

# 第三章 确定性推理

## ● 主要内容

- 命题逻辑表示法
- 谓词逻辑表示法
- 消解原理
- 消解反演求解
- 基于规则的推理
- 产生式系统

# 消解原理

上一章讨论过谓词公式、合式公式性质以及置换、合一等概念。在此基础上, 进一步介绍消解原理.

- 消解原理又称为归结原理。该原理是Robinson提出的一种基于逻辑的、采用反证法的推理方法。
- 由于其理论上的完备性, 归结原理成为机器定理证明的主要方法。

# 消解法基本原理

消解法的基本原理是**采用反证法**或者称为**反演推理方法**，将待证明的表达式（定理）转换成逻辑公式（谓词公式），然后再进行**归结**，归结能够顺利完成，则证明原公式（定理）是**正确性的**。

# 消解原理

- 证明的基本思想是：

设 $F_1, \dots, F_n, G$ 为公式， $G$ 为 $F_1, \dots, F_n$ 的逻辑推论，当且仅当公式  $((F_1 \wedge \dots \wedge F_n) \rightarrow G)$  是有效的

- 也可以采用反证法的思想：

设 $F_1, \dots, F_n, G$ 为公式， $G$ 为 $F_1, \dots, F_n$ 的逻辑推论，当且仅当公式  $(F_1 \wedge \dots \wedge F_n \wedge \neg G)$  是不可满足的

- 归结法的本质上就是一种反证法，它是在归结推理规则的基础上实现的：

为了证明一个命题 $P$ 恒真，它证明其反命题 $\neg P$ 恒假，即不存在使得 $\neg P$ 为真的解释

## 基本概念

**文字**：一个原子公式和原子公式的否定都叫做**文字**，如：

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

**子句**：由文字的**析取**组成的公式，如：

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

**空子句**：不包含任何文字的子句。

**子句集**：由子句构成的集合。

例： $\{P(x) \vee Q(x), \quad \neg P(x, f(x)) \vee Q(x, g(x))\}$

## 基本概念

- (1) **合取范式**:  $C_1 \wedge C_2 \wedge C_3 \cdots \wedge C_n$
- (2) **子句集**:  $S = \{C_1, C_2, C_3 \cdots, C_n\}$
- (3) 任何谓词公式F都可通过等价关系及推理规则化为相应的子句集S。

◇ **消解**，当消解可使用时，消解过程被应用于母体子句对，以便产生一个导出子句。

◇ 例如，如果存在某个公理  $E1 \vee E2$  和另一公理  $\sim E2 \vee E3$ ，那么，那么  $E1 \vee E3$  在逻辑上成立。这就是消解，而称  $E1 \vee E3$  为  $E1 \vee E2$  和  $\sim E2 \vee E3$  的消解式 (resolvent)

# 子句集的求取

- 任一谓词演算公式可以化成一个子句集。其变换过程由下列九个步骤组成：
- (1) 消去蕴涵符号
- 只应用  $\vee$  和  $\sim$  符号，以  $\sim A \vee B$  替换  $A \Rightarrow B$ 。
- (2) 减少否定符号的辖域
- 每个否定符号  $\sim$  最多只用到一个谓词符号上，并反复应用狄·摩根定律。例如：
- 以  $\sim A \vee \sim B$  代替  $\sim (A \wedge B)$
- 以  $\sim A \wedge \sim B$  代替  $\sim (A \vee B)$
- 以  $(\exists x) \{ \sim A \}$  代替  $\sim (\forall x) A$
- 以  $(\forall x) \{ \sim A \}$  代替  $\sim (\exists x) A$
- 以  $A$  代替  $\sim (\sim A)$

## 子句集的求取

- (3) 对变量标准化

- 在任一量词辖域内，受该量词约束的变量为一哑元（虚构变量），它可以在该辖域内处处统一地被另一个没有出现过的任意变量所代替，而不改变公式的真值。合式公式中变量的标准化，意味着对哑元改名以保证每个量词有其自己唯一的哑元。例如，把

- $$(\forall x)\{P(x) \Rightarrow (\exists x)Q(x)\}$$

- 标准化而得到：

- $$(\forall x)\{P(x) \Rightarrow (\exists y)Q(y)\}$$
-



## 子句集的求取

- (4) 消去存在量词
- (a) 如果要消去的存在量词在某些全称量词的辖域内
- **Skolem函数**：在公式  $(\forall y)[(\exists x)P(x, y)]$  中，存在量词是在全称量词的辖域内，我们允许所存在的  $x$  可能依赖于  $y$  值。令这种依赖关系明显地由函数  $g(y)$  所定义，它把每个  $y$  值映射到存在的那个  $x$ 。这种函数叫做**Skolem函数**。如果用**Skolem函数**代替存在的  $x$ ，我们就可以消去全部存在量词，并写成：

$$(\forall y)P(g(y), y)$$

- 一个公式消去一个存在量词的一般规则是**以一个Skolem函数代替每个出现的存在量词的量化变量**，而这个**Skolem函数的变量就是由那些全称量词所约束的全称量词量化变量**，这些全称量词的辖域包括要被消去的存在量词的辖域在内。
- **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。

## 子句集的求取

- (4) 消去存在量词

- 

(b) 如果要消去的存在量词不在任何一个全称量词的辖域内，那么就使用不含变量的Skolem函数即常量。

- 例如， $(\exists x)P(x)$ 化为 $P(A)$ ，其中常量符号A用来表示我们知道的存在的实体。A必须是个新的常量符号，它未曾在公式中其它地方使用过。

## 子句集的求取

- (5) 化为前束形

到这一步，已不留下任何存在量词，而且每个全称量词都有自己的变量。把所有全称量词移到公式的左边，并使每个量词的辖域包括这个量词后面公式的整个部分。所得公式称为前束形。前束形公式由前缀和母式组成，前缀由全称量词串组成，母式由没有量词的公式组成，即

前束形 = (前缀) (母式)  
          全称量词串 无量词公式

## 子句集的求取

- (6) 把母式化为合取范式
- 任何母式都可写成由一些谓词公式和(或)谓词公式的否定的析取的有限集组成的合取。这种母式叫做合取范式。我们可以反复应用分配律。把任一母式化成合取范式。例如, 我们把 $A \vee \{B \wedge C\}$ 化为 $\{A \vee B\} \wedge \{A \vee C\}$

## 子句集的求取

- (7) 消去全称量词

到了这一步，所有余下的量词均被全称量词量化了。同时，全称量词的次序也不重要了。因此，我们可以消去前缀，即消去明显出现的全称量词。

- (8) 消去连词符号  $\wedge$

- 用  $\{(A \vee B), (A \vee C)\}$  代替  $(A \vee B) \wedge (A \vee C)$ ，以消去明显的符号  $\wedge$ 。反复代替的结果，最后得到一个有限集，其中每个公式是文字的析取。任何一个只由文字的析取构成的合式公式叫做一个子句。

## 子句集的求取

- (9) 更换变量名称

可以更换变量符号的名称，**使一个变量符号不出现在一个以上的子句中**。例如，对于子集  $\{\sim P(x) \vee \sim P(y) \vee P[f(x, y)],$

- $\sim P(x) \vee Q[x, g(x)], \sim P(x) \vee \sim P[g(x)]\}$ ，在更改变量名后，可以得到子句集：

- $\{\sim P(x1) \vee \sim P(y) \vee P[f(x1, y)],$
- $\sim P(x2) \vee Q[x2, g(x2)], \sim P(x3) \vee \sim P[g(x3)]$

- 子句集的求取
- 例子，将下列谓词演算公式化为一个子句集
- 

$$(\forall x) \left\{ P(x) \Rightarrow \left\{ (\forall y) [P(y) \Rightarrow P(f(x, y))] \wedge \sim (\forall y) [Q(x, y) \Rightarrow P(y)] \right\} \right\}$$

## 子句集的求取过程

$$(\forall x) \left\{ P(x) \Rightarrow \left\{ (\forall y) [P(y) \Rightarrow P(f(x, y))] \wedge \sim (\forall y) [Q(x, y) \Rightarrow P(y)] \right\} \right\}$$

开始:

### 1. 消去蕴涵符号

只应用 $\vee$ 和 $\sim$ 符号, 以 $\sim A \vee B$ 替换 $A \rightarrow B$ 。

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

### 2. 减少否定符号的辖域

每个否定符号 $\sim$ 最多只是用到一个谓词符号上, 并反复应用狄·摩根定律。

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



## 子句集的求取过程

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

### 3. 对变量标准化

对哑元（虚构变量）改名，以保证每个量词有其自己唯一的哑元。

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

### 4. 消去存在量词

以Skolem函数代替存在量词内的约束变量，然后消去存在量词

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

式中， $w = g(x)$ 为一Skolem函数。

## 子句集的求取过程

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

式中,  $w = g(x)$  为一Skolem函数。

### 5. 化为前束形

把所有全称量词移到公式的左边, 并使每个量词的辖域包括这个量词后面公式的整个部分。

前束形 = {前缀}                      {母式}  
                    全称量词串              无量词公式

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

### 6. 把母式化为合取范式

任何母式都可写成由一些谓词公式和（或）谓词公式的否定的析取的有限集组成的合取。

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

## 子句集的求取过程

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

### 7. 消去全称量词

所有余下的量词均被全称量词量化了。消去前缀，即消去明显出现的全称量词

$$\begin{aligned} &\sim P(x) \vee \sim P(y) \vee P(f(x, y)) \\ &\quad \sim P(x) \vee Q(x, g(x)) \\ &\quad \sim P(x) \vee \sim P(g(x)) \end{aligned}$$

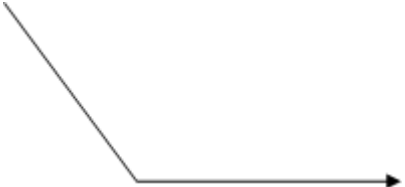
### 8. 消去连词符号 $\wedge$

用  $\{A, B\}$  代替  $(A \wedge B)$ , 消去符号  $\wedge$ . 最后得到一个有限集, 其中每个公式是文字的析取。

# 子句集的求取过程

## 9. 更换变量名称

可以更换变量符号的名称，使一个变量符号不出现在一个以上的子句中



The diagram shows a line starting from the text '可以更换变量符号的名称...' and ending in an arrow pointing to a pink box containing three logical clauses.

$$\begin{aligned} &\sim P(x1) \vee \sim P(y) \vee P[f(x1, y)] \\ &\sim P(x2) \vee Q[x2, g(x2)] \\ &\sim P(x3) \vee \sim P[g(x3)] \end{aligned}$$

## 例2

$$(\forall x) \{ [(\forall y)P(x,y)] \rightarrow \neg(\forall y)[Q(x,y) \rightarrow R(x,y)] \}$$

1.  $(\forall x) \{ \neg[(\forall y)P(x,y)] \vee [\neg(\forall y)(\neg Q(x,y) \vee R(x,y))] \}$
2.  $(\forall x) \{ [(\exists y)\neg P(x,y)] \vee [(\exists y)(Q(x,y) \wedge \neg R(x,y))] \}$
3.  $(\forall x) \{ [(\exists y)\neg P(x,y)] \vee [(\exists z)(Q(x,z) \wedge \neg R(x,z))] \}$

4. **上式中存在量词  $(\exists y)$  及  $(\exists z)$  都位于  $(\forall x)$  的辖域内，所以需要用Skolem函数替换，设替换y和z的Skolem函数分别是  $f(x)$  和  $g(x)$ ，则替换后得到**

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

5. **化为前束形**

## 举例

6.  $(\forall x) \{ [(\neg P(x, f(x)) \vee Q(x, g(x))) \wedge (\neg P(x, f(x)) \vee \neg R(x, g(x)))] \}$

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

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

9. 
$$\begin{array}{l} \neg P(x, f(x)) \vee Q(x, g(x)) \\ \neg P(y, f(y)) \vee \neg R(y, g(y)) \end{array}$$

## 练习

- (1)

$$(\forall x)(\exists y)\{P(x, y) \vee [Q(x, y) \rightarrow R(x, y)]\}$$

- (2)

$$(\exists x)(\forall y)\{[S(x) \vee T(x, y)] \rightarrow R(x)\}$$

- (3)

$$\forall x \left[ \exists y (U(x, y) \wedge I(y)) \rightarrow \exists u (P(u) \wedge Q(x, u)) \right]$$

- (4)

$$\sim \left[ \exists u P(u) \rightarrow \forall x \forall y (I(y) \rightarrow \sim U(x, y)) \right]$$

- **作业题：**

- **将下列谓词公式化为子句集：**

- (1)  $(\forall x) \{P(x) \rightarrow P(x)\}$

- (2)  $\forall x \forall y (On(x, y) \rightarrow Above(x, y))$

- (3)  $\forall x \forall y$   
 $\forall z (Above(x, y) \wedge Above(y, z) \rightarrow Above(x, z))$

- (4) **补充：**



## 消解反演

- 定理证明的任务：
  - 由前提  $A1 \wedge A2 \wedge \dots \wedge A_n$
  - 推出结论  $B$
  - 即证明： $A1 \wedge A2 \wedge \dots \wedge A_n \rightarrow B$  永真
- 转化为证明：
  - $A1 \wedge A2 \wedge \dots \wedge A_n \wedge \neg B$  为永假式
- 归结推理就是：从  $A1 \wedge A2 \wedge \dots \wedge A_n \wedge \neg B$  出发，使用归结推理规则来找出矛盾，最后证明定理  $A1 \wedge A2 \wedge \dots \wedge A_n \rightarrow B$  的成立

# 消解反演

- 归结方法是一种机械化的, 可在计算机上加以实现的推理方法
- 可认为是一种反向推理形式
- 提供了一种自动定理证明的方法

# 消解反演

- 一般过程：
  - 1) 建立子句集 $S$
  - 2) 从子句集 $S$ 出发, 仅对 $S$ 的子句间使用归结推理规则
  - 3) 如果得出空子句, 则结束; 否则转下一步
  - 4) 将所得归结式仍放入 $S$ 中
  - 5) 对新的子句集使用归结推理规则
  - 6) 转(3)
- **空子句**不含有文字, 它不能被任何解释满足, 所以空子句是永假的, **不可满足的**。
- 归结过程出现空子句, 说明出现**互补文字**, 说明 $S$ 中有矛盾, 因此 $S$ 是不可满足的。

## 消解反演

- 如欲证明Q为 $P_1, P_2, \dots, P_n$ 的逻辑结论，只需证

$$(P_1 \wedge P_2 \wedge \dots \wedge P_n) \wedge \neg Q$$

是不可满足的，或证明其子句集是不可满足的。而子句集的不可满足性可用归结原理来证明。

- 应用归结原理证明定理的过程称为归结（消解）反演。
- 设F为已知前提的公式集，Q为目标公式（结论），用归结反演进行证明的步骤是：
  1. 否定Q，得到 $\neg Q$ ；
  2. 把 $\neg Q$ 并入到公式集F中，得到 $\{F, \neg Q\}$ ；
  3. 把公式集 $\{F, \neg Q\}$ 化为子句集S；
  4. 应用**消解推理规则**对子句集S中的子句进行归结，并把每次归结得到的归结式都并入S中。如此反复进行，若出现了空子句，则停止归结。

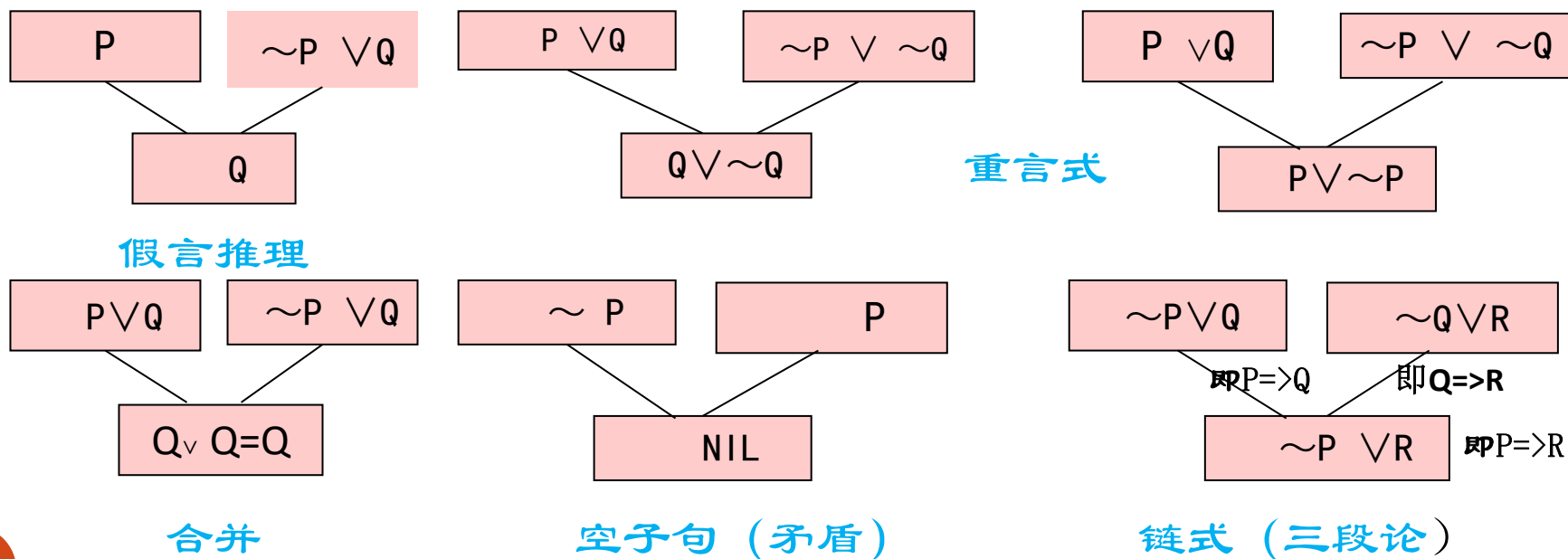
# 消解推理规则

重言式对应于所有指派，命题公式均取值真

## 消解的定义

令 $L_1, L_2$ 为两任意原子公式： $L_1$ 和 $L_2$ 具有相同的谓词符号，但一般具有不同的变量，已知两个子句 $L_1 \vee \alpha$ 和 $\sim L_2 \vee \beta$ ，如果 $L_1$ 和 $L_2$ 具有最一般合一 $\sigma$ ，那么通过消解可以从两个父辈子句推导出一个新子句 $(\alpha \vee \beta) \sigma$ 。这个新子句叫做消解式。

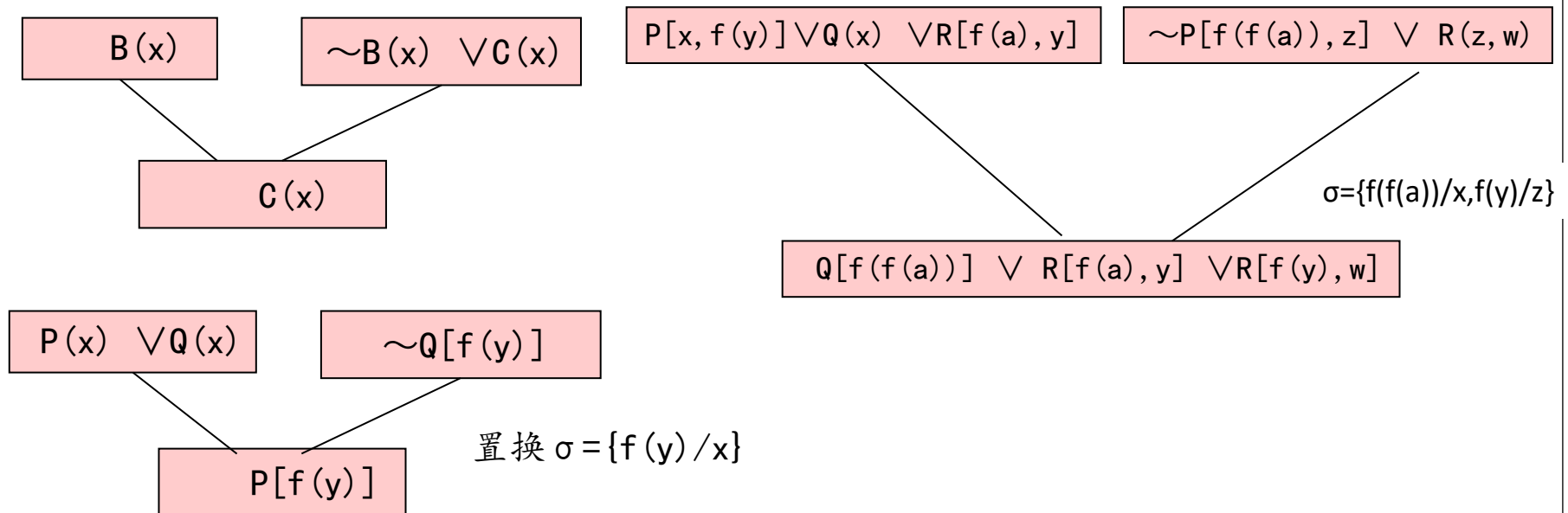
## 消解式求法



# 含有变量的消解式

## 含有变量的子句之消解式

- 要把消解推理规则推广到含有变量的子句，必须找到一个作用于父辈子句的置换，使父辈子句含有互补文字。



## 消解推理的常用规则

父 辈 子 句	消 解 式
$P$ 和 $\sim P \vee Q$ 即 $(P \Rightarrow Q)$ $P \vee Q$ 和 $\sim P \vee Q$ $P \vee Q$ 和 $\sim P \vee \sim Q$ $\sim P \vee P$ $\sim P \vee Q$ (即 $P \Rightarrow Q$ )和 $\sim Q \vee R$ (即 $Q \Rightarrow R$ ) $B(x)$ 和 $\sim B(x) \vee C(x)$ $P(x) \vee Q(x)$ 和 $\sim Q(f(y))$	$Q$ $Q$ $Q \vee \sim Q$ 和 $P \vee \sim P$ $NIL$ $\sim P \vee R$ (即 $P \Rightarrow R$ ) $C(x)$ $P(f(y)), \sigma = \{f(y)/x\}$
$P(x, f(y)) \vee Q(x) \vee R(f(a), y)$ 和 $\sim P(f(f(a)), z) \vee R(z, w)$	$Q(f(f(a))) \vee R(f(a), y) \vee R(f(y), w),$ $\sigma = \{f(f(a))/x, f(y)/z\}$

## 消解反演

- 例1: 证明

- 前提:  $(P \rightarrow Q) \wedge \sim Q$ , 结论:  $\sim P$

- 证明: 首先建立子句集:

$$S = \{\sim P \vee Q, \sim Q, P\}$$

- 对S作归结:

(1)  $\sim P \vee Q$

(2)  $\sim Q$

(3)  $P$

(4)  $\sim P$                       (1) (2) 归结

(5) NIL                        (3) (4) 归结



## 消解反演的例子

例2 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：首先把F和 $\neg G$ 化为子句集：

$$F = \{\neg A(x,y) \vee \neg B(y) \vee C(f(x)), \neg A(x,y) \vee \neg B(y) \vee D(x, f(x))\}$$

$$\neg G = \{\neg C(z), A(a,b), B(b)\}$$

然后进行归结：

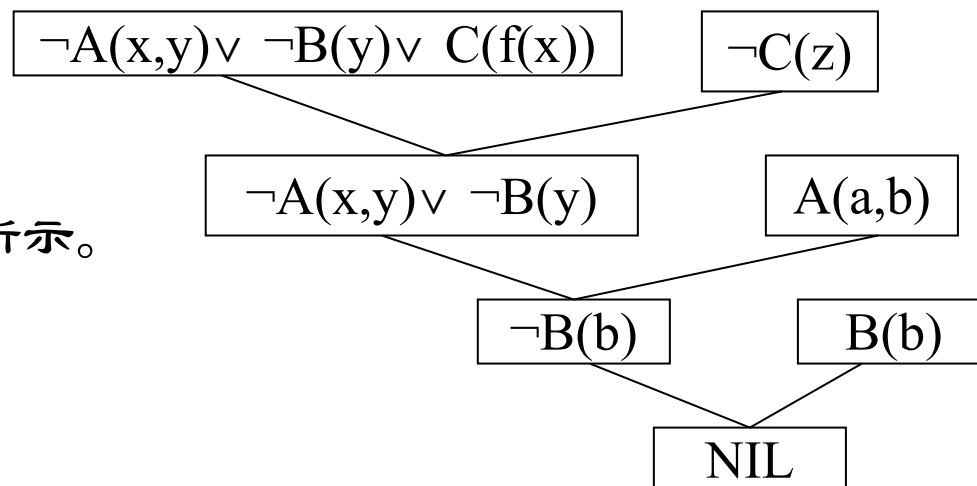
(6)  $\neg A(x,y) \vee \neg B(y)$       由(1)与(3)归结,  $\{f(x)/z\}$

(7)  $\neg B(b)$       由(4)与(6)归结,  $\{a/x, b/y\}$

(8) NIL      由(5)与(7)归结

所以G是F的逻辑结论。

上述归结过程如右图归结树所示。



## 消解反演

- 例3:
- 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。
- 求证：公司一定录取C。

应用消解原理进行消解：

(5)  $P(B) \vee P(C)$  (1)与(2)消解

(6)  $P(C)$  (3)与(5)消解

(7) nil (4)与(6)消解

所以 公司一定录取C

(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。

求证：公司一定录取C。

### ● 例3

● 证明：设用 $P(x)$ 表示录取 $x$ 。

● 把公司的想法用谓词公式表示如下：

● (1)  $P(A) \vee P(B) \vee P(C)$

● (2)  $P(A) \wedge \sim P(B) \Rightarrow P(C)$

● (3)  $P(B) \Rightarrow P(C)$

● 把要求证的问题否定，并用谓词式表示出来：

● (4)  $\sim P(C)$

化为子句集

(1)  $P(A) \vee P(B) \vee P(C)$

(2)  $\sim P(A) \vee P(B) \vee P(C)$

(3)  $\sim P(B) \vee P(C)$

(4)  $\sim P(C)$

## 消解反演

- 例4:
- 前提：每个储蓄钱的人都获得利息。  
结论：如果没有利息，那么就沒有人去储蓄钱。
- 证明：令  $S(x, y)$  表示“ $x$ 储蓄 $y$ ”
- $M(x)$  表示“ $x$ 是钱”  
 $I(x)$  表示“ $x$ 是利息”  
 $E(x, y)$  表示“ $x$ 获得 $y$ ”

## 消解反演

- **前提：每个储蓄钱的人都获得利息。**

**结论：如果没有利息，那么就沒有人去储蓄钱。**

$S(x, y)$  表示 “ $x$  储蓄  $y$ ” ;  $M(x)$  表示 “ $x$  是钱” ;  
 $I(x)$  表示 “ $x$  是利息” ;  $E(x, y)$  表示 “ $x$  获得  $y$ ” 。

- **于是上述命题写成下列形式：**  
 $(\forall x)[(\exists y)(S(x, y) \wedge M(y)) \Rightarrow (\exists y)(I(y) \wedge E(x, y))]$

$$\neg(\exists x)I(x) \Rightarrow (\forall x)(\forall y)(M(y) \Rightarrow \neg S(x, y))$$

# 证明:

化为子句形 把前提化为子句形:

$$(1) \sim S(x,y) \vee \sim M(y) \vee I(f(x))$$

$$(2) \sim S(x,y) \vee \sim M(y) \vee E(x,f(x))$$

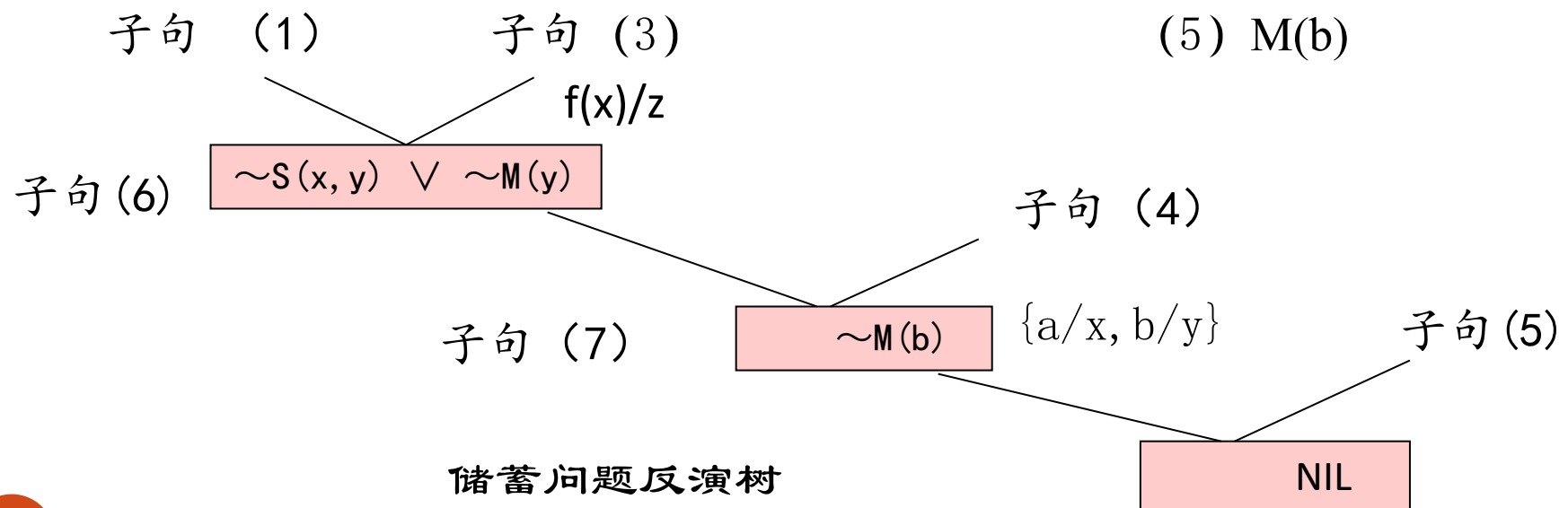
把结论化的否定为子句形:

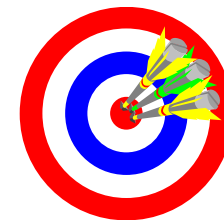
$$(3) \sim I(z)$$

$$(4) s(a,b)$$

$$(5) M(b)$$

## 消解反演求NIL





# 反演求解过程

## ■ 反演求解过程

■ 从反演树求取答案步骤：

■ (1) 把由目标公式的否定产生的每个子句添加到目标公式否定之否定的子句中去。

■ (2) 按照反演树，执行和以前相同的消解，直至在根部得到某个子句止

■ (3) 用根部的子句作为一个回答语句

## ■ 实质

■ 把一棵根部有NIL的反演树变换为根部带有回答语句的一棵证明树

## 反演求解过程

- 例：

“如果无论约翰 (John) 到哪里去，菲多 (Fido) 也就去那里，那么如果约翰在学校里，菲多在哪里呢？”
- 公式集S：
- $(\forall x) [AT(JOHN, x) \Rightarrow AT(FIDO, x)]$
- $AT(JOHN, SCHOOL)$

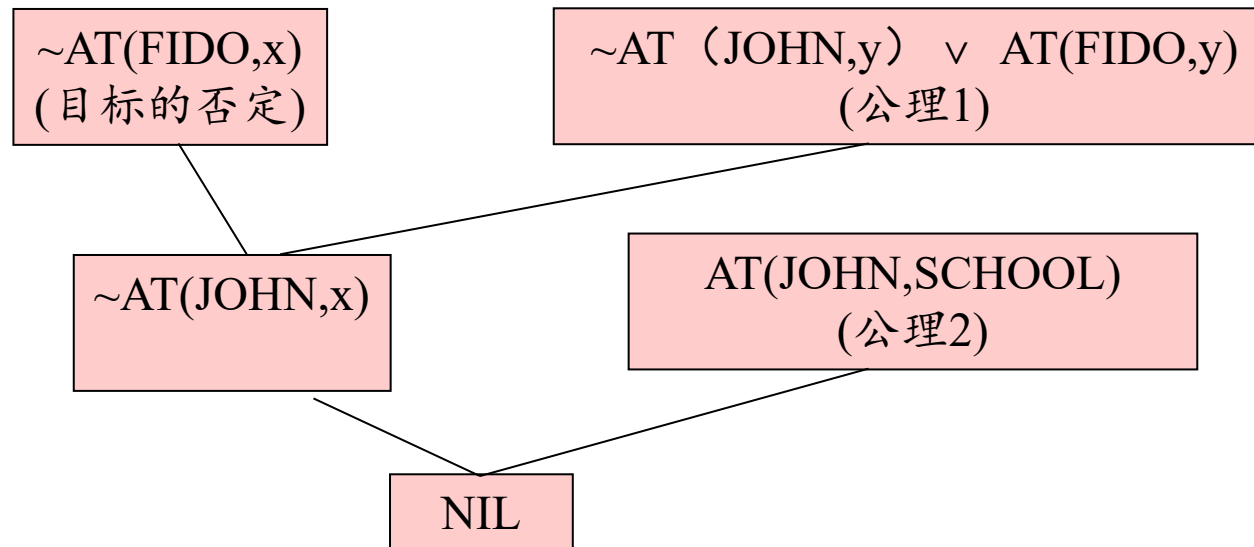


## 反演求解过程

- 如果我们首先证明公式  $(\exists x) AT(FIDO, x)$  在逻辑上遵循S，然后寻求一个存在x的例，那么就能解决“菲多在哪里”的问题。
- 关键想法是把问题化为一个包含某个存在量词的目标公式，使得此存在量词量化变量表示对该问题的一个解答。如果问题可以从给出的事实得到答案，那么按这种方法建立的目标函数在逻辑上遵循S。在得到一个证明之后，我们就试图求取存在量词量化变量的一个例，作为一个回答。

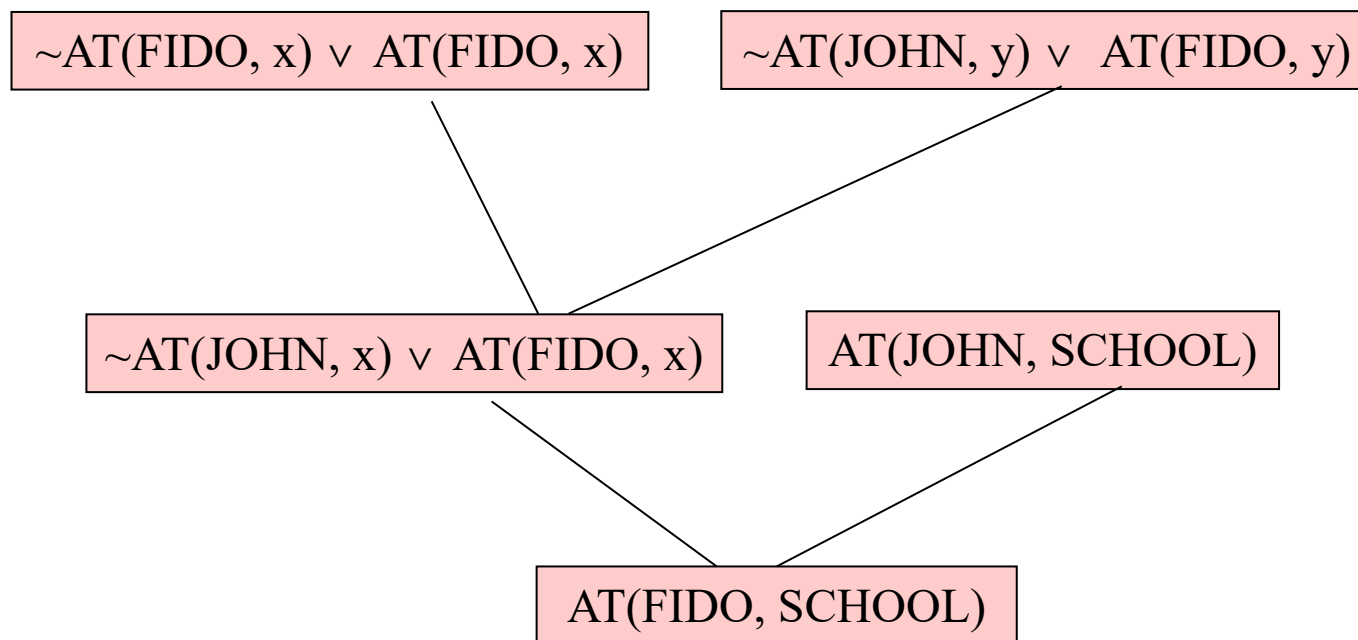
## 反演求解过程

- 目标公式( $\square x$ )AT(FIDO,X)的否定产生：
- $(\forall x) [\sim \text{AT}(\text{FIDO}, x)]$
- 其子句形式为： $\sim \text{AT}(\text{FIDO}, x)$



“菲多在哪里”例题的反演树

## 反演求解过程



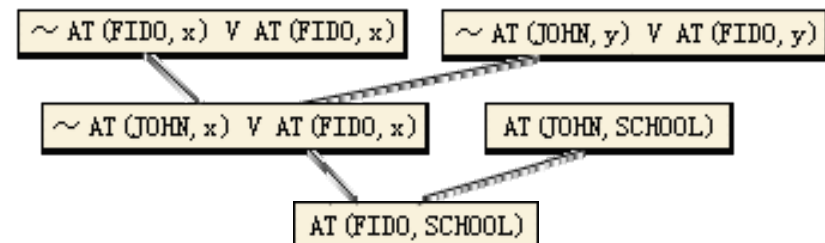
从消解求取答案例题的证明树

## 反演求解过程

- 答案求取涉及到把一棵根部有NIL的**反演树**变换为在根部带有可用作答案的某个语句的一颗**证明树**。
- 由于变换关系涉及到把由目标公式的否定产生的每个子句变换为一个**重言式**，所以被变换的证明树就是一棵消解的证明树，其在根部的语句在逻辑上遵循公理加上重言式，因而也单独地遵循公理。因此**被变换的证明树本身就证明了求取办法是正确的**。

## 反演求解过程

- (1) 目标公式否定的子句形式为  $\sim \text{AT}(\text{FIDO}, x)$  把它添加至目标公式的否定之否定的子句中, 得重言式  $\sim \text{AT}(\text{FIDO}, x) \vee \text{AT}(\text{FIDO}, x)$
- (2) 用反演树进行消解, 并在根部得到子句:  
 $\text{AT}(\text{FIDO}, \text{SCHOOL})$
- (3) 从根部求得答案  $\text{AT}(\text{FIDO}, \text{SCHOOL})$ , 用此子句作为回答语句。
- 因此, 子句  $\text{AT}(\text{FIDO}, \text{SCHOOL})$  就是这个问题的合适答案



# 反演求解过程

例2:

已知:

$(\forall x) (\forall y) (\forall z) [\text{Father}(z,x) \wedge \text{Brother}(x,y) \Rightarrow \text{Father}(z,y)]$

$\text{Brother}(\text{John}, \text{Bob})$

$\text{Father}(\text{Jim}, \text{John})$

问: 谁是Bob的父亲?

$\text{Father}(u, \text{Bob}), u=?$

构造:

$\neg \text{Father}(u, \text{Bob}) \vee \text{Father}(u, \text{Bob})$

化为子句集:

1  $\neg \text{Father}(z,x) \vee \neg \text{Brother}(x,y) \vee \text{Father}(z,y)$

2  $\text{Brother}(\text{John}, \text{Bob})$

3  $\text{Father}(\text{Jim}, \text{John})$

4  $\neg \text{Father}(u, \text{Bob}) \vee \text{Father}(u, \text{Bob})$

$\neg \text{Father}(z,x) \vee \neg \text{Brother}(x,y) \vee \text{Father}(z,y)$

$\text{Brother}(\text{John}, \text{Bob})$

$\text{John}/x, \text{Bob}/y$

$\text{Father}(\text{Jim}, \text{John})$

$\neg \text{Father}(z, \text{John}) \vee \text{Father}(z, \text{Bob})$

$\text{Jim}/z$

$\neg \text{Father}(u, \text{Bob}) \vee \text{Father}(u, \text{Bob})$   $\text{Father}(\text{Jim}, \text{Bob})$

$\text{Jim}/u$

$\text{Father}(\text{Jim}, \text{Bob})$

## 补充作业1

- 判断以下子句集是否为不可满足
- 
- $\{P(x) \vee Q(x) \vee R(x), \neg P(y) \vee R(y), \neg Q(a), \neg R(b)\}$



## 补充作业2:

已知 规则1: 任何人的兄弟不是女性。

规则2: 任何人的姐妹必是女性。

事实: Mary是Bill的姐妹。

求证: 用消解反演方法证明Mary不是Tom的兄弟。

请依次完成以下三问, 从而使问题得证。

(1) 定义谓词。将待证明的问题的前提条件和逻辑结论用谓词公式表示出来。

(2) 将上述规则与事实及求证目标的否定化成子句集。

(3) 利用消解原理对上面的子句集中的子句进行消解。

提示: 定义谓词。

Brother (x, y) : 表示x是y的兄弟。

Sister (x, y) : 表示x是y的姐妹。

Women (x) : 表示x是女性。

$$(\forall x)(\forall y)[Brother(x, y) \rightarrow \neg Woman(x)]$$
$$(\forall x)(\forall y)[Sister(x, y) \rightarrow Woman(x)]$$

## 补充作业3

- 已知规则：对于任意的 $x, y, z$ ，若 $x$ 是 $y$ 的父亲且 $z$ 是 $x$ 的父亲，则 $z$ 是 $y$ 的祖父，即  $(\forall x) (\forall y) (\forall z) [F(z, x) \wedge F(x, y)] \Rightarrow G(z, y)$ 。
- 已知事实1：John是Bob的父亲，即 $F(\text{John}, \text{Bob})$ 。
- 已知事实2：Jim是John的父亲，即 $F(\text{Jim}, \text{John})$ 。
- 用消解反演方法求解：谁是Bob的祖父？

## 规则演绎系统

- **消解反演方法的特点是简单，易于程序实现。**
- **其不足是效率低，不直观，人难于理解其“证明”过程。其原因是消解反演方法将所有的谓词公式均化简为子句，致使很多隐含在原来的谓词公式中的、对推理有利的信息得不到充分的利用。**
- **比如蕴涵关系 $P \Rightarrow Q$ ，除了其逻辑含义外，还隐含了“由 $P$ 推出 $Q$ ”这样的信息。如果有效的利用这些信息，会使得推理进行的更加合理、自然。基于规则的演绎系统将类似于 $P \Rightarrow Q$ 这样的蕴涵关系作为规则使用，直接用于推理。这类系统主要强调使用规则进行演绎，故称为规则演绎系统。**

## 规则演绎系统

- **基于规则的演绎推理**是一种直接的推理方法，它不像消解反演把知识转化为子句集，而是把有关问题的知识和信息划分为**规则**和**事实**两种类型。
- **规则**由**包含蕴含形式**的表达式表示，**事实**由**无蕴含形式**的表达式表示，并画出相应的与或图，然后通过规则进行演绎推理。
- 规则演绎系统可以分为**规则正向演绎推理**、**规则逆向演绎系统**和**规则双向演绎系统**。

# 规则演绎系统

基于规则的问题求解系统运用下述规则来建立：

- If→Then

其中，If部分可能由几个if组成，而Then部分可能由一个或一个以上的then组成。

在这种系统中，通常称每个if部分为**前项** (antecedent)，称每个then部分为**后项** (consequent)。

# 规则正向演绎系统

- 1. 定义
- **规则正向演绎系统**是从事实到目标进行操作的，即从状况条件到动作进行推理的，也就是从if到then的方向进行推理的。
- 2. 正向推理过程
- (1) 事实表达式的与或形变换
- 把事实表示为**非蕴涵形式的与或形**，作为系统的总数据库。具体变换步骤与前述化为子句形类似。
- 注意：我们不把这些事实化为子句形，而是把它们表示为谓词演算公式，并把这些公式变换为**非蕴涵形式的与或形**。

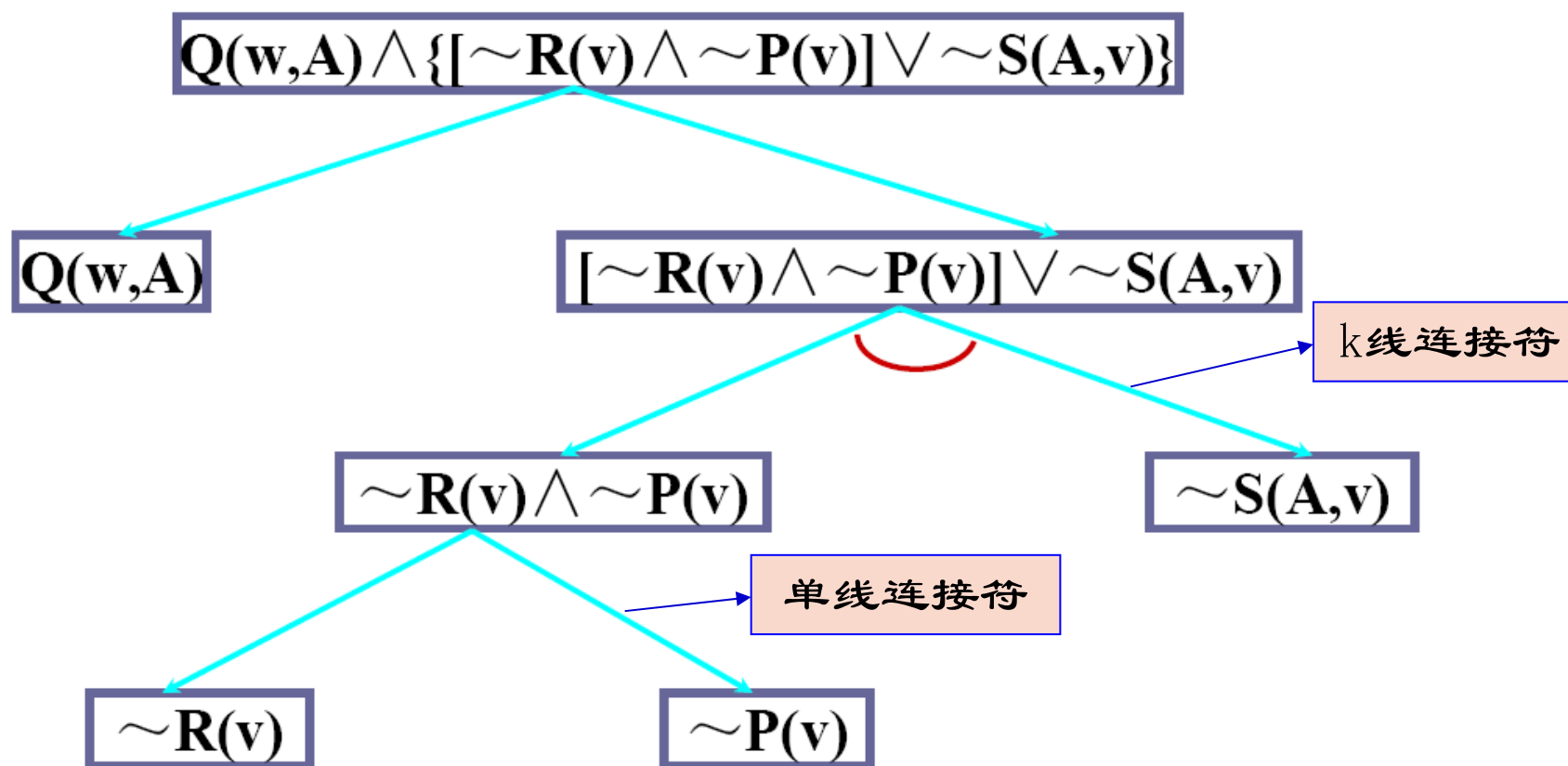
- **与或形表达式**是由符号 $\wedge$ 和 $\vee$ 连接的一些文字的子表达式组成的。呈与或形的表达式**并不是子句形**，与子句集比起来，与或形更多的保留了公式的原始形式。

## 把事实表达式变换为与或形

- 事实表达式
- $(\exists u) (\forall v) \{Q(v, u) \wedge \sim [(R(v) \vee P(v)) \wedge S(u, v)]\}$
- 把它化为
- $Q(v, A) \wedge \{[\sim R(v) \wedge \sim P(v)] \vee \sim S(A, v)\}$
- 对变量更名标准化，使得同一变量不出现在事实表达式的不同**主要合取式**中。更名后得表达式：
- $Q(w, A) \wedge \{[\sim R(v) \wedge \sim P(v)] \vee \sim S(A, v)\}$
- 注意： $Q(v, A)$  中的变量  $v$  可用新变量  $w$  代替，而合取式  $[\sim R(v) \wedge \sim P(v)]$  中的变量  $v$  却不可更名，因为后者也出现在析取式  $\sim S(A, v)$  中。

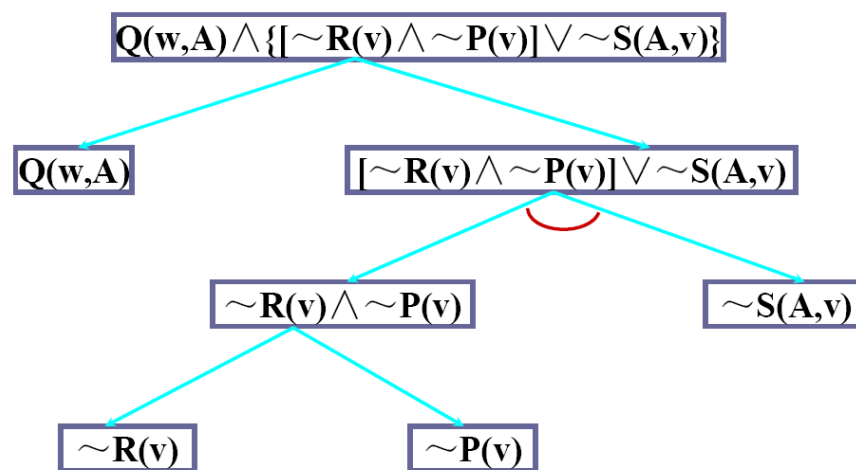


## (2) 事实表达式的与或图表示



## (2) 事实表达式的与或图表示

公式的与或图表示有个有趣的性质，即由变换该公式得到的子句集可作为此与或图的解图的集合（终止于叶节点）读出；也就是说，所得到的每个**子句**是作为解图的各个叶节点上**文字的析取**。



这样，由表达式

$$Q(w, A) \wedge \{ [\neg R(v) \wedge \neg P(v)] \vee \neg S(A, v) \}$$

得到的子句为

$$Q(w, A), \neg S(A, v) \vee \neg R(v), \neg S(A, v) \vee \neg P(v)$$

我们一般把**事实表达式的与或图**表示倒过来画，即把**根节点画在最下面**，而把其后继节点往上画。

# 规则正向演绎系统

- □ (3) 与或图的F规则变换
- 这些规则是建立在某个问题辖域中普通陈述性知识的蕴涵公式基础上的。我们把允许用作规则的公式类型限制为下列形式：
$$L \Rightarrow W$$
- 式中：L是单文字；W为与或形的公式。

### (3) 与或图的F规则变换

- **公式**  $(\forall x) \{ [(\exists y) (\forall z) P(x, y, z)] \Rightarrow (\forall u) Q(x, u) \}$   
可以通过下列步骤加以变换：

- (1) **暂时消去蕴涵符号**

$$(\forall x) \{ \sim [(\exists y) (\forall z) P(x, y, z)] \vee (\forall u) Q(x, u) \}$$

- (2) **把否定符号移进第一个析取式内，调换变量的量词**  
 $(\forall x) \{ (\forall y) (\exists z) [\sim P(x, y, z)] \vee (\forall u) Q(x, u) \}$

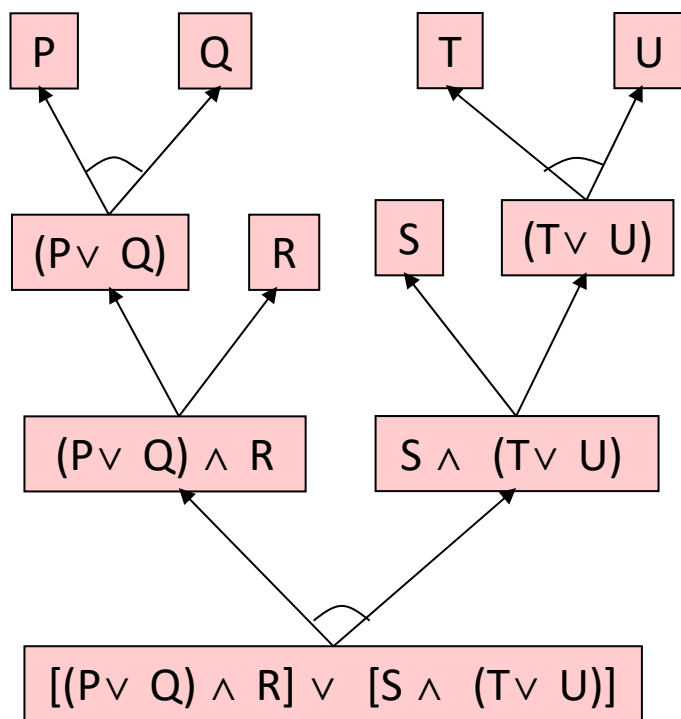
- (3) **进行Skolem化**

$$(\forall x) \{ (\forall y) [\sim P(x, y, f(x, y))] \vee (\forall u) Q(x, u) \}$$

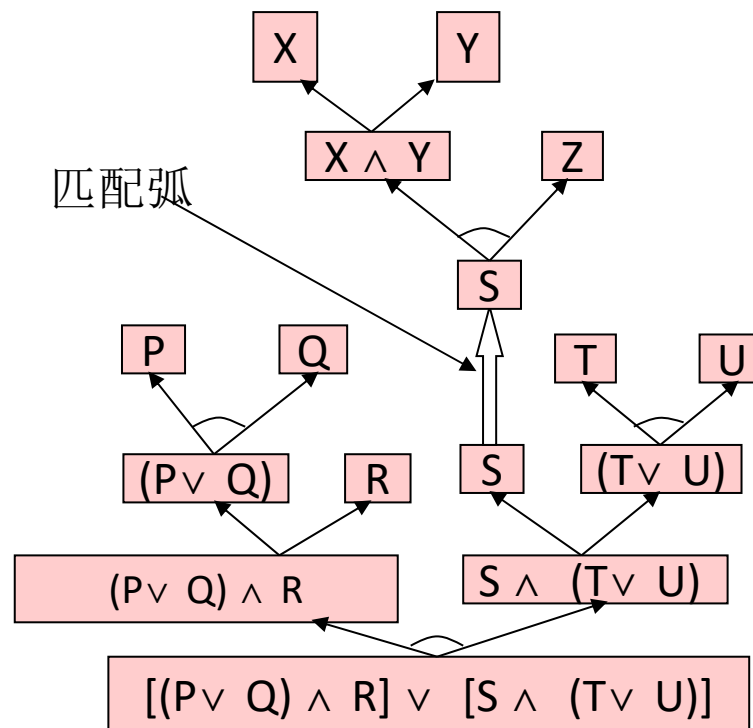
- (4) **把所有全称量词移至前面然后消去**

$$\sim P(x, y, f(x, y)) \vee Q(x, u)$$

- (5) **恢复蕴涵式**  $P(x, y, f(x, y)) \Rightarrow Q(x, u)$



不含变量的与或图



应用一条 $L \Rightarrow W$ 规则得到的与或图

把形式为 $L \Rightarrow W$ 的规则应用到任一个具有叶节点 $n$ 并由文字 $L$ 标记的与或图上，可以得到一个新的与或图。在新的图上，节点 $n$ 由一个单线连接符接到后继节点（也由 $L$ 标记），它是表示为 $W$ 的一个与或图结构的根节点。

作为例子，考虑把规则 $S \Rightarrow (X \wedge Y) \vee Z$ 应用到左图所示的与或图中标有 $S$ 的叶节点上。所得到的新与或图结构表示于右图，图中标记 $S$ 的两个节点由一条叫做匹配弧的弧线连接起来。

- **规则**  $S \Rightarrow [(X \wedge Y) \vee Z]$  的子句形是：

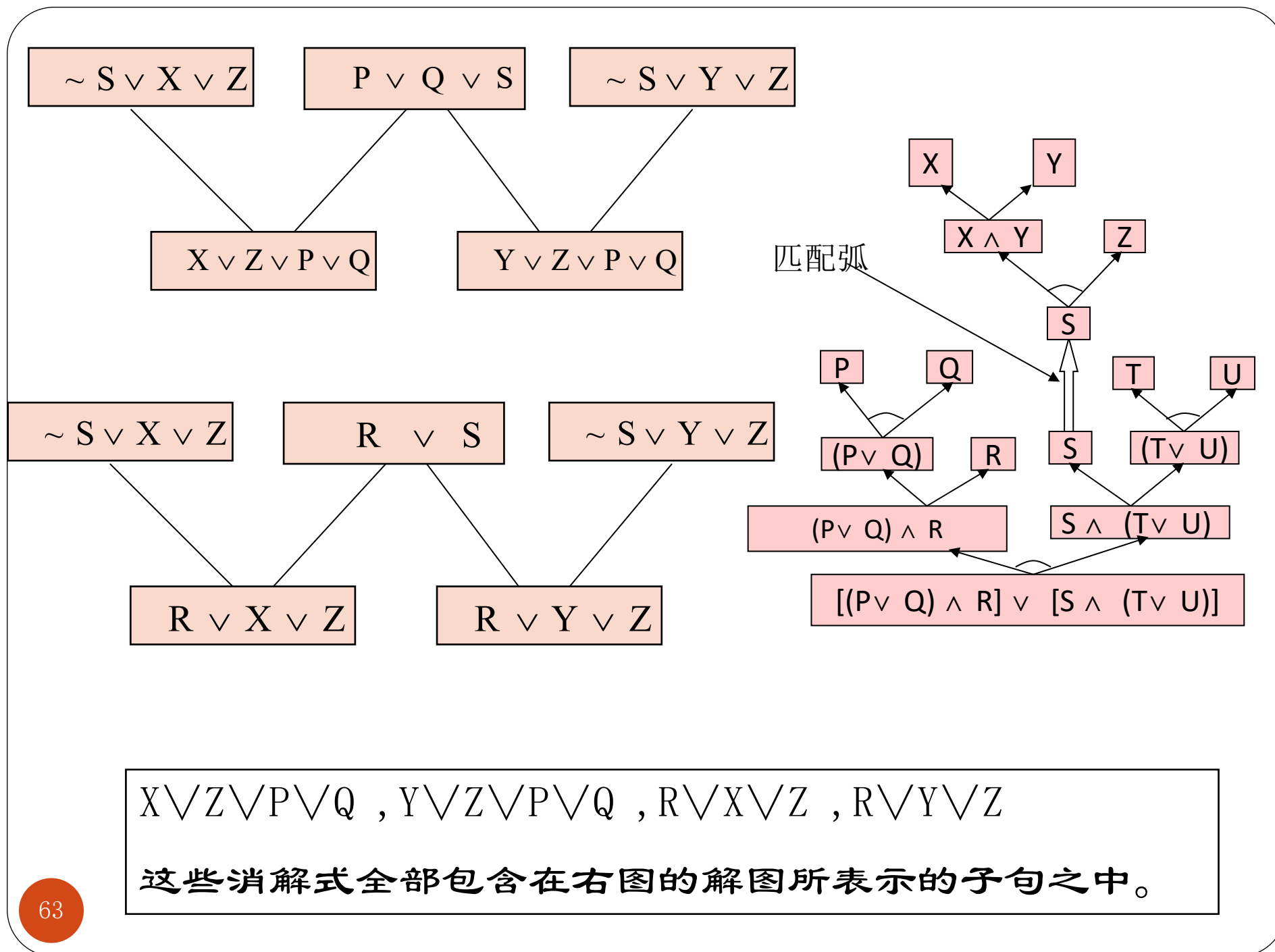
$$\sim S \vee X \vee Z, \quad \sim S \vee Y \vee Z$$

**事实表达式**  $[(P \vee Q) \wedge R] \vee [S \wedge (T \vee U)]$  的子句形解图集为：

$$P \vee Q \vee S, \quad R \vee S, \quad P \vee Q \vee T \vee U, \quad R \vee T \vee U$$

应用两个规则子句中任一个对上述子句形中的S进行消解：于是我们得到4个子句对S进行消解的消解式的完备集为：

$$X \vee Z \vee P \vee Q, \quad Y \vee Z \vee P \vee Q, \quad R \vee X \vee Z, \quad R \vee Y \vee Z$$



- 从上述讨论我们可以得出结论：**应用一条规则到与或图的过程，以极其经济的方式达到了用其它方法要进行多次消解才能达到的目的。**



## (4) 作为终止条件的目标公式

- 基于规则的正向演绎推理的基本原理是：应用F规则作用于表示事实的与或图，改变与或图的结构，从而产生新的事实，直到推出目标公式，则推理成功结束。
- 其推理过程为：
  - (1) 首先用与或图把已知事实表示出来。
  - (2) 用F规则的**左部**和与或图的**叶节点**进行**匹配**，并将匹配成功的**F规则加入到与或图**中，即利用F规则转换与或图。
  - (3) 重复第(2)步，直到产生一个含有以目标节点作为终止节点的解图为止。
- 若事实表达式、规则和目标表达式中有变量，则在推理中需要用最一般的合一进行变量的**置换**。

## (4) 作为终止条件的目标公式

证明过程如下：

事实： $A \vee B$

规则： $A \Rightarrow C \wedge D, B \Rightarrow E \wedge G$

目标： $C \vee G$

把规则化为子句形，得子句

集：

$\sim A \vee C, \sim A \vee D$

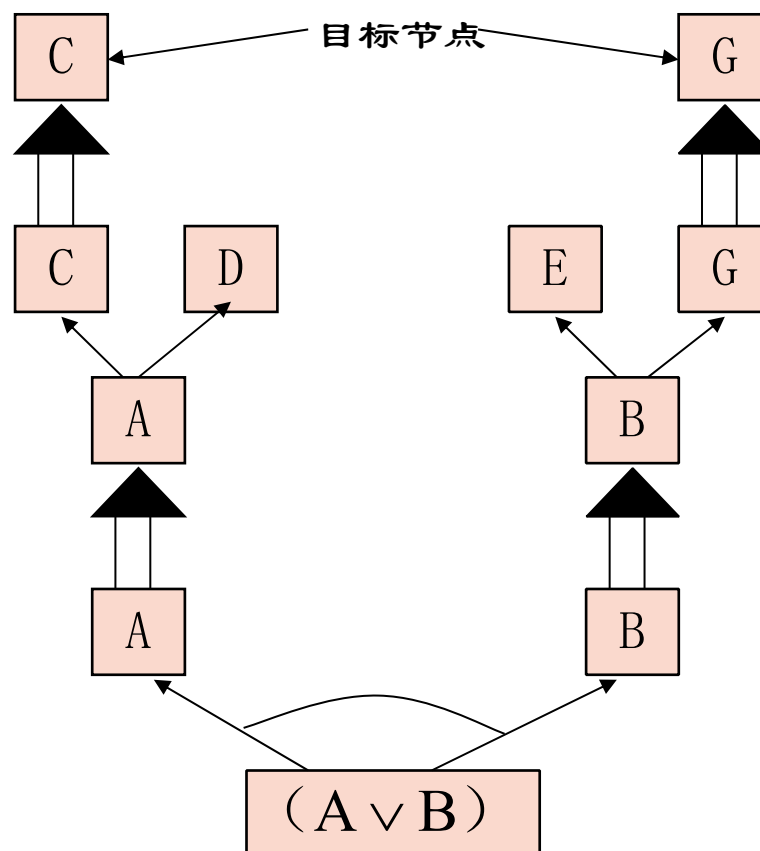
$\sim B \vee E, \sim B \vee G$

目标的否定为：

$\sim (C \vee G)$

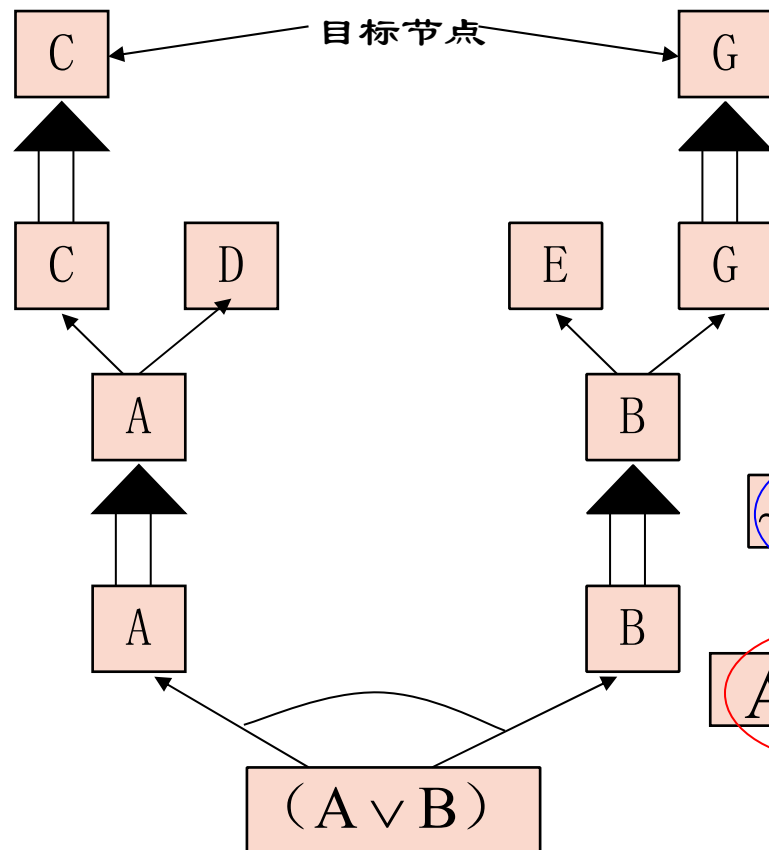
其子句形为：

$\sim C, \sim G$

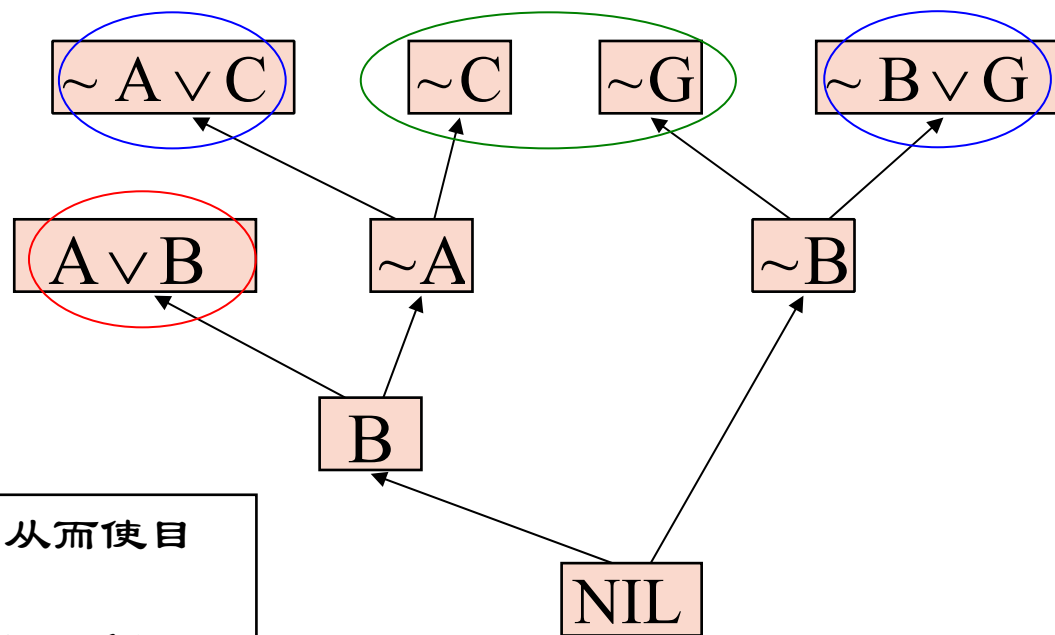


满足终止条件的与或图

正向演绎系统限制目标表达式为  
由文字析取组成的表达式



目标的否定为:  $\sim (C \vee G)$   
 其子句形为:  $\sim C, \sim G$



从右图我们推得一个空子句NIL, 从而使目标公式  $(C \vee G)$  得到证明。

我们得到的结论是: 当正向演绎系统产生一个含有以目标节点作为终止的解图时, 此系统就成功地终止。

用消解反演求证目标公式的图解

# 规则逆向演绎系统

- ◇ 定义
- 规则逆向演绎系统是从then向if进行推理的，即从目标或动作向事实或状况条件进行推理的。
- ◇ 求解过程
  - □ 目标表达式的与或形式
  - □ 与或图的B规则变换
  - □ 作为终止条件的事实节点的一致解图

# 规则逆向演绎系统

- 逆向演绎系统能够处理任意形式的目标表达式。
- 采用和变换事实表达式类似的过程，把目标公式化成与或形：
  - (1) 消去蕴涵符号
  - (2) 把否定符号移到每个谓词符号前面
  - (3) 变量标准化
  - (4) 引入Skolem函数消去全称量词
  - (5) 将公式化为前束形
  - (6) 删去存在量词。留在目标表达式与或形中的变量假定都已被存在量词量化。
  - (7) 重新命名变量，使同一变量不出现在不同的主要析取式中。

用对偶形式对目标表达式进行变换

## 把目标公式化成与或形

- 举例如下：

**目标表达式**

$$(\exists y) (\forall x) \{P(x) \Rightarrow [Q(x, y) \wedge \sim [R(x) \wedge S(y)]]\}$$

**被化成与或形：**

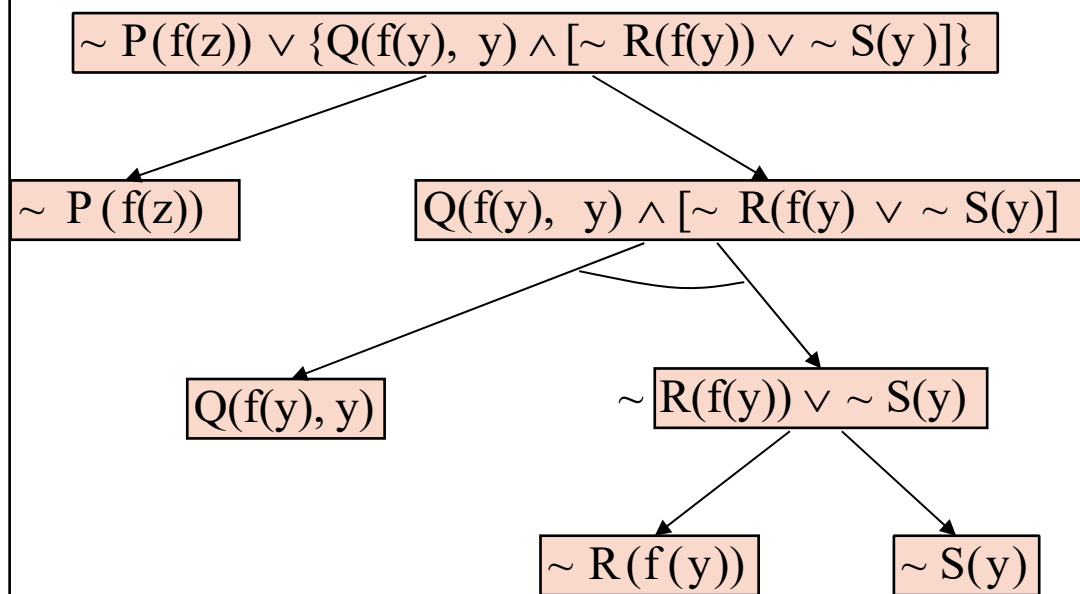
$$\sim P(f(y)) \vee \{Q(f(y), y) \wedge [\sim R(f(y)) \vee \sim S(y)]\}$$

**式中， $f(y)$  为一Skolem函数。**

- **对目标的主要析取式中的变量分离标准化可得：**
- $\sim P(f(z)) \vee \{Q(f(y), y) \wedge [\sim R(f(y)) \vee \sim S(y)]\}$

## 把目标公式化成与或形

与或形的目标公式也可以表示为与或图。不过，与事实表达式的与或图不同的是，对于目标表达式，与或图中的k线连接符用来分开合取关系的子表达式。上例所用的目标公式的与或图如右图所示。在目标公式的与或图中，我们把根节点的任一后裔叫做子目标节点，而标在这些后裔节点中的表达式叫做子目标。



这个目标公式的子句形表示中的子句集可从终止在叶节点上的解图集读出： $\sim P(f(z))$ ,  $Q(f(y), y) \wedge \sim R(f(y))$ ,  $Q(f(y), y) \wedge \sim S(y)$

可见目标子句是文字的合取，而这些子句的析取是目标公式的子句形（析取范式）。

## 与或图的B规则变换

- 现在让我们应用B规则即逆向推理规则来变换逆向演绎系统的与或图结构。
- 这个B规则是建立在确定的蕴涵式基础上的，正如正向系统的F规则一样。不过，我们现在把这些B规则限制为： $W \Rightarrow L$ 形式的表达式。
- 其中，**W为任一与或形公式**，**L为文字**，而且蕴涵式中任何变量的量词辖域为整个蕴涵式。其次，把B规则限制为这种形式的蕴涵式还可以简化匹配，使之不会引起重大的实际困难。
- 此外，可以把像 $W \Rightarrow (L1 \wedge L2)$ 这样的蕴涵式化为两个规则 $W \Rightarrow L1$ 和 $W \Rightarrow L2$ 。



## 作为终止条件的事实节点的一致解图

- 逆向系统中的**事实表达式均限制为文字合取形**，它可以表示为一个文字集。当一个事实文字和标在该图文字节点上的文字相匹配时，就可把相应的后裔事实节点添加到该与或图中去。这个事实节点通过标有mgu的匹配弧与匹配的子目标文字节点连接起来。同一个事实文字可以多次重复使用（每次用不同变量），以便建立多重事实节点。

逆向系统成功的终止条件是与或图包含有某个终止在事实节点上的一致解图。

## 举例：

- 这个例子的事实、应用规则和问题分别表示于下：

事实：

F1: DOG(FIDO); 狗的名字叫 Fido  
F2:  $\sim$ BARKS(FIDO); Fido是不叫的  
F3: WAGS-TAIL(FIDO); Fido摇尾巴  
F4: MEOWS(MYRTLE); 猫咪的名字叫Myrtle

规则：

R1:  $[WAGS-TAIL(x1) \wedge DOG(x1)] \Rightarrow FRIENDLY(x1)$ ;

- 摇尾巴的狗是温顺的狗

R2:  $[FRIENDLY(x2) \wedge \sim BARKS(x2)] \Rightarrow \sim AFRAID(y2, x2)$ ;

- 温顺而又不叫的东西是不值得害怕的

R3:  $DOG(x3) \Rightarrow ANIMAL(x3)$ ; 狗为动物

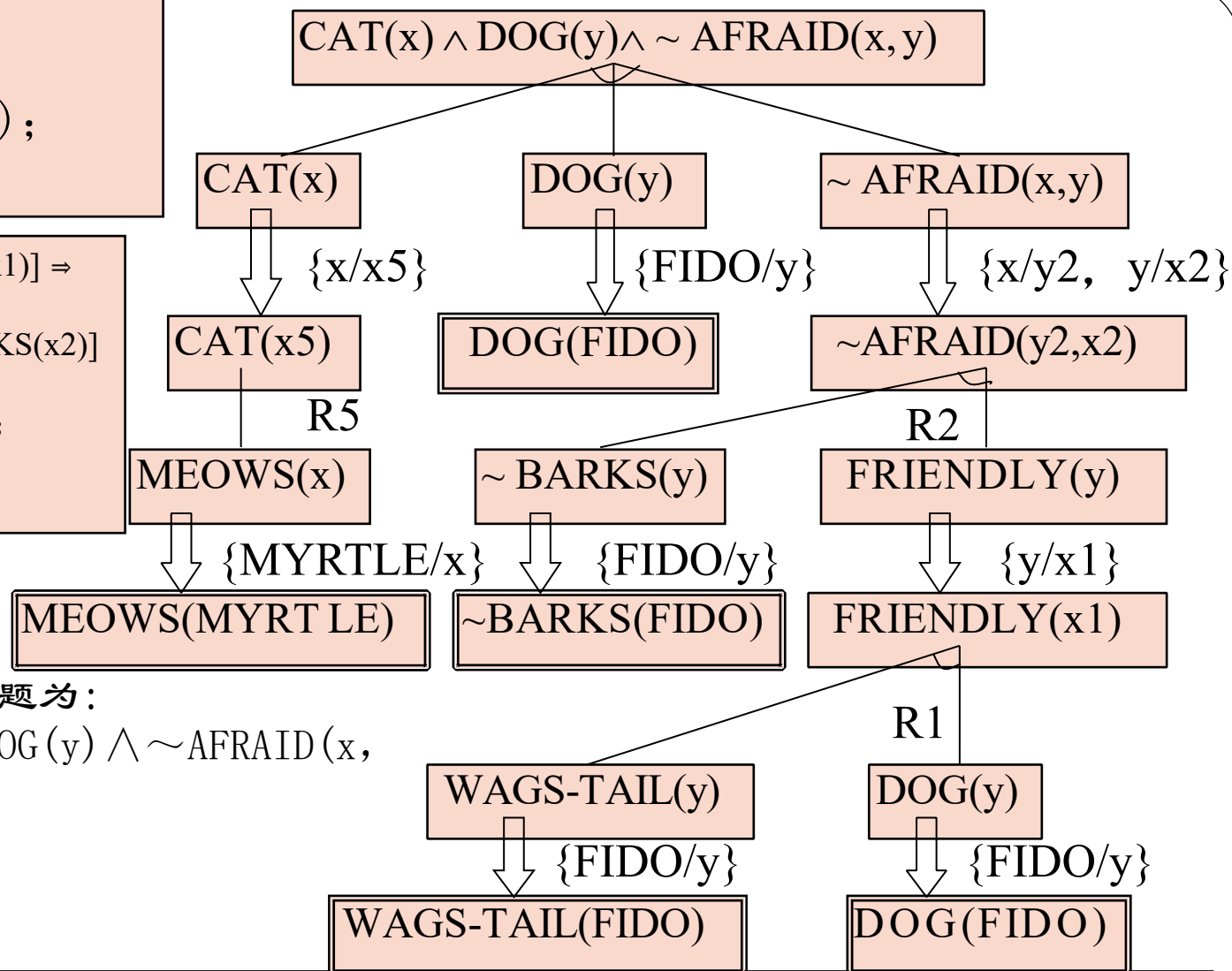
R4:  $CAT(x4) \Rightarrow ANIMAL(x4)$ ; 猫为动物

R5:  $MEOWS(x5) \Rightarrow CAT(x5)$ ; 猫咪是猫

问题：是否存在这样的一只猫和一条狗，使得这只猫不怕这条狗？

F1: DOG (FIDO) ;  
 F2:  $\sim$ BARKS (FIDO) ;  
 F3: WAGS-TAIL (FIDO) ;  
 F4: MEOWS (MYRTLE) ;

R1:  $[WAGS-TAIL(x_1) \wedge DOG(x_1)] \Rightarrow FRIENDLY(x_1)$ ;  
 R2:  $[FRIENDLY(x_2) \wedge \sim BARKS(x_2)] \Rightarrow \sim AFRAID(y_2, x_2)$ ;  
 R3:  $DOG(x_3) \Rightarrow ANIMAL(x_3)$ ;  
 R4:  $CAT(x_4) \Rightarrow ANIMAL(x_4)$ ;  
 R5:  $MEOWS(x_5) \Rightarrow CAT(x_5)$ ;



用目标表达式表示此问题为:

$(\square x) (\square y) [CAT(x) \wedge DOG(y) \wedge \sim AFRAID(x, y)]$

图中，用双线框表示事实节点，用规则编号R1、R2和R5等来标记所应用的规则。此解图中有八条匹配弧，每条匹配弧上都有一个置换。这些置换为  $\{x/x_5\}$ ， $\{MYRTLE/x\}$ ， $\{FIDO/y\}$ ， $\{x/y_2, y/x_2\}$ ， $\{FIDO/y\}$ 。由图可见，终止在事实节点前的置换为  $\{MYRTLE/x\}$  和  $\{FIDO/y\}$ 。把它应用到目标表达式，我们就得到该问题的回答语句如下：

$[CAT(MYRTLE) \wedge DOG(FIDO) \wedge \sim AFRAID(MYRTLE, FIDO)]$

# 规则双向演绎系统

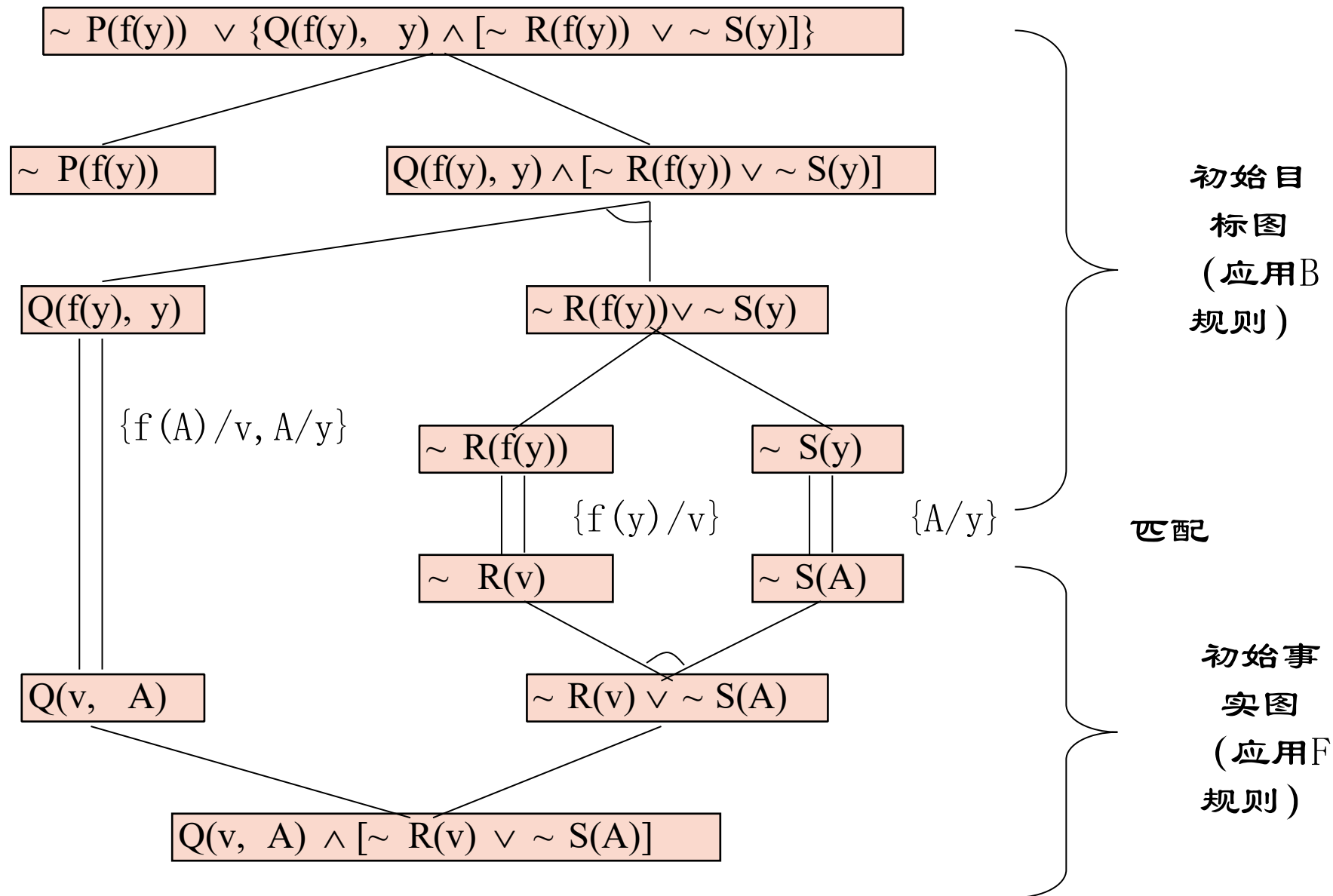
## 1. 基于规则的正向演绎系统和逆向演绎系统的特点和局限性

**正向演绎系统**能够处理任意形式的if表达式，但被限制在**then表达式**为由**文字析取**组成的一些表达式。**逆向演绎系统**能够处理任意形式的**then表达式**，但被限制在**if表达式**为**文字合取**组成的一些表达式。双向（正向和逆向）组合演绎系统具有正向和逆向两系统的优点，克服各自的缺点。

## 2. 双向（正向和逆向）组合演绎系统的构成

正向和逆向组合系统是建立在两个系统相结合的基础上的。此组合系统的总数据库由表示目标和表示事实的两个与或图结构组成，并分别用F规则和B规则来修正。

- **双向演绎系统的主要复杂之处在于其终止条件，终止涉及两个图结构之间的适当交接处。这些结构可由标有合一文字的节点上的匹配棱线来连接。**



双向演绎系统举例

# 产生式系统

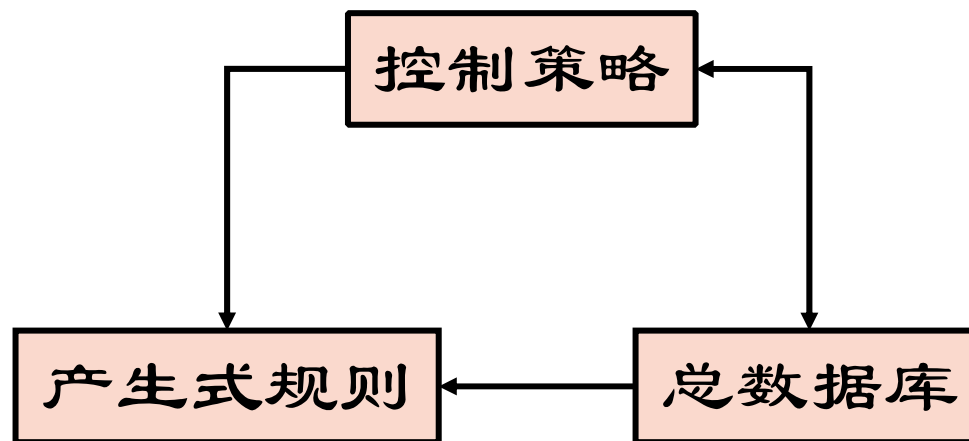
- **定义**

- 用来描述若干个不同的以一个基本概念为基础的系统。这个基本概念就是**产生式规则或产生式条件和操作对**的概念。

- **实质**

- 在产生式系统中，论域的知识分为两部分：**用事实表示静态知识**，如事物、事件和它们之间的关系；**用产生式规则表示推理过程和行为**。由于这类系统的知识库主要用于存储规则，因此又把此类系统称为基于规则的系统。

# 产生式系统的组成



产生式系统的主要组成



# 产生式系统的组成

- **(1) 产生式规则**是一个规则库，用于存放与求解问题有关的某个领域知识的规则之集合及其交换规则。规则库知识的完整性、一致性、准确性、灵活性和知识组织的合理性，将对产生式系统的运行效率和工作性能产生重要影响。
- 规则是一个以“如果满足这个条件，就应当采取某些操作”形式表示的语句。
- 例如， 规则：  
    **如果** 某种动物是哺乳动物，并且吃肉  
    **那么** 这种动物被称为食肉动物

## 产生式系统的组成

- 产生式的IF(如果)被称为**条件、前项或产生式的左边**。它说明**应用这条规则必须满足的条件**；
- THEN(那么)部分被称为**操作、结果、后项或产生式的右边**。
- 在产生式系统的执行过程中，如果某条规则的条件满足了，那么，这条规则就可以被应用；也就是说，系统的控制部分可以执行规则的操作部分。

## 产生式系统的组成

- (2) **总数据库**有时也被称作上下文，当前数据库或暂时存储器，用于存放求解过程中各种当前信息的数据结构。总数据库是产生式规则的注意中心。
- **产生式规则的左边**表示在启用这一规则之前总数据库内必须准备好的条件。
- **执行产生式规则的操作会引起总数据库的变化**，这就使其他产生式规则的条件可能被满足。

## 产生式系统的组成

**(3) 控制策略**为一推理机构，由一组程序组成，用来控制产生式系统的运行，决定问题求解过程的推理线路，实现对问题的求解。

# 控制策略的作用

- 选择规则到执行操作的步骤
- 1. 匹配
- 把当前数据库与规则的条件部分相匹配。
- 2. 冲突
- 当有一条以上规则的条件部分和当前数据库相匹配时，就需要决定首先使用哪一条规则，这称为冲突解决。
- 3. 操作
- 操作就是执行规则的操作部分。

# 产生式系统的推理

- 正向推理

- 从一组表示事实的谓词或命题出发，使用一组产生式规则，用以证明该谓词公式或命题是否成立。

- ◇实现正向推理的一般策略是：

- 先提供一批数据(事实)到总数据库中，系统利用这些事实与规则的前提匹配，触发匹配成功的规则(即启用规则)，把其结论作为新的事实添加到总数据库中。继续上述过程，用更新过的总数据库中的所有事实再与规则库中另一条规则匹配，用其结论再修改总数据库的内容，直到没有可匹配的新规则，不再有新的事实加到总数据库为止。

## 产生式系统的推理

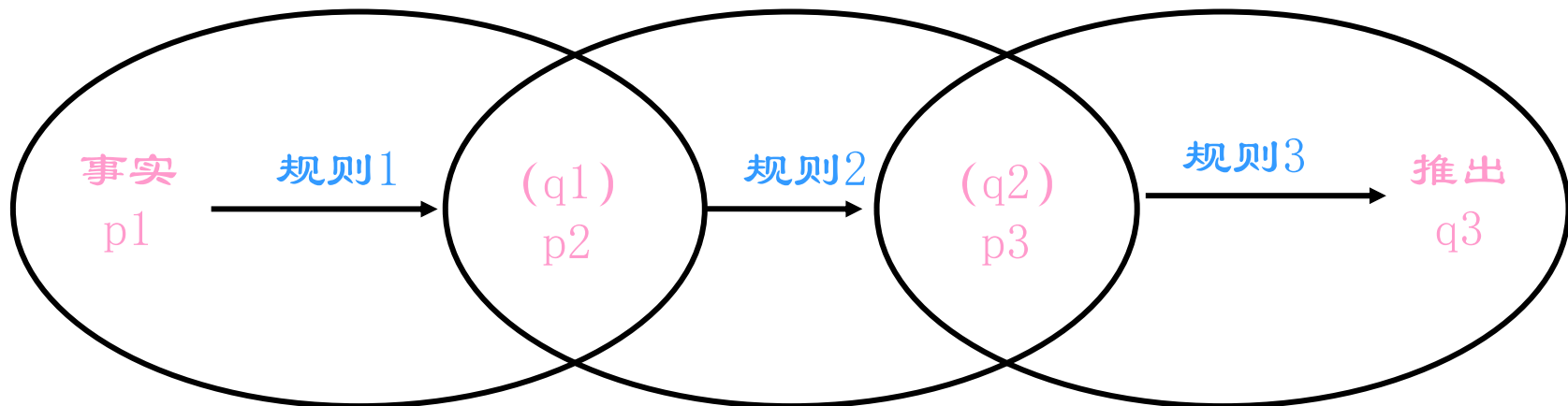
- 例如，有规则集如下：

规则1: IF p1 THEN p2

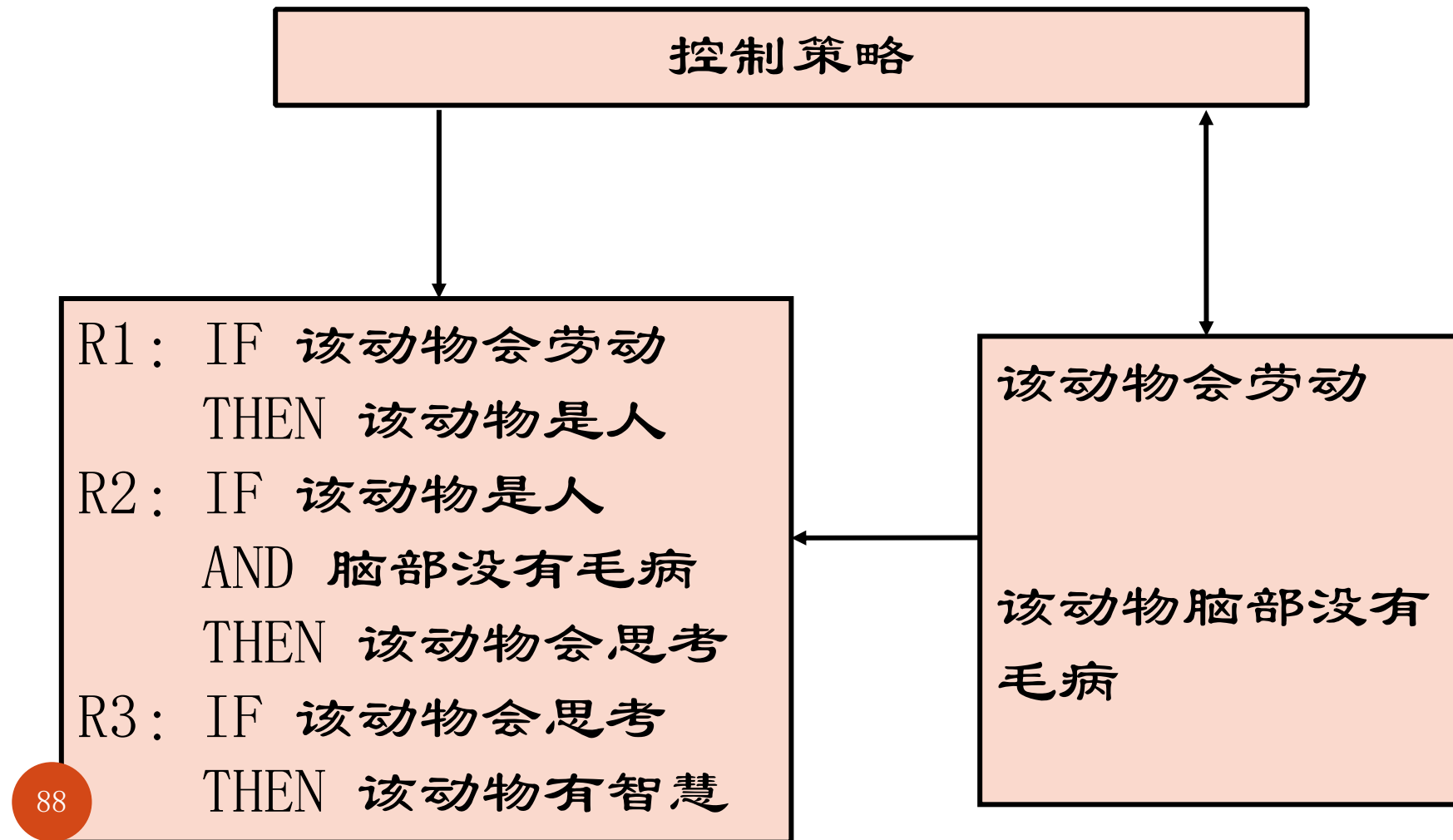
规则2: IF p2 THEN p3

规则3: IF p3 THEN q3

规则中的p1、p2、p3、q3可以是谓词公式或命题。设总数据库（工作存储器）中已有事实p1，则应用这三条规则进行正向推理，即从p1出发推导出q3的过程如图所示。



# 正向推理举例



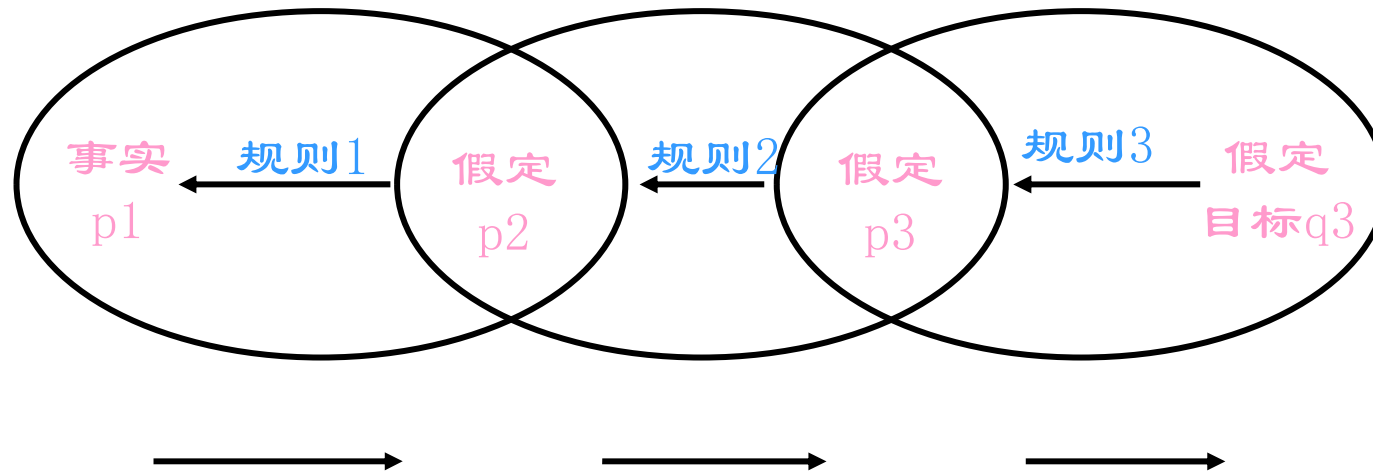


# 产生式系统的推理

- 逆向推理

□ 从表示**目标的谓词或命题**出发，使用一组产生式规则证明事实谓词或命题成立，即首先提出**一批假设目标**，然后逐一验证这些假设。

- 举例如下：仍用上述的三条规则为例，应用**反向推理方法**，从p1出发推导出q3的过程如图所示



首先假定目标q3成立，由规则3 ( $p3 \rightarrow q3$ )，为证明q3成立，须先验证p3是否成立；但总数据库没有事实p3，所以假定子目标p3成立；由规则2( $p2 \rightarrow p3$ )，应验证p2；同样，由于数据库中沒有事实p2，假定子目标p2成立；由规则1( $p1 \rightarrow p2$ )，为验证p2成立，须先验证p1。因为数据库中有事实p1，所以假定的目标p2成立，因而p3成立，最终导出结论q3确实成立。

## 双向推理

- ☐ 双向推理的推理策略是同时从目标向事实推理和从事实向目标推理，并在推理过程中的某个步骤，实现事实与目标的匹配。

# 本章小结

- 经典推理技术
- ☐ 消解反演
- 高级搜索推理技术
- ☐ 规则演绎系统
- ☐ 产生式系统



## 思考题

- 1. 求取子句集应遵循哪些步骤, 试结合例子加以应用.
- 2. 如何通过消解反演求取问题答案?
- 3. 规则演绎系统与产生式系统有哪几种推理方式? 各自有何特点.
- 4. 什么是产生式系统? 它由那些部分组成? 产生式系统如何进行推理?