第3章

确定性推理

3.1 推理概述

什么是推理

- 推理就是按某种策略由已知判断推出另一判断的思维过程
- 已知判断：包括已掌握的与求解问题有关的知识及关于问题的已知事实

  推理的结论：由已知判断推出新判断
- 推理由程序程序实现，称为推理机

推理方式及其分类

1、演绎推理、归纳推理、默认推理

- 推理的基本任务是从一种判断推出另一种判断
- 按判断推出的途径来划分，可分为演绎推理、归纳推理及默认推理

（1）演绎推理

- 演绎推理是从全称判断推导出特称判断或单称判断的过程
- 演绎推理有多种形式，经常用的是三段论式
- 三段论式包括
  - 大前提：已知的一般性知识或假设
  - 小前提：关于所研究的具体情况或个别事实的判断
  - 结论：由大前提推出的适合于小前提所示情况的新判断

推理方式及其分类

- 在任何情况下，由演绎推导出的结论都是蕴涵在大前提的一般性知识中
- 只要大前提和小前提是正确的，则由它们推出的结论必然是正确的

(2) 归纳推理

- 归纳推理是从足够多的事例中归纳出一般性结论的推理过程，是一种从个别到一般的推理
- 归纳推理：完全归纳推理、不完全归纳推理
- 完全归纳推理是在进行归纳时考察了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，从而推出这个事物是否具有这个属性
- 不完全归纳推理是指只考察了相应事物的部分对象就得出了结论

推理方式及其分类

- 枚举归纳推理：若已知某类事物的有限可数个具体事物都具有某种属性，则可推出该类事物都具有此属性
- 类比推理：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推理

(3) 默认推理

- 又称缺省推理，它是在知识不完全的情况下假设某些条件已经具备所进行的推理
- 摆脱了需要知道全部事实才能进行推理的需求，使得在知识不完全的情况下也能进行推理

推理方式及其分类

2、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）：**推理时所用的知识都是精确的，推出的结论也是确定的，其真值或者为真，或为假，没有第三种情况出现
- **不确定性推理（不精确推理）：**推理时所用的知识不都是精确的，推出的结论也不完全是肯定的，真值位于真与假之间，命题的外延模糊不清

推理方式及其分类

3、单调推理、非单调推理

- 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
- **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

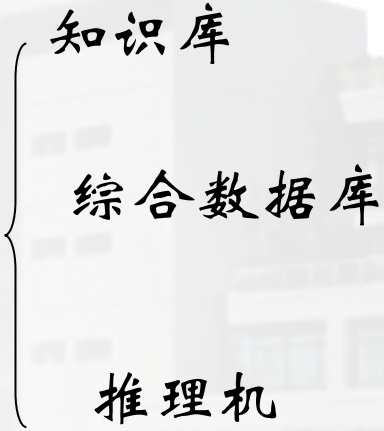
  非单调推理往往在信息不完全或者情况发生变化时出现。

推理的控制策略

- 推理过程是一个思维过程，即求解问题的过程
- 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，分为正向推理、逆向推理、混合推理及双向推理四种



推理的控制策略

① 正向推理

正向推理是从初始状态出发，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

逆向推理是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

正、逆向推理比较

项 目	正向推理	逆向推理
驱动方式	数据驱动	目标驱动
推理方法	从一组数据出发向前推导结论	从可能的解出发向后推理验证解答
启动方法	从一个事件启动	由询问关于目标状态的一个问题启动
透明程度	不能解释其推理过程	可解释其推理过程
推理方向	由底向上推理	由顶向下推理
典型系统	CLIPS, OPS	PROLOG

推理的控制策略

③ 混合推理

- 已知的事实不充分。通过正向推理先把其运用条件不能完全匹配的知识都找出来，并把这些知识可导出的结论作为假设，然后分别对这些假设进行逆向推理
- 由正向推理推出的结论可信度不高
- 希望得到更多的结论
- 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，然后再利用逆向推理中得到的信息进行正向推理，以推出更多的结论

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证据

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

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
  - ✓ 已知事实不能与知识库中的任何知识匹配成功
  - ✓ 已知事实恰好只与知识库中的一个知识匹配成功
  - ✓ 已知事实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）已知事实可与知识库中的多个知识匹配成功
- 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；或者有多组不同的已知事实都与同一条产生式规则的前件匹配成功；或者两种情况同时出现

推理的控制策略

- ✓ 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，或者有多条产生式后件可与多个假设匹配成功。

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。

推理的控制策略

③ 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。

④ 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。

⑤ 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。

推理的控制策略

⑥ 按条件个数排序

- 多条规则生成的结论相同的情况下，由于条件个数较少的规则匹配所花费的时间较少而且容易实现，所以将条件少的规则赋予较高的优先级，优先被启用。

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。

3.3 自然演绎推理

自然演绎推理的基本概念

- 定义：自然演绎推理是指从一组已知的事实出发，直接运用命题逻辑或谓词逻辑中的推理规则推出结论的过程。
- 推理规则：
  - P规则：在推理的任何步骤上都可引入前提，继续进行推理。
  - T规则：推理时，如果前面步骤中有一个或多个公式永真蕴涵公式S，则可把S引入推理过程中。
  - 反证法：$P \to Q$，当且仅当 $P \wedge \neg Q \Leftrightarrow F$ 。即：Q为P的逻辑结论，当且仅当 $P \wedge \neg Q$ 是不可满足的。

自然演绎推理的基本概念

- 假言推理

$$P, P \to Q \Rightarrow Q$$

表示：由 $P \to Q$ 及P为真，可推出Q为真

- 拒取式推理

$$P \to Q, \neg Q \Rightarrow \neg P$$

表示：由 $P \to Q$ 为真及Q为假，可推出P为假

自然演绎推理的基本概念

- 避免产生两类错误：
  - 肯定后件(Q)的错误：当P→Q为真时，希望通过肯定后件Q推出前件P为真，这是不允许的.
  - 否定前件(P)的错误：当P→Q为真时，希望通过否定前件P推出后件Q为假，这也是不允许的.

自然演绎推理的基本概念

- 如伽利略在论证哥白尼的日心说时，曾使用了下列推理：
  - 如果行星系统是以太阳为中心的，则金星会显示出位相变化。
  - 金星会显示出位相变化。
  - 所以，行星系统是以太阳为中心的。

这就是使用了肯定后件的推理，违反了经典逻辑的逻辑规则，他为此曾遭到非难。

自然演绎推理的基本概念

- 又如下列推理：

  - 如果上网，则能知道新闻。
  - 没有上网。
  - 所以，不知道新闻。

这就是使用了否定前件的推理，违反了逻辑规则，显然是不正确的，因为通过收听广播、看电视等，也会知道新闻。

自然演绎推理的优缺点

- 优点：

  定理证明过程自然，容易理解，而且它拥有丰富的推理规则，推理过程灵活，便于在它的推理规则中嵌入领域启发式知识。

- 缺点：

  容易产生组合爆炸，推理过程中得到的中间结论一般呈指数形式递增。

归结演绎推理★

- 人的问题求解行为更像是一个解答识别过程而非解答搜索过程
- 识别解答或部分解答依赖于应用领域特有的知识，
- 符号推理则成为基于知识来求解问题的主要手段。
- 符号推理的重要方式是演绎推理
- 它的基础为谓词演算——一种形式语言
  - 将各种陈述性（说明性）的描述以形式化的方式表示，以便对它们　作处理。
- 谓词演算——人工智能系统最常用的知识表示方法，
- 广泛地应用于各种人工智能系统的设计。
- 谓词演算（或更广义地，形式逻辑）是人工智能研究的重要基础之一。
- 主要内容：

  •合适公式标准化　　•H域和海伯伦定理

  •置换和合一　　•归结法　　• 归结反演

回顾谓词逻辑表示法

- 1、谓词公式
  - “谓词公式”的一般形式：
    - $P(x_1,x_2,\ldots,x_n)$，其中，
    - P——谓词符号（简称谓词）；
    - $X_i(i=1,2,\ldots,n)$——参数项（简称项），项可以是常量、变量或函数；
    - $P(x_1,x_2,\ldots,x_n)$——n元谓词公式；
  - “谓词公式”的基本组成：
    - 谓词符号、常量符号、变量符号、函数符号；
    - 用括号和逗号隔开，表示论域内的关系。
  - “谓词公式是谓词逻辑的基本单元，也称为原子公式。

- **2、连词和量词**
  - 通过引入**连词**和**量词**，可以把**谓词公式（原子公式）**组合为**复合谓词公式**。
  - **复合谓词公式**也称为**逻辑语句**。
- （1）连词

¬	（非）加在**谓词公式**前面，称为否定，或取反。
∧	（与）连接**谓词公式**，称为**合取**； 产生的**逻辑语句**称为**合取式**，每个成分成为**合取项**。
∨	（或）连接**谓词公式**，称为**析取**； 产生的**逻辑语句**称为**析取式**，每个成分成为**析取项**。
⇒	（蕴涵）连接**谓词公式**产生**蕴涵式**； 左部称为**前项**，右部称为**后项**。
⇔	（等价）连接**谓词公式**产生**等价式**；正、逆向蕴涵式的合取。

2、连词和量词

- 通过引入连词和量词，可以把原子公式组合为复合谓词公式。
- 复合谓词公式也称为逻辑语句。

（1）连词

- 通过连词产生的复合谓词公式（逻辑语句）的真值表：

P	Q	¬ P	P∧Q	P∨Q	P⇒Q	P⇔Q
T	T	F	T	T	T	T
F	T	T	F	T	T	F
T	F	F	F	T	F	F
F	F	T	F	F	T	T

- 2、连词和量词
  - 命题——不包含变量的谓词公式和逻辑语句；
  - 命题逻辑——基于命题的谓词逻辑称为命题逻辑，命题逻辑是谓词逻辑的子集。
  - 命题逻辑缺乏有效的表达一般性概念的能力
    - 无法把每个知识单元抽象、细分；
    - 如，“条条大路通罗马”。
    - Lead(Road1,Roma)
    - Lead(Road2,Roma)……
  - 谓词逻辑中引入变量和对变量进行约束的量词。
- （2）量词
  - 全称量词∀　　存在量词∃

2、连词和量词——（2）量词

- 全称量词∀
  - 符号$(\forall x)P(x)$：表示对于某个论域中的所有（任意一个）个体x, 都有P(x)真值为T。
- 存在量词∃
  - 符号$(\exists x)P(x)$：来表示某个论域中至少存在一个个体x，使P(x) 真值为T。

条条大路通罗马

$$(\forall x)[Road(x) \Rightarrow Lead(x, Roma)]$$

Mary给每个人一本书

量词可以嵌套使用

$$(\forall x)(\exists y)[Person(x) \wedge Book(y) \wedge Give(Mary, x, y)]$$

Mary给每人某个同样的东西

可以有不受量词约束的变量

$$(\forall x)[Person(x) \wedge Give(Mary, x, y)]$$

2、连词和量词——（2）量词

- 全称量词∀
  - 符号$(\forall x)P(x)$：表示对于某个论域中的所有（任意一个）个体x, 都有P(x)真值为T。
- 存在量词∃
  - 符号$(\exists x)P(x)$：来表示某个论域中至少存在一个个体x，使P(x) 真值为T。

条条大路通罗马 $(\forall x)[Road(x) \Rightarrow Lead(x, Roma)]$

所有机器人都是灰色的 $(\forall x)[Robot(x) \Rightarrow Color(x, Gray)]$

一阶谓词逻辑

- 定义：若限定不允许对谓词和函数名进行量化处理，且参数项不能是谓词公式，则这样的谓词逻辑是一阶的。
  - 谓词、函数名的出现位置不允许使用变量；
  - 参数项不能是谓词公式；
- (∀P)P(A)-谓词进行了量化；
- (∀y)Married(y(L1),Mary)-函数名进行了量化；
- P(x,Q(y))-参数项是谓词公式；

合适(式)公式

- 1、合适公式的定义
  - 合适公式适合于一阶谓词逻辑
  - 遵从以下递归方式定义的逻辑语句称为合适公式
  - ①单一谓词公式是合适公式；
  - ②若A是合适公式，则¬A也是合适公式；
  - ③若A和B都是合适公式，则A∧B、 A∨B、 A⇒B和A⇔B也都是合适公式；
  - ④若A是合适公式，x是约束变量，则(∀x)A和(∃x)A也都是合适公式；
  - ⑤只有按上述规则①-④求得的公式，才是合适公式。
  - 连词优先级别是¬，∧、∨，⇒、⇔，但可通过括号改变优先级。

合适公式的性质

合适公式等价关系：

1. 否定之否定 $\neg(\neg P) \Leftrightarrow P$
2. 蕴涵式转化 $P \Rightarrow Q \Leftrightarrow \neg P \vee Q$
3. 狄摩根定律 $\neg(P \vee Q) \Leftrightarrow \neg P \wedge \neg Q$

   $\neg(P \wedge Q) \Leftrightarrow \neg P \vee \neg Q$
4. 分配律

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

   $P \vee (Q \wedge R) \Leftrightarrow (P \vee Q) \wedge (P \vee R)$
5. 交换律 $P \vee Q \Leftrightarrow Q \vee P$

   $P \wedge Q \Leftrightarrow Q \wedge P$

合适公式的性质

6. 结合律

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

7. 逆否律

$$P \Rightarrow Q \Leftrightarrow \neg Q \Rightarrow \neg P$$

8. 量词否定

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

合适公式的性质

9.量词分配

$$(\forall x)[P(x) \wedge Q(x)] \Leftrightarrow (\forall x)P(x) \wedge (\forall x)Q(x)$$

$$(\exists x)\ [P(x) \vee Q(x)] \Leftrightarrow\ (\exists x)\ P(x) \vee (\exists x)Q(x)$$

10.约束变量的虚元性

$$(\forall x)P(x) \Leftrightarrow (\forall y)P(y)$$

$$(\exists x)P(x) \Leftrightarrow (\exists y)P(y)$$

$$(\forall x)\ [P(x)\ \wedge\ Q(x)] \Leftrightarrow (\forall x)P(x)\ \wedge (\forall y)\ Q(y)$$

$$(\exists x)\ [P(x)\ \vee\ Q(x)] \Leftrightarrow (\exists\ x)P(x)\ \vee (\exists\ y)\ Q(y)$$

合适公式的标准化★

1、标准化需求

- 常见的基于谓词演算的推理：归结反演、(正向/逆向)归结演绎推理

  - 首先要求以量词前束范式来表示合适公式

- 量词前束范式形式如下：
  - $(Q_1x_1)\ (Q_2x_2)\ldots(Q_kx_k)M$，其中
  - M ——母式，不包括任何量词；
  - Q_ix_i——Q_i可以是量词符号∀或∃；x_i是量词的约束变量 $(i=1,2,\ldots,k)$

前束范式

例1：变公式（∀x）P（x）=>（∃x）Q（x）为前束范式

~（∀x）P（x）∨（∃x）Q（x）

（∃x）（~P（x））∨（∃x）Q（x）

（∃x）（~P（x）∨Q（x））为前束范式

前束范式

例2：$(\forall x)(\forall y)\{((\exists z)(P(x, z)\wedge P(y, z))=>(\exists u)Q(x, y, u))\}$

$(\forall x)(\forall y)(\sim((\exists z)(P(x, z)\wedge P(y, z)))\vee(\exists u)Q(x, y, u))$ →

$(\forall x)(\forall y)(\forall z)(\sim P(x, z)\vee\sim P(y, z))\vee(\exists u)Q(x, y, u)$ →

$(\forall x)(\forall y)(\forall z)(\exists u)(\sim P(x, z)\vee\sim P(y, z)\vee Q(x, y, u))$

史柯伦标准型及其构造思想

- 史柯伦（Skolem）标准型：海伯伦（Herbrand）定理是归结原理的基础。海伯伦定理证明的步骤实际上是反演法，即不是证明一个公式是永真，而是证明该公式是否是永假的。反演法利用了一个标准型，这个标准型就是Skolem标准型。

一阶逻辑公式所对应的Skolem标准型基于如下思想来构造：

1. 一阶逻辑的一个公式被变换为前束范式。其中前束是一个存在量词或全称量词的序列，母式中不在含有量词。
2. 因为母式不含量词，所以可以变换为合取范式。
3. 通过使用Skolem函数，可以在前束中将存在量词消去，而不影响公式的永假性。

Skolem标准型

- 通过变换消去存在量词所得到的公式称为Skolem标准型，而拿来代替存在量词的变量的函数称Skolem函数。无参Skolem函数有时称Skolem常量。

  从一阶逻辑的公式变换到Skolem标准型不是等值变换，因为Skolem标准型与原公式不等值。但它们保持永假性。

合适公式的标准化

1、标准化需求

- 常见的基于谓词演算的推理：归结反演、(正向/逆向)演绎推理
  - 要求以量词前束范式来表示合适公式
- 量词前束范式形式如下：
  - $(Q_1x_1)\ (Q_2x_2)\ldots(Q_kx_k)M$，其中
  - M ——母式，不包括任何量词；
  - Q_ix_i——Q_i可以是量词符号$\forall$或$\exists$；x_i是量词的约束变量 $(i=1,2,\ldots,k)$
- 归结反演——要求M标准化为合取范式，定义如下：
  - $M=W_1 \wedge W_2 \wedge \ldots \wedge W_n$
  - $W_i=L_{i1} \vee L_{i2} \vee \ldots \vee L_{im} (i=1,2,\ldots,n)$
  - $L_{ij}=P_{ij} | \neg P_{ij}$:文字（Literal），是谓词公式$P_{ij}$或其取反

2、合取范式的标准化过程

- 应用合适公式性质将公式逐步转化的过程。
- 转化步骤没有严格的规定
- 较合理的化简过程，分为8步：
  - ①消去多余的量词（很少出现）；
  - ②消去蕴涵符号；
  - ③减少否定的辖域（内移否定符号）；
  - ④变量标准化（变量换名）；
  - ⑤消去存在量词（Skolem变换）；
  - ⑥全称量词前束化（化为前束形）；
  - ⑦消去全称量词；
  - ⑧把母式转化为合取范式。

- 2、合取范式的标准化过程
- ①消去多余的量词（很少出现）
  - 若一个量词的辖域内并未出现量词的约束变量，则该量词是多余的，应该删除；
  - 例，$(\exists x)P(y)$，则$(\exists x)$可以消去，得到$P(y)$；
  - 正常情况下，合适公式中不应出现多余的量词。
- ②消去蕴涵符号
  - 蕴涵式转化：$P \Rightarrow Q \Leftrightarrow \neg P \vee Q$；
  - 例，$Q(x,y) \Rightarrow P(y) \Leftrightarrow \neg Q(x,y) \vee P(y)$。

- 2、合取范式的标准化过程
- ③内移否定符号
  - 使否定只出现在原子谓词公式前，构成否定文字；
  - 狄.摩根定律：
    - $\neg(P \vee Q) \Leftrightarrow \neg P \wedge \neg Q$
    - $\neg(P \wedge Q) \Leftrightarrow \neg P \vee \neg Q$
  - 双重否定：$\neg(\neg P) \Leftrightarrow P$
  - 量词否定：
    - $\neg(\exists x)\, P(x) \Leftrightarrow (\forall x)\, (\neg P(x))$
    - $\neg(\forall x)\, P(x) \Leftrightarrow (\exists x)\, (\neg P(x))$
  - 例，$\neg(\exists y)[\neg P(y) \vee P(f(x,y))] \Leftrightarrow (\forall y)[P(y) \wedge \neg P(f(x,y))]$
- ④变量换名
  - “⑥全称量词前束化”后，同名变量的辖域无法区分，所以为避免差错，必须换名；
  - 约束变量的虚元性进行变换；

    $(\forall x)\{p(x) => (\exists x)\, Q(x)\}$

    标准化而得到 $(\forall x)\{p(x) => (\exists y)\, Q(y)\}$

- 2、合取范式的标准化过程
- ⑤消去存在量词（Skolem变换）
  - ∃在∀的辖域内
    - (∀z)(∃w)[¬Q(x,z) ∨P(z,w)]
    - w依赖于z，由函数w=g(z)来定义这种依赖关系；
    - 用g(z)来取代约束变量w，消去存在量词∃w；
    - (∀z)[¬Q(x,z) ∨P(z,g(z))]
  - ∃在多个∀的辖域内
    - (∀x)(∀y)(∀z)(∃w)P(x,y,z,w)
    - 用多元函数g(x,y,z)来取代约束变量w，消去存在量词∃w；
    - (∀x)(∀y)(∀z)P(x,y,z,g(x,y,z))
  - ∃在∀的辖域外
    - (∃w)(∀z) [¬Q(x,z) ∨P(z,w)]
    - 用任意常量A取代约束变量w，消去存在量词∃w
    - (∀z) [¬Q(x,z) ∨P(z,A)]

前两种叫Skolem函数，第三种叫Skolem常量

总结：Skolem函数和Skolem常量

在消去存在量词的过程中，需要用到Skolem函数或Skolem常量。若存在量词是在全称量词的辖域内，用Skolem函数消去存在量词。 Skolem函数所使用的函数符号必须是新的，即不允许是公式中已经出现过的函数符号。

若要消去的存在量词不在任何一个全称量词的辖域内，用不含变量的Skolem函数即Skolem常量消去存在量词。所使用的常量符号必须是新的，它未曾在公式其他地方使用过。

Skolem变换不是等价变换，但变换前后的值永假性保持不变。

- 2、合取范式的标准化过程
- ⑥全称量词前束化
  - 经过“④变量换名”后，所有量词的约束变量均有不同的名字；
  - 只要简单地将∀移到合适公式的最前面；
  - 约束变量的作用范围不会变化。
- ⑦消去全称量词
  - 经过“⑤消去存在量词”后，所有变量均受∀的约束；
  - 简单地删除∀，只留下母式。
- ⑧把母式转化为合取范式
  - 分配律： $P \vee (Q \wedge R) \Leftrightarrow (P \vee Q) \wedge (P \vee R)$

2、合取范式的标准化过程

- 例、化简合适公式

$$\neg(\forall x)\{P(x)\Rightarrow\{(\exists y)[P(y)\Rightarrow P(f(x,y))]\wedge\neg(\forall y)(\exists w)[Q(x,y)\Rightarrow P(y,w)]\}\}$$

- 2、合取范式的标准化过程
  - 例、化简合适公式

$\neg(\forall x)$

$\{$

$P(x) \Rightarrow$

$\{(\exists y)[P(y) \Rightarrow P(f(x,y))] \wedge \neg(\forall y)(\exists w)[Q(x,y) \Rightarrow P(y,w)]\}$

$\}$

②消去蕴涵符号

$\neg(\forall x)$

$\{$

$\neg P(x) \vee$

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

$\}$

2、合取范式的标准化过程

- 例、化简合适公式

$\neg(\forall x)$

$\{$

$\neg P(x) \vee$

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

$\}$

③内移否定符号

$(\exists x)$

$\{$

$P(x) \wedge$

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

$\}$

2、合取范式的标准化过程

- 例、化简合适公式

$(\exists x)$

$\{$

$P(x)\wedge$

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

$\}$

④变量换名

$(\exists x)$

$\{$

$P(x)\wedge$

$\{(\forall y)[P(y)\wedge\neg P(f(x,y))]\vee(\forall z)(\exists w)[\neg Q(x,z)\vee P(z,w)]\}$

$\}$

2、合取范式的标准化过程

- 例、化简合适公式

$(\exists x)$

$\{$

$P(x)\wedge$

$\{(\forall y)[P(y)\wedge\neg P(f(x,y))]\vee(\forall z)(\exists w)[\neg Q(x,z)\vee P(z,w)]\}$

$\}$

⑤消去存在量词

$\{$

$P(A)\wedge$

$\{(\forall y)[P(y)\wedge\neg P(f(A,y))]\vee(\forall z)[\neg Q(A,z)\vee P(z,g(z))]\}$

$\}$

2、合取范式的标准化过程

- 例、化简合适公式

$$\{$$

$$P(A)\wedge$$

$$\{(\forall y)[P(y)\wedge\neg P(f(A,y))]\vee(\forall z)[\neg Q(A,z)\vee P(z,g(z))]\}$$

$$\}$$

⑥全称量词前束化

$$(\forall y)(\forall z)$$

$$\{$$

$$P(A)\wedge$$

$$\{[P(y)\wedge\neg P(f(A,y))]\vee[\neg Q(A,z)\vee P(z,g(z))]\}$$

$$\}$$

2、合取范式的标准化过程

- 例、化简合适公式

$(\forall y)(\forall z)$

$\{$

$P(A)\wedge$

$\{[P(y)\wedge\neg P(f(A,y))]\vee[\neg Q(A,z)\vee P(z,g(z))]\}$

$\}$

⑦消去全称量词

$\{$

$P(A)\wedge$

$\{[P(y)\wedge\neg P(f(A,y))]\vee[\neg Q(A,z)\vee P(z,g(z))]\}$

$\}$

2、合取范式的标准化过程

- 例3、化简合适公式

{
P(A)∧
{[P(y)∧¬P(f(A,y))]∨[¬Q(A,z)∨P(z,g(z))]}
}

⑧把母式转化为合取范式

{
P(A)∧
{
[P(y)∨¬Q(A,z)∨P(z,g(z))]∧
[¬P(f(A,y))∨¬Q(A,z)∨P(z,g(z))]
}
}

完成标准化过程！

总结

- 合取范式的标准化过程
  - 应用合适公式性质将公式逐步转化的过程。
  - 转化步骤没有严格的规定
  - 较合理的化简过程，分为8步：
    - ①消去多余的量词（很少出现）；
    - ②消去蕴涵符号；
    - ③减少否定的辖域（内移否定符号）；
    - ④变量标准化（变量换名）；
    - ⑤消去存在量词（Skolem变换）；
    - ⑥全称量词前束化（化为前束形）；
    - ⑦消去全称量词；
    - ⑧把母式转化为合取范式。

归结演绎推理

- 自动定理证明一般表示形式为：
- $F1 \wedge F2 \wedge \ldots \wedge Fn \Rightarrow W$
  - $F_1, F_2, \ldots, F_n$都是合适公式，表示公理或事实；
  - W是合适公式，表示待证明的定理，称为目标公式；
- 证明的方法可分两大类：
  - 演绎法
    - 直接证明$F_1 \wedge F_2 \wedge \ldots \wedge F_n \Rightarrow W$为永真；
  - 反证法
    - 间接证明$\neg(F_1 \wedge F_2 \wedge \ldots \wedge F_n \Rightarrow W)$为永假；
    - 证明$F1 \wedge F2 \wedge \ldots \wedge Fn \wedge \neg W$的永假
      - 即{F1, F2,… ,Fn} ∪{¬W}是一个矛盾集。

归结演绎推理

- 海伯伦（Herbrand）
  - 提出的H域（海伯伦域）和海伯伦定理；
  - 为自动定理证明奠定了理论基础；
- 鲁宾逊（Robinson）
  - 提出的归结原理；
  - 使自动定理证明成为可能。

归结演绎推理

1）H域和海伯伦定理（了解）

- 1、子句和子句集
  - 子句——仅由文字的析取∨构成的合适公式
    - $W_i=L_{i1}\vee L_{i2}\vee\ldots\vee L_{im}$称为子句；
  - 合取范式定义：
    - $M=W_1\wedge W_2\wedge\ldots\wedge W_n$
    - $W_i=L_{i1}\vee L_{i2}\vee\ldots\vee L_{im}(i=1,2,\ldots,n)$
    - $L_{ij}=P_{ij}|\neg P_{ij}$:文字（Literal），是原子谓词公式$P_{ij}$或其取反
  - 合取范式可定义为子句的合取∧；
  - 合取范式表示为子句集，子句间隐含合取∧关系
    - 子句集$\{W_1,W_2,\ldots,W_n\}$

归结演绎推理

1）H域和海伯伦定理

- 1、子句和子句集
  - 子句——仅由文字的析取∨构成的合适公式
  - 合取范式表示为子句集，子句间隐含具有合取关系

{
P(A)∧
[P(y)∨¬Q(A,z)∨P(z,g(z))]∧
[¬P(f(A,y))∨¬Q(A,z)∨P(z,g(z))]
}

  - 可进一步表示为子句集

{
P(A),
P(y)∨¬Q(A,z)∨P(z,g(z)),
¬P(f(A,y))∨¬Q(A,z)∨P(z,g(z))
}

归结演绎推理

1）H域和海伯伦定理

- 1、子句和子句集
  - 子句——仅由文字的析取∨构成的合适公式
  - 合取范式表示为子句集，子句间隐含具有合取关系

$(\forall y)\ (\forall z)$

$\{$

$P(A) \wedge$

$[P(y) \vee \neg Q(A,z) \vee P(z,g(z))] \wedge$

$[\neg P(f(A,y)) \vee \neg Q(A,z) \vee P(z,g(z))]$

$\}$

量词分配：

$(\forall x)\ [P(x) \wedge Q(x)] \Leftrightarrow (\forall x)P(x) \wedge (\forall x)\ Q(x)$

$(\forall y)\ (\forall z)P(A) \wedge$

$(\forall y)\ (\forall z)[P(y) \vee \neg Q(A,z) \vee P(z,g(z))] \wedge$

$(\forall y)\ (\forall z)[\neg P(f(A,y)) \vee \neg Q(A,z) \vee P(z,g(z))]$

归结演绎推理

1）H域和海伯伦定理

1、子句和子句集

- 子句中的变量都是∀的约束变量

{
(∀y) (∀z)P(A),
(∀y) (∀z)[P(y)∨¬Q(A,z)∨P(z,g(z))],
(∀y) (∀z)[¬P(f(A,y))∨¬Q(A,z)∨P(z,g(z))]
}

- 为了消除子句间不必要的交互作用，保持子句的独立性，需要做变量换名

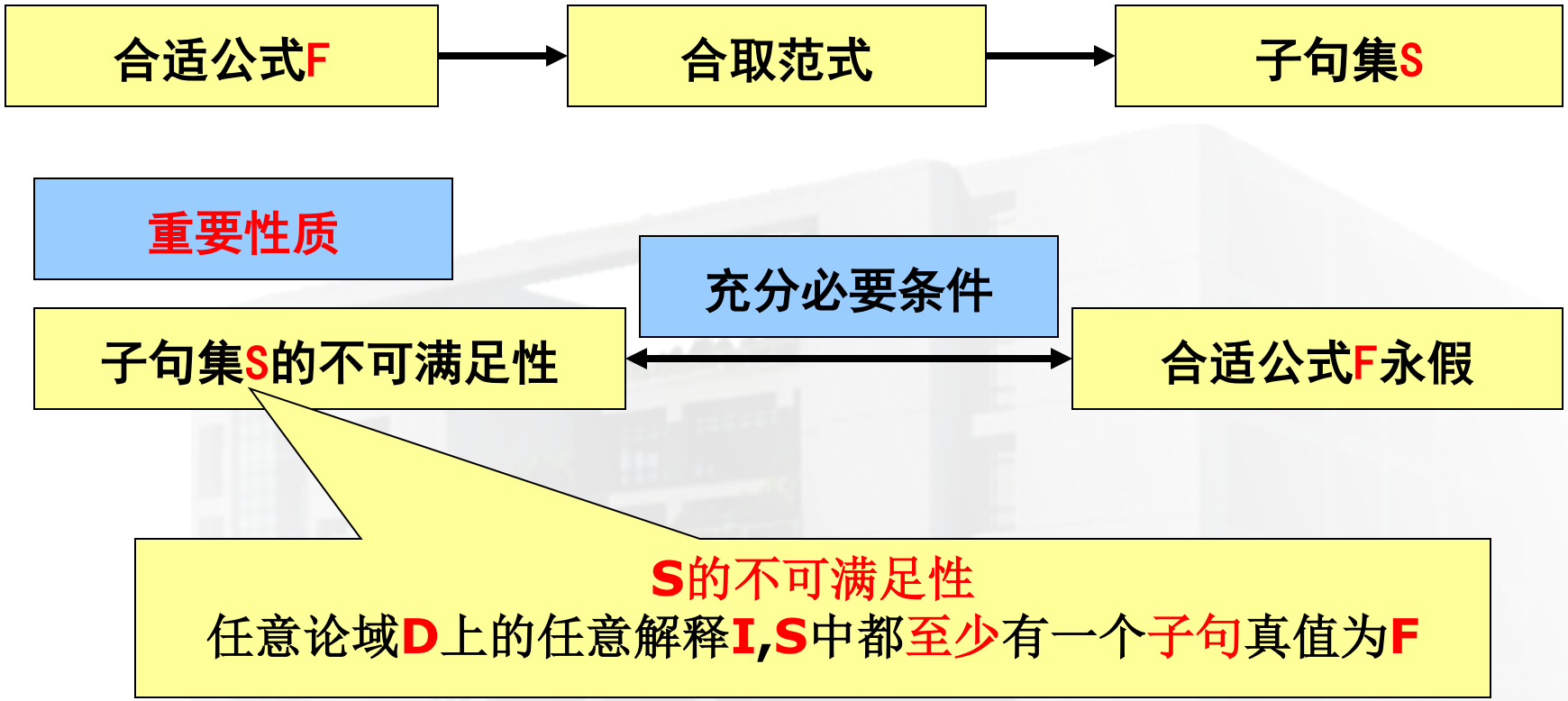
{
P(A),
P(y1)∨¬Q(A,z1)∨P(z1,g(z1)),
¬P(f(A,y2))∨¬Q(A,z2)∨P(z2,g(z2))
}

将合适公式化成字句集，只需要在化成合取范式的基础上，去掉∧符号以及进行变量换名即可。 ★

归结演绎推理

1）H域和海伯伦定理

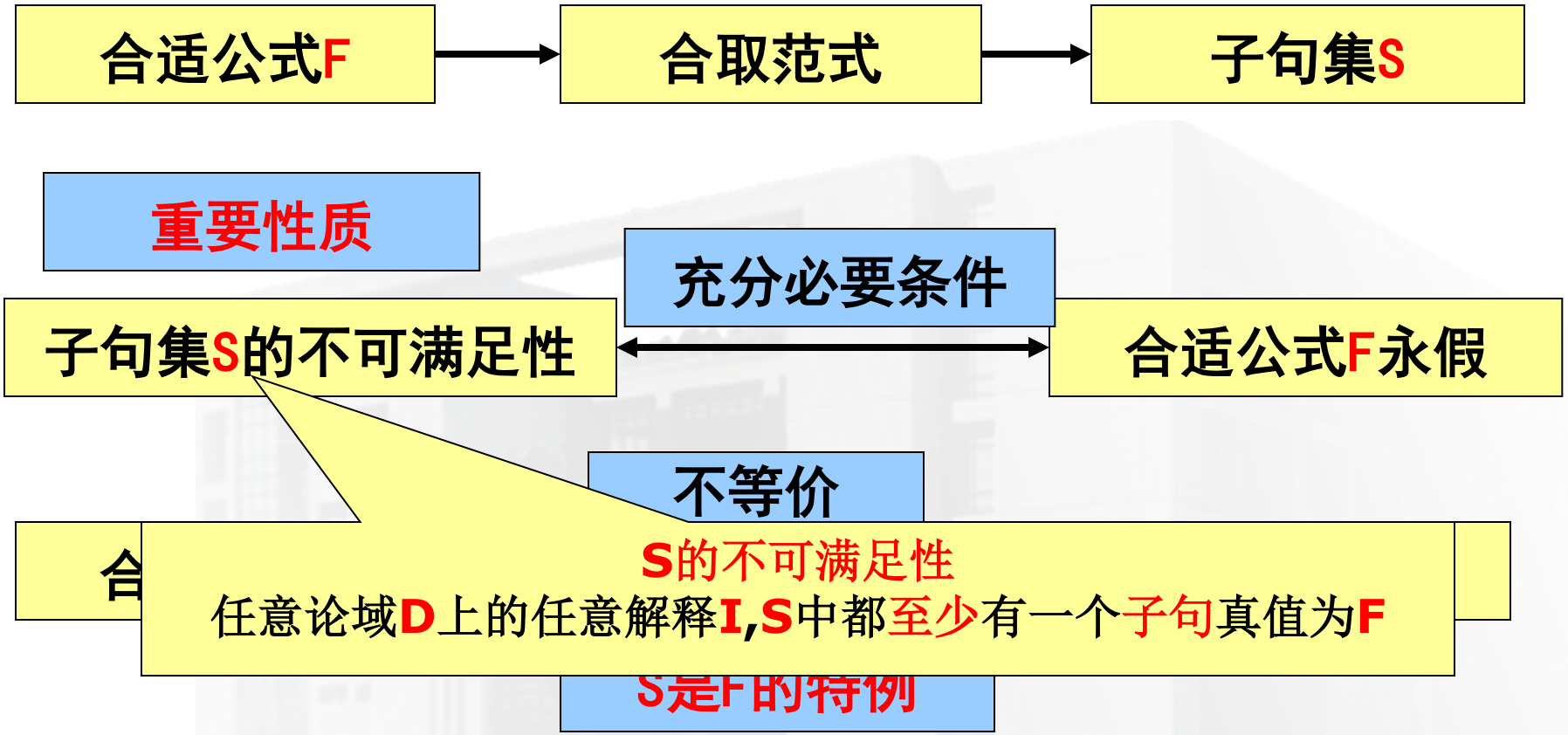
- 1、子句和子句集



归结演绎推理

1）H域和海伯伦定理

- 1、子句和子句集



归结演绎推理

1）H域和海伯伦定理

- 2、H域（了解）
  - 证明子句集S的不可满足性与证明合适公式永真性类似
    - 由于个体论域的任意性和解释个数的无限性，使得证明工作十分困难。
  - 若能建造一个较为简单的特殊论域，使得只要证明子句集S在该域不可满足，就可确保子句集在任何可能的论域上不可满足，将是十分有意义的！
  - 海伯伦建立的特殊域H就具有这样的性质。
  - H域性质——对于子句集S的任一可能论域D上的任一解释I，总能在S的H域上构造一个相应的解释I*，使子句集S具有相同的真值。
  - 证明子句集S的不可满足性
    - 确定子句集S在H域上的所有解释都不可满足。

归结演绎推理

1）H域和海伯伦定理

3、海伯伦定理（了解）

- 子句集S中一子句包含的变量用H域中元素取代后，产生的子句称为基子句。
- 海伯伦定理：
  - 子句集S不可满足的充要条件是存在一个有限的不可满足的基子句集S’。

有限的不可满足的基子句集S’ ⟷ 子句集S不可满足性

充分必要条件

归结演绎推理

2）归结原理★

- 动机
  - 为提高判定子句集S不可满足的有效性，鲁宾逊于1965年提出了归结(Resolution)原理，也称为消解原理。
  - 归结原理简单易行，便于计算机实现和执行，从而使定理的机器自动证明成为现实，也成为人工智能技术实用化的一次重要突破。

归结演绎推理

2）归结原理

- 1、归结方法
- (1)归结式（消解式*）
  - 设有两个子句$C_1=L \vee C_1'$、 $C_2=\neg L \vee C_2'$
    - ①从C_1和C_2中消去互补文字L和$\neg L$；
    - ②C_1'和C_2'通过$\vee$组成新的子句 $C=C_1' \vee C_2'$；
    - 称C为C_1和C_2的归结式（消解式）；
  - 例、两个子句
    - $C_1=P(A) \vee Q(x) \vee R(f(x))$
    - $C_2=\neg P(A) \vee Q(y) \vee R(y)$
    - 消去互补文字P(A)和$\neg P(A)$后，生成归结式：
    - $C_{12}= Q(x) \vee R(f(x)) \vee Q(y) \vee R(y)$

C1’ C2’

归结演绎推理

2）归结原理

- 1、归结方法
- (2)归结式性质
  - 定理：两个子句C_1和C_2的归结式C是C_1和C_2的逻辑推论
    - 任一使子句C_1和C_2为真的解释I，必使归结式C为真；
    - 归结式C为假的解释I，子句C_1或者C_2为假；
  - 推论：
    - 设C1 和C2 是子句集S中的两个子句，并以C作为它们的归结式；
    - 通过往S中加入C而产生的扩展子句集S’与子句集S在不可满足的意义上是等价的！

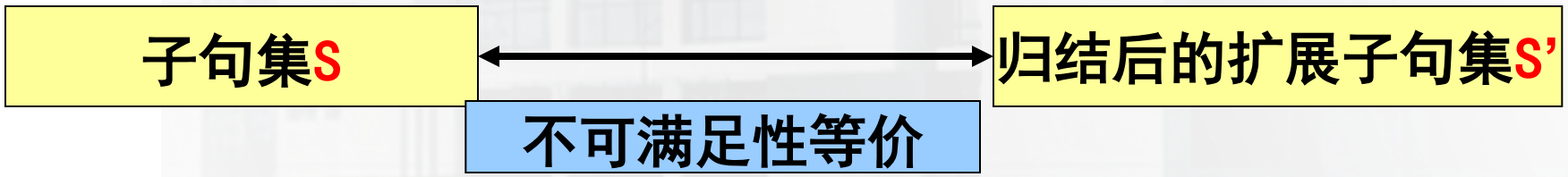
子句集S ⟷ 归结后扩展子句集S’

不可满足性等价

归结演绎推理

2）归结原理

- 1、归结方法
- (3)空子句
  - 设C_1=L、 C_2=¬L，则归结式C为空；
  - 以□表示为空的归结式C，并称C=□为空子句；
  - 因为C_1和C_2是一对矛盾子句，不可同时满足，所以□是不可满足的子句；
  - 通过往S中加入□而产生的扩展子句集S’不可满足；
  - 空子句□是用归结原理判定子句集S不可满足的成功标志。



对基子句的消解

(1) 假言推理 C1=P, C2=~P ∨Q

父辈子句P　　~P∨Q（即P=>Q）

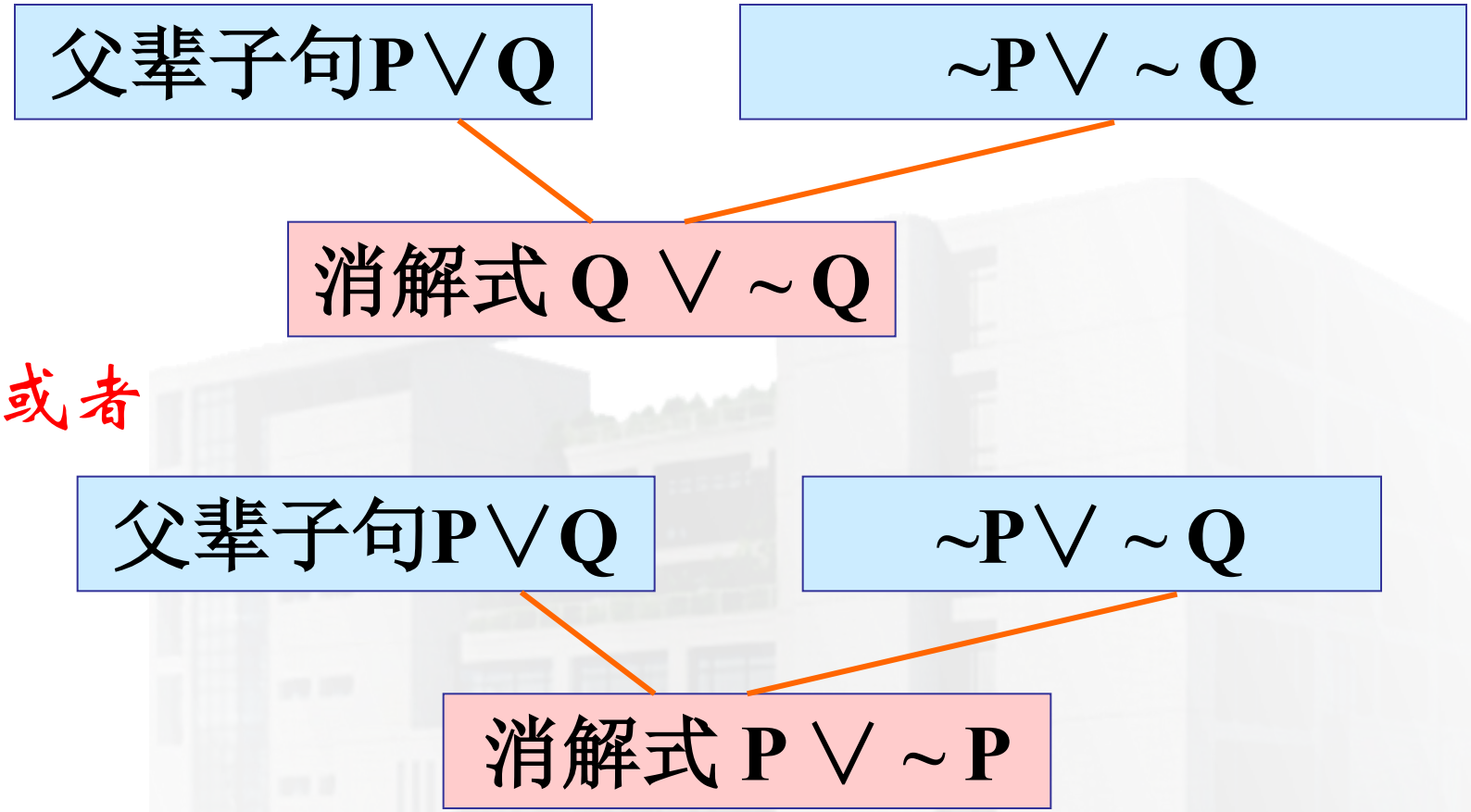
消解式 Q

（2）合并 C1=P∨Q,C2=~P∨Q

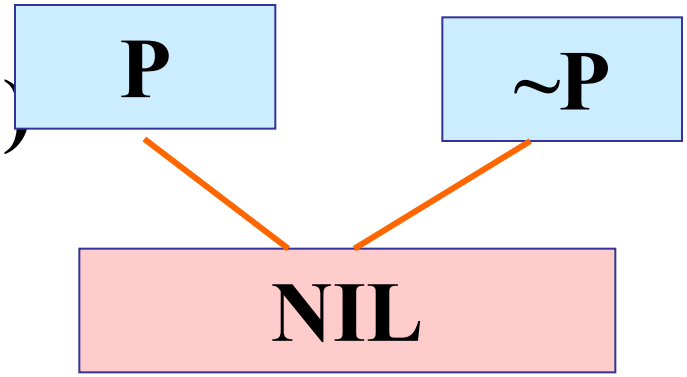
父辈子句P∨Q　　~P∨Q

消解式 Q ∨Q=Q

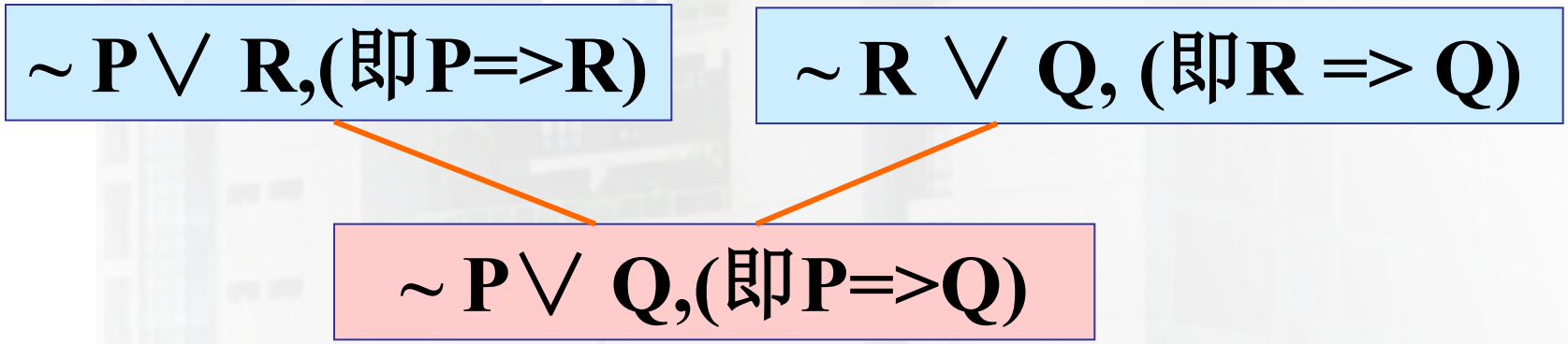
(3) 重言式 C1=P∨Q,C2=~P∨~Q



(4) 空子句(矛盾)



（5）链式（三段论）



归结演绎推理

2）归结原理

- 动机
  - 为提高判定子句集S不可满足的有效性，鲁宾逊于1965年提出了归结(Resolution)原理，也称为消解原理。
  - 归结原理简单易行，便于计算机实现和执行，从而使定理的机器自动证明成为现实，也成为人工智能技术实用化的一次重要突破。
- 基本思路★
  - 通过归结方法不断扩充待判定的子句集S，并设法使其包含进指示矛盾的空子句。
  - 空子句是不可满足（即永假）的子句；

归结演绎推理

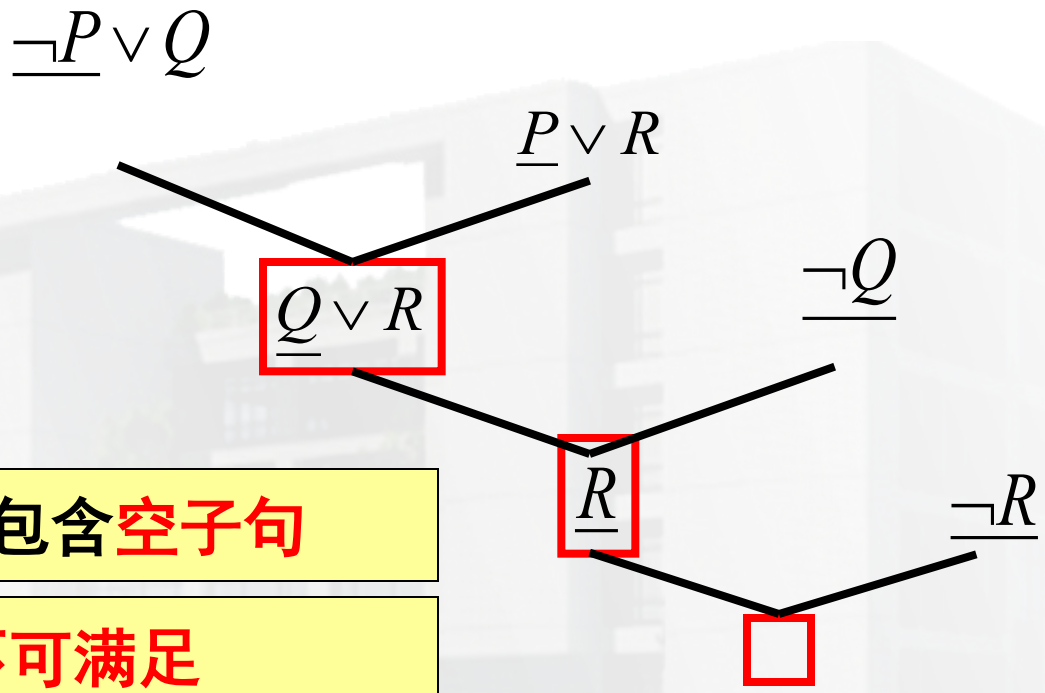
2）归结原理

- 2、归结推理过程——归结演绎树
- (1)命题逻辑中的归结推理过程
  - 子句中文字只是原子命题公式或其取反，不带变量；
  - 易于判别哪些子句对包含互补文字；
  - 例、子句集S={¬P∨Q, P∨R, ¬Q, ¬R}的归结演绎树。

归结演绎推理

2）归结原理

- 2、归结推理过程
- (1)命题逻辑中的归结推理过程
  - 例、子句集S={¬P∨Q, P∨R, ¬Q, ¬R}的归结演绎树。



扩展子句集S'包含空子句

子句集S不可满足

归结演绎推理

2）归结原理-置换和合一★

- 2、归结推理过程

$P(x, y, x, g(x)), \neg P(A, B, A, z)$

- (2)谓词逻辑中的归结推理过程
  - 子句中含有变量，不能直接发现和消去互补文字；
  - 需对潜在的互补文字先作变量置换和合一处理。
  - 变量置换：
    - 用置换项取代公式中的变量；
    - 置换项可以是变量、常量或函数。
    - 置换元素——t/v（置换项/变量）
    - 置换——若干置换元素的集合；
  - 合一处理：

S1使潜在的互补文字中的原子谓词公式变为同一，确认互补性。

$P(x, y, x, g(x)), \neg P(A, B, A, z)$

我们可以为它们建立多个置换：

$S_1 = \{A/x, B/y, g(x)/z\}$

$S_2 = \{f(w)/x, z/y, C/z\}$

$S_3 = \{B/x, f(w)/y, y/z\}$

置换结果为：

$\{P(x, y, x, g(x)), \neg P(A, B, A, z)\}S1$

$=\{P(A, B, A, g(A)), \neg P(A, B, A, g(A))\}$

$\{P(x, y, x, g(x)), \neg P(A, B, A, z)\}S2$

$=\{P(f(w)), z, f(w), g(f(w)), \neg P(A, B, A, C)\}$

$\{P(x, y, x, g(x)), \neg P(A, B, A, z)\}S3$

$=\{P(B, f(w), B, g(B)), \neg P(A, B, A, y)\}$

归结演绎推理

2）归结原理-置换和合一

- 2、归结推理过程

P(x, y, x, g(x)), ¬P(A, B, A, z)

- (2)谓词逻辑中的归结推理过程
  - 子句中含有变量，不能直接发现和消去互补文字；
  - 需对潜在的互补文字先作变量置换和合一处理。
  - 变量置换：
    - 用置换项取代公式中的变量；
    - 置换项可以是变量、常量或函数。
    - 置换元素——t/v（置换项/变量）
    - 置换——若干置换元素的集合；
  - 合一处理：
    - 通过多个变量置换，使相应于潜在互补文字的原子谓词公式同一化的过程。

归结演绎推理

2）归结原理-置换和合一

- 2、归结推理过程
- (2) 谓词逻辑中的归结推理过程
  - 通过一个匹配过程去检查两个原子谓词公式的可合一性，并同时建立用于实现合一的置换。
  - 【匹配过程】★
    - ①两个原子谓词公式必须具有相同的谓词符号和参数项个数；
    - ②从左到右逐个检查每对参数项的可合一性；
    - ③若每对参数项都可合一，则合一处理成功，并建立用于实现合一的置换。

归结演绎推理

2）归结原理-置换和合一

- 【匹配过程】
  - ②从左到右逐个检查每对参数项的可合一性；
    - 参数对中有一个是变量v（不关心另一个t是否是变量）
      - v初次出现，参数对可合一，以另一参数t为置换项，构成置换元素t/v;
      - v出现过，则已建立相应的置换元素，就取其置换项，代替该变量，检查是否与另一参数同一；若不同一，则处理失败；
    - 参数对中没有变量，则必须相同，否则合一处理失败。

归结演绎推理

2）归结原理-置换和合一

- 2、归结推理过程
- (2)谓词逻辑中的归结推理过程
  - 匹配过程的例子
  - P(x, y, x, g(x)), ¬P(A, B, A, z)
  - ①两个原子谓词公式必须具有相同的谓词和参数项个数；
  - ②从左到右逐个检查每对参数项的可合一性；
    - x和A，x初次出现，可合一，建立置换元素A/x;
    - y和B，y初次出现，可合一，建立置换元素B/y;
    - x和A，x出现过，已经建立置换元素A/x，可合一；
    - g(x)和z，z初次出现，可合一，建立置换元素g(x)/z；
  - ③若每对参数项都可合一，则合一处理成功。
  - 建立置换S_1={A/x, B/y, g(x)/z}

归结演绎推理

2）归结原理-置换和合一

- 2、归结推理过程
- (2)谓词逻辑中的归结推理过程
  - 谓词逻辑中子句归结的例子：
    - $C_1=P(x,y) \vee Q(x,f(x)) \vee R(x,f(y))$
    - $C_2=\neg P(A,B) \vee \neg Q(z,f(z)) \vee \neg R(z,f(z))$
  - $L_{11}=P(x,y)$和$L_{12}=\neg P(A,B)$是潜在的互补文字
    - L_{11}和L_{12}是可合一的；
    - 建立置换$S_1=\{A/x,B/y\}$
    - 消去互补文字L_{11}和L_{12}后，得归结式：

      $Q(A,f(A)) \vee R(A,f(B)) \vee \neg Q(z,f(z)) \vee \neg R(z,f(z))$
    - 变量的置换必须在整个子句范围内进行

归结演绎推理

2）归结原理-置换和合一

- 2、归结推理过程
- (2)谓词逻辑中的归结推理过程
  - 谓词逻辑中子句归结的例子：
    - $C_1=P(x,y) \vee Q(x,f(x)) \vee R(x,f(y))$
    - $C_2=\neg P(A,B) \vee \neg Q(z,f(z)) \vee \neg R(z,f(z))$
  - $L_{21}=Q(x,f(x))$和$L_{22}=\neg Q(z,f(z))$是潜在的互补文字
    - L_{21}和L_{22}是可合一的；
    - 建立置换$S_2=\{z/x\}$
    - 消去互补文字L_{21}和L_{22}后，得归结式：

$P(z,y) \vee R(z,f(y)) \vee \neg P(A,B) \vee \neg R(z,f(z))$

归结演绎推理

2）归结原理-置换和合一

- 2、归结推理过程
- (2)谓词逻辑中的归结推理过程
  - 谓词逻辑中子句归结的例子：
    - $C_1=P(x,y) \vee Q(x,f(x)) \vee R(x,f(y))$
    - $C_2=\neg P(A,B) \vee \neg Q(z,f(z)) \vee \neg R(z,f(z))$
  - $L_{11}=P(x,y)$和$L_{12}=\neg P(A,B)$是互补文字
  - $L_{21}=Q(x,f(x))$和$L_{22}=\neg Q(z,f(z))$是互补文字
  - 两个子句可以有多于一对的互补文字

归结演绎推理

2）归结原理

- 2、归结推理过程
- (2)谓词逻辑中的归结推理过程

$$(1)\neg R(x,y)\vee\neg Q(y)\vee P(f(x))$$

$$(2)\neg R(z,y)\vee\neg Q(y)\vee W(x,f(x))$$

$$(3)\neg P(z)$$

$$(4)R(A,B)$$

$$(5)Q(B)$$

归结演绎推理

2）归结原理

- 2、归结推理过程
- (2)谓词逻辑中的归结推理过程

$$(3)\neg P(z)$$

$$(1)\neg R(x,y)\vee\neg Q(y)\vee P(f(x))$$

$f(x)/z$

$$\neg R(x,y)\vee\neg Q(y)$$

$$(1)\neg R(x,y)\vee\neg Q(y)\vee P(f(x))$$

$$(2)\neg R(z,y)\vee\neg Q(y)\vee W(x,f(x))$$

$$(3)\neg P(z)$$

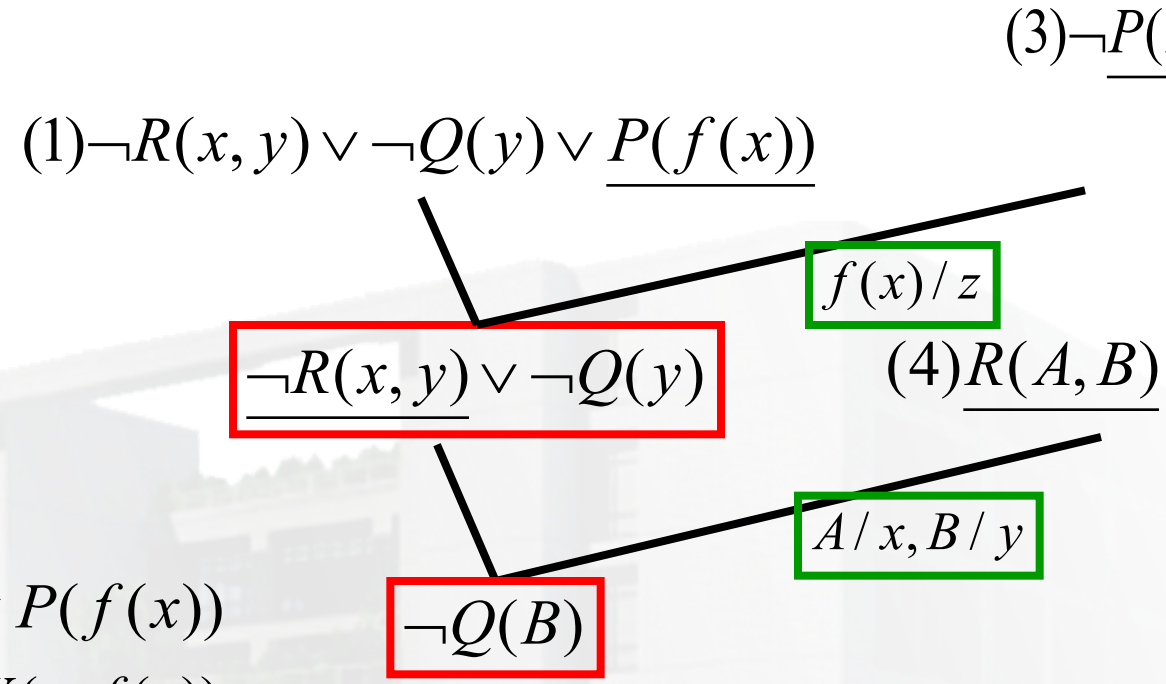
$$(4)R(A,B)$$

$$(5)Q(B)$$

归结演绎推理

2）归结原理

- 2、归结推理过程
- (2)谓词逻辑中的归结推理过程



(1)$\neg R(x,y)\vee\neg Q(y)\vee P(f(x))$

(2)$\neg R(z,y)\vee\neg Q(y)\vee W(x,f(x))$

(3)$\neg P(z)$

(4)$R(A,B)$

(5)$Q(B)$

归结演绎推理

2）归结原理

- 2、归结推理过程
- (2)谓词逻辑中的归结推理过程

子句集S不可满足

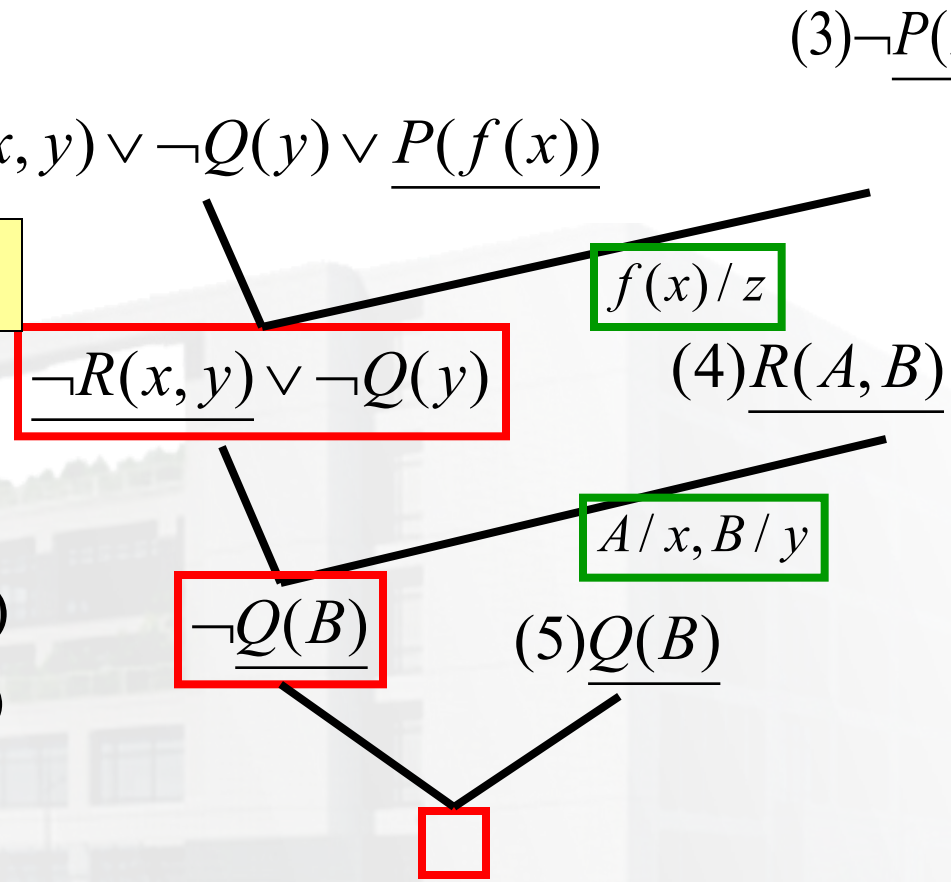
$(1)\neg R(x,y)\vee\neg Q(y)\vee P(f(x))$

$(2)\neg R(z,y)\vee\neg Q(y)\vee W(x,f(x))$

$(3)\neg P(z)$

$(4)R(A,B)$

$(5)Q(B)$



归结演绎推理
2）归结原理

- 2、归结推理过程
- (3)归结演绎的完备性
  - 基于归结的演绎推理是完备的
    - 若子句集S不可满足，就必定存在一个从S到空子句□的归结演绎；
    - 反之，若存在一个从S到空子句□的归结演绎，则S必定是不可满足的；
  - 归结演绎的完备性可用海伯伦定理进行证明，因此归结原理是建立在海伯伦定理之上的。
  - 但归结原理并不能用于解决子句集不可满足性的不可判问题，即若子句集S是永真的和可满足的，归结原理判定过程将无休止地进行下去，得不到任何结果。

归结演绎推理

3）归结反演★

- 归结演绎推理为反证法证明定理提供了有效手段。
- 应用归结演绎推理的定理证明为归结反演。
- 教学要求——掌握
- 主要内容
  - 掌握归结反演方法和提取问题回答技术；
  - 学会建立归结反演树和修改证明树。
- 学习难点
  - 从以自然语言表示的事实集证明目标公式（定理）并提取回答的综合题。

归结反演的原理

欲证F1∧F2∧...∧Fn⇒W永真，可以通过F1∧F2∧...∧Fn∧¬W的永假来证明。

子句集S的不可满足性 ⟷（充分必要条件）⟷ 合适公式F永假

有限的不可满足的基子句集S' ⟷ 子句集S不可满足性

（充分必要条件）

子句集S ⟷（不可满足性等价）⟷ 归结后扩展子句集S'

归结的演绎推理的完备性：若子句集S不可满足，就必定存在一个从S到空子句□的归结演绎；反之亦然。

归结演绎推理

3）归结反演——归结反演系统

- 归结反演的基本思路：
  - 要从作为事实的公式集F证明目标公式W为真；
  - ①先将W取反¬W，加入公式集F；
  - ②标准化F∧¬W为子句集S；
  - ③通过归结演绎证明S不可满足，得出W为真的结论。
- 归结反演系统组成：
  - 标准化部件
    - 将F∧¬W标准化为子句，并合并为子句集S；
  - 归结演绎部件
    - 按照归结演绎推理，控制定理证明的全过程。

归结演绎推理

3）归结反演——归结反演系统

- 例、归结反演的应用——食物问题
- 已知下列事实为真T，
  - 王(Wang)喜欢(Like)所有种类的食物(Food)。
  - 苹果(Apples)是食物。
  - 任何一个东西，若任何人吃了(Eat)它都不会被害死(Killed)，则该东西是食物。
  - 李(Li)吃花生(Peanuts)且仍然活着(Alive)。
  - 张(Zhang)吃任何李吃的东西。
- 证明：
  - 王喜欢花生。

归结演绎推理

3）归结反演——归结反演系统

- 例、归结反演的应用——食物问题
- ①形式化——把以自然语言表示的事实和要证明的目标形式化地表示为合适公式。
  - 王(Wang)喜欢(Like)所有种类的食物(Food)。
    - $(\forall x)[Food(x) \Rightarrow Like(Wang,x)]$
  - 苹果(Apples)是食物。
    - $Food(Apples)$
  - 任何一个东西，若任何人吃了(Eat)它都不会被害死(Killed)，则该东西是食物。
    - $(\forall x)(\forall y)[Eat(y,x) \wedge \neg Killed(y) \Rightarrow Food(x)]$
    - $(\forall x)(\forall y)[Eat(y,x) \wedge Alive(y) \Rightarrow Food(x)]$
  - 李(Li)吃花生(Peanuts)且仍然活着(Alive)。
    - $Eat(Li,Peanuts) \wedge Alive(Li)$
  - 张(Zhang)吃任何李吃的东西。
    - $(\forall x)[Eat(Li,x) \Rightarrow Eat(Zhang,x)]$
  - 王喜欢花生。
    - $Like(Wang,Peanuts)$

减少谓词数

归结演绎推理

3）归结反演——归结反演系统

- 例、归结反演的应用——食物问题
- ②标准化——将事实公式和取反后的目标公式分别标准化为子句，并组成子句集S。
  - 王(Wang)喜欢(Like)所有种类的食物(Food)。
    - $(\forall x)[Food(x) \Rightarrow Like(Wang,x)]$
    - $\neg Food(x1) \vee Like(Wang,x1)$
  - 苹果(Apples)是食物。
    - $Food(Apples)$
    - $Food(Apples)$
  - 任何一个东西，若任何人吃了(Eat)它都不会被害死(Killed)，则该东西是食物。
    - $(\forall x)(\forall y)[Eat(y,x) \wedge Alive(y) \Rightarrow Food(x)]$
    - $\neg Eat(y,x2) \vee \neg Alive(y) \vee Food(x2)$

归结演绎推理

3）归结反演——归结反演系统

- 例、归结反演的应用——食物问题
- ②标准化——将事实公式和取反后的目标公式分别标准化为子句，并组成字句集S。
  - 李(Li)吃花生(Peanuts)且仍然活着(Alive)。
    - Eat(Li,Peanuts) ∧ Alive(Li)
    - Eat(Li,Peanuts)
    - Alive(Li)
  - 张(Zhang)吃任何李吃的东西。
    - (∀x)[Eat(Li,x) ⇒ Eat(Zhang,x)]
    - ¬ Eat(Li,x3) ∨ Eat(Zhang,x3)
  - 王喜欢花生。
    - Like(Wang,Peanuts)
    - ¬ Like(Wang,Peanuts)取反

归结演绎推理

3）归结反演——归结反演系统

- 例、归结反演的应用——食物问题
- ③归结演绎——应用归结演绎推理不断生成归结式以扩展子句集S，直到生成空子句□。

$(1)\neg Food(x_1) \vee Like(Wang, x_1)$

$(2)Food(Apples)$

$(3)\neg Eat(y, x_2) \vee \neg Alive(y) \vee Food(x_2)$

$(4)Eat(Li, Peanuts)$

$(5)Alive(Li)$

$(6)\neg Eat(Li, x_3) \vee Eat(Zhang, x_3)$

$(7)\neg Like(Wang, Peanuts)$

归结演绎推理

3）归结反演——归结反演系统

- 例、归结反演的应用——食物问题
- ③归结演绎——应用归结演绎方法不断生成归结式以扩展子句集S，直到生成空子句□。

$(1)\neg Food(x_1) \vee Like(Wang, x_1)$

$(2) Food(Apples)$

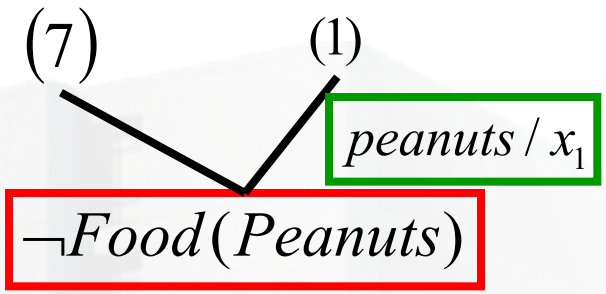
$(3)\neg Eat(y, x_2) \vee \neg Alive(y) \vee Food(x_2)$

$(4) Eat(Li, Peanuts)$

$(5) Alive(Li)$

$(6)\neg Eat(Li, x_3) \vee Eat(Zhang, x_3)$

$(7)\neg Like(Wang, Peanuts)$



归结演绎推理

3）归结反演——归结反演系统

- 例、归结反演的应用——食物问题
- ③归结演绎——应用归结演绎方法不断生成归结式以扩展子句集S，直到生成空子句□。

$(1)\neg Food(x_1) \vee Like(Wang, x_1)$

$(2)Food(Apples)$

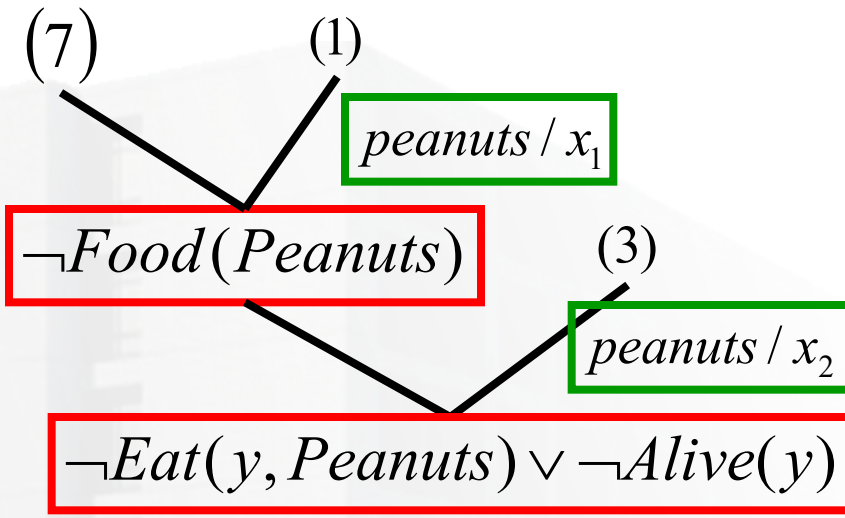
$(3)\neg Eat(y, x_2) \vee \neg Alive(y) \vee \underline{Food(x_2)}$

$(4)Eat(Li, Peanuts)$

$(5)Alive(Li)$

$(6)\neg Eat(Li, x_3) \vee Eat(Zhang, x_3)$

$(7)\neg Like(Wang, Peanuts)$



归结演绎推理

3）归结反演——归结反演系统

- 例、归结反演的应用——食物问题
- ③归结演绎——应用归结演绎方法不断生成归结式以扩展子句集S，直到生成空子句□。

(1) $\neg Food(x_1) \vee Like(Wang, x_1)$

(2) $Food(Apples)$

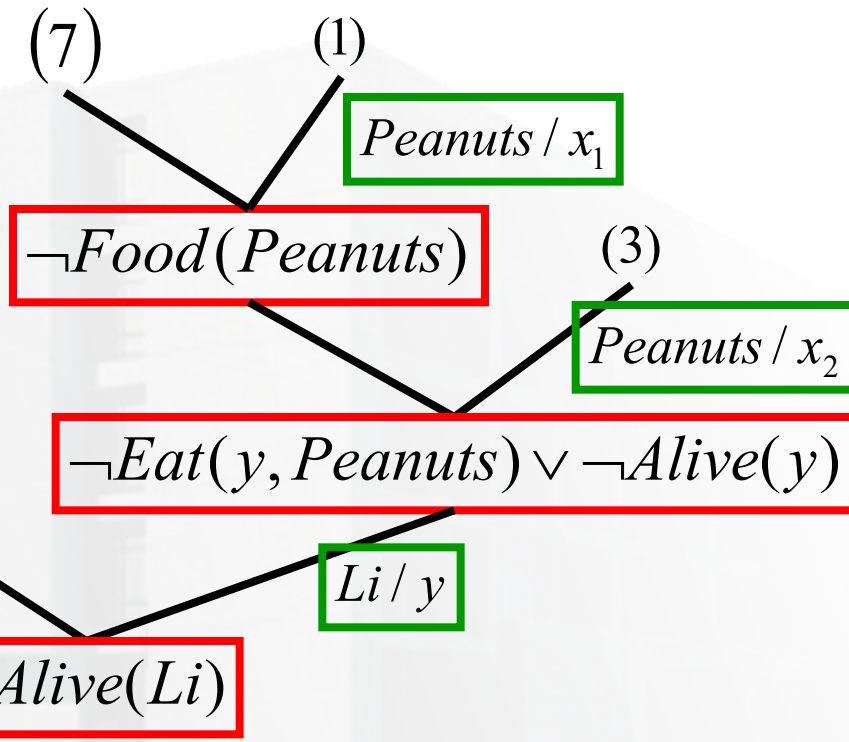
(3) $\neg Eat(y, x_2) \vee \neg Alive(y) \vee Food(x_2)$

(4) $Eat(Li, Peanuts)$

(5) $Alive(Li)$

(6) $\neg Eat(Li, x_3) \vee Eat(Zhang, x_3)$

(7) $\neg Like(Wang, Peanuts)$



归结演绎推理

3）归结

目标公式得证

归结反演树

- 例、归结反演的应用——食物问题
- ③归结演绎——应用归结演绎方法不断生成归结式以扩展子句集S，直到生成空子句□。

$(1)\neg Food(x_1) \vee Like(Wang, x_1)$

$(2)Food(Apples)$

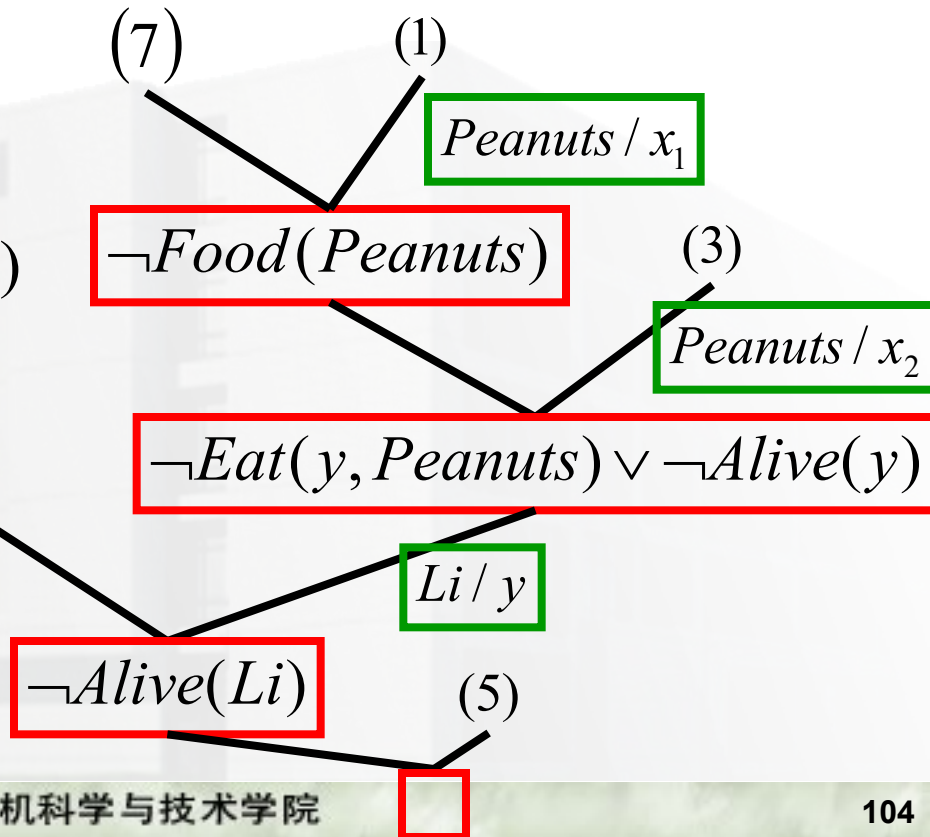
$(3)\neg Eat(y, x_2) \vee \neg Alive(y) \vee Food(x_2)$

$(4)Eat(Li, Peanuts)$

$(5)Alive(Li)$

$(6)\neg Eat(Li, x_3) \vee Eat(Zhang, x_3)$

$(7)\neg Like(Wang, Peanuts)$



设置

课堂练习

- 某公司招聘工作人员，A，B，C三人应试，经面试后，公司表示如下想法：
  - （1） 三人中至少录取一人。
  - （2） 如果录取A而不录取B，则一定录取C。
  - （3） 如果录取B，则一定录取C。
- 用归结反演法证明：公司一定录取C。
- （ 提示：设用P(x)表示录取x）

正常使用主观题需2.0以上版本雨课堂

作答

归结演绎推理

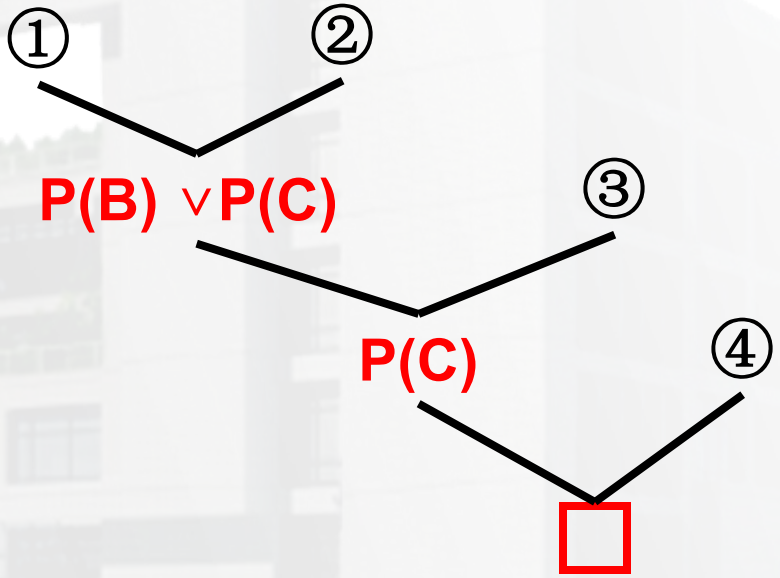
3）归结反演

- 把公司的想法用谓词公式表示如下：
  - （1） 三人中至少录取一人。
    - $P(A) \vee P(B) \vee P(C)$
  - （2） 如果录取A而不录取B，则一定录取C。
    - $P(A) \wedge \neg P(B) \Rightarrow P(C)$
  - （3） 如果录取B，则一定录取C。
    - $P(B) \Rightarrow P(C)$
- 把要求证的问题否定，并用谓词公式表示出来：
  - 公司一定录取C
    - $\neg P(C)$

归结演绎推理

3）归结反演

- 把上述公式化成子句集
  - ①P(A) ∨P(B) ∨P(C)
  - ②¬P(A) ∨P(B) ∨ P(C)
  - ③¬P(B) ∨P(C)
  - ④¬P(C)
- 应用归结反演：



归结反演的原理

欲证F1∧F2∧...∧Fn⇒W永真，可以通过F1∧F2∧...∧Fn∧¬W的永假来证明。

子句集S的不可满足性 ⟷（充分必要条件）⟷ 合适公式F永假

有限的不可满足的基子句集S' ⟷ 子句集S不可满足性

（充分必要条件）

子句集S ⟷（不可满足性等价）⟷ 归结后扩展子句集S'

归结的演绎推理的完备性：若子句集S不可满足，就必定存在一个从S到空子句□的归结演绎；反之亦然。

归结演绎推理

3）归结反演

- 某记者到一个孤岛采访，遇到一个难题，即岛上有许多人说假话，因而难以保证新闻报道的正确性，不过有一点他是清楚的，这个岛上的人有一个特点：
  - 说假话的人从来不说真话，说真话的人从来不说假话；
- 一次，记者遇到了孤岛上的3个人，为了弄清谁说真话，谁说假话，他向这3个人中的每一个都提了一个同样的问题“谁是说谎者？”
  - A回答：B和C都是说谎者。
  - B回答：A和C都是说谎者。
  - C回答：A和B中至少有一个是说谎者。
- 谁是说谎者？

归结演绎推理

3）归结反演

- 设A、B、C三个命题表示A、B、C三人是老实人。
- A回答：B和C都是说谎者。
  - $A \Rightarrow \neg B \wedge \neg C$
  - $\neg A \Rightarrow B \vee C$
- B回答：A和C都是说谎者。
  - $B \Rightarrow \neg A \wedge \neg C$
  - $\neg B \Rightarrow A \vee C$
- C回答：A和B中至少有一个是说谎者。
  - $C \Rightarrow \neg A \vee \neg B$
  - $\neg C \Rightarrow A \wedge B$

归结演绎推理

3）归结反演

- 设A、B、C三个命题表示A、B、C三人是老实人。
- 化简上述蕴含式为子句集
- ①¬ A ∨ ¬ B
- ②¬ A ∨ ¬ C
- ③A ∨ B ∨ C
- ④¬ B ∨ ¬ C
- ⑤¬ A ∨ ¬ B ∨ ¬ C
- ⑥A ∨ C
- ⑦B ∨ C

①和⑦归结：¬ A ∨ C⑧

②和⑧归结：¬ A⑨

⑥和⑨归结：C⑩

④和⑩归结：¬ B

结论：A和B都是说谎者，而C是老实人

设置

课堂练习

- **例: 王某被害，有四个嫌疑犯A，B，C，D，公安局派出五个侦察员，他们带回的信息各不一样，甲说A，B中至少有一人作案，乙说B ，C中至少有一人作案，丙说C，D中至少有一人作案，丁说A，C中至少有一人与此案无关，戊说B ，D中至少有一人与此案无关，如果这五个侦察员的话都是可靠的，那么谁是罪犯。**

正常使用主观题需2.0以上版本雨课堂

作答

（1） A ∨ B　　（2） B ∨ C

（3）C ∨ D　　（4） ~ A ∨ ~ C

（5） ~ B ∨ ~ D

（6） B ∨ ~ C　　（1）、（4）归结；

（7） B是罪犯。　　（2）、（6）归结；

（8）C ∨ ~ D　　（2）、（5）归结；

（9） C是罪犯。　　（3）、（8）归结。

归结演绎推理

3）归结反演——提取问题回答

- 归结反演可实现问题回答系统
  - 目标公式往往受存在量词约束，如$(\exists x)W(x)$；
  - 不仅证明目标公式为真T；
  - 回答提取——给出使W(x)为真T的x的某个取值。
- 问题回答系统分为2个阶段：
  - ①归结反演——证明目标公式为真T
  - ②回答提取
    - 对于标准化取反的目标公式而产生的子句（称为目标子句）G,建立其重言式（永真式）$G \vee \neg G$；
    - 以$G \vee \neg G$取代G,重复已进行过的归结演绎过程，建立修改证明树。
  - 结果——修改证明树的树根不再是空子句□，而是$\neg G$，且G中的变量已为其置换项取代，实现了回答提取。

归结演绎推理

3）归结反演——提取问题回答

- 例、提取问题回答的应用——食物问题
- 已知下列事实为真，
  - 王(Wang)喜欢(Like)所有种类的食物(Food)。
  - 苹果(Apples)是食物。
  - 任何一个东西，若任何人吃了(Eat)它都不会被害死(Killed)，则该东西是食物。
  - 李(Li)吃花生(Peanuts)且仍然活着(Alive)。
  - 张(Zhang)吃任何李吃的东西。
- 证明：
  - 王喜欢花生。
- 问题：
  - 张吃什么食物？

归结演绎推理

3）归结反演——提取问题回答

- 例、提取问题回答的应用——食物问题
- (1)形式化
  - 王(Wang)喜欢(Like)所有种类的食物(Food)。
    - $(\forall x)[Food(x) \Rightarrow Like(Wang,x)]$
  - 苹果(Apples)是食物。
    - $Food(Apples)$
  - 任何一个东西，若任何人吃了(Eat)它都不会被害死(Killed)，则该东西是食物。
    - $(\forall x)(\forall y)[Eat(y,x) \wedge Alive(y) \Rightarrow Food(x)]$
  - 李(Li)吃花生(Peanuts)且仍然活着(Alive)。
    - $Eat(Li,Peanuts) \wedge Alive(Li)$
  - 张(Zhang)吃任何李吃的东西。
    - $(\forall x)[Eat(Li,x) \Rightarrow Eat(Zhang,x)]$
  - 问题“张吃什么食物？”形式化为目标公式
    - $(\exists x)[Food(x) \wedge Eat(Zhang, x)]$

归结演绎推理

3）归结反演——提取问题回答

- 例、提取问题回答的应用——食物问题
- (2)标准化

取反

$(\exists x)[Food(x) \wedge Eat(Zhang, x)]$ ⇨ $(8) \neg Food(x_4) \vee \neg Eat(Zhang, x_4)$

目标公式

目标子句G

$(1) \neg Food(x_1) \vee Like(Wang, x_1)$

$(2) Food(Apples)$

$(3) \neg Eat(y, x_2) \vee \neg Alive(y) \vee Food(x_2)$

$(4) Eat(Li, Peanuts)$

$(5) Alive(Li)$

$(6) \neg Eat(Li, x_3) \vee Eat(Zhang, x_3)$

$(7) \neg Like(Wang, Peanuts)$

各个子句变量不要重名

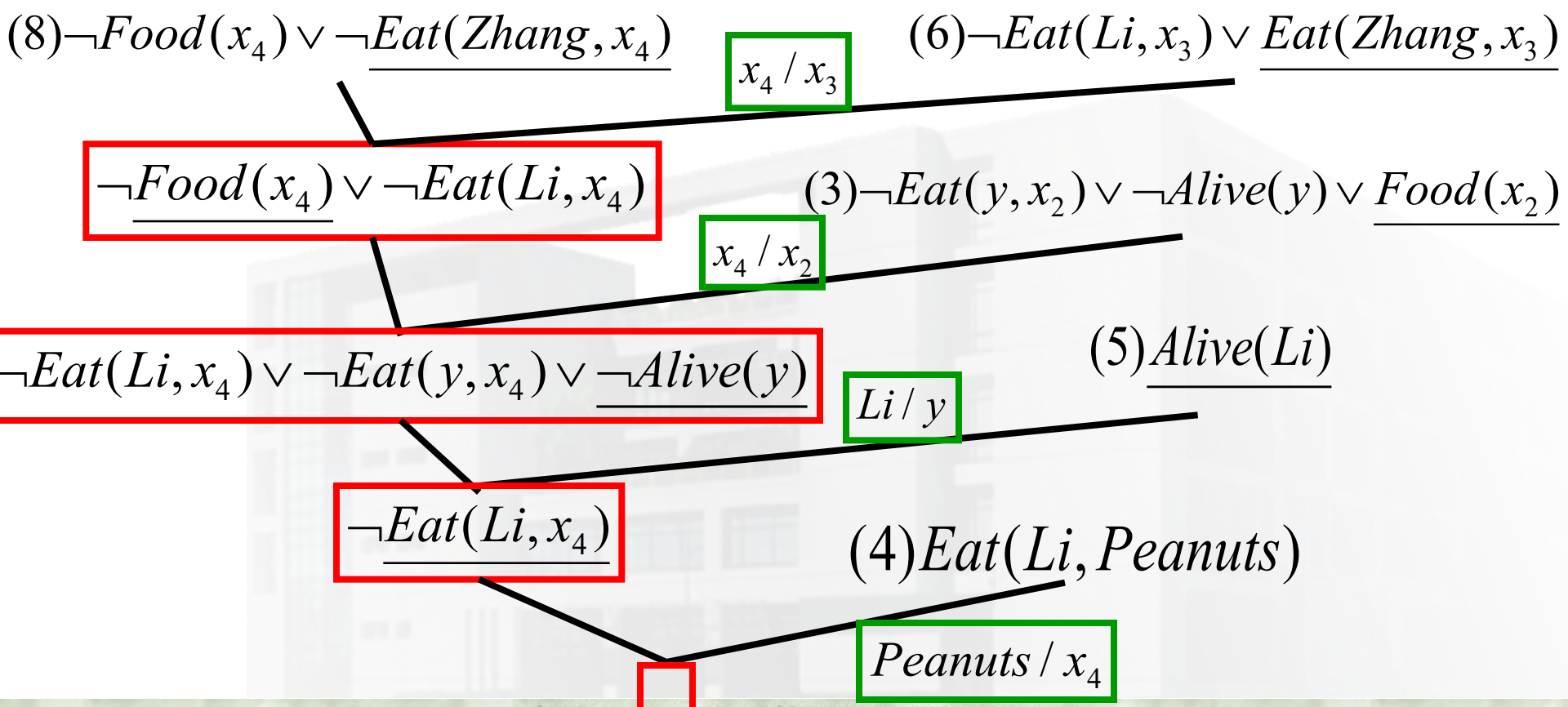
归结演绎推理

3）归结反演——提取问题回答

归结反演树

- 例、提取问题回答的应用——食物问
- ①归结反演

(∃x)[Food(x)∧Eat(Zhang, x)]



归结演绎推理

3）归结反演——提取问题回答

- 例、提取问题回答的应用——食物问题
- ②回答提取——(1)建立重言式G∨¬G

$(\exists x)[Food(x) \wedge Eat(Zhang, x)]$ ⇨ $(8)\neg Food(x_4) \vee \neg Eat(Zhang, x_4)$

目标公式　　取反　　目标子句G

$\neg Food(x_4) \vee \neg Eat(Zhang, x_4) \vee [Food(x_4) \wedge Eat(Zhang, x_4)]$

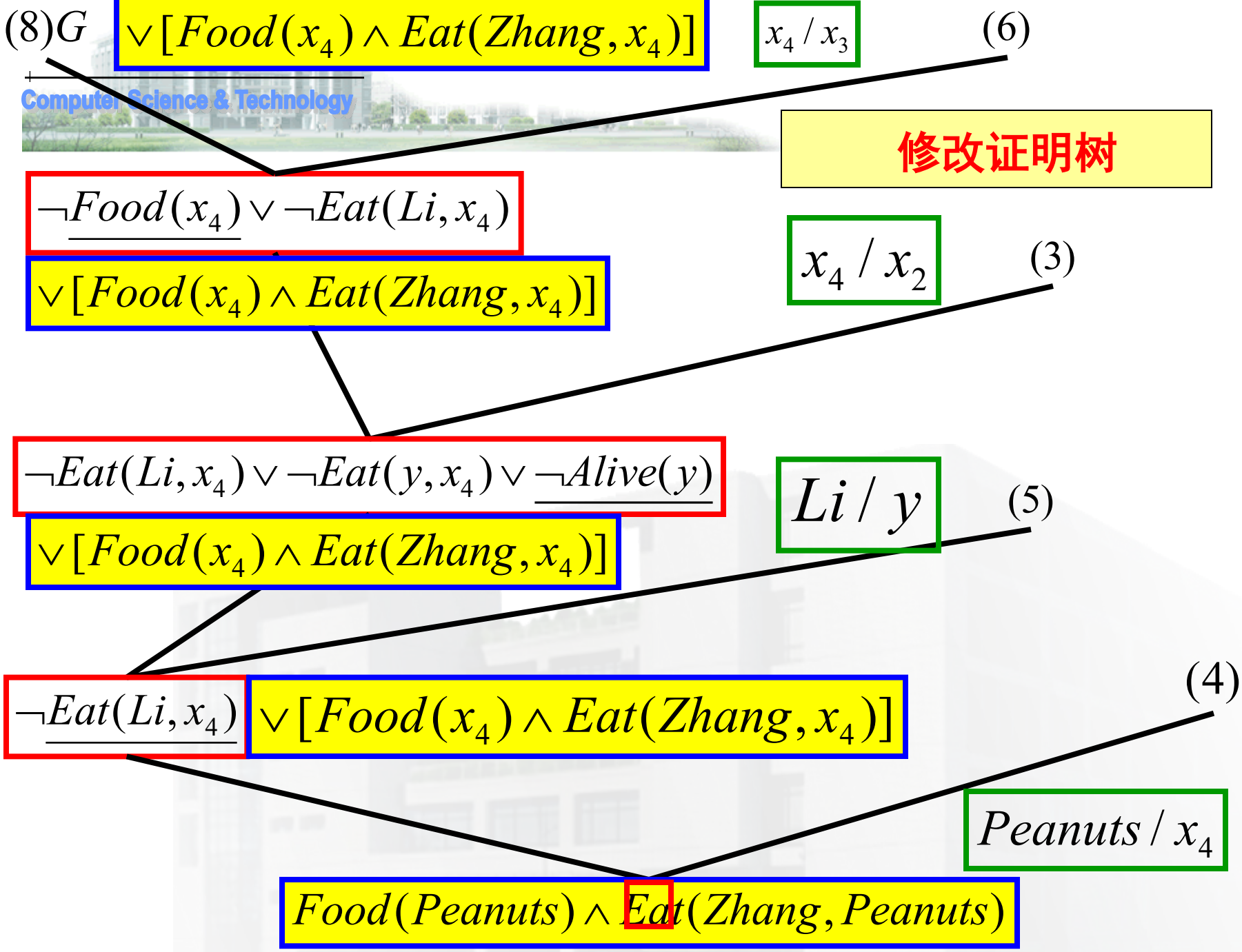
目标子句G　　¬G

建立重言式G∨¬G

归结演绎推理

3）归结反演——提取问题回答

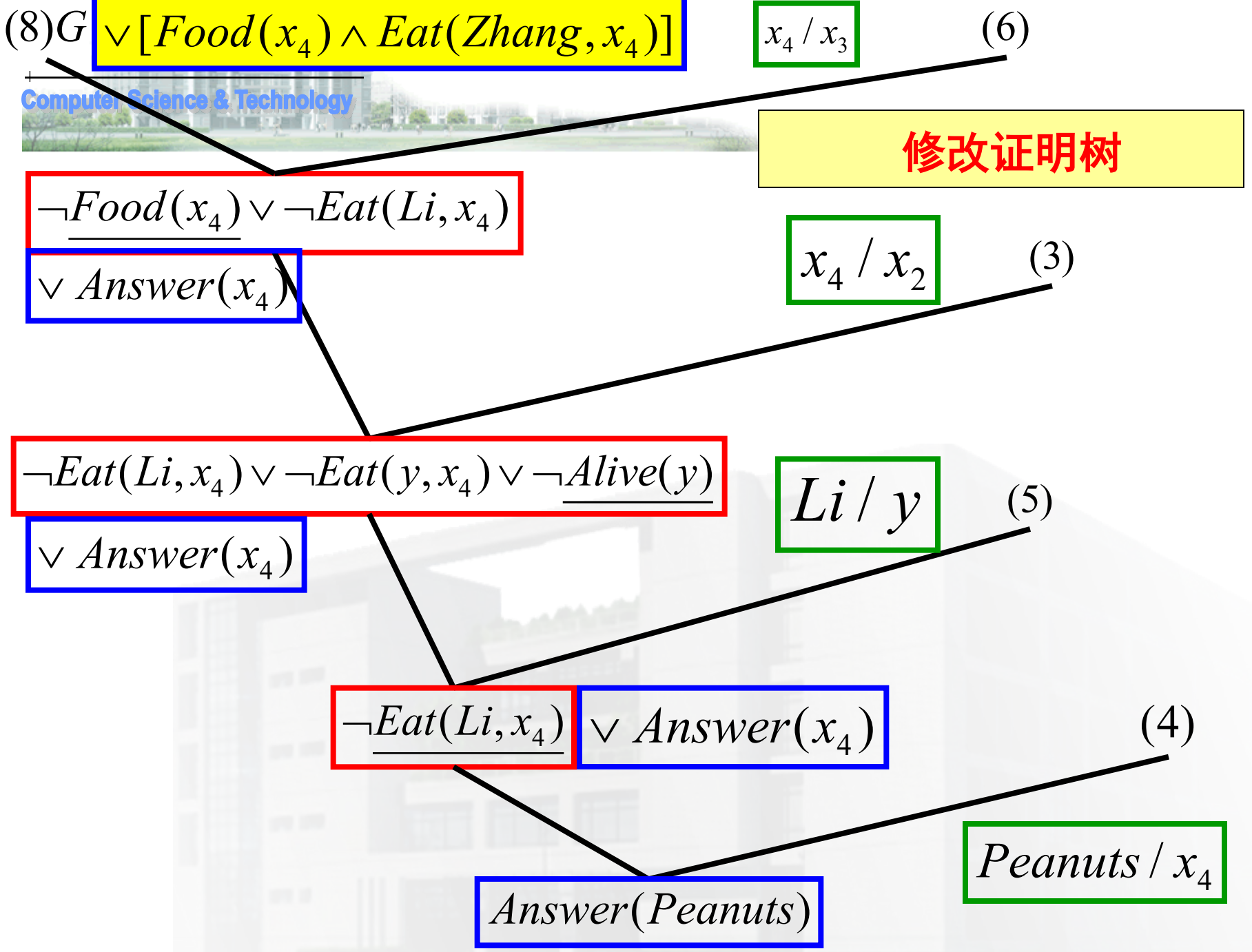
- 例、提取问题回答的应用——食物问题
- ②回答提取——(2) 以G∨¬G取代G,重复已进行过的归结演绎过程，建立修改证明树。
- 建立修改证明树过程中：
  - 1、重言式G∨¬G中的¬G并不真正参与归结演绎；
  - 2、G∨¬G取代G重复归结演绎过程，只是为了使¬G中的变量随着G中出现的变量置换而同时得到置换。


(8)G
$\vee[Food(x_4) \wedge Eat(Zhang, x_4)]$
x_4 / x_3
(6)
修改证明树
$\neg Food(x_4) \vee \neg Eat(Li, x_4)$
$\vee[Food(x_4) \wedge Eat(Zhang, x_4)]$
x_4 / x_2
(3)
$\neg Eat(Li, x_4) \vee \neg Eat(y, x_4) \vee \neg Alive(y)$
$\vee[Food(x_4) \wedge Eat(Zhang, x_4)]$
Li / y
(5)
$\neg Eat(Li, x_4)$
$\vee[Food(x_4) \wedge Eat(Zhang, x_4)]$
(4)
$Peanuts / x_4$
$Food(Peanuts) \wedge Eat(Zhang, Peanuts)$

归结演绎推理

3）归结反演——提取问题回答

- 例、提取问题回答的应用——食物问题
- ②回答提取——(2) 以G∨¬G取代G,重复已进行过的归结演绎过程，建立修改证明树。
- 建立修改证明树过程中：
  - 1、重言式G∨¬G中的¬G并不真正参与归结演绎；
  - 2、G∨¬G取代G重复归结演绎过程，只是为了使¬G中的变量随着G中出现的变量置换而同时得到置换。
  - 3、①归结反演和②回答提取可以合为一体进行，一次性生成修改证明树。
    - 避免误用¬G参与归结，可以用特定的符号替代¬G。
    - 例如、ANSWER(x1,x2,…,xn)，xi是¬G中的变量。


(8)G
$\vee[Food(x_4) \wedge Eat(Zhang, x_4)]$
x_4 / x_3
(6)
修改证明树
$\neg Food(x_4) \vee \neg Eat(Li, x_4)$
$\vee Answer(x_4)$
x_4 / x_2
(3)
$\neg Eat(Li, x_4) \vee \neg Eat(y, x_4) \vee \neg Alive(y)$
$\vee Answer(x_4)$
Li / y
(5)
$\neg Eat(Li, x_4)$
$\vee Answer(x_4)$
(4)
$Peanuts / x_4$
$Answer(Peanuts)$

归结演绎推理

3）归结反演——提取问题回答

- 从实用的角度，目标公式或许会取更为复杂的形式
  - ①目标公式取反后的化简结果是多个子句的合取
    - $(\exists x)(\exists y)\{[A(x) \wedge B(x)] \vee [C(y) \wedge D(y)]\}$
    - 取反化简后$[\neg A(x) \vee \neg B(x)] \wedge [\neg C(y) \vee \neg D(y)]$
    - 建立2个重言式
    - $\neg A(x) \vee \neg B(x) \vee [A(x) \wedge B(x)]$
    - $\neg C(y) \vee \neg D(y) \vee [C(y) \wedge D(y)]$
  - ②目标公式含有全称量词
    - $(\forall x)(\exists y)(\exists z)\ (\forall w)P(x,y,z,w)$
    - 取反后$(\exists x)(\forall y)(\forall z)\ (\exists w)P(x,y,z,w)$
    - 消去量词$P(A,y,z,g(y,z))$
    - 如何处理函数g(y,z)超出范围，不做介绍。

设置

- **求证G是F1和F2的逻辑结论。**

$$F_1 : (\forall x)(P(x) \rightarrow (\forall y)(Q(y) \rightarrow \neg L(x, y)))$$

$$F_2 : (\exists x)(P(x) \wedge (\forall y)(R(y) \rightarrow L(x, y)))$$

$$G : (\forall x)(R(x) \rightarrow \neg Q(x))$$

正常使用主观题需2.0以上版本雨课堂

作答

归结演绎推理

3）归结反演:课堂练习

- 证明:首先将F_1和 F_2化为子句集

$$F_1 : (\forall x)(P(x) \rightarrow (\forall y)(Q(y) \rightarrow \neg L(x, y)))$$

$$\Rightarrow (\forall x)(\neg P(x) \vee (\forall y)(\neg Q(y) \vee \neg L(x, y)))$$

$$\Rightarrow (\forall x)(\forall y)(\neg P(x) \vee (\neg Q(y) \vee \neg L(x, y)))$$

$$S_1 = \{\neg P(x) \vee (\neg Q(y) \vee \neg L(x, y))\}$$

归结演绎推理

3）归结反演:课堂练习

- 证明:首先将F_1和 F_2化为子句集

$$F_2 : (\exists x)(P(x) \wedge (\forall y)(R(y) \rightarrow L(x, y)))$$

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

$$\Rightarrow (\forall y)(P(\mathrm{A}) \wedge (\neg R(y) \vee L(A, y)))$$

$$S_2 = \{P(\mathrm{A}), \neg R(y) \vee L(\mathrm{A}, y)\}$$

归结演绎推理

3）归结反演:课堂练习

- 证明:G取反化为子句集

$$G:(\forall x)(R(x)\rightarrow\neg Q(x))$$

$$\Rightarrow(\forall x)(\neg R(x)\vee\neg Q(x))$$

$$\Rightarrow(\exists x)(R(x)\wedge Q(x))$$

$$\Rightarrow(R(B)\wedge Q(B))$$

$$S_3=\{R(B),Q(B)\}$$

（1） $\neg P(x) \vee (\neg Q(y1) \vee \neg L(x, y1)$ （2） $P(A)$

（3） $\neg R(y2) \vee L(A, y2)$ （4）R(B)

（5）Q(B)

（6） L(A,B) （3）、（4）归结；

（7）$\neg Q(y1) \vee \neg L(A, y1)$ （1）、（2）归结；

（8） $\neg L(A, B)$ （5）、（7）归结；

（9）Nil。 （6）、（8）归结。

设置

课堂练习

已知： 能阅读的都是有文化的；

海豚是没有文化的；

某些海豚是有智能的；

用归结反演法证明：某些有智能的并不能阅读。

- 注：谓词如下定义

R(x): x能阅读， L(x): x有文化

D(x): x是海豚， I(x): x有智能

正常使用主观题需2.0以上版本雨课堂

作答

归结演绎推理

3）归结反演:课堂练习

证明：

能阅读的都是有文化的；

$$(\forall x)(R(x) \rightarrow L(x))$$

海豚是没有文化的；

$$(\forall y)(D(y) \rightarrow \neg L(y))$$

某些海豚是有智能的；

$$(\exists z)(D(z) \wedge I(z))$$

某些有智能的并不能阅读。

$$(\exists w)(I(w) \wedge \neg R(w))$$

归结演绎推理

3）归结反演:课堂练习

化为子句集为:

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

(1)$\neg R(x)\vee L(x)$

(2)$\neg D(y)\vee\neg L(y)$

(3)$D(A)$

(4)$I(A)$

(5)$\neg I(w)\vee R(w)$

(6)$R(A)$

(7)$L(A)$

(8)$\neg D(A)$

（9）Nil。

（4）、（5）归结；

（1）、（6）归结；

（2）、（7）归结；

（3）、（8）归结。

归结演绎推理

3）归结反演——归结策略

- 归结反演面临大子句集S引起演绎效率问题
- 解决的关键是选择有利于导致快速产生空子句□的子句对进行归结
  - 支持集；
  - 线性输入；
  - 单文字子句优先；
  - 祖先过滤；
- 归结反演技术并未在定理自动证明以外的领域得到广泛应用。

归结演绎推理的归结策略 -广度优先策略

- 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;

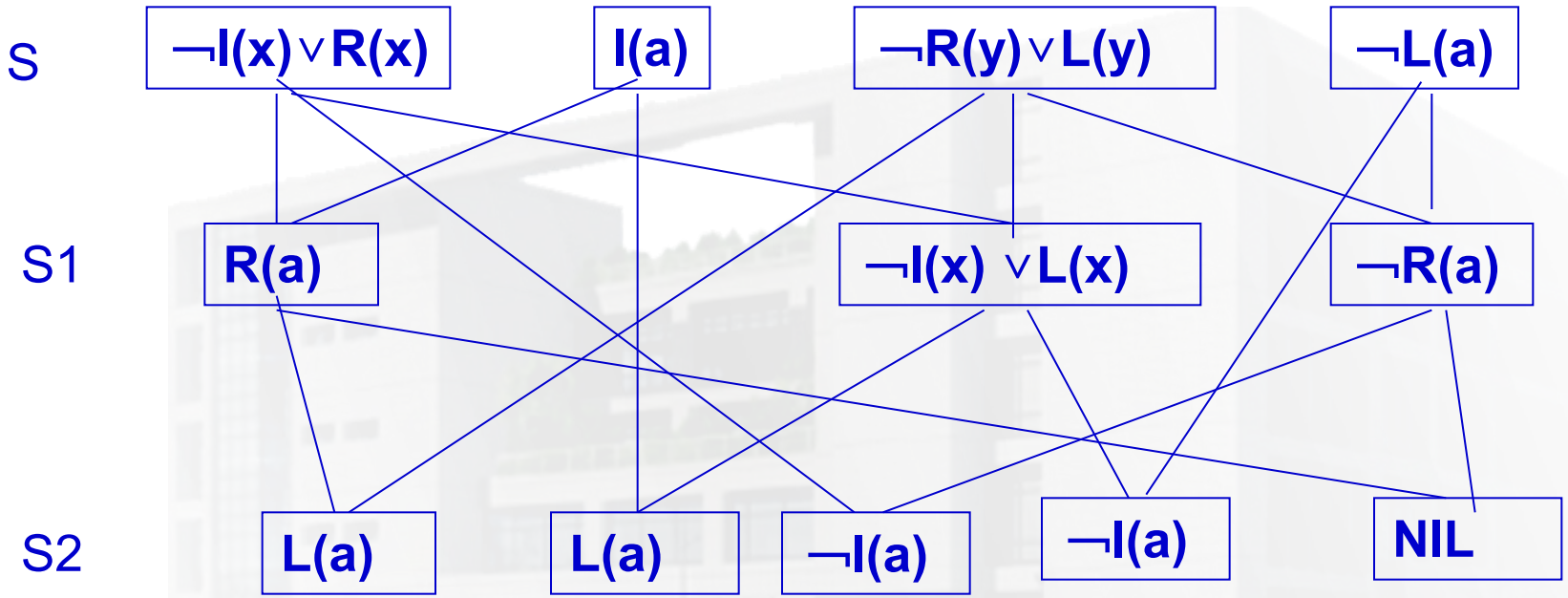
如此继续，直到得出空子句或不能再继续归结为止。

例 设有如下子句集：

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

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：



归结演绎推理的归结策略 -广度优先策略

- 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，这种策略有一个有趣的特性，就是当问题有解时保证能找到最短归结路径。
- 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。

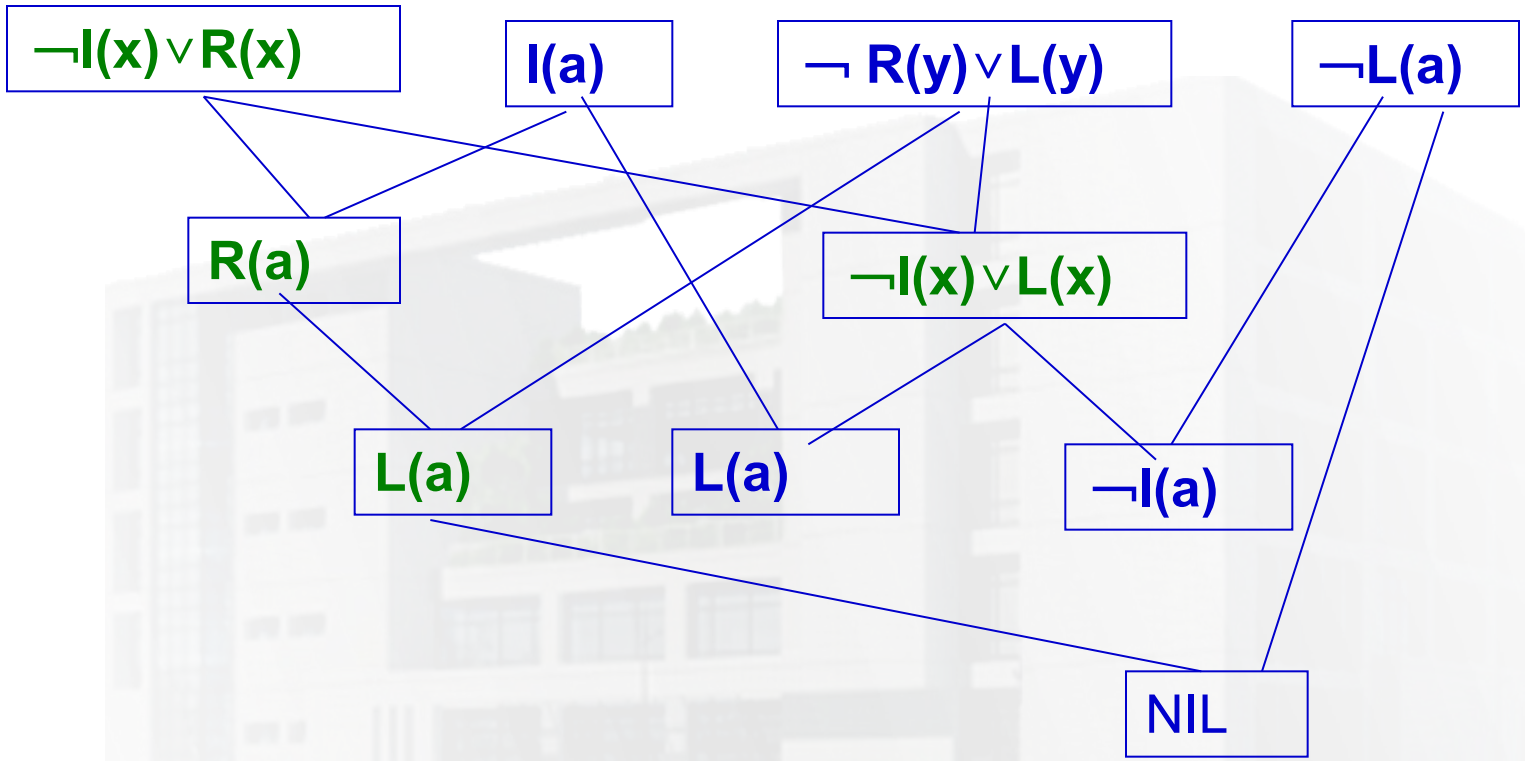
归结演绎推理的归结策略 -支持集策略

- 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

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。



归结演绎推理的归结策略
-支持集策略

- 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。
- 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。

归结演绎推理的归结策略 -删除策略

- 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

归结演绎推理的归结策略 -删除策略

▶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¬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。

归结演绎推理的归结策略 -删除策略

- **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。

归结演绎推理的归结策略 -删除策略

包孕删除法

设有子句C1和C2，如果存在一个置换σ，使得C1σ⊆C2，则称C1包孕于C2。例如

P(x)　　包孕于　P(y)∨Q(z)　　σ={y/x}

P(x)　　包孕于　P(a)　　σ={a/x}

P(x)　　包孕于　P(a)∨Q(z)　　σ={a/x}

P(x) ∨Q(a) 包孕于 P(f(a))∨Q(a)∨R(y) σ={f(a)/x}

P(x) ∨Q(y) 包孕于 P(a)∨Q(u)∨R(w)　σ={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。

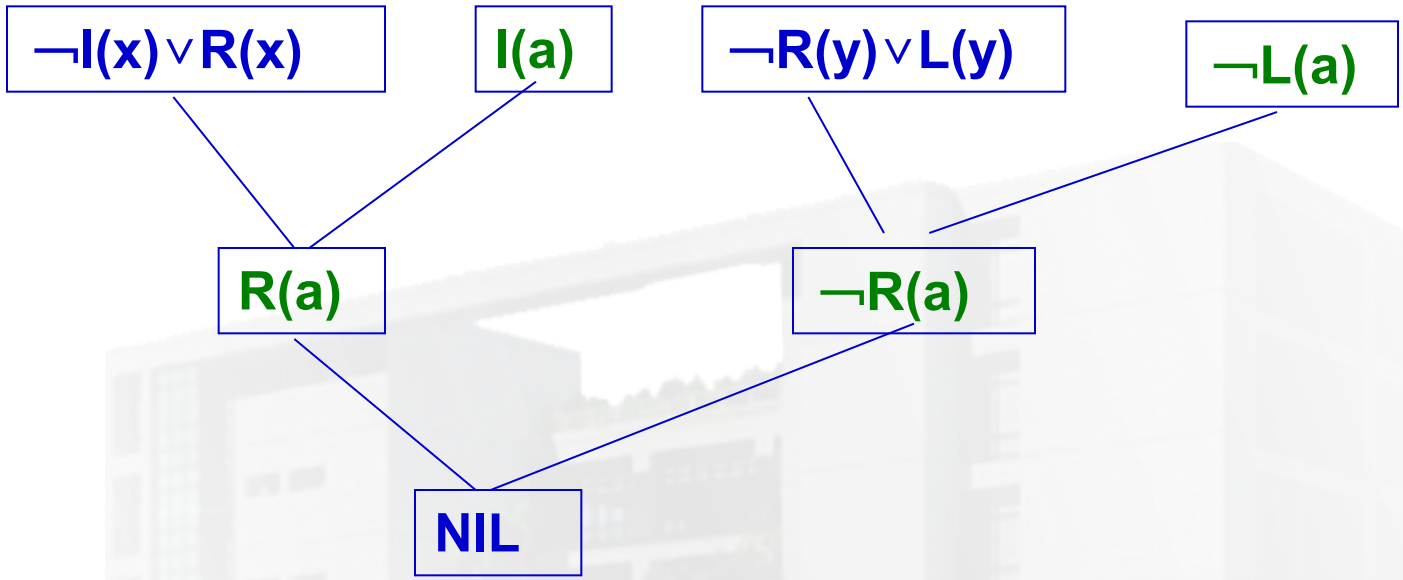
归结演绎推理的归结策略 -单文字子句策略

- 如果一个子句只包含一个文字，则称此子句为单文字子句。单文字子句策略是对支持集策略的进一步改进，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。

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。

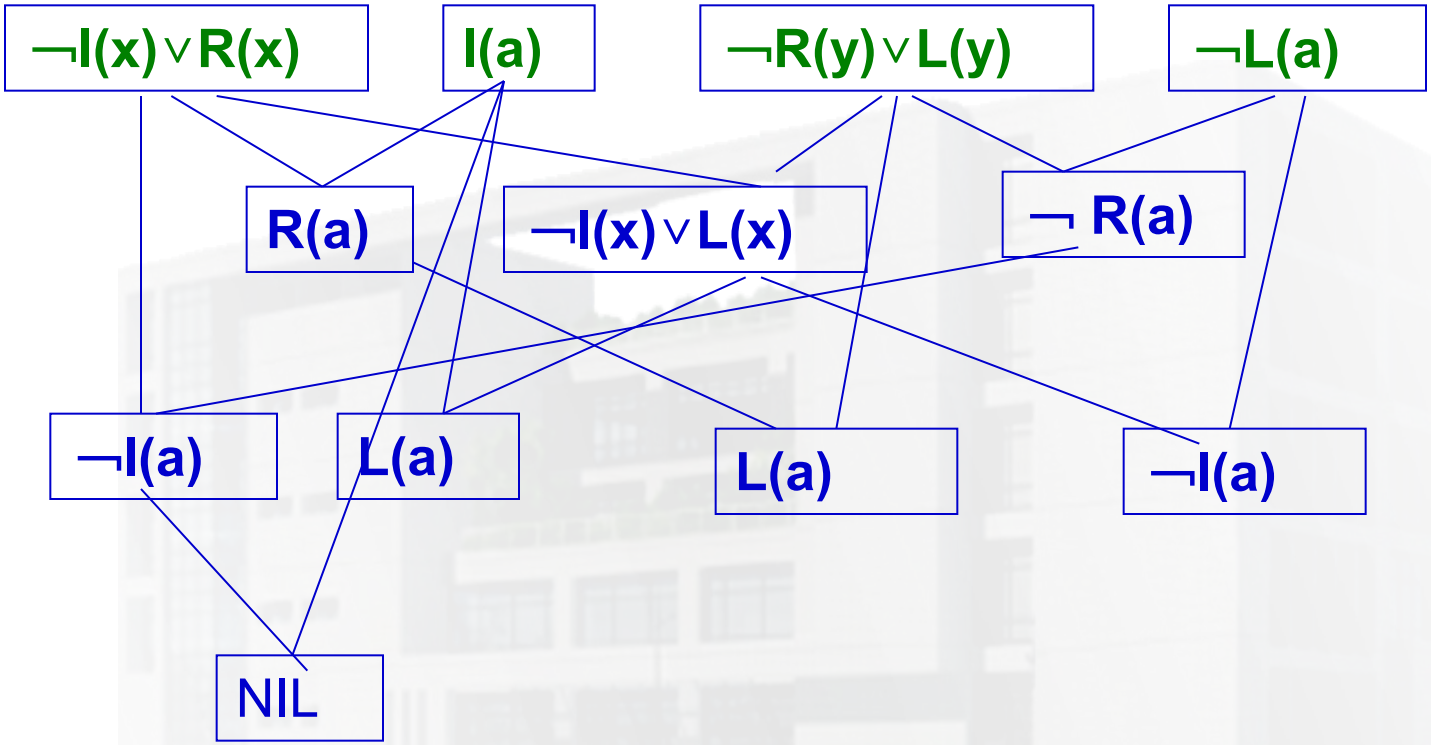
归结演绎推理的归结策略 -线形输入策略

- 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。

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。

归结演绎推理

子句演绎系统

- 归结演绎推理
  - 优点：
  - 简单易行——只要将问题求解的依据（事实或公式）和目标以合适公式形式化描述，标准化为子句集S，就可交由归结反演系统和问答系统执行。
  - 缺点：
  - ①归结演绎并非人类的自然思维方式；
    - 不利于人们从自然思维的角度提供问题求解所需的知识；
    - 难以建立高水平的问题求解系统，如专家系统；
  - ②丢失隐含于合适公式的启发式知识；
    - 合适公式标准化为高度统一的子句集，
    - $P \wedge Q \Rightarrow R$简化为$\neg P \vee \neg Q \vee R$后，丢失了“$P \wedge Q$为真导致R为真”的启发式知识；
    - 效率低下，难以适应于复杂的应用；