事实（假设）可与知识库中的多个知识匹配成功；或者有多个（组）已知事实（假设）都可与知识库中某一知识匹配成功；或者有多个（组）已知事实（假设）可与知识库中的多个知识匹配成功



3.1.3 推理的控制策略

4、冲突消解策略

■ 出现冲突的情况

- ✓ 对正向推理而言，如果有多条产生式规则的前件都和已知的事实匹配成功；或者有多条不同的已知事实都与同一条产生式规则的前件匹配成功；或者有多条产生式规则的前件与多个事实匹配成功。
- ✓ 对逆向推理而言，如果有多条产生式的后件都和同一假设匹配成功，或者有一条产生式后件可与多个假设匹配成功；或者有多条产生式规则后件可与多个假设匹配成功。



3.1.3 推理的控制策略

■ 冲突消解策略

需要按照某种策略从多条匹配的规则（事实）（假设）中选择一条最佳知识用于推理，这种解决冲突的过程称为冲突消解。所用策略称为冲突消解策略。

目前已有的多种冲突消解策略的基本思想都是对匹配的规则（事实）（假设）进行排序，以决定匹配规则等的优先级别，优先级高的规则（事实）（假设）将作为启用规则。



3.1.3 推理的控制策略

■ 冲突消解策略

1. 按就近原则排序

- 该策略把最近被使用过的规则赋予较高的优先级。

2. 按知识特殊性排序

这种策略把知识的特殊性作为知识排序的依据，具有特殊性的知识排在前面，赋予较高的优先级。

3.1.3 推理的控制策略



冲突消解策略

3. 按上下文限制排序

- 该策略将知识按照所描述的上下文分成若干组，在推理过程中根据当前数据库中的已知事实与上下文的匹配情况，确定选择某组中的某条知识。

4. 按已知事实的新鲜性排序

- 一般我们认为新鲜事实是对旧知识的更新和改进，比老知识更有效，即后生成的事实比先生成的事实具有较大的优先性。



3.1.3 推理的控制策略

冲突消解策略

5. 按知识的差异性排序

这种策略把知识的差异性作为知识排序的依据，为与上一次使用过的知识差别大的知识赋予更高的优先级。

6. 按领域问题特点排序

- 该方法按照求解问题领域的特点将知识排成固定的次序



3.1.3 推理的控制策略

■ 冲突消解策略

7.按规则的次序排序

该策略是以知识库中预先存入规则的排列顺序作为知识排序的依据，排在前面的规则具有较高的优先级。

8.按前提条件的规模排序（与2相反）

把知识的前提条件的个数作为知识排序的依据。在结论相同的多个知识中，前提条件少的知识有更高优先级。



3.1.3 推理的控制策略

冲突消解策略

9. 按匹配度排序

- 在不确定推理时，匹配度不仅可确定两个知识模式是否可匹配，还可用于冲突消解。根据匹配程度来决定哪一个产生式规则优先被应用。



第3章 确定性推理

- 3.1 推理的基本概念
- 3.2 3.3 推理的逻辑基础
- 3.4 自然演绎推理
- 3.5 3.6 归结演绎推理



3.2 3.3 推理的逻辑基础

- 置换与合一

3.3.5 置换与合一

1、置换

- 置换可简单的理解为是在一个谓词公式或者项中用置换项去替换变元。
- **定义3.12** 置换是形如 $\{t_1/x_1, t_2/x_2, \dots, t_n/x_n\}$ 的有限集合。其中， t_1, t_2, \dots, t_n 是项； x_1, x_2, \dots, x_n 是互不相同的变元； t_i/x_i 表示用 t_i 替换 x_i 。要求 x_i 不能出现在 t_i 中，并且不能出现情况：“对于不同的 i 和 j ， x_i 出现在 t_j 中且 x_j 出现在 t_i 中”。
- 例如， $\{a/x, c/y, f(b)/z\}$ 是一个置换。但 $\{g(y)/x, f(x)/y\}$ 不是一个置换。原因是它在 x 与 y 之间出现了循环置换现象。
- 通常，置换是用希腊字母 θ 、 σ 、 α 、 λ 等来表示的。
- 不含任何元素的置换称为空置换，以 ε 表示。

3.3.5 置换与合一

2、置换乘法

定义3.13 设

$$\theta = \{t_1/x_1, t_2/x_2, \dots, t_n/x_n\}$$

$$\lambda = \{u_1/y_1, u_2/y_2, \dots, u_m/y_m\}$$

是两个置换。则 θ 与 λ 的合成也是一个置换，记作 $\theta \circ \lambda$ ，也称置换乘法。它是从集合

$$\{t_1\lambda/x_1, t_2\lambda/x_2, \dots, t_n\lambda/x_n, u_1/y_1, u_2/y_2, \dots, u_m/y_m\}$$

中删去以下两种元素

① 当 $t_i\lambda = x_i$ 时，删去 $t_i\lambda/x_i$ ($i=1, 2, \dots, n$);

② 当 $y_j \in \{x_1, x_2, \dots, x_n\}$ 时，删去 u_j/y_j ($j=1, 2, \dots, m$)

最后剩下的元素所构成的集合。



3.3.5 置换与合一

2、置换乘法

- **例** 设 $\theta = \{ f(y)/x, z/y \}$, $\lambda = \{ a/x, b/y, y/z \}$, 求 θ 与 λ 的合成。
- 解:先求出集合
- $\{ f(y)\lambda/x, (z\lambda)/y, a/x, b/y, y/z \} = \{ f(b)/x, y/y, a/x, b/y, y/z \}$
- 其中, $f(b)/x$ 中的 $f(b)$ 是置换 λ 作用于 $f(y)$ 的结果; y/y 中的 y 是置换 λ 作用于 z 的结果。在该集合中, y/y 满足定义中的条件①, 需要删除; a/x 和 b/y 满足定义中的条件②, 也需要删除。最后得: $\theta \circ \lambda = \{ f(b)/x, y/z \}$



3.3.5 置换与合一

3、置换的性质

(1) 空置换 ε 是左么元和右么元，即对任意的置换 θ ，恒有

$$\varepsilon \circ \theta = \theta \circ \varepsilon = \theta$$

(2) 对任意表达式 E ，恒有 $E(\theta \circ \lambda) = (E\theta)\lambda$ 。

(3) 若对任意表达式 E 恒有 $E\theta = E\lambda$ ，则 $\theta = \lambda$ 。

(4) 对任意置换 θ, λ, μ ，恒有 $(\theta \circ \lambda) \circ \mu = \theta \circ (\lambda \circ \mu)$

即置换的合成满足结合律。

注意：置换的合成不满足交换律。



3.3.5 置换与合一

■ 4、合一

- 合一可理解为是寻找项对变量的置换，使两个谓词公式一致。可定义为：
- **定义3.14** 设有公式集 $F=\{F_1, F_2, \dots, F_n\}$ ，若存在一个置换 θ ，可使 $F_1\theta=F_2\theta=\dots=F_n\theta$ ，则称 θ 是 F 的一个合一。

称 F_1, F_2, \dots, F_n 是可合一的。

- 一般来说，一个公式集的合一不是唯一的。



Ex:

设有公式集 $F=\{P(x,y,f(y)), P(a,g(x),z)\}$, 则

$$\lambda=\{a/x, g(a)/y, f(g(a))/z\}$$

是它的一个合一。



3.3.5 置换与合一

■ 5、最一般合一(MGU, Most General Unifier)

- **定义3.15** 设 σ 是公式集F的一个合一，如果对F的任一个合一 θ 都存在一个置换 λ ，使得 $\theta = \sigma \circ \lambda$ ，则称 σ 是一个最一般合一(MGU)。

3.3.5 置换与合一

6、差异集（不一致集）

定义 在对公式集中的谓词公式中的项从左到右进行比较时，那些不相同的项所构成的集合。

例1 $W=\{P(x,f(y,z)),P(x,a),P(x,g(h(k(x))))\}$,在W的3个表达式中,前4个符号---“P(x,”是相同的,第5个符号不全相同,所以W的不一致集合(差异集)为: $\{f(y,z),a,g(h(k(x))))\}$.

例2 $W=\{P(x,y,z), P(x,f(a),g(b))\}$

不一致集合(差异集)为: $D1=\{y,f(a)\}$ $D2=\{z,g(b)\}$



3.3.5 置换与合一

假设 D 是 W 的差异集，显然有下面的结论。

(1) 若 D 中无变量符号，则 W 是不可合一的。

例： $W = \{P(a), P(b)\}$

$D = \{a, b\}$

(2) 若 D 中只有一个元素，则 W 是不可合一的

例： $W = \{P(x), P(x, y)\}$

$D = \{y\}$

(3) 若 D 中有变量符号 x 和项 t ，且 x 出现在 t 中，则 W 是不可合一的。

例： $W = \{P(x), P(f(x))\}$

$D = \{x, f(x)\}$

求最一般合一置换的算法

下面给出求公式 $\{E_1, E_2\}$ 的最一般合一置换的算法：

- ① 令 $W = \{E_1, E_2\}$ 。
- ② 令 $k = 0, W_k = W, \sigma_k = \varepsilon$; ε 是空置换, 它表示不作置换。
- ③ 如果 W_k 只有一个表达式, 则算法停止, σ_k 就是所要求的 mgu。
- ④ 找出 W_k 的不一致集 D_k 。
- ⑤ 若 D_k 中存在元素 x_k 和 t_k , 其中 x_k 是变元, t_k 是项, 且 x_k 不在 t_k 中出现, 则置:

$$\sigma_{k+1} = \sigma_k \cdot \{t_k/x_k\}$$

$$W_{k+1} = W_k \{t_k/x_k\}$$

$$k = k + 1$$

然后转步骤③。

- ⑥ 算法终止, W 的 mgu 不存在。

可将此处的两个公式的集合 W 推广为有限个公式的集合 W , 后面的步骤都一样。

例 3.5 设 $E_1 = P(a, v, f(g(y)))$, $E_2 = P(z, f(a), f(u))$, 求 E_1 和 E_2 的 mgu。

解 依据算法:

① 令 $W = \{P(a, v, f(g(y))), P(z, f(a), f(u))\}$ 。

② 令 $\sigma_0 = \varepsilon$, $W_0 = W$ 。

③ 由于 W_0 中含有两个表达式, 即未合一。

④ 从左到右找不一致集, 得 $D_0 = \{a, z\}$ 。

⑤ 取 $x_0 = z, t_0 = a$, 则

$$\sigma_1 = \sigma_0 \cdot \{t_0/x_0\} = \sigma_0 \cdot \{a/z\} = \{a/z\}$$

$$W_1 = W_0 \sigma_1 = \{P(a, v, f(g(y))), P(a, f(a), f(u))\}$$

第一次循环:

③ W_1 未合一, 因为其中含有两个表达式。

④ 从左到右找不一致集: $D_1 = \{v, f(a)\}$ 。

⑤ 取 $x_1 = v, t_1 = f(a)$, 则

$$\sigma_2 = \sigma_1 \cdot \{f(a)/v\} = \{a/z, f(a)/v\}$$

$$W_2 = W_1 \sigma_2 = \{P(a, f(a), f(g(y))), P(a, f(a), f(u))\}$$

第二次循环:

③ W_2 未合一, 因为其中仍含有两个不同的表达式。

④ 从左到右找不一致集: $D_2 = \{g(y), u\}$ 。

⑤ 取 $x_2 = u, t_2 = g(y)$, 则

$$\sigma_3 = \sigma_2 \cdot \{g(y)/u\} = \{a/z, f(a)/v, g(y)/u\}$$

$$W_3 = W_2 \sigma_3 = \{P(a, f(a), f(g(y))), P(a, f(a), f(g(y)))\}$$

第三次循环:

③ W_3 已合一, 因为其中包含的表达式相同, 这时

$$\sigma_3 = \{a/z, f(a)/v, g(y)/u\}$$

σ_3 就是 E_1 和 E_2 的 mgu。



第3章 确定性推理

- 3.1 推理的基本概念
- 3.2 3.3 推理的逻辑基础
- 3.4 自然演绎推理
- 3.5 3.6 归结演绎推理



3.4 自然演绎推理

- **自然演绎推理**：从一组已知为真的事实出发，直接运用经典逻辑中的推理规则推出结论的过程称为自然演绎推理。

交换律	$P \vee Q \Leftrightarrow Q \vee P$		
	$P \wedge Q \Leftrightarrow Q \wedge P$	补余律	$P \vee \sim P \Leftrightarrow T$
结合律	$(P \vee Q) \vee R \Leftrightarrow P \vee (Q \vee R)$		$P \wedge \sim P \Leftrightarrow F$
	$(P \wedge Q) \wedge R \Leftrightarrow P \wedge (Q \wedge R)$	逆否定律	$P \rightarrow Q \Leftrightarrow \sim Q \rightarrow \sim P$
分配律	$P \vee (Q \wedge R) \Leftrightarrow (P \vee Q) \wedge (P \vee R)$	连接词化归律	$P \rightarrow Q \Leftrightarrow \sim P \vee Q$
	$P \wedge (Q \vee R) \Leftrightarrow (P \wedge Q) \vee (P \wedge R)$		$P \leftrightarrow Q \Leftrightarrow (P \rightarrow Q) \wedge (Q \rightarrow P)$
德·摩根定律	$\sim (P \vee Q) \Leftrightarrow \sim P \wedge \sim Q$	量词转换律	$P \leftrightarrow Q \Leftrightarrow (P \wedge Q) \vee (\sim P \wedge \sim Q)$
	$\sim (P \wedge Q) \Leftrightarrow \sim P \vee \sim Q$		$\sim (\exists x) P \Leftrightarrow (\forall x) (\sim P)$
否定之否定律	$\sim (\sim P) \Leftrightarrow P$		$\sim (\forall x) P \Leftrightarrow (\exists x) (\sim P)$
吸收律	$P \vee (P \wedge Q) \Leftrightarrow P$	量词分配律	$(\forall x) (P \wedge Q) \Leftrightarrow (\forall x) P \wedge (\forall x) Q$
	$P \wedge (P \vee Q) \Leftrightarrow P$		$(\exists x) (P \vee Q) \Leftrightarrow (\exists x) P \vee (\exists x) Q$

化简式	$P \wedge Q \Rightarrow P, P \wedge Q \Rightarrow Q$
附加式	$P \Rightarrow P \vee Q, Q \Rightarrow P \vee Q$
析取三段论	$\sim P, P \vee Q \Rightarrow Q$
假言推理	$P, P \rightarrow Q \Rightarrow Q$
拒取式	$\sim Q, P \rightarrow Q \Rightarrow \sim P$
假言三段论	$P \rightarrow Q, Q \rightarrow R \Rightarrow P \rightarrow R$
二难推论	$P \vee Q, P \rightarrow R, Q \rightarrow R \Rightarrow R$
全称固化	$(\forall x)P(x) \Rightarrow P(y)$, 其中 y 是个体域上的任一个体, 利用此永真蕴涵式可以消去公式中的全称量词。
存在固化	$(\exists x)P(x) \Rightarrow P(y)$, 其中 y 是个体域中某个使 $P(y)$ 为真的个体, 利用此式可消去公式中的存在量词。



① P 规则:在推理的任何步骤上都可引入前提。

② T 规则:推理时,如果前面步骤中有一个或多个永真蕴涵公式 S ,则可将 S 引入推理过程中。

③ CP 规则:如果能从 R 和前提集合中推出 S 来,则可从前提集合推出 $R \rightarrow S$ 来。

④ 反证法: $P \Rightarrow Q$,当且仅当 $P \wedge \sim Q \Leftrightarrow F$,即 Q 为 P 的逻辑结论,当且仅当 $P \wedge \sim Q$ 是不可满足的。

若推广,可得如下定理。

定理 3.1 Q 为 P_1, P_2, \dots, P_n 的逻辑结论,当且仅当 $(P_1 \wedge P_2 \wedge \dots \wedge P_n) \wedge \sim Q$ 是不可满足的。



3.4 自然演绎推理

- 避免两类错误：肯定后件的错误和否定前件的错误。
- 肯定后件的错误：当 $P \rightarrow Q$ 为真时，希望通过肯定后件 Q 为真来推出前件 P 为真，这是不允许的。原因是指当 $P \rightarrow Q$ 及 Q 为真时，前件 P 既可能为真，也可能为假。
- 否定前件的错误：当 $P \rightarrow Q$ 为真时，希望通过否定前件 P 为假来推出后件 Q 为假，这也是不允许的。原因是指当 $P \rightarrow Q$ 及 P 为假时，后件 Q 既可能为真，也可能为假。

3.4 自然演绎推理

- 自然演绎推理的例子
- 例 设已知如下事实：
 - $A, B, A \rightarrow C, B \wedge C \rightarrow D, D \rightarrow Q$
 - 求证：Q为真。
 - 证明： $A, A \rightarrow C \Rightarrow C$ 假言推理
 - $B, C \Rightarrow B \wedge C$ 引入合取词
 - $B \wedge C, B \wedge C \rightarrow D \Rightarrow D$ 假言推理，P规则，T规则
 - $D, D \rightarrow Q \Rightarrow Q$ 假言推理，P规则，T规则
- 因此，Q为真

3.4 自然演绎推理

- 例 设已知如下事实：
- (1) 只要是需要编程序的课，王程都喜欢。
- (2) 所有的程序设计语言课都是需要编程序的课。
- (3) C是一门程序设计语言课。
- 求证：王程喜欢C这门课。

证明：首先定义谓词

$\text{Prog}(x)$ x 是需要编程序的课。 $\text{Like}(x, y)$ x 喜欢 y 。

$\text{Lang}(x)$ x 是一门程序设计语言课。

把已知事实及待求解问题用谓词公式表示如下：

$\text{Prog}(x) \rightarrow \text{Like}(\text{Wang}, x)$

$(\forall x) (\text{Lang}(x) \rightarrow \text{Prog}(x))$

$\text{Lang}(C)$

$\text{Like}(\text{Wang}, C)$

应用推理规则进行推理：

$\text{Lang}(y) \rightarrow \text{Prog}(y)$

全称固化

$\text{Lang}(C), \text{Lang}(y) \rightarrow \text{Prog}(y) \Rightarrow \text{Prog}(C)$

假言推理 $\{C/y\}$ 及T、P规则

$\text{Prog}(C), \text{Prog}(x) \rightarrow \text{Like}(\text{Wang}, x) \Rightarrow \text{Like}(\text{Wang}, C)$

假言推理 $\{C/x\}$ 及T、P规则

因此，王程喜欢C这门课。



3.4 自然演绎推理

- **优点：**定理证明过程自然，易于理解，并且有丰富的推理规则可用。
- **缺点：**是容易产生知识爆炸，推理过程中得到的中间结论一般按指数规律递增，对于复杂问题的推理不利，甚至难以实现。



第3章 确定性推理

- 3.1 推理的基本概念
- 3.2 3.3 推理的逻辑基础
- 3.4 自然演绎推理
- 3.5 3.6 归结演绎推理



3.5 归结演绎推理

- 归结演绎推理是一种基于鲁宾逊（Robinson）归结原理的机器推理技术。鲁宾逊归结原理亦称为消解原理，是鲁宾逊于1965年在海伯伦（Herbrand）理论的基础上提出的一种基于逻辑的“反证法”。
- 定理证明的实质，就是要对前提F和结论G，证明 $F \rightarrow G$ 永真。
- 由3.3.3节可知，要证明 $F \rightarrow G$ 永真，就是要证明 $F \rightarrow G$ 对于任何一个赋值都是真的。这将是非常困难的，甚至是不可实现的。
- 为此，人们进行了大量的探索，后来发现可以采用反证法的思想，把关于永真性的证明转化为关于不可满足性的证明。
- 即要证明 $F \rightarrow G$ 永真，只要能够证明 $F \wedge \neg G$ 是不可满足的就可以了(原因是 $\neg(F \rightarrow G) \Leftrightarrow \neg(\neg F \vee G) \Leftrightarrow F \wedge \neg G$)。
-



命题逻辑 子句集

1. 子句与子句集

- **定义** 原子公式及其否定统称为文字。

- 例如， p 、 q 、 $\neg p$ 、 $\neg q$ 等都是文字。

- **定义** 任何文字的析取式称为子句。

- 例如， $p \vee q$ 是子句。

- **定义** 不含任何文字的子句称为空子句。

- 由于空子句不含有任何文字，也就不能被任何真假赋值所满足，因此空子句是永假的，不可满足的。

- 空子句一般被记为 \square 或NIL。

- **定义** 由子句或空子句所构成的集合称为子句集。

命题逻辑 子句集

2. 命题公式转化为子句集 (1/4)

- 在命题逻辑中，任何一个命题公式都可以通过应用等价关系及推理规则化成相应的子句集。其化简步骤如下：
 - (1) 消去连接词 “ \rightarrow ” 和 “ \leftrightarrow ”
 - 反复使用如下等价公式：
 - $P \rightarrow Q \Leftrightarrow \neg P \vee Q$
 - $P \leftrightarrow Q \Leftrightarrow (P \wedge Q) \vee (\neg P \wedge \neg Q) \Leftrightarrow (\neg P \wedge Q) \vee (P \wedge \neg Q)$
 - 即可消去谓词公式中的连接词 “ \rightarrow ” 和 “ \leftrightarrow ”。
 - 例如公式
 - $P \rightarrow \neg (Q \rightarrow R)$
 - 经等价变化后为
 - $\neg P \vee \neg(\neg Q \vee R)$

命题逻辑 子句集

2. 命题公式转化为子句集(2/4)

(2) 减少否定符号的辖域

反复使用双重否定率

$$\neg(\neg P) \Leftrightarrow P$$

摩根定律

$$\neg(P1 \wedge P2 \wedge P3...) \Leftrightarrow \neg P1 \vee \neg P2 \vee \neg P3...$$

$$\neg(P1 \vee P2 \vee P3...) \Leftrightarrow \neg P1 \wedge \neg P2 \wedge \neg P3...$$

例如，上式经等价变换后为

$$\neg P \vee (Q \wedge \neg R)$$

命题逻辑 子句集

2. 命题公式转化为子句集(3/4)

(3) 从 \vee 的辖域中消去 \wedge

使用

$$P \vee (Q1 \wedge Q2 \wedge Q3 \dots) \Leftrightarrow (P \vee (Q1)) \wedge (P \vee (Q2)) \wedge (P \vee (Q3)) \dots$$

例如，将上式变为：

$$(\neg P \vee Q) \wedge (\neg P \vee \neg R)$$



命题逻辑 子句集

2. 命题公式转化为子句集(4/4)

(4) 消去合取词

- 消去所有合取词，用子句集的形式表示出来。其中，子句集中的每一个元素都是一个子句。

- 例如，上式的子句集中包含以下两个子句

- $\neg P \vee Q$

- $\neg P \vee \neg R$



谓词逻辑 子句集

1. 子句与子句集

- **定义3.16** 原子谓词公式及其否定统称为文字。

- 例如， $P(x)$ 、 $Q(x)$ 、 $\neg P(x)$ 、 $\neg Q(x)$ 等都是文字。

- **定义3.17** 任何文字的析取式称为子句。

- 例如， $P(x) \vee Q(x)$ ， $P(x, f(x)) \vee Q(x, g(x))$ 都是子句。

- **定义3.18** 不含任何文字的子句称为空子句。

- 由于空子句不含有任何文字，也就不能被任何赋值所满足，因此空子句是永假的，不可满足的。

- 空子句一般被记为 \square 或NIL。

- **定义3.19** 由子句或空子句所构成的集合称为子句集。

谓词逻辑 子句集

2. 谓词公式转化为子句集 (1/6)

- 在谓词逻辑中，任何一个谓词公式都可以通过应用等价关系及推理规则化成相应的子句集。其化简步骤如下：
 - (1) 消去连接词 “ \rightarrow ” 和 “ \leftrightarrow ”
 - 反复使用如下等价公式：
 - $P \rightarrow Q \Leftrightarrow \neg P \vee Q$
 - $P \leftrightarrow Q \Leftrightarrow (P \wedge Q) \vee (\neg P \wedge \neg Q) \Leftrightarrow (\neg P \wedge Q) \vee (P \wedge \neg Q)$
 - 即可消去谓词公式中的连接词 “ \rightarrow ” 和 “ \leftrightarrow ”。
 - 例如公式
 - $(\forall x)((\forall y)P(x,y) \rightarrow \neg (\forall y)(Q(x,y) \rightarrow R(x,y)))$
 - 经等价变化后为
 - $(\forall x)(\neg(\forall y)P(x,y) \vee \neg (\forall y)(\neg Q(x,y) \vee R(x,y)))$

谓词逻辑 子句集

2. 谓词公式转化为子句集(2/6)

(2) 减少否定符号的辖域

反复使用双重否定率

$$\neg(\neg P) \Leftrightarrow P$$

摩根定律

$$\neg(P \wedge Q) \Leftrightarrow \neg P \vee \neg Q$$

$$\neg(P \vee Q) \Leftrightarrow \neg P \wedge \neg Q$$

量词转换率

$$\neg(\forall x)P(x) \Leftrightarrow (\exists x)\neg P(x)$$

$$\neg(\exists x)P(x) \Leftrightarrow (\forall x)\neg P(x)$$

将每个否定符号“ \neg ”移到仅靠谓词的位置，使得每个否定符号最多只作用于一个谓词上。

例如，上式经等价变换后为

$$(\forall x)((\exists y)\neg P(x, y) \vee (\exists y)(Q(x, y) \wedge \neg R(x, y)))$$

谓词逻辑 子句集

2. 谓词公式转化为子句集(3/6)

(3) 对变元标准化

使不同量词约束的变元有不同的名字。即使所有不相干的变元的名字均不同，并且自由变元及约束变元亦不同。

例如，上式经变换后为

$$(\forall x)((\exists y)\neg P(x, y) \vee (\exists z)(Q(x, z) \wedge \neg R(x, z)))$$

(4) 化为前束范式

化为前束范式的方法:把所有量词都移到公式的左边，并且在移动时不能改变其相对顺序。由于第(3)步已对变元进行了标准化，每个量词都有自己的变元，这就消除了任何由变元引起冲突的可能，因此这种移动是可行的。

例如，上式化为前束范式后为

$$(\forall x)(\exists y)(\exists z)(\neg P(x, y) \vee (Q(x, z) \wedge \neg R(x, z)))$$

谓词逻辑 子句集

2. 谓词公式转化为子句集(4/6)

(5) 消去存在量词

消去存在量词时，需要区分以下两种情况：

若存在量词不出现在全称量词的辖域内（即它的左边没有全称量词），只要用一个新的个体常量替换受该存在量词约束的变元，就可消去该存在量词。

若存在量词位于一个或多个全称量词的辖域内，例如

$$(\forall x_1) \dots (\forall x_n) (\exists y) P(x_1, x_2, \dots, x_n, y)$$

则需要用Skolem函数 $f(x_1, x_2, \dots, x_n)$ 替换受该存在量词约束的变元 y ，然后再消去该存在量词。

例如，上步所得公式中存在量词 $(\exists y)$ 和 $(\exists z)$ 都位于 $(\forall x)$ 的辖域内，因此都需要用Skolem函数来替换。设替换 y 和 z 的Skolem函数分别是 $f(x)$ 和 $g(x)$ ，则替换后的式子为

$$(\forall x) (\neg P(x, f(x)) \vee (Q(x, g(x)) \wedge \neg R(x, g(x))))$$

谓词逻辑 子句集

2. 谓词公式转化为子句集(5/6)

(6) 化为Skolem标准形

Skolem标准形的一般形式为

$$(\forall x_1) \dots (\forall x_n) M(x_1, x_2, \dots, x_n)$$

其中, $M(x_1, x_2, \dots, x_n)$ 是Skolem标准形的母式, 它由子句的合取所构成。

把谓词公式化为Skolem标准形需要使用以下等价关系

$$P \vee (Q \wedge R) \Leftrightarrow (P \vee Q) \wedge (P \vee R)$$

例如, 前面的公式化为Skolem标准形后为

$$(\forall x)((\neg P(x, f(x)) \vee Q(x, g(x))) \wedge (\neg P(x, f(x)) \vee \neg R(x, g(x))))$$

(7) 消去全称量词

由于母式中的全部约束变元均受全称量词的约束, 并且全称量词的次序已无关紧要, 因此可以省掉全称量词。但剩下的母式, 仍假设其变元是被全称量词量化的。

例如, 上式消去全称量词后为

$$(\neg P(x, f(x)) \vee Q(x, g(x))) \wedge (\neg P(x, f(x)) \vee \neg R(x, g(x)))$$

谓词逻辑 子句集

2. 谓词公式转化为子句集(6/6)

(8) 消去合取词

在母式中消去所有合取词，把母式用子句集的形式表示出来。其中，子句集中的每一个元素都是一个子句。

例如，上式的子句集中包含以下两个子句

$$\neg P(x, f(x)) \vee Q(x, g(x))$$

$$\neg P(x, f(x)) \vee \neg R(x, g(x))$$

(9) 更换变量名称

对子句集中的某些变量重新命名，使任意两个子句中不出现相同的变量名。这样，更换变量名是不会影响公式的真值的。

例如，对前面的公式，可把第二个子句集中的变元名 x 更换为 y ，得到如下子句集

$$\neg P(x, f(x)) \vee Q(x, g(x))$$

$$\neg P(y, f(y)) \vee \neg R(y, g(y))$$



例

例 将合式公式化为子句形。

$$\forall x[P(x) \rightarrow [\forall y[P(y) \rightarrow P(f(x, y))] \wedge \sim \forall y[Q(x, y) \rightarrow P(y)]]]$$

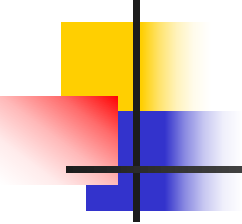
解：（1）消去蕴涵符号：

这可以利用等价式： $P \rightarrow Q \Leftrightarrow \sim P \vee Q$

得到：

$$\forall x[(\sim P(x) \vee [\forall y[\sim P(y) \vee P(f(x, y))] \wedge \sim \forall y[\sim Q(x, y) \vee P(y)]]]$$

（2）减少否定符号的辖域，把“ \sim ”移到紧靠谓词的位置上：



得到:

$$\forall x[(\sim P(x) \vee [\forall y[\sim P(y) \vee P(f(x,y))]] \wedge \exists y[Q(x,y) \wedge \sim P(y)])]$$

(3) 变量标准化: 重新命名变元名, 使不同量词约束的变元有不同的名字:

$$\forall x[\sim P(x) \vee [\forall y[\sim P(y) \vee P(f(x,y))]] \wedge \exists w[Q(x,w) \wedge \sim P(w)]]]$$

(4) 消去存在量词:

$$\forall x[\sim P(x) \vee [\forall y[\sim P(y) \vee P(f(x,y))]] \wedge [Q(x,g(x)) \wedge \sim P(g(x))]]]$$



(5) 化为前束形:

$$\forall x \forall y [\sim P(x) \vee [[\sim P(y) \vee P(f(x,y))] \wedge [Q(x,g(x)) \wedge \sim P(g(x))]]]$$

(6) 化为skolem标准型:

$$\forall x \forall y [\sim P(x) \vee \sim P(y) \vee P(f(x,y))] \wedge [\sim P(x) \vee Q(x,g(x))] \\ \wedge [\sim P(x) \vee \sim P(g(x))]$$

(7) 消去全称量词和 (8) 合取连接词:

$$[\sim P(x) \vee \sim P(y) \vee P(f(x,y))]$$

$$[\sim P(x) \vee Q(x,g(x))]$$

$$[\sim P(x) \vee \sim P(g(x))]$$



(9) 更改变量名，有时称为变量分离标准化。
于是有：

$$\sim P(x_1) \vee \sim P(y) \vee P(f(x_1, y))$$

$$\sim P(x_2) \vee Q(x_2, g(x_2))$$

$$\sim P(x_3) \vee \sim P(g(x_3))$$

注意：一个子句内的文字可以含有变量，但这些变量总是被理解为全称量词量化了的变量。



子句集

- 在上述化简过程中，由于在消去存在量词时所用的Skolem函数可以不同，因此化简后的标准子句集是不唯一的。
-
- **定理** 设有命题公式 F ，其标准子句集为 S ，则 F 为不可满足的充要条件是 S 为不可满足的。
- **定理3.2** 设有谓词公式 F ，其标准子句集为 S ，则 F 为不可满足的充要条件是 S 为不可满足的。



■ $P = P_1 \wedge P_2 \wedge \dots \wedge P_n$

设P的子句集是 S_p ， P_i 的子句集是 S_i

一般情况下， S_p 并不等于 $S_1 \cup S_2 \dots \cup S_n$ ，而是比 $S_1 \cup S_2 \dots \cup S_n$ 复杂得多。但在不可满足的意义下，子句集 S_p 和 $S_1 \cup S_2 \dots \cup S_n$ 是一致的，即：

S_p 不可满足 $\Leftrightarrow S_1 \cup S_2 \dots \cup S_n$ 不可满足

这样对于 S_p 的讨论就可以用 $S_1 \cup S_2 \dots \cup S_n$ 来代替



Herbrand理论

- **Herbrand**定理只是从理论上给出证明子句集不可满足性的可行性及方法，从而给出了证明谓词公式的不可满足性的可行性及方法。
- **1965年，Robinson** 提出了归结原理，才让机器定理证明变为现实。

归结原理

首先注意以下关键问题：

- 第一，子句集中的子句之间是合取关系。因此，子句集中只要有一个子句为不可满足，则整个子句集就是不可满足的；
- 第二，空子句是不可满足的。
- 因此，一个子句集中如果包含有空子句，则此子句集就一定是不可满足的。



归结原理

1. 基本思想

- 归结原理基本思想
- 首先把欲证明问题的结论否定转换为子句集 S'' ，并加入前提的子句集 S' ，得到一个扩充的子句集 S 。然后设法检验子句集 S 是否含有空子句，若含有空子句，则表明 S 是不可满足的；若不含有空子句，则使用归结法，在子句集中选择合适的子句进行归结，直至导出空子句或不能继续归结为止。
- 归结原理包括
- 命题逻辑归结原理
- 谓词逻辑归结原理

归结原理

2. 命题逻辑的归结 (1/8)

(1) 归结式的定义及性质

定义3.25 若 P 是原子命题公式, 则称 P 与 $\neg P$ 为互补文字。

- **定义3.26** 设 C_1 和 C_2 是子句集中的任意两个子句, 如果 C_1 中的文字 L_1 与 C_2 中的文字 L_2 互补, 那么可从 C_1 和 C_2 中分别消去 L_1 和 L_2 , 并将 C_1 和 C_2 中余下的部分按析取关系构成一个新的子句 C_{12} , 则称这一过程为归结, 称 C_{12} 为 C_1 和 C_2 的归结式, 称 C_1 和 C_2 为 C_{12} 的亲本子句。
- 例 设 $C_1 = P \vee Q \vee R$, $C_2 = \neg P \vee S$, 求 C_1 和 C_2 的归结式 C_{12} 。
- 解: 这里 $L_1 = P$, $L_2 = \neg P$, 通过归结可以得到
$$C_{12} = Q \vee R \vee S$$

归结原理

2. 命题逻辑的归结 (2/8)

■ (1) 归结式的定义及性质

■ 定理3.6 归结式 C_{12} 是其亲本子句 C_1 和 C_2 的逻辑结论。

这个定理是归结原理中一个重要定理，由它可得到如下两个推论：

推论1：设 C_1 和 C_2 是子句集 S 中的两个子句， C_{12} 是 C_1 和 C_2 的归结式，若用 C_{12} 代替 C_1 和 C_2 后得到新的子句集 S_1 ，则由 S_1 的不可满足性可以推出原子句集 S 的不可满足性。即：

S_1 的不可满足性 $\Rightarrow S$ 的不可满足性

推论2：设 C_1 和 C_2 是子句集 S 中的两个子句， C_{12} 是 C_1 和 C_2 的归结式，若把 C_{12} 加入 S 中得到新的子句集 S_2 ，则 S 与 S_2 的不可满足性是等价的。即：

S_2 的不可满足性 $\Leftrightarrow S$ 的不可满足性

归结原理

2. 命题逻辑的归结 (4/8)

- 由此得到下列结论：
- (1) 为证明子句集 S 的不可满足性，只要对其中可进行归结的子句进行归结，并把归结式加入到子句集 S 中；或者用归结式代替它的亲本子句，然后对新的子句集证明其不可满足性就可以了。
- (2) 如果经归结得到空子句，根据空子句的不可满足性，即可得到原子句集 S 是不可满足的结论。

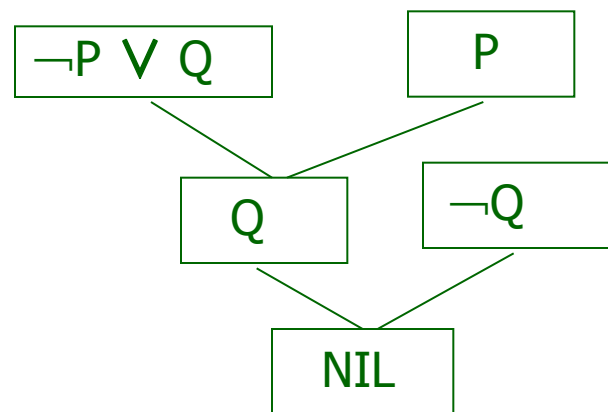
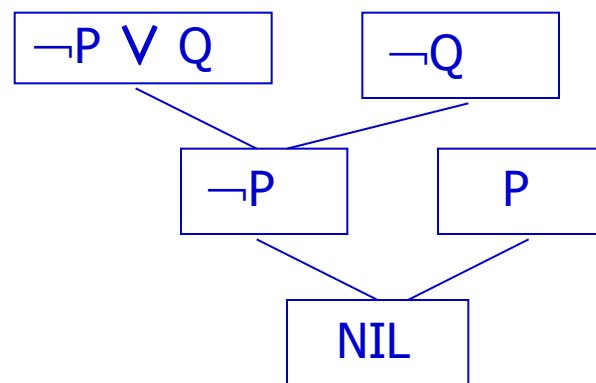
命题逻辑的归结原理的可靠性和完备性：

- **定理：** 子句集 S 是不可满足的，当且仅当存在一个从 S 到空子句的归结过程。

归结原理

2. 命题逻辑的归结 (5/8)

- 例4.9 假设 $C_1 = \neg P \vee Q$, $C_2 = \neg Q$, $C_3 = P$, 求 C_1 、 C_2 、 C_3 的归结式 C_{123} 。
- 解：若先对 C_1 、 C_2 归结，可得到
- $C_{12} = \neg P$
- 然后再对 C_{12} 和 C_3 归结，得到
- $C_{123} = \text{NIL}$
- 如果改变归结顺序，同样可以得到相同的结果，即其归结过程是不唯一的。
- 其归结过程可用右图来表示，该树称为归结树。





归结原理

2. 命题逻辑的归结 (6/8)

- (2) 命题逻辑的归结反演

- 应用归结原理证明定理的过程称为归结反演。

归结原理

2. 命题逻辑的归结 (7/8)

命题逻辑归结反演的步骤:

在命题逻辑中, 已知**F**, 证明**G**为真的归结反演过程如下:

- ①否定目标公式**G**, 得 $\neg G$;
- ②把 $\neg G$ 并入到公式集**F**中, 得到 $\{F, \neg G\}$;
- ③把 $\{F, \neg G\}$ 化为子句集**S**。
- ④应用归结原理对子句集**S**中的子句进行归结, 并把每次得到的归结式并入**S**中。如此反复进行, 若出现空子句, 则停止归结, 此时就证明了**G**为真。

归结原理

2. 命题逻辑的归结 (8/8)

■ 例 设已知的公式集为:

$\{P, (P \wedge Q) \rightarrow R, (S \vee T) \rightarrow Q, T\}$

求证结论R。

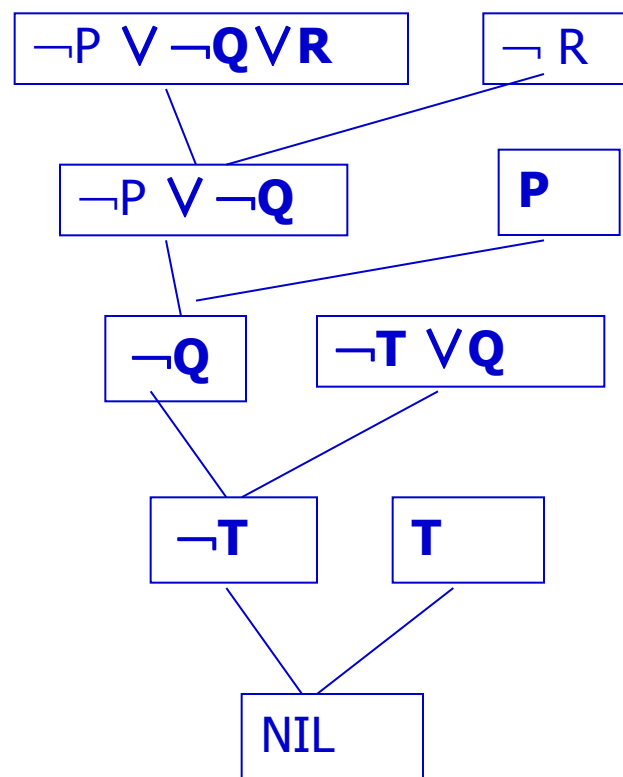
■ 解: 假设结论R为假, 将 $\neg R$ 加入公式集, 并化为子句集

■ $S = \{P, \neg P \vee \neg Q \vee R, \neg S \vee Q, \neg T \vee Q, T, \neg R\}$

■ 其归结过程如右图的归结演绎树所示。该树根为空子句。

■ 其含义为: 先假设子句集S中的所有子句均为真, 即原公式集为真, $\neg R$ 也为真; 然后利用归结原理, 对子句集进行归结, 并把所得的归结式并入子句集中; 重复这一过程, 最后归结出了空子句。

■ 根据归结原理的可靠性, 可知子句集S是不可满足的, 即开始时假设 $\neg R$ 为真是错误的, 这就证明了R为真。





归结原理

3. 谓词逻辑的归结 (1/20)

- 在谓词逻辑中，由于子句集中的谓词一般都含有常量、函数和变量，因此不能象命题逻辑那样直接消去互补文字。而需要先用一个**最一般合一 (MGU)** 对变元进行代换，然后才能进行归结。
- 可见，**谓词逻辑的归结**要比**命题逻辑的归结**麻烦一些。

归结原理

3. 谓词逻辑的归结 (2/20)

- 谓词逻辑的归结原理：
- 谓词逻辑中的归结式可用如下定义来描述：
- **定义3.27** 设 C_1 和 C_2 是两个没有公共变元的子句， L_1 和 L_2 分别是 C_1 和 C_2 中的文字。如果 L_1 和 $\neg L_2$ 存在最一般合一 σ ，则称
$$C_{12} = (C_1\sigma - L_1\sigma) \cup (C_2\sigma - L_2\sigma)$$
为 C_1 和 C_2 的二元归结式，而 L_1 和 L_2 为归结式上的文字。

归结原理

3. 谓词逻辑的归结 (3/20)

- 谓词逻辑归结举例:
- 例 设 $C_1=P(a) \vee R(x)$, $C_2=\neg P(y) \vee Q(b)$, 求 C_{12}
- 解: 取 $L_1=P(a)$, $L_2=\neg P(y)$, 则 L_1 和 $\neg L_2$ 的最一般合一是 $\sigma=\{a/y\}$ 。根据定义3.27,可得
- $$C_{12}=(C_1\sigma-L_1\sigma) \cup (C_2\sigma-L_2\sigma)$$
- $$=({P(a), R(x)}-\{P(a)\}) \cup (\{\neg P(a), Q(b)\}-\{\neg P(a)\})$$
- $$=({R(x)}) \cup (\{Q(b)\})= \{R(x), Q(b)\}$$
- $$\rightarrow R(x) \vee Q(b)$$

归结原理

3. 谓词逻辑的归结 (4/20)

- 谓词逻辑归结举例：
- 例 设 $C_1 = P(x) \vee Q(a)$, $C_2 = \neg P(b) \vee R(x)$, 求 C_{12}
- 解：由于 C_1 和 C_2 有相同的变元 x , 不符合定义3.27的要求。为了进行归结, 需要修改 C_2 中变元 x 的名字为 y , 令 $C_2 = \neg P(b) \vee R(y)$ 。此时 $L_1 = P(x)$, $L_2 = \neg P(b)$, L_1 和 $\neg L_2$ 的最一般合一 $\sigma = \{b/x\}$ 。则有
- $$C_{12} = (C_1\sigma - L_1\sigma) \cup (C_2\sigma - L_2\sigma)$$
- $$= (\{P(b), Q(a)\} - \{P(b)\}) \cup (\{\neg P(b), R(y)\} - \{\neg P(b)\})$$
- $$= (\{Q(a)\}) \cup (\{R(y)\}) = \{Q(a), R(y)\}$$
- $$\rightarrow Q(a) \vee R(y)$$

归结原理

3. 谓词逻辑的归结 (5/20)

- 对以上讨论做以下两点说明:
- (1) 这里之所以使用集合符号和集合的运算, 目的是为了说明问题的方便。
■ 即先将子句 $C_i\sigma$ 和 $L_i\sigma$ 写成集合的形式, 在集合表示下做减法和并集运算, 然后再写成子句集的形式。
- (2) 定义中还要求 C_1 和 C_2 无公共变元, 这也是合理的。
- 例 $C_1=P(x)$, $C_2=\neg P(f(x))$, 而 $S=\{C_1, C_2\}$ 是不可满足的。但由于 C_1 和 C_2 的变元相同, 就无法合一了。没有归结式, 就不能用归结法证明 S 的不可满足性, 这就限制了归结法的使用范围。
- 如果对 C_1 或 C_2 的变元进行换名, 便可通过合一, 对 C_1 和 C_2 进行归结。如上例, 若先对 C_2 进行换名, 即 $C_2=\neg P(f(y))$, 则可对 C_1 和 C_2 进行归结, 得到一个空子句, 从而证明了 S 是不可满足的。
- 事实上, 在由公式集化为子句集的过程中, 其最后一步就是做换名处理。因此, 定义中假设 C_1 和 C_2 没有相同变元是可以的。

归结原理

3. 谓词逻辑的归结 (6/20)

- (3) 再看几个归结例子:
- 例 设 $C_1 = P(x) \vee \neg Q(b)$, $C_2 = \neg P(a) \vee Q(y) \vee R(z)$
- 解: 对 C_1 和 C_2 通过最一般合一 ($\sigma = \{a/x, b/y\}$) 的作用, 可以得到两个互补对。
- 注意:
- 求归结式不能同时消去两个互补对, 这样的结果不是二元归结式。如在 $\sigma = \{a/x, b/y\}$ 下, 若同时消去两个互补对, 所得的 $R(z)$ 不是 C_1 和 C_2 的二元归结式。

归结原理

3. 谓词逻辑的归结 (7/20)

(4) 例 设 $C_1 = P(x) \vee P(f(a)) \vee Q(x)$, $C_2 = \neg P(y) \vee R(b)$, 求 C_{12}

解: 对参加归结的某个子句, 若其内部有可合一的文字, 则在进行归结之前应先对这些文字进行合一。本例的 C_1 中有可合一的文字 $P(x)$ 与 $P(f(a))$, 若用它们的最一般合一 $\sigma = \{f(a)/x\}$ 进行代换, 可得到

- $C_1\sigma = P(f(a)) \vee Q(f(a))$
- 此时可对 $C_1\sigma$ 与 C_2 进行归结。选 $L_1 = P(f(a))$, $L_2 = \neg P(y)$, L_1 和 L_2 的最一般合一是 $\sigma = \{f(a)/y\}$, 则可得到 C_1 和 C_2 的二元归结式为
- $C_{12} = R(b) \vee Q(f(a))$
- 我们把 $C_1\sigma$ 称为 C_1 的因子。
- 一般来说, 若子句 C 中有两个或两个以上的文字具有最一般合一 σ , 则称 $C\sigma$ 为子句 C 的因子。如果 $C\sigma$ 是一个单文字, 则称它为 C 的单元因子。
- 应用因子概念, 可对谓词逻辑中的归结原理给出如下定义:

归结原理

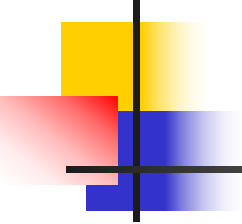
3. 谓词逻辑的归结 (8/20)

- 谓词逻辑中的二元归结原理（应用因子的概念）
- **定义3.28** 若 C_1 和 C_2 是无公共变元的子句，则
 - ① C_1 和 C_2 的二元归结式；
 - ② C_1 和 C_2 的因子 $C_2\sigma_2$ 的二元归结式；
 - ③ C_1 的因子 $C_1\sigma_1$ 和 C_2 的二元归结式；
 - ④ C_1 的因子 $C_1\sigma_1$ 和 C_2 的因子 $C_2\sigma_2$ 的二元归结式。
- 这四种二元归结式都是子句 C_1 和 C_2 的**二元归结式**，记为 C_{12} 。

归结原理

3. 谓词逻辑的归结 (9/20)

- 例 设 $C_1 = P(y) \vee P(f(x)) \vee Q(g(x))$, $C_2 = \neg P(f(g(a))) \vee Q(b)$, 求 C_{12} 。
- 解: 对 C_1 , 取最一般合一 $\sigma_1 = \{f(x)/y\}$, 得 C_1 的因子
$$C_1\sigma_1 = P(f(x)) \vee Q(g(x))$$
- 对 C_1 的因子和 C_2 归结 ($\sigma_2 = \{g(a)/x\}$) , 可得到 C_1 和 C_2 的二元归结式
$$C_{12} = Q(g(g(a))) \vee Q(b)$$



对谓词逻辑，**定理3.6**仍然适用，即**归结式 C_{12} 是其亲本子句 C_1 和 C_2 的逻辑结论**。用归结式取代它在子句集 S 中的亲本子句（或者将归结式加入子句集 S ），所得到的子句集仍然保持着原子句集 S 的不可满足性。

此外，从不可满足的意义上说，一阶谓词逻辑的归结原理也是可靠的和完备的：

定理：子句 S 集是不可满足的，当且仅当存在一个从 S 到空子句的归结过程。

谓词逻辑的归结反演过程与命题逻辑的归结反演过程相比，其步骤基本相同，但每步的处理对象不同。

归结原理

3. 谓词逻辑的归结 (10/20)

谓词逻辑归结反演的步骤：

在谓词逻辑中，已知**F**，证明**G**为真的归结反演过程如下：

①否定目标公式**G**，得 $\neg G$ ；

②把 $\neg G$ 并入到公式集**F**中，得到 $\{F, \neg G\}$ ；

③把 $\{F, \neg G\}$ 化为子句集**S**。

④应用归结原理对子句集**S**中的子句进行归结（考虑两个亲本子句的最一般合一），并把每次得到的归结式并入**S**中。如此反复进行，若出现空子句，则停止归结，此时就证明了**G**为真。

归结原理

3. 谓词逻辑的归结 (11/20)

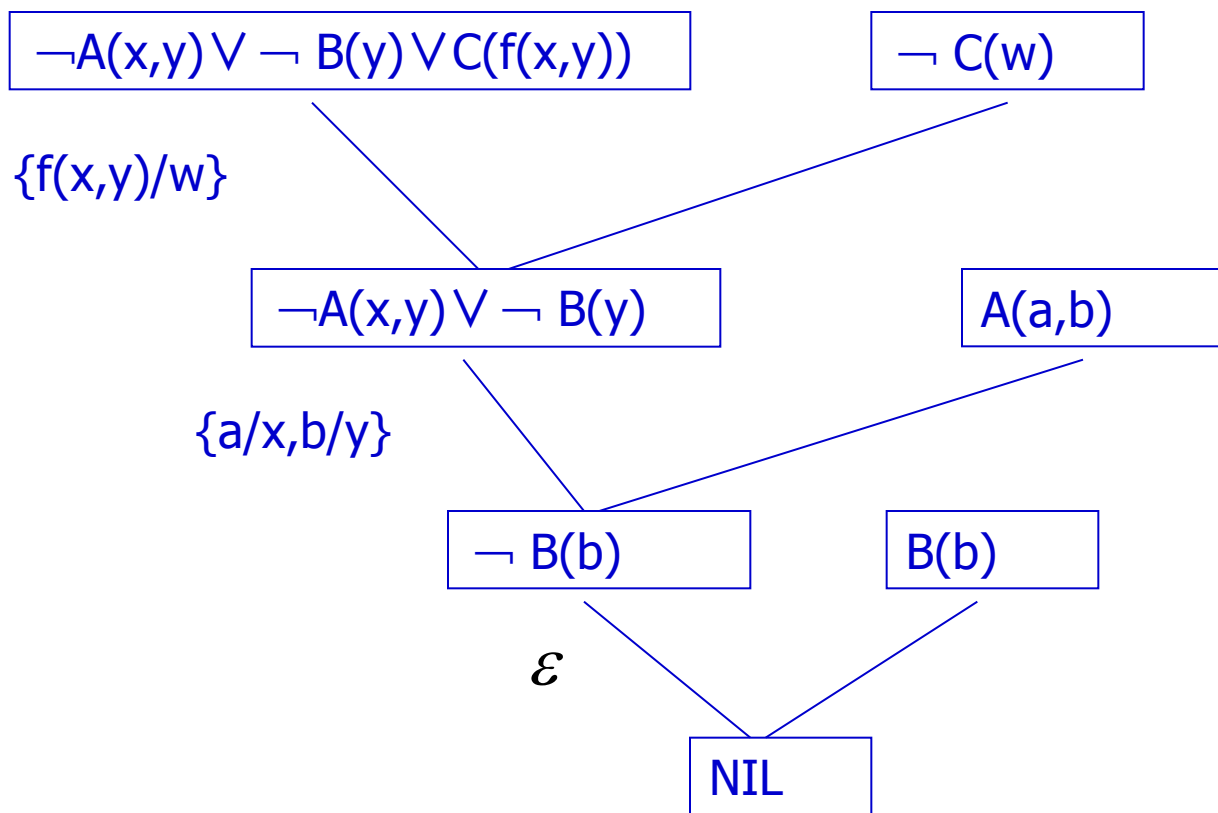
例 已知 $F: (\forall x)((\exists y)(A(x, y) \wedge B(y)) \rightarrow (\exists y)(C(y) \wedge D(x, y)))$

$G: \neg(\exists x)C(x) \rightarrow (\forall x)(\forall y)(A(x, y) \rightarrow \neg B(y))$

■ 求证G是F的逻辑结论。

- **证明:** 先把G否定, 并放入F中, 得到的 $\{F, \neg G\}$ 为
- $\{(\forall x)((\exists y)(A(x, y) \wedge B(y)) \rightarrow (\exists y)(C(y) \wedge D(x, y))),$
- $\neg(\neg(\exists x)C(x) \rightarrow (\forall x)(\forall y)(A(x, y) \rightarrow \neg B(y)))\}$
- 再把 $\{F, \neg G\}$ 化成子句集, 得到
- (1) $\neg A(x, y) \vee \neg B(y) \vee C(f(x, y))$
- (2) $\neg A(u, v) \vee \neg B(v) \vee D(u, f(u, v))$
- (3) $\neg C(w)$
- (4) $A(a, b)$
- (5) $B(b)$
- 其中, (1)、(2)是由F化出的两个子句, (3)、(4)、(5)是由 $\neg G$ 化出的3个子句。
- 最后应用谓词逻辑的归结原理对上述子句集进行归结, 其过程为
- (6) $\neg A(x, y) \vee \neg B(y)$ 由(1)和(3)归结, 取 $\sigma = \{f(x, y)/w\}$
- (7) $\neg B(b)$ 由(4)和(6)归结, 取 $\sigma = \{a/x, b/y\}$
- (8) NIL 由(5)和(7)归结, 取 $\sigma = \mathcal{E}$
- 因此, G是F的逻辑结论。
- 上述归结过程可用如下归结树来表示

谓词逻辑的归结 (12/20)



归结原理

3. 谓词逻辑的归结 (13/20)

■ 为了进一步加深对谓词逻辑归结的理解，下面再给出一个经典的归结问题

■ 例 “快乐学生” 问题

■ 假设：任何通过计算机考试并获奖的人都是快乐的，任何肯学习或幸运的人都可以通过所有考试，张不肯学习但他是幸运的，任何幸运的人都能获奖。

■ 求证：张是快乐的。

■ 解：先定义谓词：

■ $\text{Pass}(x, y)$ x 可以通过 y 考试

■ $\text{Win}(x, \text{prize})$ x 能获得奖励

■ $\text{Study}(x)$ x 肯学习

■ $\text{Happy}(x)$ x 是快乐的

■ $\text{Lucky}(x)$ x 是幸运的

归结原理

3. 谓词逻辑的归结 (14/20)

- 再将问题用谓词表示如下：
- “任何通过计算机考试并奖的人都是快乐的”
 $(\forall x)(\text{Pass}(x, \text{computer}) \wedge \text{Win}(x, \text{prize}) \rightarrow \text{Happy}(x))$
- “任何肯学习或幸运的人都可以通过所有考试”
 $(\forall x) (\forall y) (\text{Study}(x) \vee \text{Lucky}(x) \rightarrow \text{Pass}(x, y))$
- “张不肯学习但他是幸运的”
 $\neg \text{Study}(\text{zhang}) \wedge \text{Lucky}(\text{zhang})$
- “任何幸运的人都能获奖”
 $(\forall x) (\text{Lucky}(x) \rightarrow \text{Win}(x, \text{prize}))$
- 结论 “张是快乐的” 的否定
 $\neg \text{Happy}(\text{zhang})$

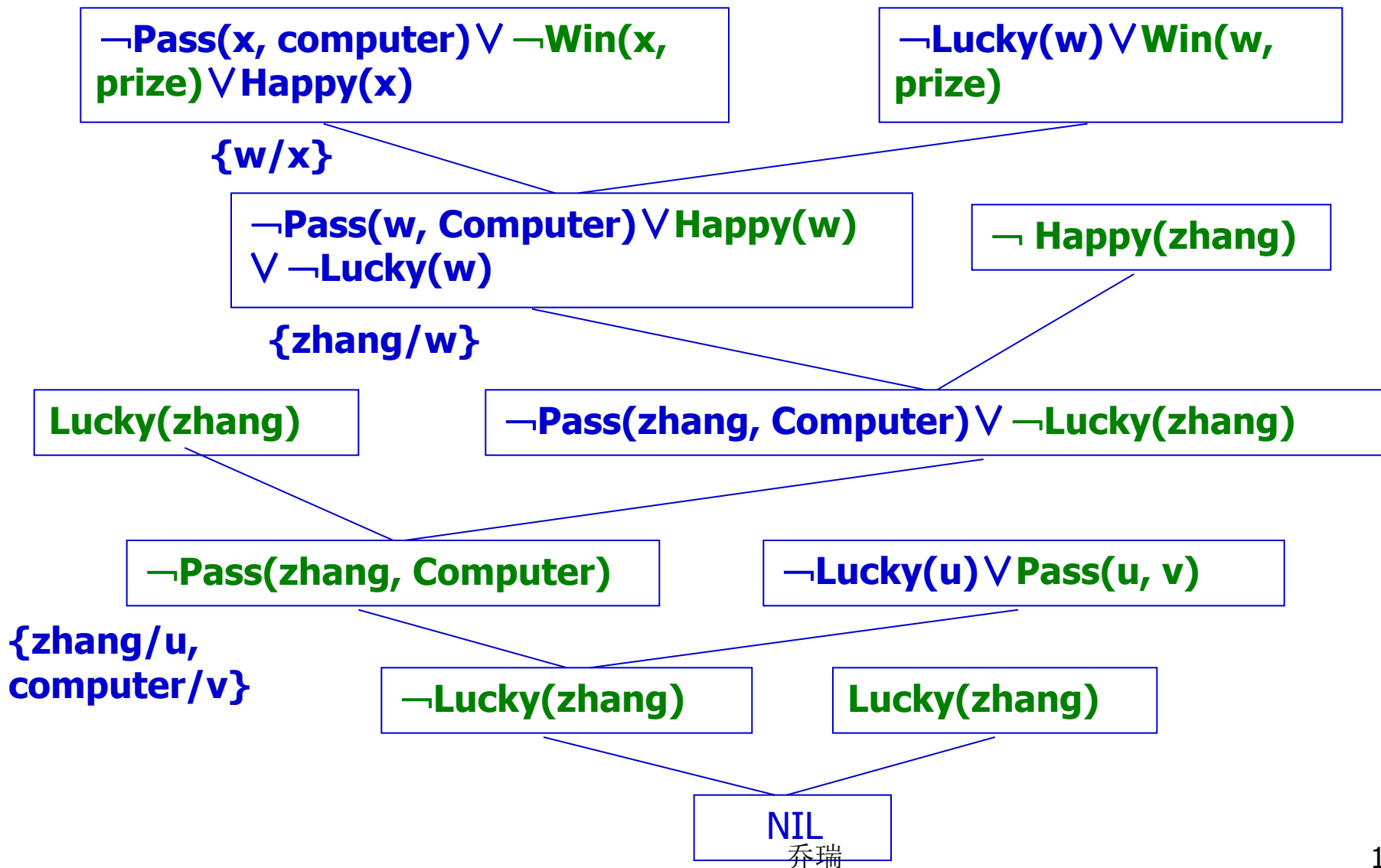
归结原理

3. 谓词逻辑的归结 (15/20)

- 将上述谓词公式转化为子句集如下：
 - (1) $\neg \text{Pass}(x, \text{computer}) \vee \neg \text{Win}(x, \text{prize}) \vee \text{Happy}(x)$
 - (2) $\neg \text{Study}(y) \vee \text{Pass}(y, z)$
 - (3) $\neg \text{Lucky}(u) \vee \text{Pass}(u, v)$
 - (4) $\neg \text{Study}(\text{zhang})$
 - (5) $\text{Lucky}(\text{zhang})$
 - (6) $\neg \text{Lucky}(w) \vee \text{Win}(w, \text{prize})$
 - (7) $\neg \text{Happy}(\text{zhang})$ (结论的否定)

归结原理

谓词逻辑的归结 (16/20)



归结原理

3. 谓词逻辑的归结 (17/20)

■ 为了进一步加深对谓词逻辑归结的理解，下面再给出一个经典的归结问题

■ 例“激动人心的生活”问题

■ 假设：所有不贫穷并且聪明的人都是快乐的，那些看书的人是聪明的。李明能看书且不贫穷，快乐的人过着激动人心的生活。

■ 求证：李明过着激动人心的生活。

■ 解：先定义谓词：

■ $Poor(x)$ x 是贫穷的

■ $Smart(x)$ x 是聪明的

■ $Happy(x)$ x 是快乐的

■ $Read(x)$ x 能看书

■ $Exciting(x)$ x 过着激动人心的生活

归结原理

3. 谓词逻辑的归结 (18/20)

- 再将问题用谓词表示如下：
- “所有不贫穷并且聪明的人都是快乐的”
- $(\forall x)((\neg \text{Poor}(x) \wedge \text{Smart}(x)) \rightarrow \text{Happy}(x))$
- “那些看书的人是聪明的”
- $(\forall y)(\text{Read}(y) \rightarrow \text{Smart}(y))$
- “李明能看书且不贫穷”
- $\text{Read}(\text{Liming}) \wedge \neg \text{Poor}(\text{Liming})$
- “快乐的人过着激动人心的生活”
- $(\forall z)(\text{Happy}(z) \rightarrow \text{Exciting}(z))$
- 目标 “李明过着激动人心的生活” 的否定
- $\neg \text{Exciting}(\text{Liming})$

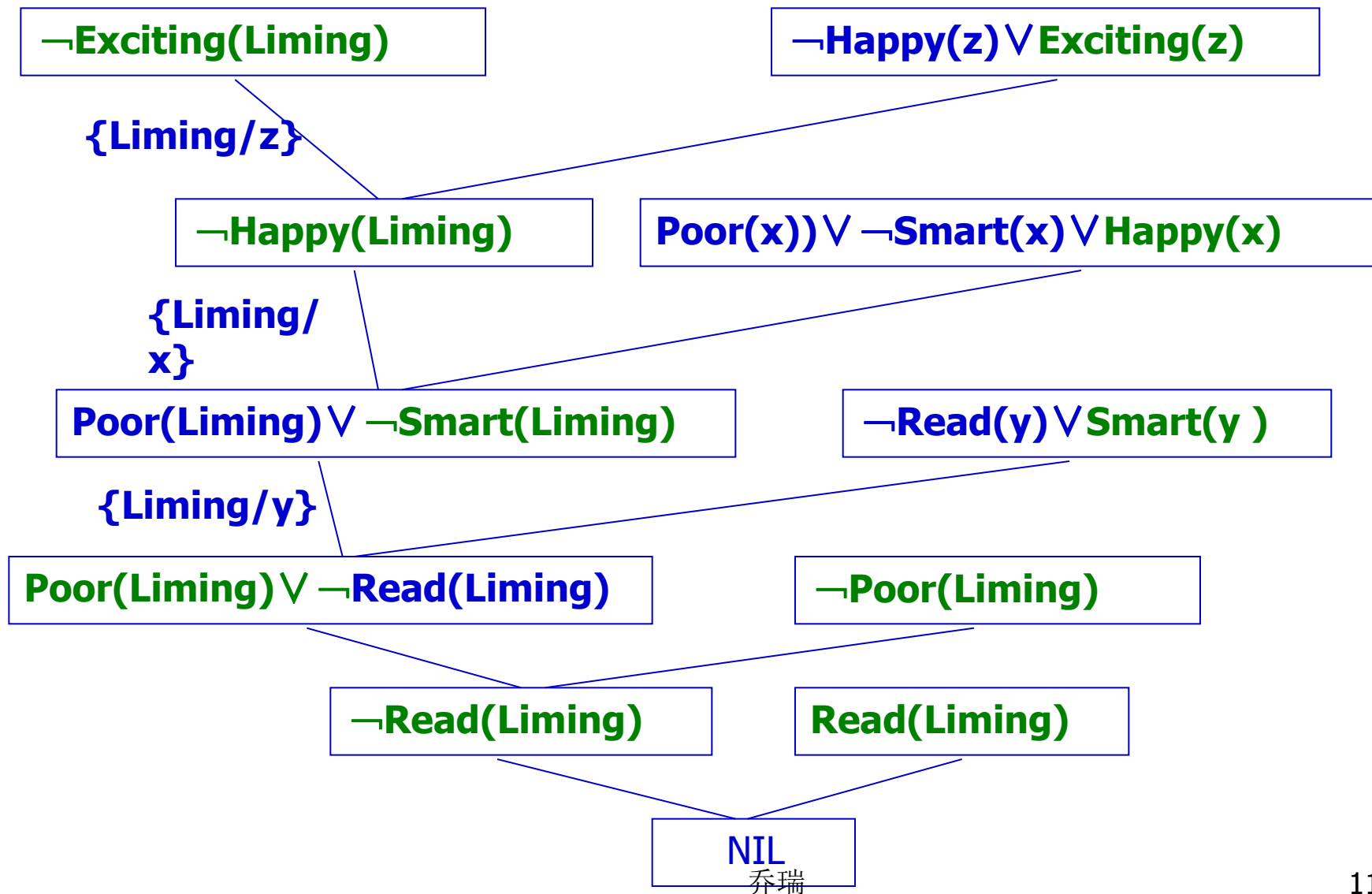
归结原理

3. 谓词逻辑的归结 (19/20)

- 将上述谓词公式转化为子句集如下：
- (1) $\text{Poor}(x) \vee \neg \text{Smart}(x) \vee \text{Happy}(x)$
- (2) $\neg \text{Read}(y) \vee \text{Smart}(y)$
- (3) $\text{Read}(\text{Liming})$
- (4) $\neg \text{Poor}(\text{Liming})$
- (5) $\neg \text{Happy}(z) \vee \text{Exciting}(z)$
- (6) $\neg \text{Exciting}(\text{Liming})$ (结论的否定)

归结原理

3. 谓词逻辑的归结 (20/20)



用归结反演求取问题的答案

(1/4)

- 归结原理除了可用于定理证明外，还可用来求取问题答案，其思想与定理证明相似。其一般步骤为：
- **(1)** 把问题的已知条件用谓词公式表示出来，并化为相应的子句集，设该子句集的名字为**S1**；
- **(2)** 把待求解的问题也用谓词公式表示出来，然后将其否定，并与一谓词**ANSWER**构成析取式。**ANSWER**的变量必须与问题公式的变量完全一样。
- **(3)** 把问题公式与谓词**ANSWER**构成的析取式化为子句集，并把该子句集与**S1**合并构成子句集**S**。
- **(4)** 对子句集**S**应用谓词归结原理进行归结，在归结的过程中，通过合一置换，改变**ANSWER**中的变元。
- **(5)** 如果得到归结式**ANSWER**，则问题的答案即在**ANSWER**谓词中。

用归结反演求取问题的答案

(2/4)

- 下面再通过一个例子来说明如何求取问题的答案。
- **例** 已知：“张和李是同班同学，如果x和y是同班同学，则x的教室也是y的教室，现在张在302教室上课。”
- 问：“现在李在哪个教室上课？”
- **解**：首先定义谓词：
 - $C(x, y)$ x和y是同班同学；
 - $At(x, u)$ x在u教室上课。
- 把已知前提用谓词公式表示如下：
 - $C(zhang, li)$
 - $(\forall x) (\forall y) (\forall u) (C(x, y) \wedge At(x, u) \rightarrow At(y, u))$
 - $At(zhang, 302)$
- 把目标的否定用谓词公式表示如下：
 - $\neg At(li, v)$

用归结反演求取问题的答案

(3/4)

- 把上述公式化为子句集:

1 $C(\text{zhang}, \text{li})$

2 $\neg C(x, y) \vee \neg At(x, u) \vee At(y, u)$

3 $At(\text{zhang}, 302)$

- 把目标的否定化成子句式，并与Answer(v)析取:

4 $\neg At(\text{li}, v) \vee Answer(v)$

5 $\neg C(x, \text{li}) \vee \neg At(x, v) \vee Answer(v)$ 由2和4归结 $(\text{li}/y, v/u)$

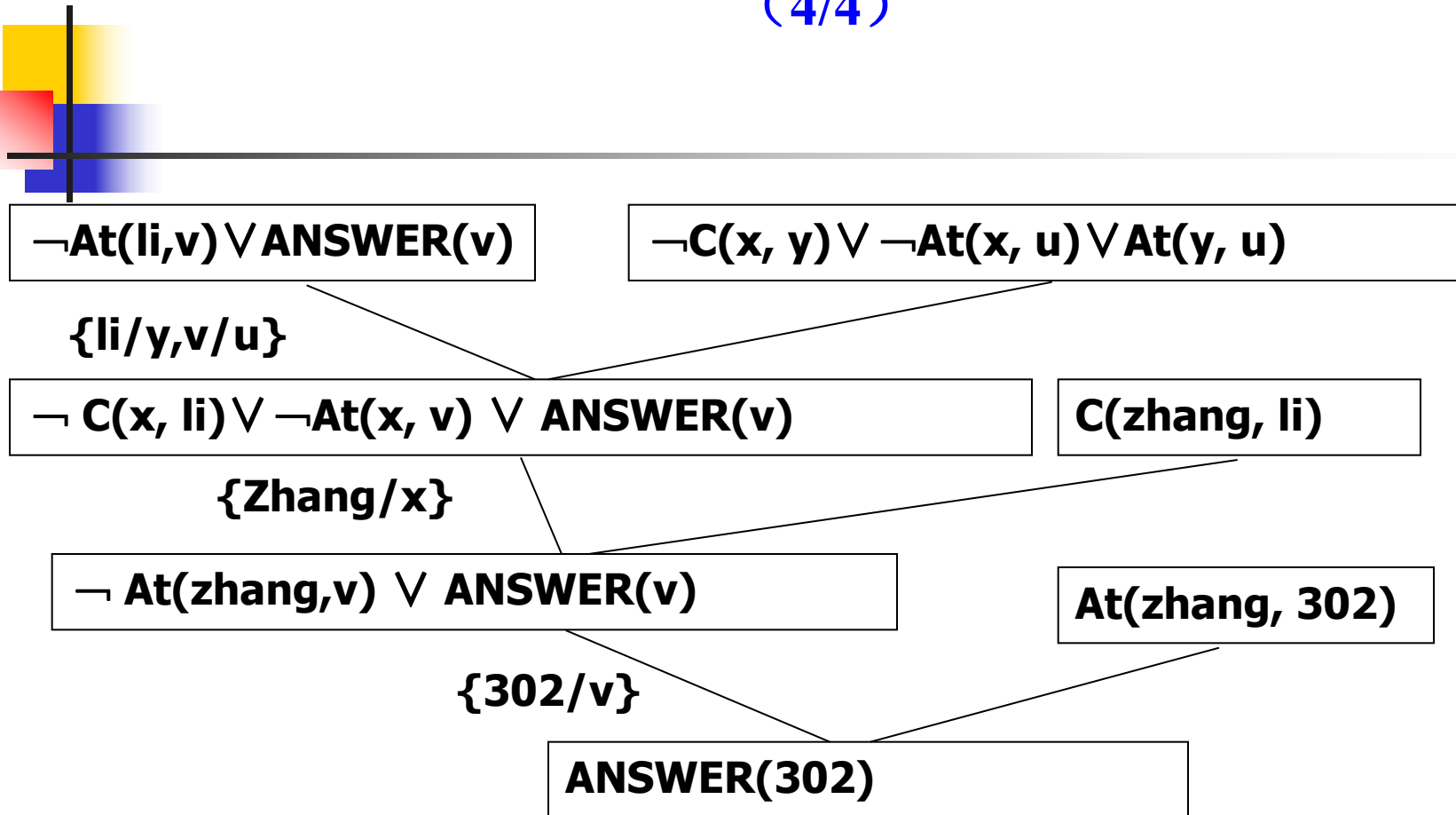
6 $\neg At(\text{zhang}, v) \vee Answer(v)$ 由1和5归结 (zhang/x)

7 $Answer(302)$ 由3和6归结 $(302/v)$

所以 $v=302$

用归结反演求取问题的答案

(4/4)





把归结反演的证明算法为:

规则集合: IF $c_x (\in S_i)$ 和 $c_y (\in S_i)$ 有归结式 r_{xy}
THEN $S_{i+1} = S_i \cup \{r_{xy}\}$

目标条件: S_n 中出现空子句

过程 RESOLUTION

① CLAUSES := S; S 为初始的基本子句集。

② until NIL 是 CLAUSES 的元素, do:

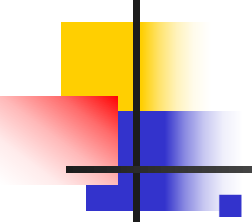
③ begin

④ 在 CLAUSES 中选两个不同的可归结的子句 c_i 、 c_j ; c_i 、 c_j 称母子句。

⑤ 求 c_i 、 c_j 的归结式 r_{ij} ;

⑥ CLAUSES := $\{r_{ij}\} \cup$ CLAUSES;

⑦ end;

- 
- 归结方法的基本算法：每次从子句集中选择两个可进行归结的子句，求它们的归结式，如果归结式为**NIL**，则算法结束，结论得证。如果归结式不为空，则将该归结式加入到子句集中，继续以上过程。

当归结出空子句**NIL**时，说明结论是正确的，此时算法结束。但是算法并没有说明什么情况下结论不正确。或者说，如果不能证明结论正确时，算法何时结束。

事实上在一阶谓词逻辑的范围内，这一点是做不到的。已经证明，在给定的已知条件下，如果结论成立，则归结法可以在有限的步数内证明该结论成立。但是如果在给定的条件下，结论不成立，则不存在一个通用的方法，可以在有限的步骤内证明该结论不成立。



3.6 归结演绎推理的归结策略

- 归结演绎推理实际上就是从子句集中不断寻找可进行归结的子句对，并通过对这些子句对的归结，最终得出一个空子句的过程。归结方法很简单，但是即便是对于一个比较简单的问题，往往可以进行归结的子句也比较多。如何从众多的可归结的子句中选择两个子句，即为搜索策略问题。不同的搜索策略，会影响到系统的效率和开销，同时也会涉及到完备性问题。

3.6 归结演绎推理的归结策略

■ 目前，常用的归结策略可分为两大类，一类是删除策略，另一类是限制策略。

删除策略是通过删除某些无用的子句来缩小归结范围；

限制策略是通过参加归结的子句进行某些限制，来减少归结的盲目性，以尽快得到空子句。

■ 当给定的问题在已知条件下成立时，如果某种归结策略一定可以在有限步内证明问题是成立的，则该策略是完备的，否则是不完备的。这里介绍的是几种在归结过程中常用的搜索策略。这些策略中有些是完备的，有些是非完备的，应该注意每种策略的完备性。

■ 在讨论各种常用的归结策略之前，还是先提一下广度优先策略。

3.6 归结演绎推理的归结策略

1. 广度优先策略 (1/3)

■ 广度优先是一种穷尽子句比较的复杂搜索方法。设初始子句集为 S_0 ，广度优先策略的归结过程可描述如下：

■ (1) 从 S_0 出发，对 S_0 中的全部子句作所有可能的归结，得到第一层归结式，把这些归结式的集合记为 S_1 ；

■ (2) 用 S_0 中的子句与 S_1 中的子句进行所有可能的归结，得到第二层归结式，把这些归结式的集合记为 S_2 ；

■ (3) 用 S_0 和 S_1 中的子句与 S_2 中的子句进行所有可能的归结，得到第三层归结式，把这些归结式的集合记为 S_3 ；

■ 如此继续，直到得出空子句或不能再继续归结为止。

■ 例4.19 设有如下子句集：

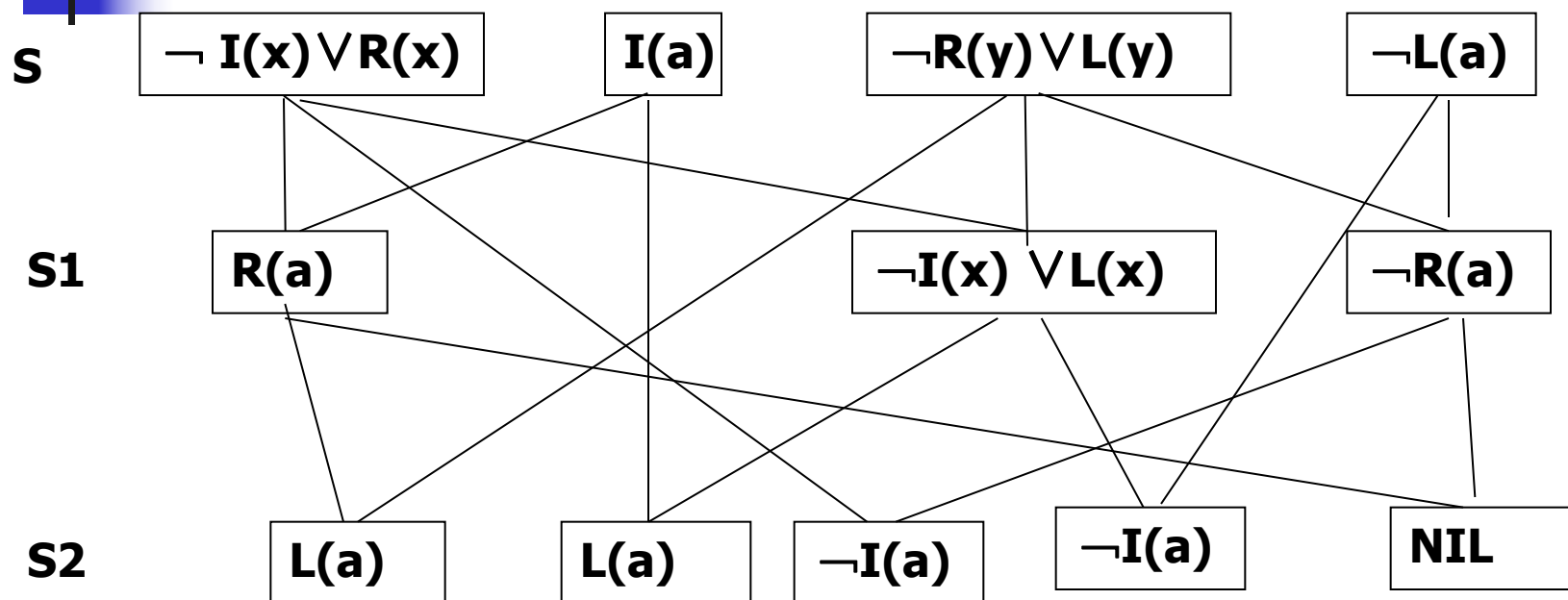
■
$$S = \{ \neg I(x) \vee R(x), I(a), \neg R(y) \vee L(y), \neg L(a) \}$$

■ 用宽度优先策略证明 S 为不可满足。

■ 宽度优先策略的归结树如下： 乔瑞

3.6 归结演绎推理的归结策略

1. 广度优先策略 (2/3)





3.6 归结演绎推理的归结策略

1. 广度优先策略（3/3）

- 从这个例子可以看出，宽度优先策略归结出了许多无用的子句，既浪费时间，又浪费空间。但是，这种策略由一个有趣的特性，就是当问题有解时保证能找到最短归结路径。
- 因此，它是一种完备的归结策略。宽度优先对大问题的归结容易产生组合爆炸，但对小问题却仍是一种比较好的归结策略。

3.6 归结演绎推理的归结策略

2. 删除策略 (1/3)

■ **主要想法是：**归结过程在寻找可归结子句时，子句集中的子句越多，需要付出的代价就会越大。如果在归结时能把子句集中无用的子句删除掉，这就会缩小搜索范围，减少比较次数，从而提高归结效率。

■ 常用的删除方法有以下几种：纯文字、重言式、包孕

3.6 归结演绎推理的归结策略

2. 删除策略 (1/3)

■ 纯文字删除法

- 如果某文字 L 在子句集中不存在可与其互补的文字 $\neg L$ ，则称该文字为**纯文字**。
- 在归结过程中，纯文字不可能被消除，用包含纯文字的子句进行归结也不可能得到空子句，**因此对包含纯文字的子句进行归结是没有意义的，应该把它从子句集中删除。**
- 对子句集而言，删除包含纯文字的子句，是不影响其不可满足性的。例如，对子句集
- $S = \{P \vee Q \vee R, \neg Q \vee R, Q, \neg R\}$
- 其中 **P 是纯文字**，因此可以将子句 $P \vee Q \vee R$ 从子句集 S 中删除。

3.6 归结演绎推理的归结策略

2. 删除策略 (2/3)

■ 重言式删除法

■ 如果一个子句中**包含有互补的文字对**，则称该子句为**重言式**。例如

$$P(x) \vee \neg P(x), P(x) \vee Q(x) \vee \neg P(x)$$

■ 都是重言式，不管 $P(x)$ 的真值为真还是为假， $P(x) \vee \neg P(x)$ 和 $P(x) \vee Q(x) \vee \neg P(x)$ 都均为真。

■ 重言式是真值为真的子句。对于一个子句集来说，不管是增加还是删除一个真值为真的子句，都不会影响该子句集的不可满足性。

■ **因此，可从子句集中删去重言式。**

3.6 归结演绎推理的归结策略

2. 删除策略 (3/3)

■ 包孕删除法

- 设有子句 C_1 和 C_2 ，如果存在一个置换 σ ，使得 $C_1\sigma \subseteq C_2$ ，则称 C_1 包孕于 C_2 。例如
- $P(x)$ 包孕于 $P(y) \vee Q(z)$ $\sigma = \{x/y\}$
- $P(x)$ 包孕于 $P(a)$ $\sigma = \{a/x\}$
- $P(x)$ 包孕于 $P(a) \vee Q(z)$ $\sigma = \{a/x\}$
- $P(x) \vee Q(a)$ 包孕于 $P(f(a)) \vee Q(a) \vee R(y)$ $\sigma = \{f(a)/x\}$
- $P(x) \vee Q(y)$ 包孕于 $P(a) \vee Q(u) \vee R(w)$ $\sigma = \{a/x, u/y\}$
- 对子句集来说，把其中包孕的子句删去后，不会影响该子句集的不可满足性。
- 因此，可从子句集中删除那些包孕的子句。

3.6 归结演绎推理的归结策略

3. 限制策略----支持集策略（1/3）

■ 支持集策略是沃斯(Wos)等人在1965年提出的一种归结策略。它要求每一次参加归结的两个亲本子句中，至少应该有一个是由目标公式的否定所得到的子句或它们的后裔。

■ 可以证明支持集策略是完备的，即当子句集为不可满足时，则由支持集策略一定能够归结出一个空子句。也可以把支持集策略看成是在宽度优先策略中引入了某种限制条件，这种限制条件代表一种启发信息，因而有较高的效率

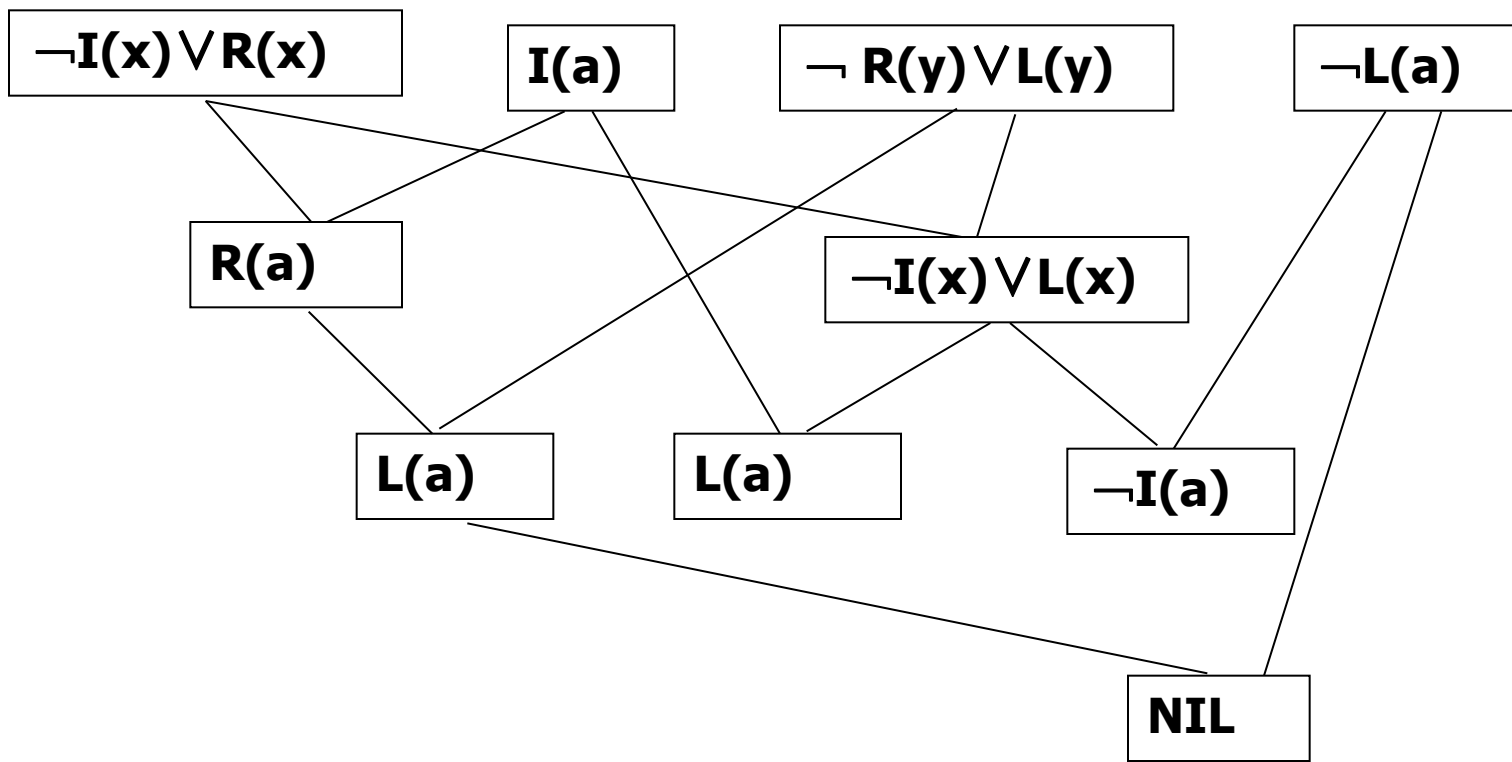
■ 例4.20 设有如下子句集：

$$S = \{ \neg I(x) \vee R(x), I(a), \neg R(y) \vee L(y), \neg L(a) \}$$

■ 其中， $\neg I(x) \vee R(x)$ 为目标公式的否定。用支持集策略证明S为不可满足。

3.6 归结演绎推理的归结策略

3. 限制策略-----支持集策略 (2/3)





3.6 归结演绎推理的归结策略

3. 限制策略-----支持集策略（3/3）

从上述归结过程可以看出，各级归结式数目要比宽度优先策略生成的少，但在第二级还没有空子句。就是说这种策略限制了子句集元素的剧增，但会增加空子句所在的深度。此外，支持集策略具有逆向推理的含义，由于进行归结的亲本子句中至少有一个与目标子句有关，因此推理过程可以看作是沿目标、子目标的方向前进的。

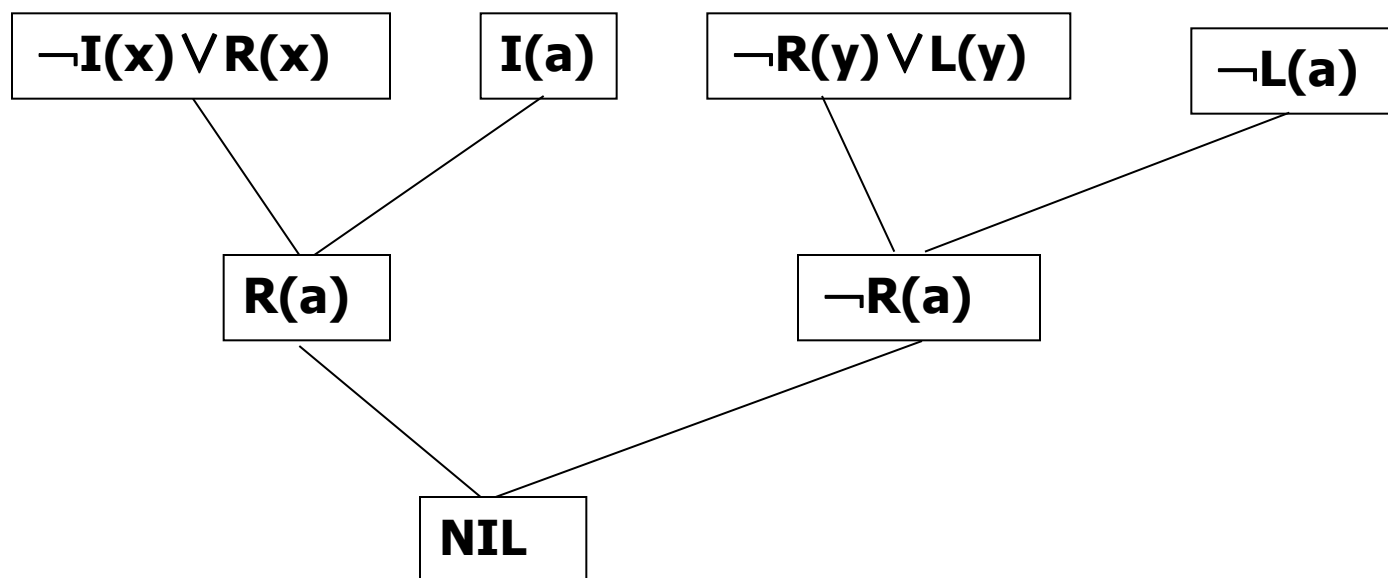
3.6 归结演绎推理的归结策略

3. 限制策略-----单文字子句策略（1/2）

- 如果一个子句只包含一个文字，则称此子句为单文字子句。它要求每次参加归结的两个亲本子句中至少有一个子句是单文字子句。
- 例 设有如下子句集：
$$S = \{ \neg I(x) \vee R(x), I(a), \neg R(y) \vee L(y), \neg L(a) \}$$
- 用单文字子句策略证明S为不可满足。
- 采用单文字子句策略，归结式包含的文字数将少于其亲本子句中的文字数，这将有利于向空子句的方向发展，因此会有较高的归结效率。但这种策略是不完备的，即当子句集为不可满足时，用这种策略不一定能归结出空子句。

3.6 归结演绎推理的归结策略

3. 限制策略----- 单文字子句策略 (2/2)



3.6 归结演绎推理的归结策略

3. 限制策略-----线形输入策略（1/2）

- 这种策略要求每次参加归结的两个亲本子句中，至少应该有一个是初始子句集中的子句。所谓初始子句集是指开始归结时所使用的子句集。

- 例4.22 设有如下子句集：

- $$S = \{ \neg I(x) \vee R(x), I(a), \neg R(y) \vee L(y), \neg L(a) \}$$

- 用线性输入策略证明S为不可满足。

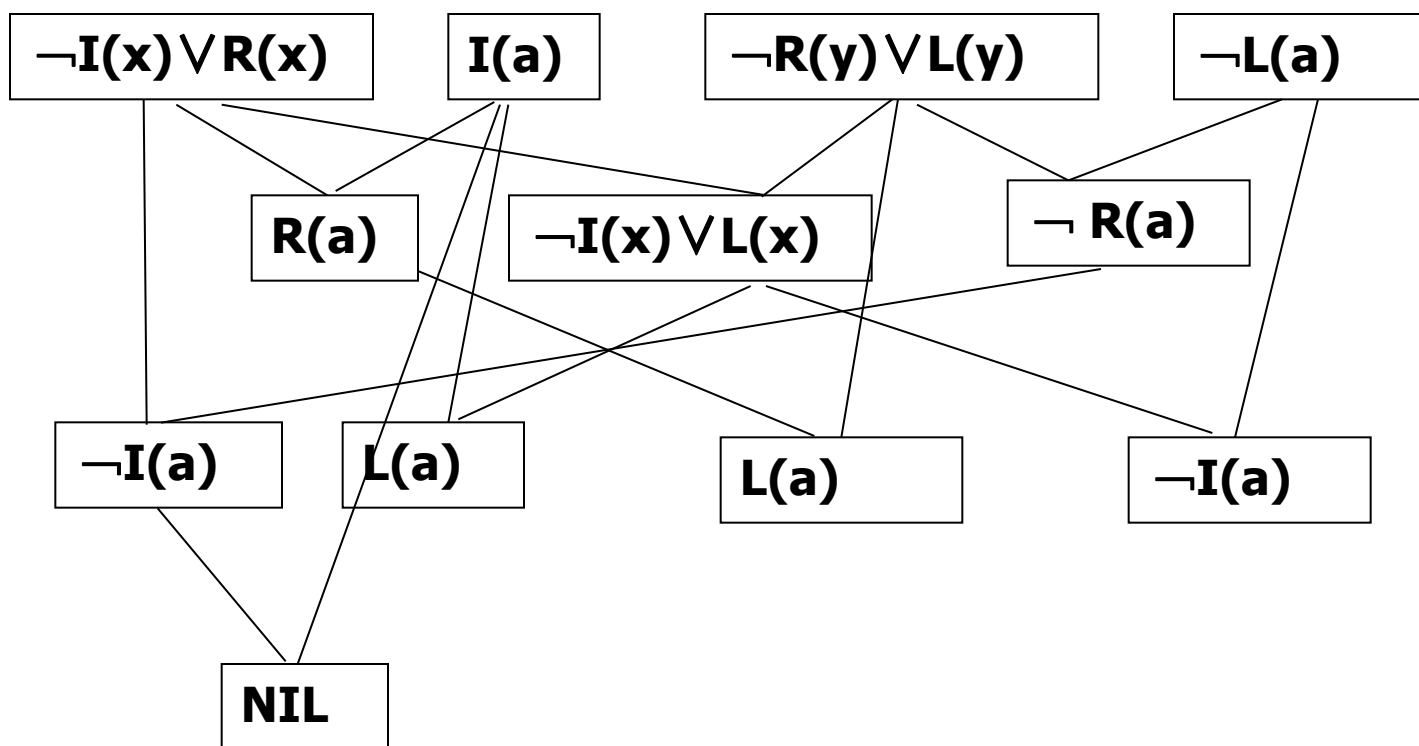
- 线性输入策略可限制生成归结式的数目，具有简单和高效的优点。但是，这种策略也是一种不完备的策略。例如，子句集

- $$S = \{ Q(u) \vee P(a), \neg Q(w) \vee P(w), \neg Q(x) \vee \neg P(x), Q(y) \vee \neg P(y) \}$$

- 从S出发很容易找到一棵归结反演树，但却不存在线性输入策略的归结反演树。

3.6 归结演绎推理的归结策略

3. 限制策略-----线形输入策略（2/2）



3.6 归结演绎推理的归结策略

3. 限制策略-----祖先过滤策略（1/2）

这种策略与线性输入策略有点相似，但是，放宽了对子句的限制。每次参加归结的两个亲本子句，只要满足以下两个条件中的任意一个就可进行归结：

(1) 两个亲本子句中至少有一个是初始子句集中的子句。

(2) 如果两个亲本子句都不是初始子句集中的子句，则一个子句应该是另一个子句的先辈子句。所谓一个子句(例如 C_1)是另一个子句(例如 C_2)的先辈子句是指 C_2 是由 C_1 与别的子句归结后得到的归结式。

例 设有如下子句集：

$S = \{ \neg Q(x) \vee \neg P(x), Q(y) \vee \neg P(y), \neg Q(w) \vee P(w), Q(a) \vee P(a) \}$

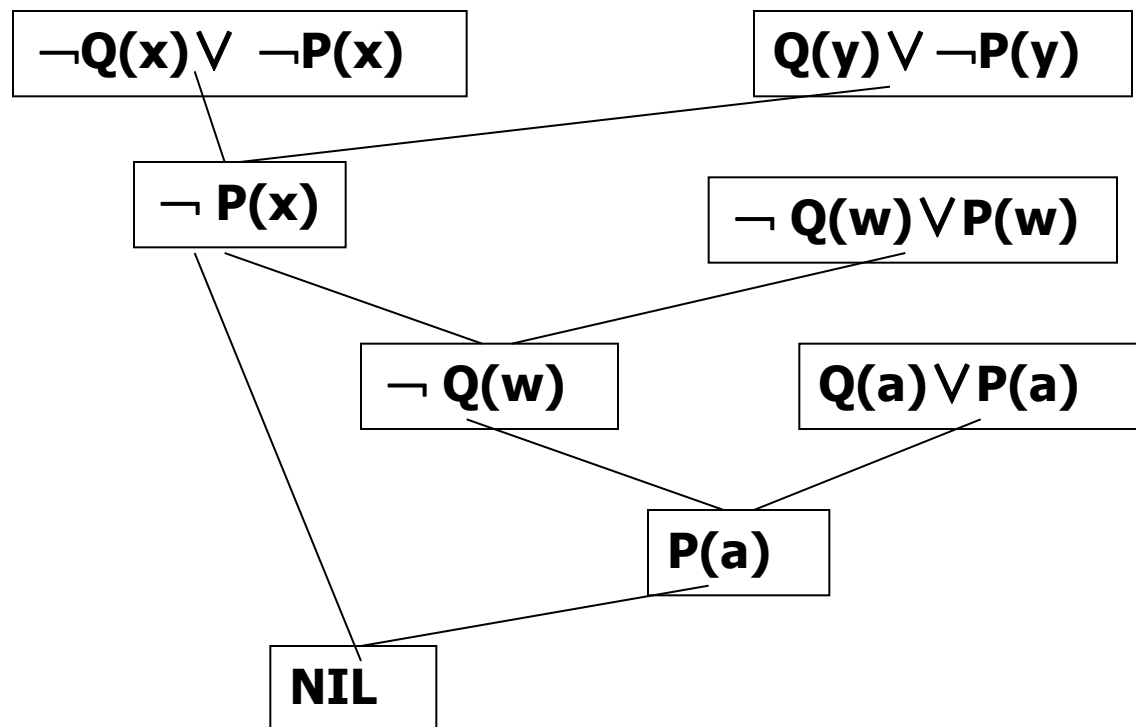
用祖先过滤策略证明 S 为不可满足

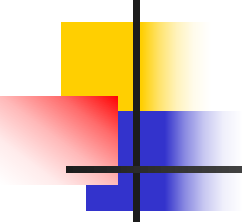
证明：从 S 出发，按祖先过滤策略归结过程如下图所示。

可以证明祖先过滤策略也是完备的。

3.6 归结演绎推理的归结策略

3. 限制策略-----祖先过滤策略（2/2）



- 
-
- 上面分别讨论了几种基本的归结策略，在系统实现时，我们当然希望选择完备的搜索策略，但一些非完备的搜索策略往往具有较高的求解效率，因此也有使用的必要性。在实际应用中，还可以把几种策略结合起来使用。