人工智能

知识推理方法

主讲: 赵国亮

内蒙古大学电子信息工程学院

April 17, 2020

知识推理方法

推理的概念——推理指按照某种策略从已知事实出发去推出结论的过程.

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

例 1.1

医疗专家系统, 专家知识保存在知识库中. 在推理开始之前, 先把病人的症状和检查结果放到综合数据库中, 然后再从综合数据库的初始证据出发, 按照某种策略在知识库中寻找, 并使用知识, 直到推出最终结论为止.

推理的两个基本问题

推理方法

主要用来表达前提和结论的逻辑关系.

推理的控制策略

确定推理方向, 给出消解冲突策略.

按推理的逻辑基础分类

演绎推理

演绎推理是从已知的知识出发，去推理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，其核心是三段论：假言推理、拒取式和假言三段论.

例 1.2

假言三段论

$$A \to B, B \to C \Rightarrow A \to C.$$

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.

例 1.3

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)

推理的基本思想

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.

归纳推理

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、统计和差异归纳推理等.

完全、不完全和枚举归纳推理

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.

不完全归纳推理

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.

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个具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.

例 1.4

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.

类比归纳推理

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

设 A、B 分别是两类事物的集合:

$$A = a_1, a_2, \cdots$$
$$B = b_1, b_2, \cdots$$

并设 a_i 与 b_i 总是成对出现, 且当 a_i 有属性 P 时, b_i 就有属性 Q 与此对应, 即

$$P(a_i) \to Q(b_i),\ i = 1, 2, \cdots .$$

则当 A 与 B 中有一新的元素对出现时, 若已知 a' 有属性 P, b' 有属性 Q, 即

$$P(a') \to Q(b')$$

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, 或者这两个或两类事物的相同属性与推出新的属性之间的相关程度.

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，演绎推理只不过是将已有事实揭露出来，因此它不能增殖新知识.

- 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.

演绎推理

演绎推理

演绎推理是一种以严格的经典逻辑为基础的推理，而不确定性推理本质上是一种非演绎推理，虽然演绎推理也是一种人类的智能活动.

人工智能

人工智能中的推理主要是指不确定性逻辑推理，也称之为常识推理.

演绎推理是在抽象的逻辑结构上进行的，是一种从普遍性到特殊性的推理，它不承认任何由已知前提推不出的结论，不承认任何不经过演绎推理的假设，不承认任何含有例外事例的结论，演绎推理是一种让人“放心”，而偏于”保守“的推理机制.

常识推理是在实践和经验的基础上进行的，是一种从特殊性到普遍性的推理，具有”容错“的特征，通过不断的发现矛盾与限制矛盾，修正和维护知识，其知识的正确性，既不取决于逻辑的推理规则，也不取决于以知识为前提的推理过程，而取决于推理的结果与事实的符合程度.

演绎推理的真值只能是或“真”或“假”，而常识推理的真值可以是从“真”到“假”之间的一种过渡值.

在常识推理中，往往会用一种确定度来表示结果的真实程度.

正畸思维

往往是这样一种常识推理的心理思维活动在其中起决定性作用，因此，如何将常识推理，或说不确定性推理的思维活动采用定量的形式让计算机处理，将会大大提高数字化人工智能正畸技术的发展.

- 演绎推理使用的是抽象而严格的定义以及在一定理论框架下的绝对正确的定理和公式. 而常识推理使用的是具有局部与暂时合理性的知识和超知识，即人们的共识和个人的经验.

- 演绎推理有一定抽象的理论承诺，它所使用的概念是清晰的，因而是确定的. 而常识推理使用的概念是模糊的，不确定的，具有不肯定性，而知识中的不肯定性表现了人类认识的局限性，作为普遍的知识中可能有一部分具体对象不正确.

- 演绎推理是一种“保真”的推理，也就是前提正确，那么由前提经演绎推出的结论也正确，而常识推理是一种“未必保真”的推理，即使前提正确，其结论也未必正确，是一种近似保真而具有直观合理性的结论.

- 演绎推理是具有相容性的推理，在一个理论体系中不可能同时推出两个相反的命题成立，而常识推理具有矛盾性，是一种非协调推理，在一个知识系统中可能会得到两个相反的结论.

- 演绎推理是完全的，具有单调性，增加新的事例不会影响原有的结论，而常识推理是不完全的非单调性推理，增加新的事例可能会影响原有的结论.

演绎推理例子

例 1.5

一位计算机维修员, 从书本知识, 到通过大量实例积累经验, 是一种归纳推理方式.

例 1.6

一位计算机维修员运用这些一般性知识知识去维修计算机的过程则是演绎推理.

推理的控制策略

- 推理过程不仅依赖于所用的推理方法, 同时也依赖于推理的控制策略.
- 推理的控制策略是指如何使用领域知识使推理过程尽快达到目标的策略.

控制策略的分类

智能系统的推理控制策略又可分为推理策略和搜索策略.

- 推理策略: 主要解决推理方向、推理冲突等问题, 如推理方向控制策略、求解策略、限制策略、冲突消解策略等.

> - 推理方向控制策略用于确定推理的控制方向, 可分为正向推理、逆向推理、混合推理及双向推理.
> - 求解策略是指仅求一个解, 还是求所有解或最优解等.
> - 限制策略是指对推理的深度、宽度、时间、空间等进行的限制.
> - 冲突消解策略是指当推理过程有多条知识可用时, 如何从这多条可用知识中选出一条最佳知识用于推理的策略.

搜索策略主要解决推理线路、推理效果、推理效率等问题.

本章主要讨论推理策略, 至于搜索策略将放到第 4 章单独讨论.
从已知事实出发、正向使用推理规则, 亦称为数据驱动推理或前向链推理.

- sta 把用户提供的初始证据放入综合数据库;
- stb 检查综合数据库中是否包含了问题的解, 若已包含, 则求解结束, 并成功退出; 否则执行下一步;
- stc 检查知识库中是否有可用知识, 若有, 形成当前可用知识集, 执行下一步; 否则转 (5).

1 按照某种冲突消解策略, 从当前可用知识集中选出一条规则进行推理, 并将推出的新事实加入综合数据库中, 然后转 (2).

2 询问用户是否可以进一步补充新的事实, 若可补充, 则将补充的新事实加入综合数据库中, 然后转 (3); 否则表示无解, 失败退出.

如何根据综合数据库中的事实到知识库中选取可用知识, 当知识库中有多条知识可用时应该先使用哪一条知识等. 这些问题涉及到了知识的匹配方法和冲突消解策略, 以后将会分别讨论.

正向推理

流程图如下：

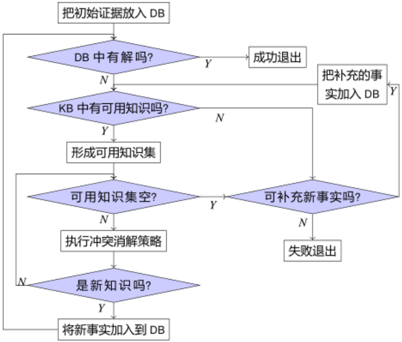


图 1: 正向推理流程图

例 1.7

请用正向推理完成以下问题的求解. 假设知识库中包含有以下 2 条规则:

$$r_1: \text{IF B THEN C}$$
$$r_2: \text{IF A THEN B}$$

已知初始证据 A, 求证目标 C.

本例推理过程

- 推理开始前, 综合数据库为空.
- 推理开始后, 先把 A 放入综合数据库, 然后检查综合数据库中是否含有该问题的解, 回答为“N”.

- 接着检查知识库中是否有可用知识, 显然 r_2 可用, 形成仅含 r_2 的知识集. 从该知识集中取出 r_2, 推出新的实事 B, 将 B 加入综合数据库, 检查综合数据库中是否含有目标 C, 回答为“N”.

- 再检查知识库中是否有可用知识, 此时由于 B 的加入使得 r_1 为可用, 形成仅含 r_1 的知识集.
  - 从该知识集中取出 r_1, 推出新的实事 C, 将 C 加入综合数据库;
  - 检查综合数据库中是否含有目标 C, 回答为 “Y” .

  此时, 综合数据库中已经含有问题的解, 推理成功结束, 目标 C 得证.

正向推理的主要优点

正向推理的主要优点

比较直观, 允许用户主动提供有用的事实信息, 适合于诊断、设计、预测、监控等领域的问题求解.

正向推理的主要缺点

推理无明确目标, 求解问题是可能会执行许多与解无关的操作, 导致推理效率较低.

逆向推理算法

从某个假设目标出发，逆向使用规则，亦称为目标驱动推理或逆向链推理.

1. 将要求证的目标 (假设) 构成一个假设集;
2. 从假设集中选出一个假设，检查该假设是否在综合数据库中，若在，则该假设成立，此时，若假设集为空，则成功退出，否则仍执行 (2); 若该假设不在数据库中，则执行下一步;

检查该假设是否可由知识库的某个知识导出，若不能由某个知识导出，则询问用户该假设是否为可由用户证实的原始事实，若是，该假设成立，并将其放入综合数据库，再重新寻找新的假设，若不是，则转 (5); 若能由某个知识导出，则执行下一步;

1 将知识库中可以导出该假设的所有知识构成一个可用知识集;

1 检查可用知识集是否为空, 若是, 失败退出; 否则执行下一步;

1 按冲突消解策略从可用知识集中取出一个知识, 继续;

1 将该知识的前提中的每个子条件都作为新的假设放入假设集, 然后转 (2).

其流程图如下:

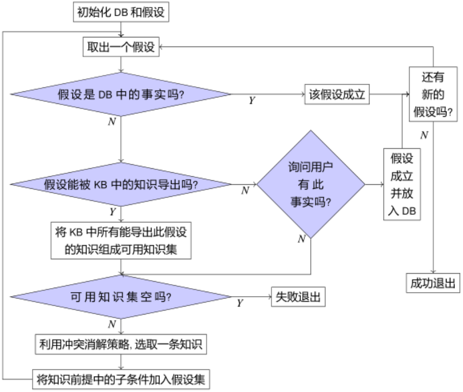


图 2: 反向推理流程图

逆向推理过程

- 推理开始前, 综合数据库和假设集均为空.

- 推理开始后, 先将初始证据 A 和目标 C 分别放入综合数据库和假设集, 然后从假设集中取出一个假设 C, 查找 C 是否为综合数据库中的已知事实, 回答为"N".

- 再检查 C 是否能被知识库中的知识所导出, 发现 C 可由 r_1 导出, 于是 r_1 被放入可用知识集. 由于知识库中只有 r_1 可用, 故可用知识集中仅含 r_1.

- 接着从可用知识集中取出 r_1, 将其前提条件 B 作为新的假设放入假设集. 从假设集中取出 B, 检查 B 是否为综合数据库中的实事, 回答为 “N”. 再检查 B 是否能被知识库中的知识所导出, 发现 B 可由 r_2 导出, 于是 r_2 被放入可用知识集. 由于知识库中只有 r_2 可用, 故可用知识集中仅含 r_2.

- 从可用知识集中取出 r_2, 将其前提条件 A 作为新的假设放入假设集. 然后从假设集中取出 A, 检查 A 是否为综合数据库中的实事, 回答为 “Y”.

该假设成立, 由于无新的假设, 故推理过程成功结束, 于是目标 C 得证.

逆向推理的优缺点

- 逆向推理的主要优点:
  - 不必寻找和使用那些与假设目标无关的信息和知识
  - 推理过程的目标明确
  - 也有利于向用户提供解释, 在诊断性专家系统中较为有效.

- 逆向推理的主要缺点:
  - 当用户对推理过程认识不清楚时, 由系统自主选择假设目标，盲目性比较大, 若选择不好, 可能需要多次提出假设, 会影响系统效率.

混合推理

混合推理概念

把正向推理和逆向推理结合起来所进行的推理称为混合推理. 是一种解决较复杂问题的方法.

- 混合推理的方法
  - 先正向后逆向: 这种方法先进行正向推理, 从已知事实出发推出部分结果, 然后再用逆向推理对这些结果进行证实或提高它们的可信度.
  - 先逆向后正向: 这种方法先进行逆向推理, 从假设目标出发推出一些中间假设, 然后再用正向推理对这些中间假设进行证实.
  - 双向混合: 是指正向推理和逆向推理同时进行, 使推理过程在中间的某一步结合起来. 对于这些方法不再详细讨论.

逻辑学与人工智能

人工智能的产生与发展与逻辑学密不可分. 逻辑学为人工智能的研究提供了根本观点与方法，而逻辑方法则是人工智能研究中的主要工具.

从逻辑学为人工智能的研究提供理论基础出发，本节讨论经典逻辑和非经典逻辑在人工智能中的应用，以及人工智能在逻辑学发展方向上的影响.

人工智能主要研究用计算机方法模拟和扩展人的智能，最终实现机器智能. 人工智能研究与人的思维研究密切相关. 逻辑学始终是人工智能研究中的基础科学问题，它为人工智能研究提供了根本方法.

人工智能学科的诞生

12 世纪末 13 世纪初，西班牙罗门 · 卢乐提出制造可解决各种问题的通用逻辑机.

17 世纪，英国培根在《新工具》中提出了归纳法. 随后，德国莱布尼兹做出了四则运算的手摇计算器，并提出了“通用符号”和“推理计算”的思想.

19 世纪，英国布尔创立了布尔代数，奠定了现代形式逻辑研究的基础. 德国弗雷格完善了命题逻辑，创建了一阶谓词演算系统.

20 世纪，哥德尔对一阶谓词完全性定理与 N 形式系统的不完全性定理进行了证明. 在此基础上，克林对一般递归函数理论作了深入的研究，建立了演算理论. 英国图灵建立了描述算法的机械性思维过程，提出了理想计算机模型(即图灵机)，创立了自动机理论.

1945 年匈牙利冯 · 诺依曼提出存储程序的思想和建立通用电子数字计算机的冯 · 诺依曼型体系结构，

1946 年美国的莫克利和埃克特成功研制世界上第一台通用电子数学计算机 ENIAC 做出了开拓性的贡献.

现代逻辑发展动力主要来自于数学中的公理化运动. 20 世纪逻辑研究严重数学化，发展出来的逻辑被恰当地称为“数理逻辑”，它增强了逻辑研究的深度，使逻辑学的发展继古希腊逻辑、欧洲中世纪逻辑之后进入第三个高峰期，并且对整个现代科学产生了非常重要的影响.

逻辑学的发展

- 逻辑学的分类
  - 逻辑学大体上可分为经典逻辑、非经典逻辑和现代逻辑. 经典逻辑与模态逻辑都是二值逻辑. 多值逻辑，是具有多个命题真值的逻辑，是向模糊逻辑的逼近. 模糊逻辑是处理具有模糊性命题的逻辑. 概率逻辑是研究基于逻辑的概率推理.

- 泛逻辑的基本原理
  - 泛逻辑是从高层研究一切逻辑的一般规律，建立能包容一切逻辑形态和推理模式，并能根据需要自由伸缩变化的柔性逻辑学，刚性逻辑学将作为一个最小的内核存在其中，这就是提出泛逻辑的根本原因，也是泛逻辑的最终历史使命.

逻辑学在人工智能学科的研究方面的应用

逻辑方法是人工智能研究中的主要工具，逻辑学的研究成果不但为人工智能学科的诞生奠定了理论基础，而且它们还作为重要的成分被应用于人工智能系统中.

经典逻辑的应用

- 人工智能诞生后的 20 年间是逻辑推理占统治地位的时期. 1963 年，纽厄尔、西蒙等人编制的“逻辑理论机”数学定理证明程序 (LT). 在此基础之上，纽厄尔和西蒙编制了通用问题求解程序 (GPS)，开拓了人工智能“问题求解”的一大领域. 经典数理逻辑只是数学化的形式逻辑，只能满足人工智能的部分需要.

经典逻辑的应用

人工智能诞生后的 20 年间是逻辑推理占统治地位的时期.

1963 年，纽厄尔、西蒙等人编制的“逻辑理论机”数学定理证明程序 (LT).

纽厄尔和西蒙编制了通用问题求解程序 (GPS)，开拓了人工智能“问题求解”的一大领域.

经典数理逻辑只是数学化的形式逻辑，只能满足人工智能的部分需要.

非经典逻辑的应用

不确定性的推理研究

人工智能发展了用数值的方法表示和处理不确定的信息，即给系统中每个语句或公式赋一个数值，用来表示语句的不确定性或确定性.

比较具有代表性的有：1976 年杜达提出的主观贝叶斯模型，1978 年查德提出的可能性模型，1984 年邦迪提出的发生率计算模型，以及假设推理、定性推理和证据空间理论等经验性模型.

不完全信息的推理研究

常识推理是一种非单调逻辑，即人们基于不完全的信息推出某些结论，当人们得到更完全的信息后，可以改变甚至收回原来的结论. 非单调逻辑可处理信息不充分情况下的推理.

多值逻辑和模糊逻辑也已经被引入到人工智能中来处理模糊性和不完全性信息的推理. 多值逻辑的三个典型系统是克林、卢卡西维兹和波克万的三值逻辑系统. 模糊逻辑的研究始于 20 世纪 20 年代卢卡西维兹的研究. 1972 年，扎德提出了模糊推理的关系合成原则，现有的大多数模糊推理方法都是关系合成规则的变形或扩充.

人工智能 — 当代逻辑发展的动力

现代逻辑

创始于 19 世纪末叶和 20 世纪早期，其发展动力主要来自于数学中的公理化运动.

21 世纪逻辑发展的主要动力来自哪里？

计算机科学和人工智能将至少是 21 世纪早期逻辑学发展的主要动力源泉，并将由此决定 21 世纪逻辑学的另一幅面貌.

由于人工智能要模拟人的智能，它的难点不在于人脑所进行的各种必然性推理，而是最能体现人的智能特征的能动性、创造性思维，这种思维活动中包括学习、抉择、尝试、修正、推理诸因素.

例 2.1

选择性地搜集相关的经验证据，在不充分信息的基础上做出尝试性的判断或抉择，不断根据环境反馈调整、修正自己的行为，由此达到实践的成功.

逻辑学将不得不比较全面地研究人的思维活动，并着重研究人的思维中最能体现其能动性特征的各种不确定性推理，由此发展出的逻辑理论也将具有更强的可应用性.

人工智能的产生与发展和逻辑学的发展密不可分

一方面我们试图找到一个包容一切逻辑的泛逻辑，使得形成一个完美统一的逻辑基础.

另一方面，我们还要不断地争论、更新、补充新的逻辑. 如果二者能够有机地结合，将推动人工智能进入一个新的阶段.

概率逻辑大都是基于二值逻辑的，目前许多专家和学者又在基于其他逻辑的基础上研究概率推理，使得逻辑学尽可能满足人工智能发展的各方面的需要.

就目前来说，一个新的泛逻辑理论的发展和完善需要一个比较长的时期，那何不将“百花齐放”与“一统天下”并行进行，各自发挥其优点，为人工智能的发展做出贡献.

许多制约人工智能发展的问题仍有待于解决，技术上的突破，还有赖于逻辑学研究上的突破.

在对人工智能的研究中，我们只有重视逻辑学，努力学习与运用并不断深入挖掘其基本内容，拓宽其研究领域，才能更好地促进人工智能学科的发展 (机器人网 2017-11-08).

推理的逻辑基础

- 命题公式的解释
  - 在命题逻辑中，命题公式的一个解释就是对该命题公式中各个命题变元的一次真值指派．有了命题公式的解释，就可据此求出该命题公式的真值．

- 谓词公式的解释
  - 由于谓词公式中可能包含有个体常量、变元或函数，因此，必须先考虑这些个体常量和函数在个体域上的取值，然后才能根据它们的具体取值为谓词分别指派真值．

解释

1 设 D 是谓词公式 P 的非空个体域, 若对 P 中的个体常量、函数和谓词规定赋值:
(1) 为每个个体常量指派 D 中的一个元素;

(2) 为每个 n 元函数指派一个从 D_n 到 D 的一个映射, 其中

$$D^n = \{(x_1, x_2, \cdots, x_n) | x_1, x_2, ..., x_n \in D\} \tag{1}$$

(3) 为每个 n 元谓词指派一个从 D_n 到 $\{F, T\}$ 的映射. 则称这些指派为 P 在 D 上的一个解释 I.

例 2.2

设个体域 $D = \{1, 2\}$, 求公式 $A = (\forall x)(\forall y)P(x, y)$ 在 D 上的解释, 并指出在每一种解释下公式 A 的真值.

解: 由于公式 A 中没有包含个体常量和函数, 故可直接为谓词指派真值, 设有

表 1: 谓词的真值指派

$P(1,1)$	$P(1,2)$	$P(2,1)$	$P(2,2)$
T	F	T	F

这就是公式 A 在 D 上的一个解释.

从这个解释可以看出:

- 当 $x=1, y=1$ 时, 有 $P(x,y)$ 的真值为 T;
- 当 $x=2, y=1$ 时, 有 $P(x,y)$ 的真值为 T;

即对 x 在 D 上的任意取值, 都存在 $y=1$ 使 $P(x,y)$ 的真值为 T. 因此, 在此解释下公式 A 的真值为 T.

注

一个谓词公式在其个体域上的解释不是唯一的. 例如, 对公式 A, 若给出另一组真值指派.

表 2: 谓词的真值指派

$P(1,1)$	$P(1,2)$	$P(2,1)$	$P(2,2)$
T	T	F	F

这也是公式 A 在 D 上的一个解释. 从这个解释可以看出:

- 当 $x = 1, y = 1$ 时, 有 $P(x, y)$ 的真值为 T;
- 当 $x = 2, y = 1$ 时, 有 $P(x, y)$ 的真值为 F;

即对 x 在 D 上的任意取值, 不存在一个 y 使得 $P(x, y)$ 的真值为 T. 因此, 在此解释下公式 A 的真值为 F. 实际上, A 在 D 上共有 16 种解释, 这里就不再一一列举.

例 2.3

设个体域 $D = \{1, 2\}$, 求公式 $B = (\forall x)\, P(f(x), a)$ 在 D 上的解释, 并指出在该解释下公式 B 的真值.

解: 设对个体常量 a 和函数 f(x) 的值指派为:

表 3: 真值指派方案

a	f(1)	F(2)
1	1	2

对谓词的真值指派

表 4: 谓词的真值指派

$\mathsf{P}(1,1)$	$\mathsf{P}(1,2)$	$\mathsf{P}(2,1)$	$\mathsf{P}(2,2)$
T	$\times$	T	$\times$

由于已指派 $\mathsf{a}=1$, 因此 $\mathsf{P}(1,2)$ 和 $\mathsf{P}(2,2)$ 不可能出现, 故没给它们指派真值.

上述指派是公式 B 在 D 上的一个解释. 在此解释下有

- 当 $x=1$ 时, $a=1$ 使 $P(1,1)=T$
- 当 $x=2$ 时, $a=1$ 使 $P(2,1)=T$.

即 x 在 D 上的任意取值, 都存在 $a=1$ 使 $P(f(x),a)$ 的真值为 T. 因此, 在此解释下公式 B 的真值为 T.

由上面的例子可以看出, 谓词公式的真值都是针对某一个解释而言的, 它可能在某一个解释下真值为 T, 而在另一个解释下为 F.

谓词公式的永真性和可满足性

先后续的推理需要一些相关定义:

永真定义

1 如果谓词公式 P 对非空个体域 D 上的任一解释 I 都取得真值 T, 则称 P 在 D 上是永真的; 如果 P 在任何非空个体域上均是永真的, 则称 P 永真.

要判定谓词公式为永真, 必须对每个非空个体域上的每个解释逐一进行判断. 当解释的个数为有限时, 尽管工作量大, 公式的永真性毕竟还可以判定, 但当解释个数为无限时, 其永真性就很难判定.

永假的定义

如果谓词公式 P 对非空个体域 D 上的任一解释都取真值 F, 则称 P 在 D 上是永假的; 如果 P 在任何非空个体域上均是永假的, 则称 P 永假.

永假的判定

如果谓词公式 P 对非空个体域 D 上的任一解释都取真值 F, 则称 P 在 D 上是永假的; 如果 P 在任何非空个体域上均是永假的, 则称 P 永假.

注

谓词公式的永假性又称不可满足性. 后面给出的归结推理, 就是采用一种逻辑上的反证法, 将永真性转换为不可满足性的证明.

逻辑符号及其含义

表 5: 逻辑符号及其含义

$\neg$	$\wedge$	$\vee$	$\forall$	$\exists$	$\in$	$\subset$
否定	合取	析取	全称量词	存在量词	属于	包含于
$\supset$	$\cup$	$\cap$	$\rightarrow$	$\leftarrow$	$\leftrightarrow$	$\Rightarrow$
包含	并集	交集	蕴涵	逆蕴涵	等值	严格蕴含

谓词公式的等价

- 谓词公式的等价性和永真蕴含性可分别用相应的等价式和永真蕴含式来表示, 这些等价式和永真蕴含式都是演绎推理的主要依据, 因此也称它们为推理规则.

谓词公式的等价式

设 P 与 Q 是 D 上的两个谓词公式, 若对 D 上的任意解释, P 与 Q 都有相同的真值, 则称 P 与 Q 在 D 上是等价的. 如果 D 是任意非空个体域, 则称P 与 Q 是等价的, 记作 $P \Leftrightarrow Q$.

1. 双重否定律 $\neg\neg P \Leftrightarrow P$.
2. 交换律 $(P \vee Q) \Leftrightarrow (Q \vee P), (P \wedge Q) \Leftrightarrow (Q \wedge P)$.
3. 结合律 $(P \vee Q) \vee R \Leftrightarrow P \vee (Q \vee R), (P \wedge Q) \wedge R \Leftrightarrow P \wedge (Q \wedge R)$.
4. 分配律 $P \vee (Q \wedge R) \Leftrightarrow (P \vee Q) \wedge (P \vee R), P \wedge (Q \vee R) \Leftrightarrow (P \wedge Q) \vee (P \wedge R)$.
5. 摩根定律 $\neg(P \vee Q) \Leftrightarrow \neg P \wedge \neg Q, \neg(P \wedge Q) \Leftrightarrow \neg P \vee \neg Q$.
6. 吸收律 $P \vee (P \wedge Q) \Leftrightarrow P \wedge Q, P \wedge (P \vee Q) \Leftrightarrow P$.
7. 补余律 $P \vee \neg P \Leftrightarrow T, P \wedge \neg P \Leftrightarrow F$.
8. 连词化归律
   $P \rightarrow Q \Leftrightarrow \neg P \vee Q, P \longleftrightarrow Q \Leftrightarrow (P \rightarrow Q) \wedge (Q \rightarrow P), P \longleftrightarrow Q \Leftrightarrow (P \wedge Q) \vee (Q \wedge P)$.
9. 量词转换律 $\neg(\exists x)P \leftrightarrow (\forall x)(\neg P), \neg(\forall x)P \leftrightarrow (\exists x)(\neg P)$.
10. 量词分配律 $(\forall x)(P \wedge Q) \Leftrightarrow (\forall x)P \wedge (\forall x)Q, (\exists x)(P \vee Q) \Leftrightarrow (\exists x)P \vee (\exists x)Q$.

逻辑与格论

连词化归率

$P \rightarrow Q \Leftrightarrow \neg P \vee Q, P \longleftrightarrow Q \Leftrightarrow (P \rightarrow Q) \wedge (Q \rightarrow P), P \longleftrightarrow Q \Leftrightarrow (P \wedge Q) \vee (Q \wedge P)$

格论中的两个元素也可记为 0,1, 看成数字后，是良序集上的完全序; 定义关系"$\leq$", 即$0 \leq 1$ 且 0 和 1 能比较大小.

例 2.4

运用连词化归率说明结论: $P \rightarrow Q \Leftrightarrow \neg P \vee Q$.

永真蕴含

对谓词公式 P 和 Q, 如果 $P \to Q$ 永真, 则称 P 永真蕴含 Q, 且称 Q 为 P 的逻辑结论, P 为 Q 的前提, 记作 $P \Rightarrow Q$.

常用的永真蕴含式如下:

1. 化简式 $P \wedge Q \Rightarrow P, P \wedge Q \Rightarrow Q$
2. 附加式 $P \Rightarrow P \vee Q, Q \Rightarrow P \vee Q$
3. 析取三段论 $\neg P, P \vee Q \Rightarrow Q$
4. 假言推理 $P, P \to Q \Rightarrow Q$
5. 拒取式 $\neg Q, P \to Q \Rightarrow \neg P$
6. 假言三段论 $P \to Q, Q \to R \Rightarrow P \to R$
7. 二难推理 $P \vee Q, P \to R, Q \to R \Rightarrow R$
8. 全称固化 $(\forall x)P(x) \Rightarrow P(y)$, 其中, y 是个体域中的任意个体, 依此可消去谓词公式中的全称量词.
9. 存在固化 $(\exists x)P(x) \Rightarrow P(y)$, 其中, y 是个体域中某一个可以使 P(y) 为真的个体, 依此可消去谓词公式中的存在量词.

谓词公式的范式

范式

范式是谓词公式的标准形式.

谓词逻辑中的范式

在谓词逻辑中, 范式分为两种: 前束范式和 Skolem 范式.

前束范式

前束范式

设 F 为一谓词公式, 如果其中的所有量词均非否定地出现在公式的最前面, 且它们的辖域为整个公式, 则称 F 为前束范式.

一般形式

$$(Q_1x_1)\cdots(Q_nx_n)\,M(x_1,x_2,\cdots,x_n)$$

其中, $Q_i\ (i=1,2,\cdots,n)$ 为前缀, 它是一个由全称量词或存在量词组成的量词串; $M(x_1,x_2,\cdots,x_n)$ 为母式, 它是一个不含任何量词的谓词公式.

例 2.5

$(\forall x)(\forall y)(\exists z)(P(x) \wedge Q(y,z) \vee R(x,z))$ 是前束范式.

任一谓词公式均可化为与其对应的前束范式

Skolem 范式

如果前束范式中所有的存在量词都在全称量词之前, 则称这种形式的谓词公式为 Skolem 范式.

例 2.6

$(\exists x)(\exists z)(\forall y)(P(x) \vee Q(y,z) \wedge R(x,z))$ 是 Skolem 范式.

任一谓词公式均可化为与其对应的 Skolem 范式.

其化简方法也将在后面子句集的化简中讨论.

置换与合一

在不同谓词公式中，往往会出现谓词名相同但其个体不同的情况，此时推理过程是不能直接进行匹配的，需要先进行置换.

例 2.7

可根据全称固化推理和假言推理由谓词公式

$$W_1(A)\text{和}(\forall x)(W_1(x) \to W_2(x))$$

推出 $W_2(A)$.

对谓词 $W_1(A)$ 可看作是由全程固化推理 (即 $(\forall x)(W_1(x) \Rightarrow W_1(A))$ 推出的, 其中 A 是任一个体常量.

要使用假言推理, 首先需要找到项 A 对变元 x 的置换, 使 $W_1(A)$ 与 $W_1(x)$ 一致.

合一过程

寻找项对变元的置换, 使谓词一致的过程叫做合一的过程.

置换与合一的有关概念与方法

置换可简单的理解为是在一个谓词公式中用置换项去替换变量.

置换

置换是形如

$$\{t_1/x_1, t_2/x_2, \cdots, t_n/x_n\}$$

的有限集合. 其中, $t_1, t_2, \cdots, t_n$ 是项; $x_1, x_2, \cdots, x_n$ 是互不相同的变元; t_i/x_i 表示用 t_i 替换 x_i. 并且要求 t_i 与 x_i 不能相同, x_i 不能循环地出现在另一个 t_i 中.

例 2.8

$\{a/x, c/y, f(b)/z\}$ 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. 即当用 g(y) 置换 x, n 用 f(g(y)) 置换 y 时, 既没有消去 x, 也没有消去 y.

若改为 $\{g(a)/x, f(x)/y\}$ 即可, 原因是用 g(a) 置换 x, 用 f(g(a)) 置换 y, 则消去了 x 和 y.

注

置换

设

$$\theta = \{t_1/x_1, t_2/x_2, \cdots, t_n/x_n\} \quad (2)$$
$$\lambda = \{u_1/y_1, u_2/y_2, \cdots, u_m/y_m\} \quad (3)$$

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

$$\{t_1\lambda/x_1, t_2\lambda/x_2, \cdots, t_n\lambda/x_n, u_1/y_1, u_2/y_2, \cdots, u_m/y_m\} \quad (4)$$

中删去以下两种元素
① 当 $t_i\lambda = x_i$ 时, 删去 $t_i\lambda/x_i$ $(i = 1, 2, \cdots, n)$;
② 当 $y_j \in \{x_1, x_2, \cdots, x_n\}$时, 删去 u_j/y_j $(j = 1, 2, \cdots, m)$
最后剩下的元素所构成的集合.

例 2.9

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

$$\{f(y)\lambda/x, (y\lambda)/y, a/x, b/y, y/z\} = \{f(b)/x, y/y, a/x, b/y, y/z\} \tag{5}$$

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

$$\theta \circ \lambda = \{f(b)/x, y/z\}. \tag{6}$$

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.

合一

设有公式集 $\mathsf{F} = \{\mathsf{F}_1, \mathsf{F}_2, \cdots, \mathsf{F}_n\}$, 若存在一个置换 θ, 可使

$$\mathsf{F}_1\theta = \mathsf{F}_2\theta = \cdots = \mathsf{F}_n\theta, \tag{7}$$

则称 θ 是 F 的一个合一. 称 $\mathsf{F}_1, \mathsf{F}_2, \cdots, \mathsf{F}_n$ 是可合一的.

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.

例 2.10

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

$$\lambda=\{a/x,g(a)/y,f(g(a))/z\} \tag{8}$$

是它的一个合一.

一般来说, 一个公式集的合一不是唯一的.

最一般合一

设 σ 是公式集 F 的一个合一, 如果对 F 的任一合一 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.

一个公式集的最一般合一是唯一的.

对如何求取最一般合一的问题, 不再讨论.

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.

自然演绎推理规则的三段论推理

- 假言推理 $P, P \rightarrow Q \Rightarrow Q$
- 拒取式 $\neg Q, P \rightarrow Q \Rightarrow \neg P$
- 假言三段论 $P \rightarrow Q, Q \rightarrow R \Rightarrow P \rightarrow R$

自然演绎推理的例子

例 3.1

设已知如下事实:

$$A, B, A \to C, B \wedge C \to D, D \to Q$$

求证: Q 为真.

证明: 因为

- $A, A \rightarrow C \Rightarrow C$ 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
- $D, D \rightarrow Q \Rightarrow Q$ 假言推理

因此, Q 为真.

例 3.2

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

Prog(x)　x 是需要编程序的课.

Like(x, y)　x 喜欢 y.

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:

$$Prog(x) \rightarrow Like(Wang, x)$$
$$(\forall x)(Lang(x) \rightarrow Prog(x))$$
$$Lang(C).$$

应用推理规则进行推理

$Lang(y) \rightarrow Prog(y)$ 全称固化

$Lang(C), Lang(y) \rightarrow Prog(y) \Rightarrow Prog(C)$ 假言推理 $\{C/y\}$

$Prog(C), Prog(x) \rightarrow Like(Wang, x) \Rightarrow Like(Wang, C)$ 假言推理 $\{C/x\}$.

因此, 王程喜欢 C 这门课.

推理的优点和缺点

推理的优点

定理证明过程自然, 易于理解, 并且有丰富的推理规则可用.

推理的缺点

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.

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”.

在人工智能使用的命题和陈述中, 几乎所有的问题都可以转化为一个定理证明问题. 定理证明的实质, 就是要对前提 P 和结论 Q, 证明 $P \rightarrow Q$ 永真.

由 1.3 节可知, 要证明 $P \to Q$ 永真, 就是要证明 $P \to Q$ 在任何一个非空的个体域上都是永真的. 这将是非常困难的, 甚至是不可实现的.

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.

子句集及其化简

即要证明 $P \to Q$ 永真, 只要能够证明 $P \wedge \neg Q$ 是不可满足的就可以了 (原因是 $\neg(P \to Q) \Leftrightarrow \neg(\neg P \vee Q) \Leftrightarrow P \wedge \neg Q$. 这方面最有成效的工作就是鲁宾逊归结原理. 它使定理证明的机械化成为现实.

鲁滨逊归结原理是在子句集的基础上讨论问题的. 因此, 讨论归结演绎推理之前, 需要先讨论子句集的有关概念.

文字

文字

原子谓词公式及其否定统称为文字.

例 3.3

$P(x)$、$Q(x)$、$\neg P(x)$、$\neg Q(x)$ 等都是文字.

子句

任何文字的析取式称为子句.

子句集

例 3.4

$P(x) \vee Q(x)$, $P(x, f(x)) \vee Q(x, g(x))$ 都是子句.

空子句和子句集

不含任何文字的子句称为空子句. 由子句或空子句所构成的集合称为子句集.

由于空子句不含有任何文字, 也就不能被任何解释所满足, 因此空子句是永假的, 不可满足的. 空子句一般被记为 NIL.

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. 化简步骤如下:

(1) 消去连接词 “$\rightarrow$” 和 “$\longleftrightarrow$”

反复使用如下等价公式:

$$P \rightarrow Q \Leftrightarrow \neg P \vee Q \tag{9}$$

$$P \longleftrightarrow Q \Leftrightarrow (P \wedge Q) \vee (\neg P \wedge \neg Q) \tag{10}$$

即可消去谓词公式中的连接词 “$\rightarrow$” 和 “$\longleftrightarrow$” .

归结举例

例 3.5

公式

$$(\forall x)((\forall y)P(x,y) \to \neg(\forall y)(Q(x,y) \to R(x,y))), \tag{11}$$

经等价变化后为

$$(\forall x)(\neg(\forall y)P(x,y) \vee \neg(\forall y)(\neg Q(x,y) \vee R(x,y)))。 \tag{12}$$

(3) 减少否定符号的辖域

反复使用双重否定率

$$\neg(\neg P) \Leftrightarrow P。 \qquad (13)$$

摩根定律

$$\neg(P \wedge Q) \Leftrightarrow \neg P \vee \neg Q \qquad (14)$$

$$\neg(P \vee Q) \Leftrightarrow \neg P \wedge \neg Q \qquad (15)$$

量词转换律

$$\neg(\forall x)P(x) \Leftrightarrow (\exists x)\neg P(x) \tag{16}$$

$$\neg(\exists x)P(x) \Leftrightarrow (\forall x)\neg P(x) \tag{17}$$

将每个否定符号“¬”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.

(4) 变元标准化

(3) 对变元标准化

在一个量词的辖域内, 把谓词公式中受该量词约束的变元全部用另外一个没有出现过的任意变元代替, 使不同量词约束的变元有不同的名字.

例 3.6

上式经变换后为

$$(\forall x)((\exists y)\neg P(x,y) \vee (\exists z)(Q(x,z) \wedge \neg R(x,z))) \tag{18}$$

(5) 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.

(5) 消去存在量词

消去存在量词时

- 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 3.7

$$(\forall x_1)\cdots(\forall x_n)(\exists y)\,P(x_1, x_2, \cdots, x_n, y), \tag{19}$$

则需要用 Skolem 函数 $f(x_1, x_2, \cdots, x_n)$ 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 3.8

上步公式中存在量词 $(\exists y)$ 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 $(\forall x)(\neg P(x, f(x)) \vee (Q(x, g(x)) \wedge \neg R(x, g(x))))$.

(6) 化为 Skolem 标准形

Skolem 标准形的一般形式为

$$(\forall x_1)\cdots(\forall x_n)M(x_1,x_2,\cdots,x_n), \tag{20}$$

其中, $M(x_1,x_2,\cdots,x_n)$ 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

$$P\vee(Q\wedge R)\Leftrightarrow(P\vee Q)\wedge(P\vee R). \tag{21}$$

例如, 前面的公式化为 Skolem 标准形后为

$$(\forall x)((\neg P(x,f(x))\vee Q(x,g(x)))\wedge(\neg P(x,f(x))\vee\neg R(x,g(x)))) \tag{22}$$

(7) 消去全称量词

由于母式中的全部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 3.9

上式消去全称量词后为

$$(\neg P(x, f(x)) \vee Q(x, g(x)) \wedge (\neg P(x, f(x)) \vee \neg R(x, g(x))) \qquad (23)$$

(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 3.10

上式的子句集中包含以下两个子句

$$\neg P(x, f(x)) \lor Q(x, g(x)) \quad (24)$$

$$\neg P(x, f(x)) \lor \neg R(x, g(x)) \quad (25)$$

(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. 由于每一个子句都对应着母式中的一个合取元, 并且所有变元都是由全称量词量化的, 因此任意两个不同子句的变量之间实际上不存在任何关系. 这样, 更换变量名是不会影响公式的真值的.

例 3.11

对前面的公式, 可把第二个子句集中的变元名 x 更换为 y, 子句集

$$\neg P(x, f(x)) \vee Q(x, g(x)) \tag{26}$$

$$\neg P(y, f(y)) \vee \neg R(y, g(y)) \tag{27}$$

子句集及其化简

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. 这样, 当原谓词公式为非永假时, 它与其标准子句集并不等价.

当原谓词公式为永假 (不可满足) 时, 其标准子句集则一定是永假的, 即 Skolem 化并不影响原谓词公式的永假性.

这个结论是归结原理的主要依据, 可用定理的形式来描述.

Theorem

不可满足的充要条件: 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.

为证明此定理, 先作如下说明: 为讨论问题方便, 设给定的谓词公式 F 已为前束形

$$(Q_1x_1)\cdots(Q_rx_r)\cdots(Q_nx_n)M(x_1,x_2,\cdots,x_n) \tag{28}$$

其中, $M(x_1,x_2,\cdots,x_n)$ 已化为合取范式. 由于将 F 化为这种前束形是一种很容易实现的等价运算, 因此这种假设是可以的.
又设 (Q_rx_r) 是第一个出现的存在量词 $(\exists x_r)$, 即 F 为

$$F=(x_1)\cdots(x_{r-1})(\exists x_r)(Q_{r+1}x_{r+1})\cdots(Q_nx_n)M(x_1,\cdots,x_{r-1},x_r,x_{r+1},\cdots,x_n) \tag{29}$$

为把 F 化为 Skolem 形, 需要先消去这个 $(\exists x_r)$, 并引入 Skolem 函数, 得到

$$F1 = (x_1)\cdots(x_{r-1})(Q_{r+1}x_{r+1})\cdots(Q_n x_n)M(x_1,\cdots,x_{r-1},f(x_1,\cdots x_{r-1}),x_{r+1}\cdots,x_n) \tag{30}$$

若能证明

$$F\text{不可满足} \Leftrightarrow F_1\text{不可满足} \tag{31}$$

则同理可证

$$F_1\text{不可满足} \Leftrightarrow F_2\text{不可满足} \tag{32}$$

重复这一过程, 直到证明了

$$F_{m-1}\text{不可满足} \Leftrightarrow F_m\text{不可满足} \tag{33}$$

为止. 此时, F_m 已为 F 的 Skolem 标准形. 而 S 只不过是 F_m 的一种集合表示形式. 因此有

$$F_m\text{不可满足} \Leftrightarrow \text{不可满足} \tag{34}$$

下面开始用反证法证明

$$F\text{不可满足} \Leftrightarrow F_1\text{不可满足} \tag{35}$$

先证明 $\Rightarrow$
已知 F 不可满足, 假设 F_1 是可满足的, 则存在一个解释 I, 使 F_1 在解释 I 下为真. 即对任意 $x_1, \cdots, x_{r-1}$ 在 I 的设定下有

$$(Q_{r+1}x_{r+1})\cdots(Q_n x_n)M(x_1,\cdots,x_{r-1},f(x_1,\cdots,x_{r-1}),x_{r+1},\cdots,x_n) \tag{36}$$

为真. 亦即对任意的 $x_1, \cdots, x_{r-1}$ 都有一个 $f(x_1, \cdots, x_{r-1})$ 使

$$(Q_{r+1}x_{r+1})\cdots(Q_n x_n)M(x_1,\cdots,x_{r-1},f(x_1,\cdots,x_{r-1}),x_{r+1}\cdots,x_n) \tag{37}$$

为真. 即在 I 下有

$$(\forall x_1)\cdots(\forall x_{r-1})(\exists x_r)(Q_{r+1}x_{r+1})\cdots(Q_n x_n)M(x_1,\cdots,x_{r-1},x_r,x_{r+1},\cdots,x_n) \tag{38}$$

为真. 即 F 在 I 下为真.
但这与前提 F 相矛盾, 即假设 F_1 为可满足是错误的. 从而可以得出“若 F 不可满足, 则必有 F_1 不可满足”.

再证明

已知 F_1 不可满足, 假设 F 是可满足的. 于是便有某个解释 I 使 F 在 I 下为真. 即对任意的 $x_1, \cdots, x_{r-1}$ 在 I 的设定下都可找到一个 x_r 使

$$(Q_{r+1}x_{r+1}) \cdots (Q_n x_n) M(x_1, \cdots, x_{r-1}, x_r, x_{r+1} \cdots, x_n) \tag{39}$$

为真. 若扩充 I, 使它包含一个函数 $f(x_1, \cdots, x_{r-1})$, 且有

$$x_r = f(x_1, \cdots, x_{r-1}) \tag{40}$$

这样, 就可以把所有的 $(f(x_1, \cdots, x_{r-1}))$ 映射到 x_r, 从而得到一个新的解释 I', 并且在此解释下对任意的 $x_1, \cdots, x_{r-1}$ 都有

$$(Q_{r+1}x_{r+1}) \cdots (Q_n x_n) M(x_1, \cdots, x_{r-1}, f(x_1, \cdots x_{r-1}), x_{r+1} \cdots, x_n) \tag{41}$$

为真. 即在 I' 下有

$$(\forall x_1) \cdots (\forall x_{r-1})(Q_{r+1}x_{r+1}) \cdots (Q_n x_n) M(x_1, \cdots, x_{r-1}, f(x_1, \cdots x_{r-1}), x_{r+1} \cdots, x_n) \tag{42}$$

为真.

基本思想

它说明 F_1 在解释 I′ 下为真. 但这与前提 F_1 是不可满足的相矛盾, 即假设 F 为可满足是错误的. 从而可以得出 "若F_1不可满足, 则必有F不可满足". 于是, 定理得证. 由此定理可知, 证明一个谓词公式是不可满足的, 只要证明其相应的标准子句集是不可满足的就可以了. 至于如何证明一个子句集的不可满足性, 由下面的海伯伦理论和鲁宾逊归结原理来解决.

1 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;

1 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.

鲁滨逊归结原理基本思想

首先把欲证明问题的结论否定, 并加入子句集, 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, 则继续使用归结法, 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.

鲁滨逊归结原理包括

命题逻辑归结原理, 谓词逻辑归结原理.

命题逻辑的归结

归结推理的核心是求两个子句的归结式

互补文字

若 P 是原子谓词公式, 则称 P 与 $\neg P$ 为互补文字.

亲本子句

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 4.1

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

$$C_{12} = Q \vee R \vee S$$

例 4.2

设 $C_1 = \neg Q$, $C_2 = Q$, 求 C_1 和 C_2 的归结式 C_{12}.

解: 这里 $L_1 = \neg Q$, $L_2 = Q$, 通过归结可以得到 $C_{12} = NIL$.

例 4.3

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

$$C_{12} = \neg P \tag{43}$$

然后再对 C_{12} 和 C_3 归结, 得到

$$C_{123} = NIL \tag{44}$$

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.

归结归结过程可用下图来表示，该树称为归结树.

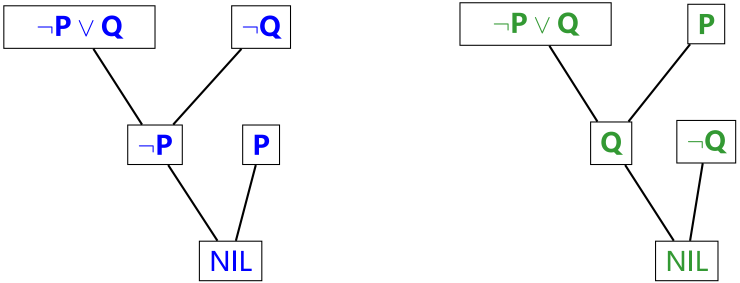


图 3: 例111归结树, $C_1 = \neg P \vee Q$, $C_2 = \neg Q, C_3 = P$,

Theorem

全概率公式: 归结式 C_{12} 是其亲本子句 C_1 和 C_2 的逻辑结论.

Proof.

(按定义) 设 $C_1 = L \vee C_1'$, $C_2 = \neg L \vee C_2'$ 关于解释 I 为真, 则只需证明 $C_{12} = C_1' \vee C_2'$ 关于解释 I 也为真. 对于解释 I, L 和 $\neg L$ 中必有一个为假.

若 L 为假, 则必有 C_1' 为真, 不然就会使 C_1 为假, 这将与前提假设 C_1 为真矛盾, 因此只能有 C_1' 为真.

同理, 若 $\neg L$ 为假, 则必有 C_2' 为真.

因此, 必有 $C_{12} = C_1' \vee C_2'$ 关于解释 I 也为真. 即 C_{12} 是 C_1 和 C_2 的逻辑结论. □

上述定理是归结原理中的一个重要定理, 由它可得到以下两个推论:

归结式的不可满足性

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

证明: 设 $S = \{C_1, C_2, C_3, \cdots\cdots, C_n\}$, C_{12} 是 C_1 和 C_2 的归结式, 则用 C_{12} 代替 C_1 和 C_2 后可得到一个新的子句集

$$S_1 = \{C_{12}, C_3, \cdots, C_n\}$$

设 S_1 是不可满足的, 则对不满足 S_1 的任一解释 I, 都可能有以下两种情况:

1. 解释 I 使 C_{12} 为真, 则 $C_3, \cdots\cdots, C_n$ 中必有一个为假, 即 S 是不可满足的.
2. 解释 I 使 C_{12} 为假, 即 $\neg C_{12}$ 为真, 根据 $(C_1 \wedge C_2 \Rightarrow C_{12} \Leftrightarrow \neg C_{12} \Rightarrow \neg C_1 \vee \neg C_2)$, 即 $\neg C_1 \vee \neg C_2$ 永真, 它说明解释 I 使 C_1 为假, 或 C_2 为假. 即 S 也是不可满足的. 因此可以得出

$$S_1 \text{ 的不可满足性} \Rightarrow S \text{ 的不可满足性}$$

归结式的不可满足性

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:

$$S_2 \text{ 的不可满足性} \Leftrightarrow S \text{ 的不可满足性}.$$

先证明

设 $S = \{C_1, C_2, C_3, \cdots, C_n\}$ 是不可满足的, 则 $C_1, C_2, C_3, \cdots, C_n$ 中必有一子句为假. 不妨设 C_1, C_2 可归结, 因而 $S2 = \{C_{12}, C_1, C_2, C_3, \cdots\cdots, C_n\}$ 必为不可满足.

再证明 $\Rightarrow$.

设 S_2 是不可满足的, 则对不满足 S_2 的任一解释 I, 都可能有以下两种情况:
① 解释 I 使 C_{12} 为真, 则 $C_1, C_2, C_3, \cdots, C_n$ 中必有一个为假, 即 S 是不可满足的.
② 解释 I 使 C_{12} 为假, 即 $\neg C_{12}$ 为真, 根据定理 3.2 有 $\neg(C_1 \wedge C_2)$ 永真, 即 $\neg C_1 \vee \neg C_2$ 永真, 它说明解释 I 使 C_1 为假, 或 C_2 为假. 即 S 也是不可满足的

由此可见 S 与 S_2 的不可满足性是等价的. 即

$$S \text{ 的不可满足性} \Leftrightarrow S_2 \text{ 的不可满足性}$$

性质说明

为证明子句集 S 的不可满足性, 只要对其中可进行归结得子句进行归结, 并把归结式加入到子句集 S 中, 或者用归结式代替他的亲本子句, 然后对新的子句集证明其不可满足性就可以了.

如果经归结能得到空子句, 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.

在命题逻辑中, 对不可满足的子句集 S, 其归结原理是完备的. 这种不可满足性可用如下定理描述:

Theorem

子句集的不可满足: 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.

命题逻辑的归结反演

归结原理: 假设 F 为已知前提, G 为欲证明的结论, 归结原理把证明 G 为 F 的逻辑结论转化为证明公式集 $F \wedge \neg G$ 为不可满足.

有前面的内容可知, 在不可满足的意义下, 公式集 $F \wedge \neg G$ 与其子句集是等价的, 即把公式集上的不可满足转化为子句集上的不可满足.

归结反演 应用归结原理证明定理的过程称为归结反演.

在命题逻辑中, 已知 F, 证明 G 为真的归结反演过程如下:

1. 把公式集 $\{F, \neg G\}$ 化为子句集 S.
2. 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.

1. 把公式集 $\{F, \neg G\}$ 化为子句集 S.
2. 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.

例 4.4

设已知的公式集为 $\{P, (P \land Q) \to R, (S \lor T) \to Q, T\}$, 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

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

其归结过程如下图的归结演绎树所示. 该树根为空子句.

空子句

假设结论 R 为假, 将 $\neg R$ 加入公式集, 并化为子句集

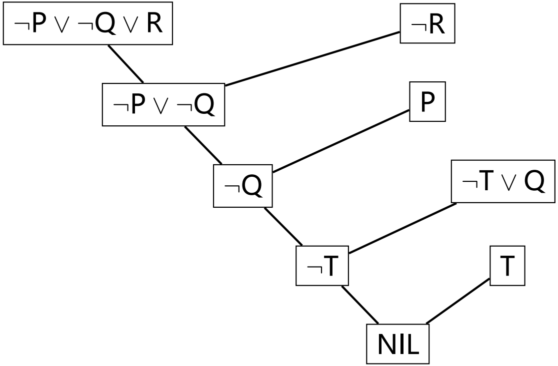
$$S = \{P, \neg P \vee \neg Q \vee R, \neg S \vee Q, \neg T \vee Q, T, \neg R\}.$$

其归结过程如下图的归结演绎树所示. 该树根为空子句.

含义

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.

根据归结原理的完备性, 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.



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

图 4: 归结树

谓词逻辑的归结

在谓词逻辑中, 由于子句集中的谓词一般都含有变元, 因此不能像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.

- 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:

归结式上的文字

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 L_2 存在最一般合一 σ, 则称

$$C_{12} = (C_1\sigma - \{L_1\sigma\}) \cup (C_2\sigma - \{L_2\sigma\}) \tag{45}$$

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.

例 4.5

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 L_2 的最一般合一是 $\sigma = \{a/y\}$. 根据前面的定义, 可得

$$
\begin{aligned}
C_{12} &= (C_1\sigma - \{L_1\sigma\}) \cup (C_2\sigma - \{L_2\sigma\}) \\
&= (\{P(a), R(x)\} - \{P(a)\}) \cup (\{\neg P(a), Q(b)\} - \{\neg P(a)\}) \\
&= (\{R(x)\}) \cup (\{Q(b)\}) = \{R(x), Q(b)\} \\
&= R(x) \vee Q(b)
\end{aligned}
\tag{46}
$$

例 4.6

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 3.20 的要求. 为了进行归结, 需要修改 C_2 中变元的名字, 令 $C_2 = \neg P(b) \vee R(y)$. 此时 $L_1 = P(x)$, $L_2 = \neg P(b)$, L_1 和 L_2 的最一般合一是 $\sigma = \{b/x\}$. 则有

$$\begin{aligned} C_{12} &= (\{C_1\sigma\} - \{L_1\sigma\}) \cup (\{C_2\sigma\} - \{L_2\sigma\}) \\ &= (\{P(b), Q(a)\} - \{P(b)\}) \cup (\{\neg P(b), R(y)\} - \{\neg P(b)\}) \\ &= (\{Q(a)\}) \cup (\{R(y)\}) = \{Q(a), R(y)\} \\ &= Q(a) \vee R(y) \end{aligned} \tag{47}$$

两点说明

1. 这里之所以使用集合符号和集合的运算, 目的是为了方便说明问题. 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 4.7

$C_1 = P(x), C_2 = \neg P(f(x))$, 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.

例 4.8

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.

例 4.9

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

$$C_1\sigma = P(f(a)) \vee Q(f(a)) \tag{48}$$

因子

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

$$C_{12} = R(b) \vee Q(f(a)) \tag{49}$$

我们把 $C_1\sigma$ 称为 C_1 的因子.

单元因子

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.

谓词逻辑中的归结原理

应用因子概念, 可对谓词逻辑中的归结原理给出如下定义:

互补文字

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 $C_2\sigma$ 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}.

例 4.10

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 $\sigma = \{f(x)/y\}$, 得 C_1 的因子

$$C_1\sigma = P(f(x)) \vee Q(g(x)). \tag{50}$$

对 C_1 的因子和 C_2 归结 $(\sigma = \{g(a)/x\})$, 可得到 C_1 和 C_2 的二元归结式

$$C_{12} = Q(g(g(a))) \vee Q(b). \tag{51}$$

注

对谓词逻辑, 定理 3.2 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, 所得到的子句集仍然保持着原子句集 S 的不可满足性.

归结原理的完备性

此外, 对谓词逻辑定理 3.3 也仍然适用, 即从不可满足的意义上说, 一阶谓词逻辑的归结原理也是完备的.

谓词逻辑的归结反演

谓词逻辑的归结反演

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.

例 4.11

- 在步骤 (3) 化简子句集时, 谓词逻辑需要把由谓词构成的公式集化为子句集;
- 在步骤 (4) 按归结原理进行归结时, 谓词逻辑的归结原理需要考虑两个亲本子句的最一般合一.

例 4.12

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) $\neg A(x,y) \vee \neg B(y) \vee C(f(x))$,
(2) $\neg A(u,v) \vee \neg B(v) \vee D(u,f(u))$,
(3) $\neg C(z)$,
(4) $A(m,n)$,
(5) $B(k)$.

(1)、(2) 是由 F 化出的两个子句,
(3)、(4) 和 (5) 是由 $\neg G$ 化出的 3 个子句.

应用谓词逻辑的归结原理对子句集进行归结

(6) $\neg A(x,y) \vee \neg B(y)$, 由 (1) 和 (3) 归结, 取 $\sigma = \{f(x)/z\}$.
(7) $\neg B(n)$, 由 (4) 和 (6) 归结, 取 $\sigma = \{m/x, n/y\}$.
(8) NIL, 由 (5) 和 (7) 归结, 取 $\sigma = \{n/k\}$.

因此, G 是 F 的逻辑结论.

用如下归结树来表示

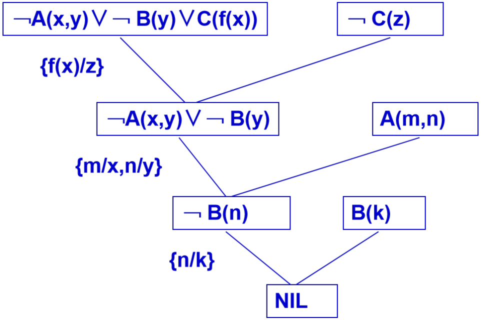


图 5: 归结树

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.

“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:

表 6: 谓词表

Pass(x, y)	x 可以通过 y 考试
Win(x, prize)	x 能获得奖励
Study(x)	x 肯学习
Happy(x)	x 是快乐的
Lucky(x)	x 是幸运的

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”.

$$(\forall x)(Pass(x, computer) \wedge Win(x, prize) \rightarrow Happy(x)). \tag{52}$$

“任何肯学习或幸运的人都可以通过所有考试”.

$$(\forall x)(\forall y)(Study(x) \vee Lucky(x) \rightarrow Pass(x, y)). \tag{53}$$

“张不肯学习但他是幸运的”.

$$\neg Study(zhang) \wedge Lucky(zhang). \tag{54}$$

“任何幸运的人都能获奖”.

$$(\forall x)(Lucky(x) \rightarrow Win(x, prize)). \tag{55}$$

结论“张是快乐的”的否定:

将上述谓词公式转化为子句集:

1. $\neg Pass(x, computer) \vee \neg Win(x, prize) \vee Happy(x)$.
2. $\neg Study(y) \vee Pass(y, z)$.
3. $\neg Lucky(u) \vee Pass(u, v)$.

1. $\neg Study(zhang)$.
2. $Lucky(zhang)$.
3. $\neg Lucky(w) \vee Win(w, prize)$.
4. $\neg Happy(zhang)$(结论的否定).

“幸福的生活”问题

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 4.13

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，快乐的人过着幸福的生活．求证：李明过着幸福的生活．

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 过着幸福的生活

将问题用谓词表示

“所有不贫穷并且聪明的人都是快乐的”

$$(\forall x)((\neg \mathrm{Poor}(x) \wedge \mathrm{Smart}(x)) \to \mathrm{Happy}(x)). \tag{57}$$

“那些看书的人是聪明的”

$$(\forall y)(\mathrm{Read}(y) \to \mathrm{Smart}(y)). \tag{58}$$

“李明能看书且不贫穷”

$$\mathrm{Read}(\mathrm{Liming}) \wedge \neg \mathrm{Poor}(\mathrm{Liming}). \tag{59}$$

“快乐的人过着幸福的生活”

$$(\forall z)(\mathrm{Happy}(z) \to \mathrm{Exciting}(z)). \tag{60}$$

目标“李明过着幸福的生活”的否定:

将谓词公式转化为子句集

1. Poor(x) ∨ ¬Smart(x) ∨ Happy(x).
2. ¬Read(y) ∨ Smart(y).
3. Read(Liming)

1. ¬Poor(Liming).
2. ¬Happy(z) ∨ Exciting(z).
3. ¬Exciting(Liming) (结论的否定).

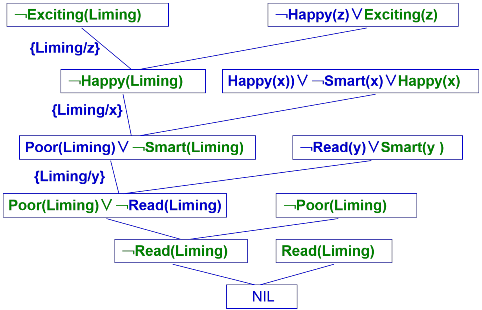

¬Exciting(Liming)
¬Happy(z)∨Exciting(z)
{Liming/z}
¬Happy(Liming)
Happy(x))∨¬Smart(x)∨Happy(x)
{Liming/x}
Poor(Liming)∨¬Smart(Liming)
¬Read(y)∨Smart(y)
{Liming/y}
Poor(Liming)∨¬Read(Liming)
¬Poor(Liming)
¬Read(Liming)
Read(Liming)
NIL

图 6: 归结树

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.

由于事先并不知道哪些子句对可进行归结, 更不知道通过对哪些子句对的归结能尽快得到空子句, 因此就需要对子句集中的所有子句逐对进行比较, 直到得出空子句为止.

盲目的全面进行归结的方法, 不仅会产生许多无用的归结式, 更严重的是会产生组核爆炸问题. 因此, 需要研究有效的归结策略来解决这些问题.

删除和限制策略

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;
- 限制策略是通过对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.

为了说明选择归结策略的重要性，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.

广度优先

广度优先

是一种穷尽子句比较的复杂搜索方法.

广度优先策略的归结过程

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;

例 5.1

设有如下子句集:

$$S = \{\neg I(x) \vee R(x), I(a), \neg R(y) \vee L(y), \neg L(a)\}. \quad (62)$$

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

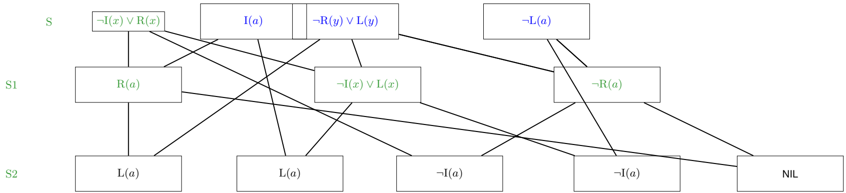


图 7: 广义归结树

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, 既浪费事间, 又浪费空间. 但是, 这种策略在当问题有解时保证能找到最短归结路径.

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.

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.

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 5.2

设有如下子句集:

$$S = \{\neg I(x) \vee R(x), I(a), \neg R(y) \vee L(y), \neg L(a)\}. \qquad (63)$$

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.

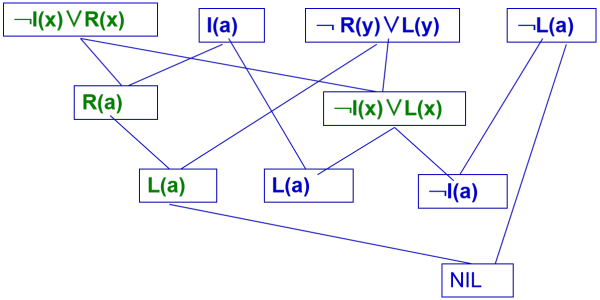


图 8: 支持集策略

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.

删除策略

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

常用的三种删除方法

有纯文字、重言式和包孕.

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 5.3

对子句集

$$S = \{P \vee Q \vee R, \neg Q \vee R, Q, \neg R\}. \tag{64}$$

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.

重言式删除法

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 5.4

$$P(x) \vee \neg P(x), P(x) \vee Q(x) \vee \neg P(x) \tag{65}$$

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. 可从子句集中删去重言式.

包孕删除法

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 5.5

$P(x)$ 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 $P(a) \vee Q(u) \vee R(w)$ $\sigma = \{a/x, u/y\}$

单文字子句策略

包孕子句的删除

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.

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.

例 5.6

设有如下子句集:

$$S = \{\neg I(x) \vee R(x), I(a), \neg R(y) \vee L(y), \neg L(a)\}. \tag{66}$$

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.

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.

单文字子句策略

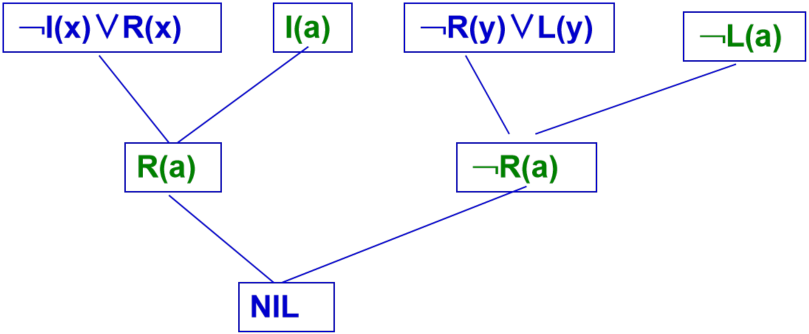


图 9: 单文字子句策略

线形输入策略

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.

例 5.7

设有如下子句集:

$$S = \{\neg I(x) \vee R(x), I(a), \neg R(y) \vee L(y), \neg L(a)\} \tag{67}$$

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 5.8

子句集

$$S = \{Q(u) \vee P(a), \neg Q(w) \vee P(w), \neg Q(x) \vee \neg P(x), Q(y) \vee \neg P(y)\} \tag{68}$$

从 S 出发很容易找到一棵归结反演树, 但不存在线性输入策略的归结反演树.

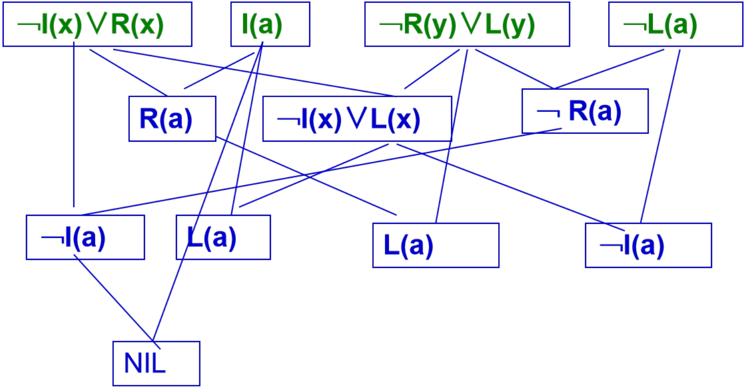


图 10: 线形输入策略

祖先过滤策略

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 5.9

设有如下子句集:

$$S = \{\neg Q(x) \vee \neg P(x), Q(y) \vee \neg P(y), \neg Q(w) \vee P(w), Q(a) \vee P(a)\} \tag{69}$$

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图11所示.

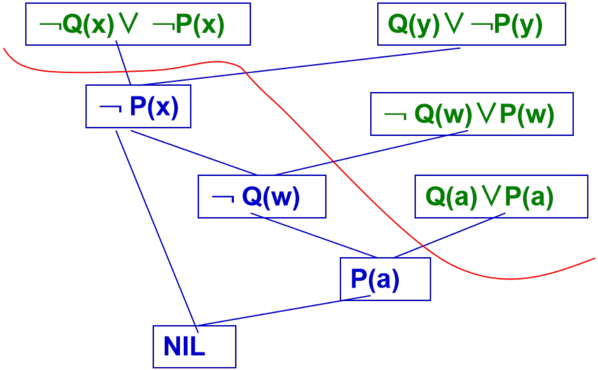


图 11：祖先过滤策略

证毕

用归结反演求取问题的答案

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.

归结原理的一般步骤

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;
2. 把问题的目标的否定用谓词公式表示出来, 并化为子句集;
3. 对目标否定子句集中的每个子句, 构造该子句的重言式，即把该目标否定子句和此目标否定子句的否定之间再进行析取所得到的子句, 用这些重言式代替相应的目标否定子句式, 并把这些重言式加入到前提子句集中, 得到一个新的子句集;

1 对这个新的子句集, 应用归结原理求出其证明树, 这时证明树的根子句不为空, 称这个证明树为修改的证明树;

1 用修改证明树的根子句作为回答语句, 则答案就在此根子句中.

例 6.2

设已知事实为: $A \vee B$, F 规则为:

$$r_1 : A \rightarrow C \wedge D \tag{88}$$

$$r_2 : B \rightarrow E \wedge G \tag{89}$$

目标公式为: $C \vee G$

先将已知事实用与/或树表示出来, 然后再用匹配弧把 r_1 和 r_2 分别连接到事实与/或树中与 r_1 和 r_2 前件匹配的两个不同端节点, 由于出现了以目标节点为终节点的解树, 故推理过程结束. 这一证明过程可用图18表示. 在该图中, 双箭头表示匹配弧, 它相当于一个单线连接符.

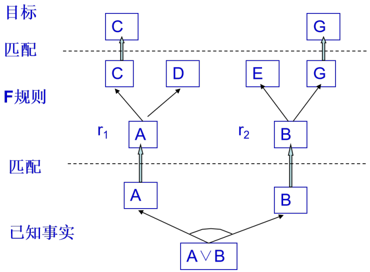


图 14: 与或树

为了验证上述推理的正确性，下面再用归结演绎推理给于证明. 由已知事实、规则及目标的否定可得到如下子句集:

$$A \vee B, \neg A \vee C, \neg A \vee D, \neg B \vee E, \neg B \vee G, \neg C, \neg G \tag{90}$$

其归结过程如图19所示.

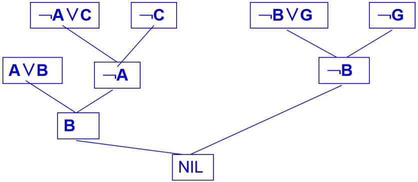


图 15: 与或树

可见，用归结演绎推理对已知事实、F 规则集目标的否定所过程的子句集进行归结，得到了空子句 NIL，从而证明了目标公式. 它与正向演绎推理所得到的结果是一致的.

谓词逻辑情况

由于事实、F 规则及目标中均含有变元, 因此, 其规则演绎过程还需要用最一般合一对变进行置换.

例 6.3

设已知事实的与/或形表示为: $P(x, y) \vee (Q(x) \wedge R(v, y))$.

F 规则为: $P(u, v) \to (S(u) \vee N(v))$.

目标公式为: $S(a) \vee N(b) \vee Q(c)$.

推理过程

如下图20所示.

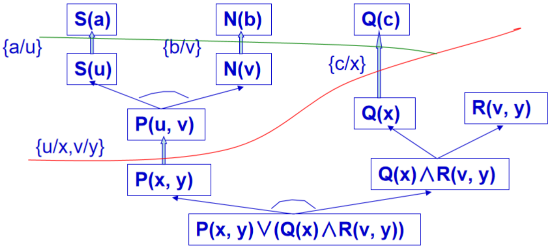


图 16: 与或树

规则逆向演绎推理

规则逆向演绎推理过程是从目标公式的与/或树出发, 反向使用规则 (B 规则), 对目标公式的与/或树进行变换, 直到得出含有事实节点一致解图为止.

步骤包括

- 目标公式的与/或形变换.
- 目标公式的与/或树表示.
- B 规则的表示形式.
- 规则逆向演绎推理过程.

逆向系统中的目标公式采用与/或形表示, 其化简采用与正向系统中对事实表达式处理的对偶形式.

要用存在量词约束变元的 Skolem 函数来替换由全称量词约束的相应变元, 消去全称量词, 再消去存在量词, 并进行变元换名, 使主析取元之间具有不同的变元名.

例 6.4

有如下目标公式:

$$(\exists y)(\forall x)(P(x) \to (Q(x) \land \neg(R(x) \land S(y)))). \tag{91}$$

Skolem 化后为

$$\neg P(f(y)) \lor (Q(f(y), y) \land (\neg R(f(y)) \lor \neg S(y))). \tag{92}$$

变元换名后为

$$\neg P(f(z)) \lor (Q(f(y), y) \land (\neg R(f(y)) \lor \neg S(y))). \tag{93}$$

目标公式的与/或形也可用与/或树表示出来, 其表示方法与正向演绎推理中事实的与或树表示略有不同.

规定

- 子表达式之间的析取关系用单一连接符连接, 表示或的关系;
- 子表达式之间的合取关系则用 k 线连接符连接, 表示与的关系.

对上述目标公式的与/或形, 可用如下的与/或树21表示. 在目标公式的与/或树中, 若把叶节点用它们之间的合取及析取关系连接起来.

原目标公式的三个子目标

$\neg P(f(z))$

$Q(f(y), y) \wedge \neg R(f(y))$

$Q(f(y), y) \wedge \neg S(y)$

归结演绎推理需要把谓词公式化为子句形，尽管这种转化在逻辑上是等价的，但是原来蕴含在谓词公式中的一些重要信息却会在求取子句形的过程中被丢失.

Example

下面的几个蕴含式

$$\neg A \wedge \neg B \to C, \neg A \wedge \neg C \to B, \neg A \to B \vee C, \tag{94}$$

$$\neg B \to A \vee C, \tag{95}$$

都与子句 $A \vee B \vee C$ 等价.

但在 $A \vee B \vee C$ 中, 是根本得不到原逻辑公式中所蕴含的哪些超逻辑的含义的. 况且, 在不少情况下人们多希望使用那种接近于问题原始描述的形式来进行求解, 而不希望把问题描述化为子句集.

规则是一种比较接近于人们习惯的问题描述方式, 按照这种问题描述方式进行求解的系统称为基于规则的系统, 或者叫做基于规则的演绎系统.

规则演绎系统的推理可分为正向演绎推理、逆向演绎推理和双向演绎推理三种形式.

规则正向演绎对应前面所讨论过的正向推理.

从已知事实出发, 正向使用规则, 直接进行演绎, 直至到达目标为止的一种证明方法. 一个直接演绎系统不一定比反演系统更有效, 但其演绎过程容易被人们所理解.

事实表达式的与/或形变换

在一个基于规则的正向演绎系统中, 其前提条件中的事实表达式是作为系统的初始综合数据库来描述的. 对事实的化简, 只须转换成不含蕴含符号“$\rightarrow$”的与/或形表示即可, 而不必化成子句集. 把事实表达式化为非蕴含形式的与/或形的主要步骤如下:

(1) 利用连接词化规律“$P \to Q \Leftrightarrow \neg P \vee Q$”, 消去蕴含符号. 事实上, 在事实表达式中很少有含蕴含符号“$\to$”出现, 因为可把蕴含式表示成规则.

(2) 利用狄 · 摩根定律及量词转换率把“$\neg$”移到紧靠谓词的位置, 直到每个否定符号的辖域最多只含一个谓词为止.

(3) 对所得到的表达式进行前束化.

(4) 对全称量词辖域内的变量进行改名和标准化, 对存在量词量化用 Skolem 函数代替, 使不同量词约束的变元有不同的名字.

(5) 消去全称量词, 而余下的变量都被认为是全程量词量化的变量.
(6) 对变量进行换名, 使主合取元具有不同的变量名.

Example

有如下表达式

$$(\exists x)(\forall y)(Q(y,x) \wedge \neg((R(y) \vee P(y)) \wedge S(x,y))), \tag{96}$$

可把它转化为:

$$Q(z,a) \wedge ((\neg R(y) \wedge \neg P(y)) \vee \neg S(a,y)). \tag{97}$$

这就是与/或形表示.

与或树表示

对上例所给出的与/或式，可用如下与/或树来表示.

事实表达式的与/或形可用一棵与/或树表示出来.

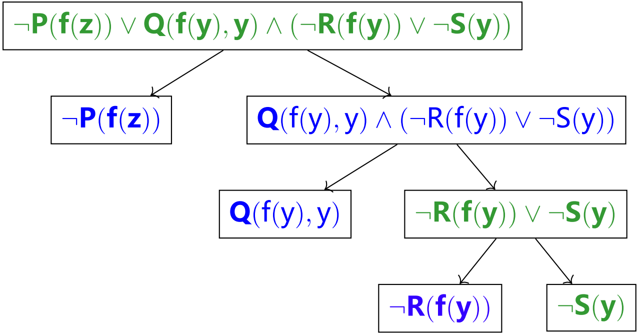


图 17: 与或树

表达式的子表达式——图 17 中每个节点表示事实的一个子式

当某个表达式为子表达式的析取时

$E_1 \vee E_2 \vee \cdots \vee E_k$, 其中的每个子表达式 E_i 均被表示为 $E_1 \vee E_2 \vee \cdots \vee E_k$ 的后继节点, 并由一个 k 线连接符将这些后继节点都连接到其父节点, 即表示成" 与" 的关系.

当某个表达式为子表达式的合取时

$E_1 \wedge E_2 \wedge \cdots \wedge E_k$, 其中的每个子表达式 E_i 均被表示成 $E_1 \wedge E_2 \wedge \cdots \wedge E_k$ 的一个单一的后继节点, 并由一个单一连接符连接到其父节点, 即表示成" 或" 的关系.

与或树的根节点就是整个事实表达式, 端节点均为事实表达式中的一个文字.

有了与或树的表示, 就可以求出其解树 (结束于文字节点上的子树) 集.

并且可以发现, 事实表达式的子句集与解树集之间存在着一一对应关系, 即解树集中的每个解树都对应着子句集中的一个子句. 其对应方式为: 解树集中每个解树的端节点上的文字的析取就是子句集中的一个子句.

与或树的 3 个解树

上图所示的与或树有 3 个解树, 分别对应这以下 3 个子句:

$$Q(z, a);$$
$$\neg R(y) \vee \neg S(a, y);$$
$$\neg P(y) \vee \neg S(a, y).$$

利用与/或树的这个性质, 可以把与/或树看作是对子句集的简洁表示. 不过, 表达式的与/或树表示要比子句集表示的通用性差一些, 原因是与/或树中的合取元没有进一步展开, 因此不能象在子句形中那样对某些变量进行改名, 这就限制了与/或树表示的灵活性.

Example

上面的最后一个子句, 在子句集中其变量 y 可全部改名为 x, 但却无法在与/或树中加以表示, 因而失去了通用性, 并且可能带来一些困难.

还需要指出, 这里的与/或树是作为综合数据库的一种表示, 其中的变量受全称量词的约束. 而在第 2 章可分解产生式系统中, 所描述的与/或树则是搜索过程的一种表示, 两者有着不同的目的和含义, 因此应用时应加以区分.

规则的与或形变换

与/或形正向演绎系统中是以正向方式使用规则 (F 规则) 对综合数据库中的与/或树结构进行变换的. 为简化这种演绎过程, 通常要求 F 规则应具有如下形式:

$$L \rightarrow W, \tag{98}$$

其中, L 为单文字, W 为与/或形公式. 假定出现在蕴含式中的任何变量全都受全称量词的约束, 并且这些变量已经被换名, 使得他们与事实公式和其他规则中的变量不同.

如果领域知识的规则表示形式与上述要求不同, 则应将它转换成要求的形式. 其变换步骤如下:

(1) 暂时消去蕴含符号“$\rightarrow$”. 设有如下公式:

$$(\forall x)(((\exists y)(\forall z)P(x,y,z)) \rightarrow (\forall u)Q(x,u)). \tag{99}$$

运用等价关系“$P \rightarrow Q \Leftrightarrow \neg P \vee Q$”, 可将上式变为:

$$(\forall x)(\neg((\exists y)(\forall z)P(x,y,z)) \vee (\forall u)Q(x,u)). \tag{100}$$

(2) 把否定符号 “¬” 移到紧靠谓词的位置上, 使其作用域仅限于单个谓词. 通过使用狄. 摩根定律及量词转换率可把上式转换为:

$$(\forall x)((\forall y)(\exists z)\neg P(x,y,z)) \vee (\forall u)Q(x,u)). \tag{101}$$

(3) 引入 Skolem 函数, 消去存在量词. 消去存在量词后, 上式可变为:

$$(\forall x)((\forall y)(\neg P(x,y,f(x,y))) \vee (\forall u)Q(x,u)). \tag{102}$$

化成前束式, 消去全部全称量词. 消去全称量词后, 上式变为:

$$\neg P(x,y,f(x,y)) \vee Q(x,u), \tag{103}$$

此公式中的变元都被视为受全称量词约束的变元.

(4) 恢复蕴含式表示. 利用等价关系 "$\neg P \vee Q \Leftrightarrow P \to Q$", 将上式变为:

$$P(x, y, f(x, y)) \to Q(x, u). \tag{104}$$

在上述对 F 规则的要求中, 之所以限制其前件为单文字, 是因为在进行正向演绎推理时要用 F 规则作用于表示事实的与/或树, 而该与/或树的叶节点都是单文字, 这样就可用 F 规则的前件与叶节点进行简单匹配. 对非单文字情况, 若形式为 $L_1 \vee L_2 \to W$, 则可将其转换成与之等价的两个规则 $L_1 \to W$ 与 $L_2 \to W$ 进行处理.

目标公式的表示形式

与/或树正向演绎系统要求目标公式用子句形表示.

如果目标公式不是子句形, 则需要化成子句形. 把一个目标公式转化为子句形的步骤与第 3 节所述的化简子句形的步骤类似, 但 Skolem 化的过程不同.

目标公式的 Skolem 化过程是化简子句形的 Skolem 过程的对偶形式, 即把目标公式中属于存在量词辖域内的全称变量用存在量词量化变量的 Skolem 函数来代替, 使经 Skolem 化目标公式只剩下存在量词, 然后再对析取元作变量替换, 最后把存在量词消掉.

设目标公式为

$$(\exists y)(\forall x)(P(x,y) \vee Q(x,y)). \tag{105}$$

用 Skolem 函数消去全称量词后有

$$(\exists y)(P(f(y),y) \vee Q(f(y),y)), \tag{106}$$

进行变量换名, 使每个析取元具有不同的变量符号, 于是有

$$(\exists y)P(f(y), y) \vee (\exists z)Q(f(z), z). \tag{107}$$

最后, 消去存在量词得

$$P(f(y), y) \vee Q(f(z), z), \tag{108}$$

这样, 目标公式中的变量都假定受存在量词的约束.

规则正向演绎系统

规则正向演绎推理过程是从已知事实出发, 不断运用 F 规则, 推出欲证明目标公式的过程. 即先用与/或树把已知事实表示出来, 然后再用 F 规则的前件和与或树的叶节点进行匹配, 并通过一个匹配弧把匹配成功的 F 规则加入到与/或树中, 依此使用 F 规则, 直到产生一个含有以目标节点为终止节点解图为止.

下面分别在命题逻辑和谓词逻辑的情况下来讨论规则正向演绎过程.

由于命题逻辑中的公式不含变元, 因此其规则演绎过程比较简单.

命题逻辑的规则演绎过程

Example

设已知事实为: $A \vee B$, F 规则为:

$$r_1 : A \rightarrow C \wedge D; \qquad (109)$$

$$r_2 : B \rightarrow E \wedge G. \qquad (110)$$

目标公式为: $C \vee G$.

Proof.

先将已知事实用与/或树表示出来, 然后再用匹配弧把 r_1 和 r_2 分别连接到事实与/或树中与 r_1 和 r_2 前件匹配的两个不同端节点, 由于出现了以目标节点为终节点的解树, 故推理过程结束.

这一证明过程可用图18表示.

图中, 双箭头表示匹配弧, 它相当于一个单线连接符. □

与或树

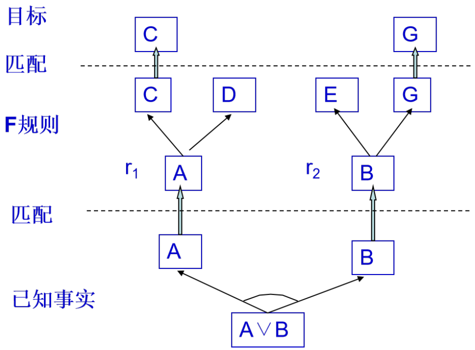
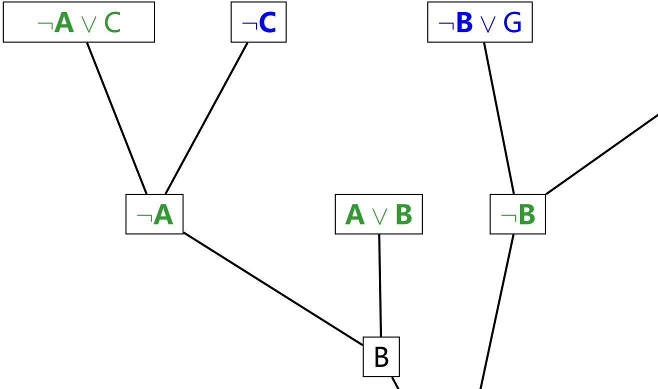


图 18: 与或树

为了验证上述推理的正确性，下面再用归结演绎推理给于证明. 由已知事实、规则及目标的否定可得到如下子句集:

$$\{A \vee B, \neg A \vee C, \neg A \vee D, \neg B \vee E, \neg B \vee G, \neg C, \neg G\}. \tag{111}$$

其归结过程如图19所示.



谓词逻辑的归结演绎推理

可见, 用归结演绎推理对已知事实、F 规则集目标的否定所过程的子句集进行归结, 得到了空子句 NIL, 从而证明了目标公式. 它与正向演绎推理所得到的结果是一致的.

由于事实、F 规则及目标中均含有变元, 因此, 其规则演绎过程还需要用最一般合一对变进行置换.

Example

设已知事实的与/或形表示为: $P(x, y) \vee (Q(x) \wedge R(v, y))$.
F 规则为: $P(u, v) \rightarrow (S(u) \vee N(v))$.
目标公式为: $S(a) \vee N(b) \vee Q(c)$.

推理过程如下图20所示.

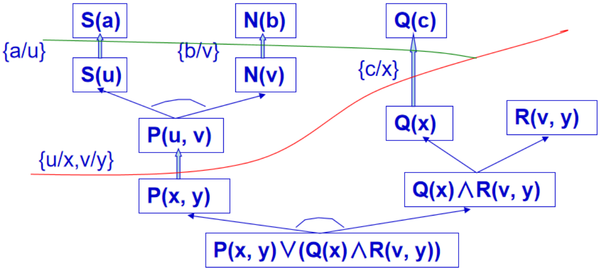


图 20: 与或树

规则逆向演绎推理过程

规则逆向演绎推理过程是从目标公式的与/或树出发, 反向使用规则 (B 规则), 对目标公式的与/或树进行变换, 直到得出含有事实节点一致解图为止.

规则逆向演绎推理过程包括的内容

- 目标公式的与/或形变换.
- 目标公式的与/或树表示.
- B 规则的表示形式.
- 规则逆向演绎推理过程.

逆向系统中的目标公式采用与/或形表示, 其化简采用与正向系统中对事实表达式处理的对偶形式.

即要用存在量词约束变元的 Skolem 函数来替换由全称量词约束的相应变元, 消去全称量词, 再消去存在量词, 并进行变元换名, 使主析取元之间具有不同的变元名.

Example

目标公式:

$$(\exists y)(\forall x)(P(x) \rightarrow (Q(x) \wedge \neg(R(x) \wedge S(y)))). \tag{112}$$

Skolem 化后为

$$\neg P(f(y)) \vee (Q(f(y), y) \wedge (\neg R(f(y)) \vee \neg S(y))). \tag{113}$$

变元换名后为

$$\neg P(f(z)) \vee (Q(f(y), y) \wedge (\neg R(f(y)) \vee \neg S(y))). \tag{114}$$

目标公式的与或形也可用与或树表示出来, 其表示方法与正向演绎推理中事实的与或树表示略有不同.

目标公式与或形的与或树表示规定

1) 表达式之间的析取关系用单一连接符连接, 表示为或的关系;
2) 子表达式之间的合取关系则用 k 线连接符连接, 表示为与的关系.

子目标是文字的合取式

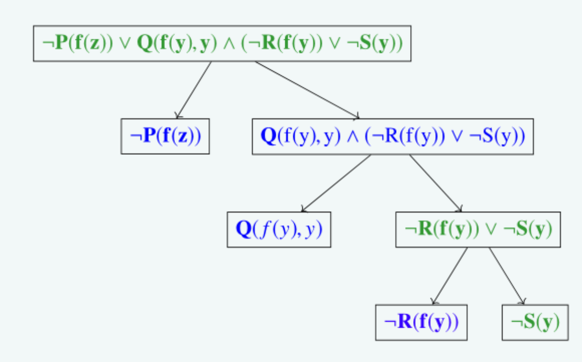


图 21: 与或树