

# Artificial Intelligence 人工智能

## 第2章 知识表示和推理

### 谓词逻辑、归结推理

# 知识表示和推理

- 1 概述
- 2 命题逻辑
- 3 谓词逻辑
- 4 归结推理

# 谓词逻辑

✓ “ Robot A is to the right of robot B”

✓  $\forall u \forall v \text{ is\_further\_right}(u, v) \Leftrightarrow$

$$\begin{aligned} \exists x_u \exists y_u \exists x_v \exists y_v & \text{Position}(u, x_u, y_u) \wedge \text{Position}(v, x_v, y_v) \\ & \wedge \text{Larger}(x_u, x_v) \end{aligned}$$

• Typically,  $\Rightarrow$  is the main connective with  $\forall$ ;

$\wedge$  is the main connective with  $\exists$

◦  $\forall x \text{ At}(x, \text{SYSU}) \Rightarrow \text{Smart}(x)$

◦  $\exists x \text{ At}(x, \text{SYSU}) \wedge \text{Smart}(x)$

• **Morgan's law**

◦  $\forall x L \equiv \neg \exists x \neg L$

◦  $\neg(\forall x L) \equiv \exists x \neg L$

“Not everyone likes cat”

$$\neg(\forall x, \text{Likes}(x, \text{cat}))$$

$$\exists x, \neg \text{Likes}(x, \text{cat})$$

# 字母表

Logical symbols (fixed meaning and use):

- Punctuation:  $(, ), \dots$
- Connectives and quantifiers:  $=, \neg, \wedge, \vee, \forall, \exists$
- Variables:  $x, x_1, x_2, \dots, x', x'', \dots, y, \dots, z, \dots$

Non-logical symbols (domain-dependent meaning and use):

- Predicate symbols
  - arity: number of arguments
  - arity 0 predicates: propositional symbols
- Function symbols
  - arity 0 functions: constant symbols

# 项和公式

- Every variable is a term
- If  $t_1, \dots, t_n$  are terms and  $f$  is a function symbol of arity  $n$ , then  $f(t_1, \dots, t_n)$  is a term
- If  $t_1, \dots, t_n$  are terms and  $P$  is a predicate symbol of arity  $n$ , then  $P(t_1, \dots, t_n)$  is an atomic formula
- If  $t_1$  and  $t_2$  are terms, then  $(t_1 = t_2)$  is an atomic formula
- If  $\alpha$  and  $\beta$  are formulas, and  $v$  is a variable, then  $\neg\alpha, (\alpha \wedge \beta), (\alpha \vee \beta), \exists v.\alpha, \forall v.\alpha$  are formulas

# 谓词逻辑的应用

例1 “某些患者喜欢所有医生。没有患者喜欢庸医。所以没有医生是庸医。”

解：P(x)表示“x是患者”，

D(x)表示“x是医生”，

Q(x)表示“x是庸医”，

L(x, y)表示“x喜欢y”。

目的是证明G是F1和F2的逻辑结论。

# 谓词逻辑的应用

例1 “某些患者喜欢所有医生。没有患者喜欢庸医。所以没有医生是庸医。”

解：P(x)表示“x是患者”，

D(x)表示“x是医生”，

Q(x)表示“x是庸医”，

L(x, y)表示“x喜欢y”。

$$F_1 \quad (\exists x)(P(x) \wedge (\forall y)(D(y) \rightarrow L(x, y)))$$

$$F_2 : (\forall x)(P(x) \rightarrow (\forall y)(Q(y) \rightarrow \neg L(x, y)))$$

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

目的是证明G是F1和F2的逻辑结论。

# 记法

- Occasionally add or omit  $(,)$
- Use  $[,]$  and  $\{, \}$
- Abbreviation:  $(\alpha \rightarrow \beta)$  for  $(\neg\alpha \vee \beta)$
- Abbreviation:  $(\alpha \leftrightarrow \beta)$  for  $(\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$
- Predicates: mixed case capitalized, e.g., Person, OlderThan
- Functions (and constants): mixed case uncapitalized, e.g., john, father,



# 变量范围

- Free and bound occurrences of variables
- *e.g.*,  $P(x) \wedge \exists x[P(x) \vee Q(x)]$
- A sentence: formula with no free variables
- Substitution:  $\alpha[v/t]$  means  $\alpha$  with all free occurrences of the  $v$  replaced by term  $t$
- In general,  $\alpha[v_1/t_1, \dots, v_n/t_n]$

# 解释

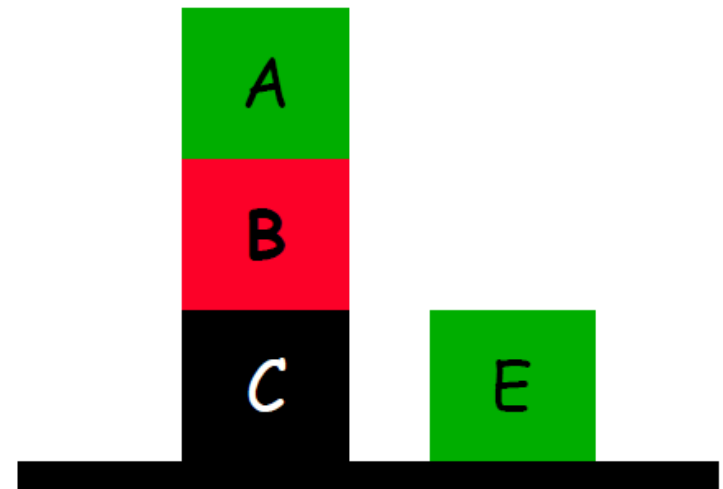
An interpretation is a pair  $\mathfrak{I} = \langle D, I \rangle$

- $D$  is the domain, can be any non-empty set
- $I$  is a mapping from the set of predicate and function symbols
- If  $P$  is a predicate symbol of arity  $n$ ,  $I(P)$  is an  $n$ -ary relation over  $D$ , i.e.,  $I(P) \subseteq D^n$ 
  - If  $p$  is a 0-ary predicate symbol, i.e., a propositional symbol,  $I(p) \in \{true, false\}$
- If  $f$  is a function symbol of arity  $n$ ,  $I(f)$  is an  $n$ -ary function over  $D$ , i.e.,  $I(f) : D^n \rightarrow D$ 
  - If  $c$  is a 0-ary function symbol, i.e., a constant symbol,  $I(c) \in D$

# 积木世界例子

- $D = \{\underline{A}, \underline{B}, \underline{C}, \underline{E}\}$
- $\Phi(a) = \underline{A}, \Phi(b) = \underline{B},$   
 $\Phi(c) = \underline{C}, \Phi(e) = \underline{E}.$
- $\Psi(\text{on}) = \{(\underline{A}, \underline{B}), (\underline{B}, \underline{C})\}$
- $\Psi(\text{above}) =$   
 $\{(\underline{A}, \underline{B}), (\underline{B}, \underline{C}), (\underline{A}, \underline{C})\}$
- $\Psi(\text{clear}) = \{\underline{A}, \underline{E}\}$
- $\Psi(\text{ontable}) = \{\underline{C}, \underline{E}\}$

- Constants: a,b,c,e
- Functions:
  - No function
- Predicates:
  - on: binary
  - above: binary
  - clear: unary
  - ontable: unary



# 项的指称（指派）

- Terms denote elements of the domain
- A variable assignment  $\mu$  is a mapping from the set of variables to the domain  $D$
- $\|v\|_{\mathfrak{S}, \mu} = \mu(v)$
- $\|f(t_1, \dots, t_n)\|_{\mathfrak{S}, \mu} = I(f)(\|t_1\|_{\mathfrak{S}, \mu}, \dots, \|t_n\|_{\mathfrak{S}, \mu})$

# 满足： 原子公式

$\mathfrak{S}, \mu \models \alpha$  is read “ $\mathfrak{S}, \mu$  satisfies  $\alpha$ ”

- $\mathfrak{S}, \mu \models P(t_1, \dots, t_n)$  iff  $\langle \|t_1\|_{\mathfrak{S}, \mu}, \dots, \|t_n\|_{\mathfrak{S}, \mu} \rangle \in I(P)$
- $\mathfrak{S}, \mu \models (t_1 = t_2)$  iff  $\|t_1\|_{\mathfrak{S}, \mu} = \|t_2\|_{\mathfrak{S}, \mu}$

# 满足：联结词

- $\mathfrak{S}, \mu \models \neg\alpha$  iff  $\mathfrak{S}, \mu \not\models \alpha$
- $\mathfrak{S}, \mu \models (\alpha \wedge \beta)$  iff  $\mathfrak{S}, \mu \models \alpha$  and  $\mathfrak{S}, \mu \models \beta$
- $\mathfrak{S}, \mu \models (\alpha \vee \beta)$  iff  $\mathfrak{S}, \mu \models \alpha$  or  $\mathfrak{S}, \mu \models \beta$

# 满足：量词

$\mu\{d; v\}$  denotes a variable assignment just like  $\mu$ , except that it maps  $v$  to  $d$

- $\mathfrak{S}, \mu \models \exists v. \alpha$  iff for some  $d \in D$ ,  $\mathfrak{S}, \mu\{d; v\} \models \alpha$
- $\mathfrak{S}, \mu \models \forall v. \alpha$  iff for all  $d \in D$ ,  $\mathfrak{S}, \mu\{d; v\} \models \alpha$

Let  $\alpha$  be a sentence. Then whether  $\mathfrak{S}, \mu \models \alpha$  is independent of  $\mu$ .  
Thus we simply write  $\mathfrak{S} \models \alpha$

# 积木世界例子

- $D = \{\underline{A}, \underline{B}, \underline{C}, \underline{E}\}$
- $\Phi(a) = \underline{A}, \Phi(b) = \underline{B},$   
 $\Phi(c) = \underline{C}, \Phi(e) = \underline{E}.$
- $\Psi(\text{on}) = \{(\underline{A}, \underline{B}), (\underline{B}, \underline{C})\}$
- $\Psi(\text{above}) =$   
 $\{(\underline{A}, \underline{B}), (\underline{B}, \underline{C}), (\underline{A}, \underline{C})\}$
- $\Psi(\text{clear}) = \{\underline{A}, \underline{E}\}$
- $\Psi(\text{ontable}) = \{\underline{C}, \underline{E}\}$

$\forall X, Y. \text{on}(X, Y) \rightarrow \text{above}(X, Y)$

✓  $X = \underline{A}, Y = \underline{B}$

✓  $X = \underline{C}, Y = \underline{A}$

✓ ...

$\forall X, Y. \text{above}(X, Y) \rightarrow \text{on}(X, Y)$

✓  $X = \underline{A}, Y = \underline{B}$

✗  $X = \underline{A}, Y = \underline{C}$



# 积木世界例子

- $D = \{\underline{A}, \underline{B}, \underline{C}, \underline{E}\}$
- $\Phi(a) = \underline{A}, \Phi(b) = \underline{B},$   
 $\Phi(c) = \underline{C}, \Phi(e) = \underline{E}.$
- $\Psi(\text{on}) = \{(\underline{A}, \underline{B}), (\underline{B}, \underline{C})\}$
- $\Psi(\text{above}) =$   
 $\{(\underline{A}, \underline{B}), (\underline{B}, \underline{C}), (\underline{A}, \underline{C})\}$
- $\Psi(\text{clear}) = \{\underline{A}, \underline{E}\}$
- $\Psi(\text{ontable}) = \{\underline{C}, \underline{E}\}$

$\forall X \exists Y. (\text{clear}(X) \vee \text{on}(Y, X))$

- ✓  $X = \underline{A}$
- ✓  $X = \underline{C}, Y = \underline{B}$
- ✓ ...

$\exists Y \forall X. (\text{clear}(X) \vee \text{on}(Y, X))$

- ✗  $Y = \underline{A} ?$  No! ( $X = \underline{C}$ )
- ✗  $Y = \underline{C} ?$  No! ( $X = \underline{B}$ )
- ✗  $Y = \underline{E} ?$  No! ( $X = \underline{B}$ )
- ✗  $Y = \underline{B} ?$  No! ( $X = \underline{B}$ )

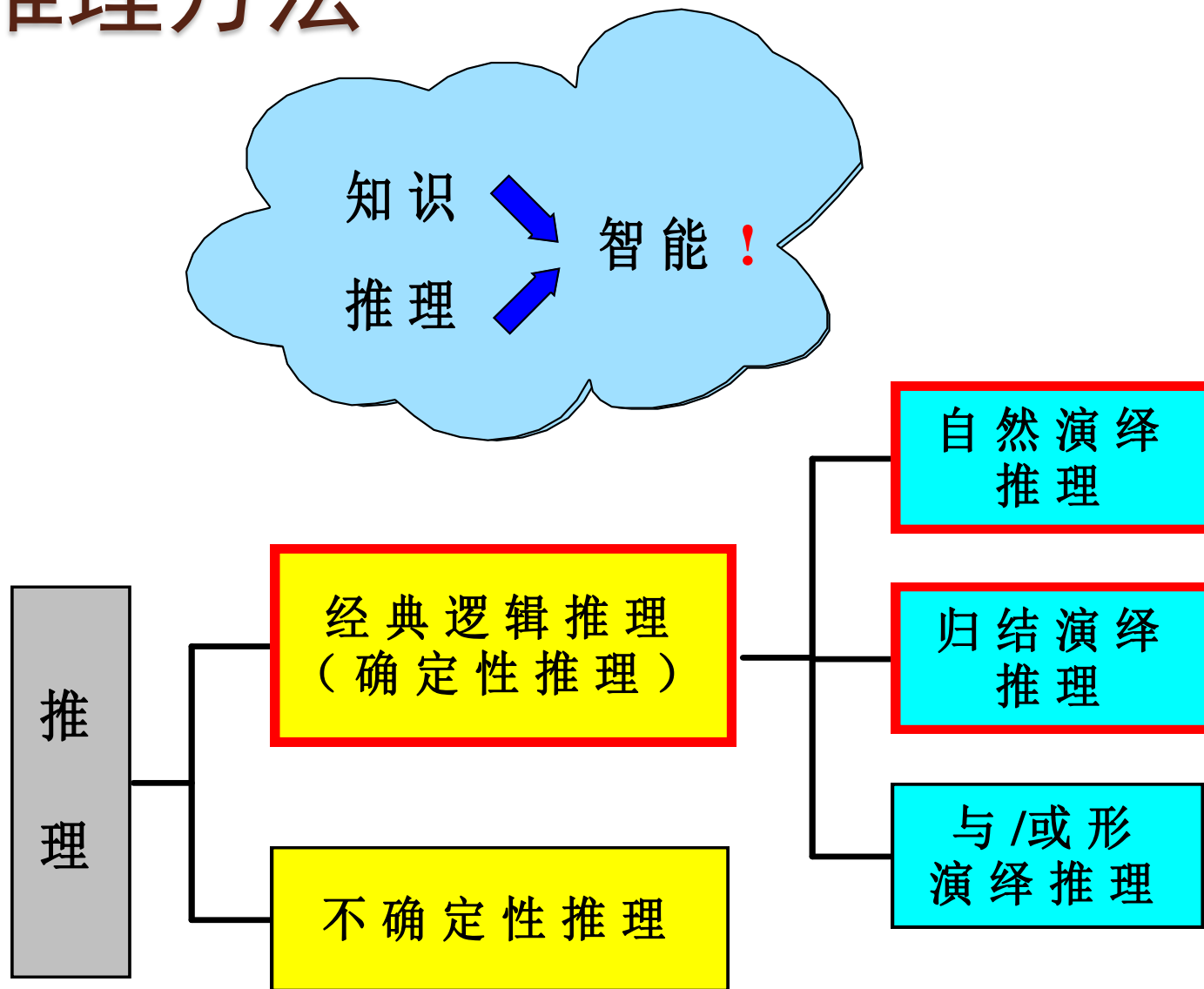
# 可满足性

- Let  $S$  be a set of sentences
- $\mathfrak{S} \models S$ , read  $\mathfrak{S}$  satisfies  $S$ , if for every  $\alpha \in S$ ,  $\mathfrak{S} \models \alpha$
- If  $\mathfrak{S} \models S$ , we say  $\mathfrak{S}$  is a model of  $S$
- We say that  $S$  is satisfiable if there is  $\mathfrak{S}$  s.t.  $\mathfrak{S} \models S$ , and
- e.g., is  $\{\forall x(P(x) \rightarrow Q(x)), P(a), \neg Q(a)\}$  satisfiable?

# 推理方法

- ✿ 前面讨论了把知识用某种模式表示出来存储到计算机中去。但是，为使计算机具有智能，还必须使它具有思维能力。推理是求解问题的一种重要方法。因此，推理方法成为人工智能的一个重要研究课题。
- ✿ 下面首先讨论关于推理的基本概念，然后介绍鲁宾逊归结原理及其在机器定理证明和问题求解中的应用。鲁宾逊归结原理使定理证明能够在计算机上实现。

# 推理方法



# 推理的定义

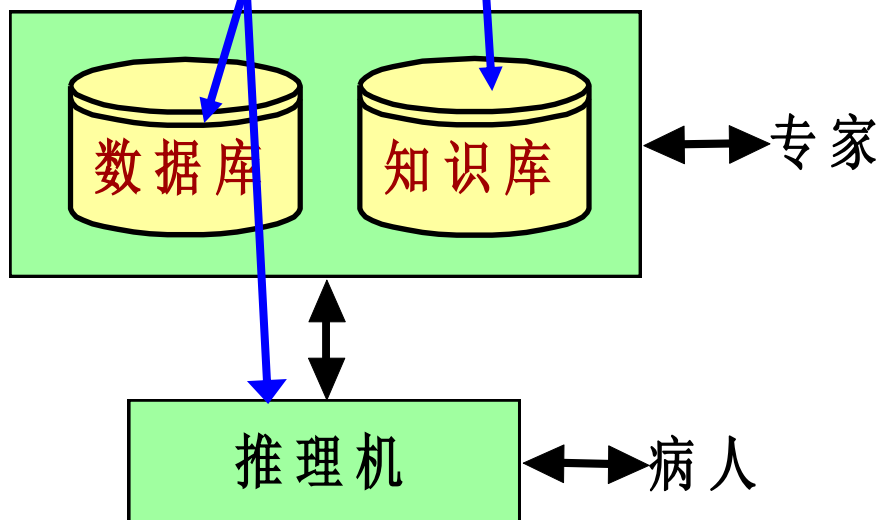
推理：

已知事实  
(证据)

某种策略

结论

知 识



医疗专家系统

知识	专家的经验、医学常识
初始证据	病人的症状、化验结果
证据	中间结论

# 推理方式及其分类

## 1. 演绎推理、归纳推理、默认推理

(1) **演绎推理** (deductive reasoning) : 一般  $\rightarrow$  个别

■ **三段论式** (三段论法)

① 足球运动员的身体都是强壮的 ; ( **大前提** )

② 高波是一名足球运动员 ; ( **小前提** )

---

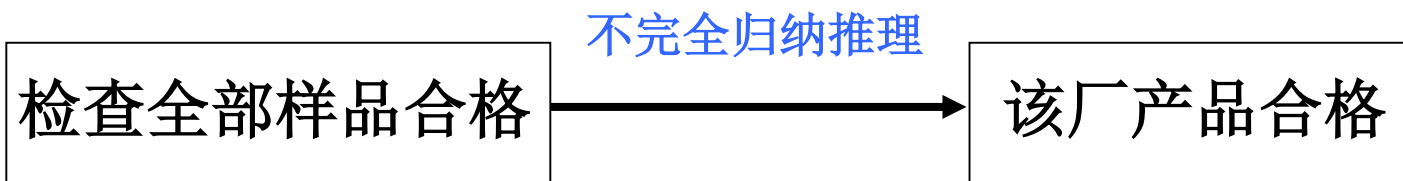
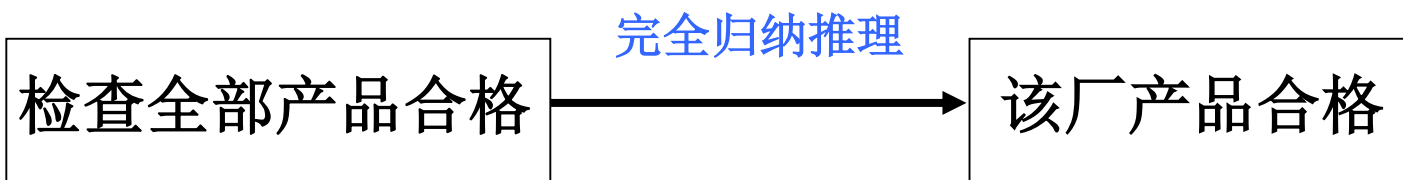
③ 所以, 高波的身体是强壮的。 ( **结 论** )

# 推理方式及其分类

## 1. 演绎推理、归纳推理、默认推理

(2) **归纳推理** (inductive reasoning): 个别 → 一般

{ 完全归纳推理 (必然性推理)  
不完全归纳推理 (非必然性推理)





# 推理方式及其分类

## 1. 演绎推理、归纳推理、默认推理

### (3) 默认推理 (default reasoning, 缺省推理)

- 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。

A 成立  
B 成立?  结 论  
(默认B成立)

制造鸟笼  
鸟会飞?  鸟笼要  
有盖子  
(默认成立)

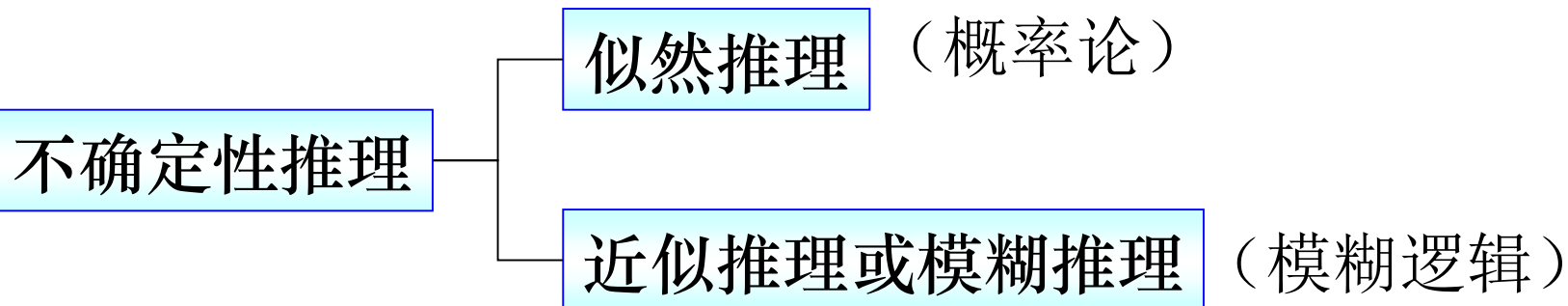


# 推理方式及其分类

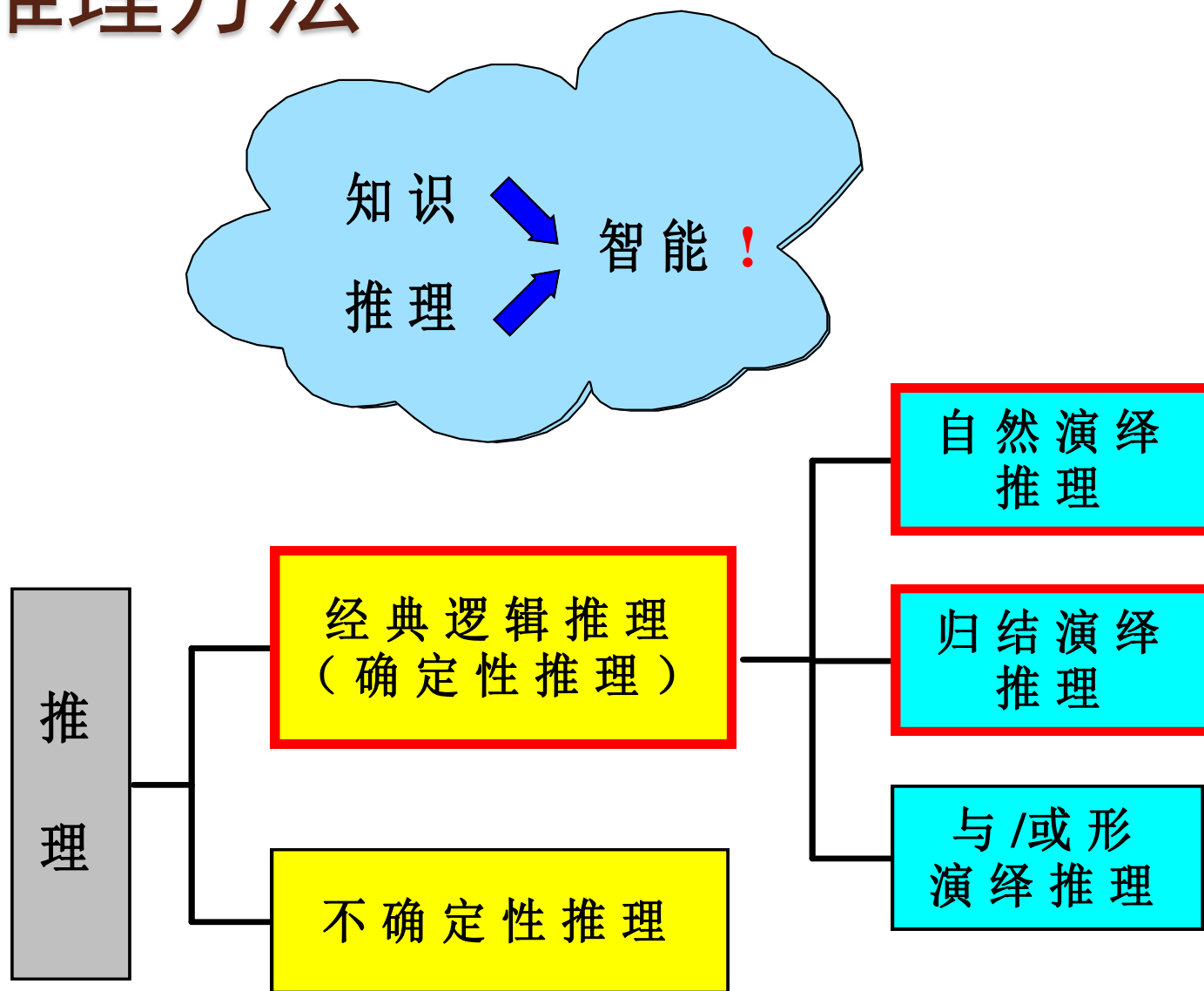
## 2. 确定性推理、不确定性推理

(1) **确定性推理**：推理时所用的知识与证据都是确定的，推出的结论也是确定的，其真值或者为真或者为假。

(2) **不确定性推理**：推理时所用的知识与证据不都是确定的，推出的结论也是不确定的。



# 推理方法



# 自然演绎推理

- 自然演绎推理：从一组已知为真的事实出发，运用**经典逻辑的推理规则**推出结论的过程。
- 推理规则： $P$ 规则、 $T$ 规则、假言推理、拒取式推理  
(p54. 55.)

■ **假言推理**：  $P, P \rightarrow Q \Rightarrow Q$

■ “如果 $x$ 是金属，则 $x$ 能导电”，“铜是金属” 推出 “铜能导电”

■ **拒取式推理**：  $P \rightarrow Q, \neg Q \Rightarrow \neg P$

■ “如果下雨，则地下就湿”，“地上不湿” 推出 “没有下雨”

# 自然演绎推理

**错误1**——否定前件： $P \rightarrow Q, \neg P \not\Rightarrow \neg Q$

- (1) 如果下雨，则地上是湿的 ( $P \rightarrow Q$ )；
- (2) 没有下雨 ( $\neg P$ )；
- (3) 所以，地上不湿 ( $\neg Q$ )。

**错误2**——肯定后件： $P \rightarrow Q, Q \not\Rightarrow P$

- (1) 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 ( $P \rightarrow Q$ )；
- (2) 金星显示出位相变化 ( $Q$ )；
- (3) 所以，行星系统是以太阳为中心 ( $P$ )。

# 自然演绎推理

- 例1 已知事实：
  - (1) 凡是容易的课程小王( Wang )都喜欢；
  - (2) C 班的课程都是容易的；
  - (3) AI 是 C 班的一门课程。
- 求证：小王喜欢 AI 这门课程。

# 自然演绎推理

- 证明:

- 定义谓词:

$EASY(x)$ :  $x$  是容易的

$LIKE(x, y)$ :  $x$  喜欢  $y$

$C(x)$ :  $x$  是  $C$  班的一门课程

- 已知事实和结论用谓词公式表示:

$(\forall x)(EASY(x) \rightarrow LIKE(Wang, x))$


$(\forall x)(C(x) \rightarrow EASY(x))$


$C(AI)$

$LIKE(Wang, AI)$

# 自然演绎推理

## ■ 应用推理规则进行推理：

  $(\forall x) (EASY(x) \rightarrow LIKE(Wang, x))$   
 $EASY(z) \rightarrow LIKE(Wang, z)$       全称固化

  $(\forall x) (C(x) \rightarrow EASY(x))$   
 $C(y) \rightarrow EASY(y)$       全称固化

所以  $C(AI), C(y) \rightarrow EASY(y)$   
 $\Rightarrow EASY(AI)$        $P$ 规则及假言推理

所以  $EASY(AI), EASY(z) \rightarrow LIKE(Wang, z)$   
 $\Rightarrow LIKE(Wang, AI)$        $T$ 规则及假言推理

# 谓词逻辑的应用

例2 每个去临潼游览的人或者参观秦始皇兵马俑，或者参观华清池，或者洗温泉澡。凡去临潼游览的人，如果爬骊山就不能参观秦始皇兵马俑，有的游览者既不参观华清池，也不洗温泉澡。因而有的游览者不爬骊山。

解：定义  $G(x)$  表示“ $x$ 去临潼游览”；

$A(x)$  表示“ $x$ 参观秦始皇兵马俑”；

$B(x)$  表示“ $x$ 参观华清池”；

$C(x)$  表示“ $x$ 洗温泉澡”；

$D(x)$  表示“ $x$ 爬骊山”。



# 谓词逻辑的应用

例2 每个去临潼游览的人或者参观秦始皇兵马俑，或者参观华清池，或者洗温泉澡。凡去临潼游览的人，如果爬骊山就不能参观秦始皇兵马俑，有的游览者既不参观华清池，也不洗温泉澡。因而有的游览者不爬骊山。

解：定义  $G(x)$  表示“ $x$ 去临潼游览”；

$A(x)$  表示“ $x$ 参观秦始皇兵马俑”；

$B(x)$  表示“ $x$ 参观华清池”；

$C(x)$  表示“ $x$ 洗温泉澡”；

$D(x)$  表示“ $x$ 爬骊山”。

前提：  $\forall x (G(x) \rightarrow A(x) \vee B(x) \vee C(x))$  (1)

$\forall x (G(x) \wedge D(x) \rightarrow \neg A(x))$  (2)

$\exists x (G(x) \wedge \neg B(x) \wedge \neg C(x))$  (3)

结论：  $\exists x (G(x) \wedge \neg D(x))$

# 谓词逻辑的应用

**前提:**  $\forall x (G(x) \rightarrow A(x) \vee B(x) \vee C(x))$  (1)

$\forall x (G(x) \wedge D(x) \rightarrow \neg A(x))$  (2)

$\exists x (G(x) \wedge \neg B(x) \wedge \neg C(x))$  (3)

**结论:**  $\exists x (G(x) \wedge \neg D(x))$

**证明:** (4)  $G(a) \wedge \neg B(a) \wedge \neg C(a)$  由(3)

(5)  $G(a) \rightarrow A(a) \vee B(a) \vee C(a)$  由(1)

(6)  $G(a) \wedge D(a) \rightarrow \neg A(a)$  由(2)

(7)  $A(a) \rightarrow \neg G(a) \vee \neg D(a)$  由(6)

(8)  $G(a)$  由(4)

(9)  $A(a) \vee B(a) \vee C(a)$  由(5) (8)

(10)  $\neg B(a), \neg C(a)$  由(4)

(11)  $A(a)$  由(9) (10)

(12)  $\neg D(a)$  由(7) (8) (11)

(13)  $\exists x (G(x) \wedge \neg D(x))$  由(8) (12)

# 自然演绎推理

## ■ 优点：

- 表达定理证明过程自然，易理解。
- 拥有丰富的推理规则，推理过程灵活。
- 便于嵌入领域启发式知识。

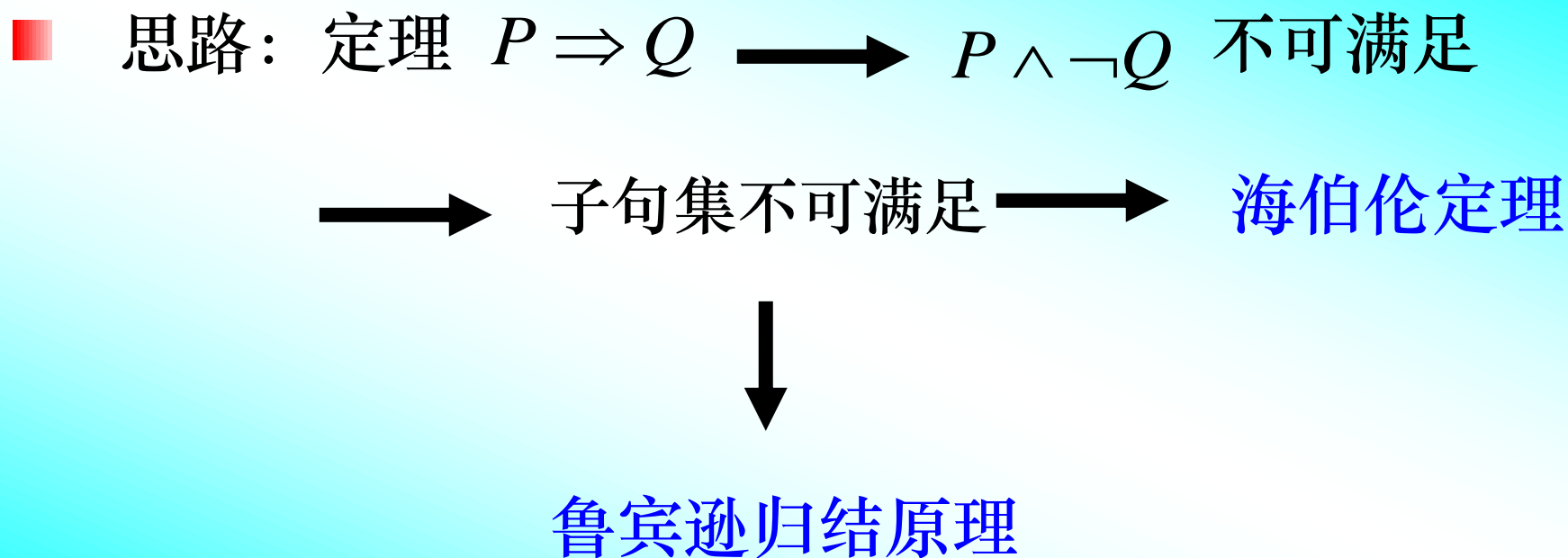
■ 缺点：易产生组合爆炸，得到的中间结论一般呈指数形式递增。

# 归结演绎推理

■ 反证法：  $P \Rightarrow Q$  ，当且仅当  $P \wedge \neg Q \Leftrightarrow F$  ，  
即  $Q$  为  $P$  的逻辑推论，当且仅当  $P \wedge \neg Q$  是不可满足的。

■ 定理：  $Q$  为  $P_1, P_2, \dots, P_n$  的逻辑推论，当且仅当  
 $(P_1 \wedge P_2 \wedge \dots \wedge P_n) \wedge \neg Q$  是不可满足的。

# 归结演绎推理



# 知识表示和推理

- 1 概述
- 2 命题逻辑
- 3 谓词逻辑
- 4 归结推理

# 逻辑蕴涵和基于知识的系统

- Start with KB representing explicit beliefs, usually what the agent has been told or has learned
- Implicit beliefs:  $\{\alpha \mid KB \models \alpha\}$
- Actions depend on implicit beliefs, rather than explicit beliefs

# 推理程序

- 我们希望找到一种自动的推理程序来判断“ $KB$ 逻辑上蕴涵 $\alpha$ ”是否成立。
- 对于一个推理程序：
- 它是合理的（Sound）是指：如果该推理程序认为答案为yes，那么“ $KB$ 逻辑上蕴涵 $\alpha$ ”是成立的；
- 它是完备的（Complete）是指：如果“ $KB$ 逻辑上蕴涵 $\alpha$ ”，那么该推理程序会认为答案为yes。



# 归结推理

- 1965年，由Robinson (鲁宾逊)提出归结法
- 归结法的基本思想
- 命题逻辑归结原理和过程
- 谓词逻辑归结原理和过程
- 应用归结原理求解问题
- 归结反演

# 什么是归结原理

- 在定理证明系统中，已知一个公式集  $F_1, F_2, \dots, F_n$ ，要证明一个公式  $W$  (定理) 是否成立，即要证明  $W$  是公式集的逻辑推论时，一种证明法就是要证明  $F_1 \wedge F_2 \wedge \dots \wedge F_n \rightarrow W$  为永真式。
- **反证法**：证明  $F = F_1 \wedge F_2 \wedge \dots \wedge F_n \wedge \neg W$  为永假，这等价于证明  $F$  对应的 **子句集**  $S = \{F_1, F_2, \dots, F_n, \neg W\}$  为不可满足的。

# 子句集

- 文字 (literal) : 原子公式及其否定。例如,  $P$ : 正文字,  $\neg P$ : 负文字。
- 子句 (clause) : 任何文字的析取。某个文字本身也都是子句。

$$P \vee Q \vee \neg R, \text{ 记作 } (P, Q, \neg R)$$

- 空子句 (NIL) : 不包含任何文字的子句。

空子句是永假的, 不可满足的。

- 子句集: 由子句构成的集合 (子句的合取)。

# 归结式的定义及性质

- 对于任意两个子句 $C_1$ 和 $C_2$ ，若 $C_1$ 中有一个文字 $L$ ，而 $C_2$ 中有一个与 $L$ 成互补的文字 $\neg L$ ，则分别从 $C_1$ 和 $C_2$ 中删去 $L$ 和 $\neg L$ ，并将其剩余部分组成新的析取式。这个新的子句被称为 $C_1$ 和 $C_2$ 关于 $L$ 的**归结式**， $C_1$ 和 $C_2$ 则是该归结式的亲本子句
- 例如， $P$ 和 $\neg P$ 的归结式为空子句，记作 $()$ 、 $\square$ 或 $NIL$ ； $(W, R, Q)$ 和 $(W, S, \neg R)$ 关于 $R$ 的归结式为 $(W, Q, S)$
- **定理：两个子句的归结式是这两个子句的逻辑推论，**如 $\{(P, C_1), (\neg P, C_2)\} \vdash (C_1, C_2)$

# 鲁宾逊归结原理

◆ 子句集中子句之间是合取关系，只要有一个子句不可满足，则子句集就不可满足。

◆ 鲁宾逊归结原理（消解原理）的基本思想：

- 检查子句集  $S$  中是否包含空子句，若包含，则  $S$  不可满足。
- 若不包含，在  $S$  中选择合适的子句进行归结，一旦归结出空子句，就说明  $S$  是不可满足的。

# 推导

- 从一个子句集 $S$ （如 $KB$ ）推导出一个子句 $C$ 的过程中会产生一系列子句 $C_1, C_2, \dots, C_n$ ，其中 $C_n = C$ ，且对于 $C_i$  ( $i = 1, 2, \dots, n-1$ )均有：
  - $C_i \in S$
  - 或者 $C_i$ 是推导过程中产生的某两个子句的归结式
- 从 $S$ 推导出 $C$ 记为： $S \vdash C$

# 推导的合理性

- 定理：如果  $S \vdash C$ ，那么  $S \models C$
- 证明：
  - 令  $S$  推导出  $C$  产生的子句序列为  $C_1, C_2, \dots, C_n$
  - 通过数学归纳法证明对于  $i \in [1, n]$ ， $S \models C_i$  均成立
- 反之，若  $S \models C$ ，则从  $S$  中不一定能够推导出  $C$ 。  
例如， $P \models (P, Q)$ ，但是  $P$  不能推导出  $(P, Q)$

# 归结法的合理性和完备性

- 定理：  $S \vdash ()$ ，当且仅当  $S \models ()$ ，当且仅当  $S$  不可满足
- 由前文可知， $KB \models \alpha$ ，当且仅当  $KB \wedge \neg \alpha$  不可满足。结合上述定理，我们通过下述过程来判断  $KB \models \alpha$  是否成立：
  - 记  $KB \wedge \neg \alpha$  的子句集为  $S$
  - 判断  $S \vdash ()$  是否成立，即从  $S$  中能否推导出空子句
- 归结法的过程比较单纯，只涉及归结推理规则的应用问题，因而便于实现机器证明。



# 命题逻辑的归结原理和过程

命题逻辑中，若给定前提集 $F$ 和命题 $P$ ，则归结证明过程可归纳如下：

- (1) 把 $F$ 转化成子句集表示，得到子句集 $S_0$ ；
- (2) 把命题 $P$ 的否定式 $\neg P$ 也转化成子句集表示，并将其加到 $S_0$ 中，得 $S = S_0 \cup S_{\neg P}$ ；
- (3) 对子句集 $S$ 反复应用归结推理规则（推导），直至导出含有空子句的扩大子句集为止。即出现归结式为空子句时，表明已找到矛盾，证明过程结束。

# 命题逻辑的归结原理和过程

例1： 设已知前提集为

$P \dots\dots\dots(1)$        $(P \wedge Q) \rightarrow R \dots\dots(2)$

$(S \vee T) \rightarrow Q \dots(3)$        $T \dots\dots\dots(4)$

求证 $R$ 。

证明：化成子句集

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

- 归结可用图的演绎树表示，  
由于根部出现空子句，因此  
命题 $R$ 得证。

# 命题逻辑的归结原理和过程

例1: 设已知前提集为

$P \dots\dots\dots(1)$

$(P \wedge Q) \rightarrow R \dots\dots(2)$

$(S \vee T) \rightarrow Q \dots(3)$

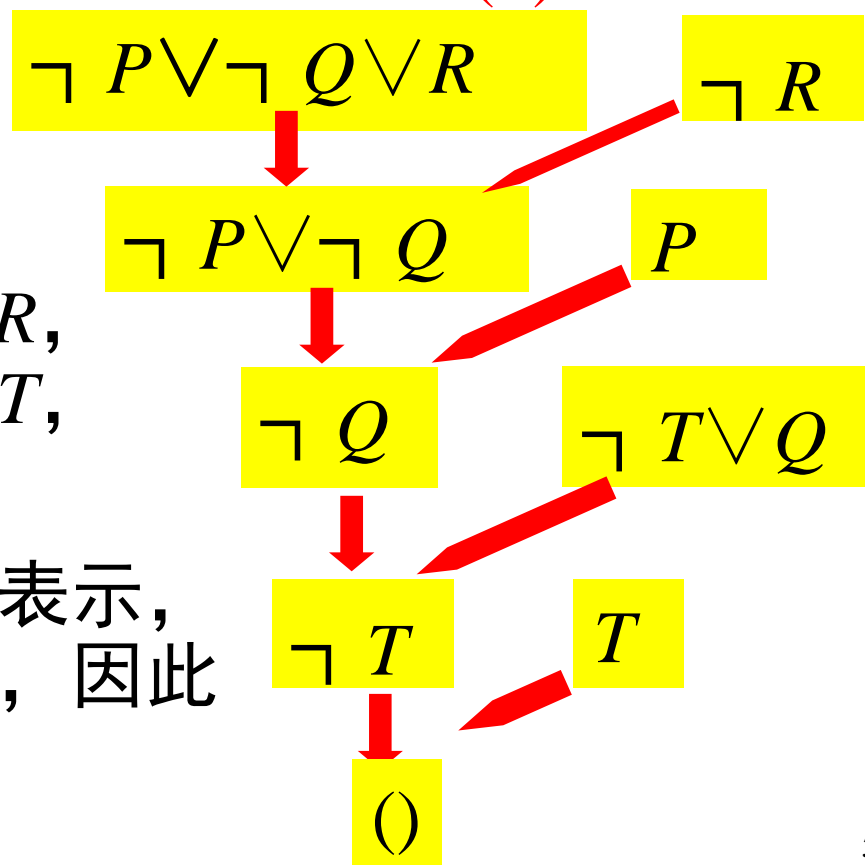
$T \dots\dots\dots(4)$

求证 $R$ 。

证明: 化成子句集

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

- 归结可用图的演绎树表示, 由于根部出现空子句, 因此命题 $R$ 得证。



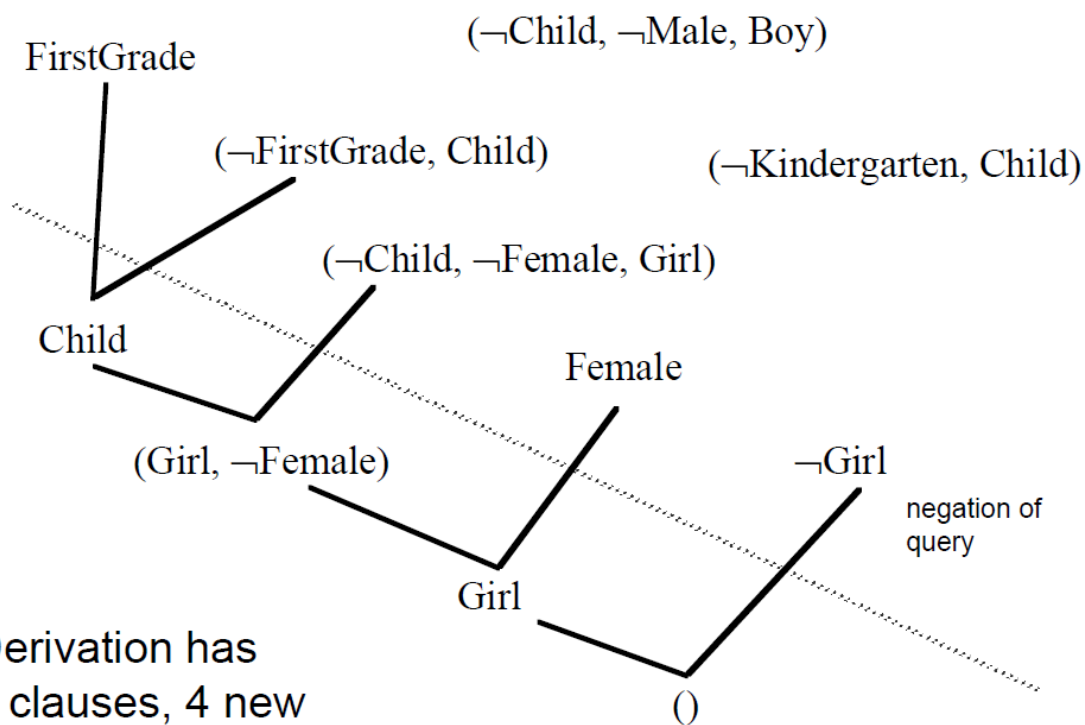
# 命题逻辑的归结原理和过程

例2:

KB

FirstGrade  
FirstGrade  $\rightarrow$  Child  
Child  $\wedge$  Male  $\rightarrow$  Boy  
Kindergarten  $\rightarrow$  Child  
Child  $\wedge$  Female  $\rightarrow$  Girl  
Female

Show that  $KB \models \text{Girl}$




Derivation has  
9 clauses, 4 new

# 谓词逻辑的归结原理和过程

在谓词逻辑中应用归结法时，首先需要：

- (1) 将所有谓词公式（包括知识库 $KB$ 和查询 $\alpha$ ）化为子句集
- (2) 通过合一，对含有变量的子句进行归结


$$C_1 = P(x) \vee Q(x)$$
$$C_2 = \neg P(a) \vee R(y)$$

?

# 谓词逻辑的归结原理和过程

$$\forall x(\forall yP(x, y) \rightarrow \neg\forall y(Q(x, y) \rightarrow R(x, y)))$$

谓词公式化为子句集的步骤：

(1) **消去蕴涵和等价符号** ( $\rightarrow$ 和 $\leftrightarrow$ 联结词)。

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

$$\iff \forall x(\neg\forall yP(x, y) \vee \neg\forall y(\neg Q(x, y) \vee R(x, y)))$$

(2) **内移否定符号 $\neg$** ，将其移到紧靠谓词的位置上。

$$\text{双重否定律 } \neg(\neg P) \Leftrightarrow P$$

$$\text{德.摩根律 } \neg(P \wedge Q) \Leftrightarrow \neg P \vee \neg Q, \quad \neg(P \vee Q) \Leftrightarrow \neg P \wedge \neg Q$$

$$\text{量词转换律 } \neg\exists xP \Leftrightarrow \forall x\neg P, \quad \neg\forall xP \Leftrightarrow \exists x\neg P$$

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

# 谓词逻辑的归结原理和过程

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

谓词公式化为子句集的步骤：

- (3) **变量标准化**。对变量作必要的换名，使每一量词只约束一个唯一的变量名。

$$\exists xP(x) \equiv \exists yP(y), \quad \forall xP(x) \equiv \forall yP(y)$$

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

- (4) **消去存在量词 (Skolemize)**。对于待消去的存在量词，若不在任何全称量词辖域之内，则用Skolem常量替代公式中存在量词约束的变量；若受全称量词约束，则要用Skolem函数替代存在量词约束的变量，然后就可消去存在量词。

# 谓词逻辑的归结原理和过程

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

Skolemize:

对于一般情况

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

存在量词 $y$ 的Skolem函数为 $y = f(x_1, x_2, \cdots, x_n)$

Skolem化：用Skolem函数替代存在量词约束的变量的过程。

$$\begin{array}{l} y = f(x), \\ z = g(x) \end{array} \iff \forall x(\neg P(x, f(x)) \vee (Q(x, g(x)) \wedge \neg R(x, g(x))))$$

(5) **化为前束型**。前束型=（前缀）{母式}。其中，前缀为全称量词串，母式为不含量词的谓词公式。



# 谓词逻辑的归结原理和过程

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

谓词公式化为子句集的步骤：

(6) **把母式化成合取范式**。反复使用结合律和分配律，将母式表达成合取范式的Skolem标准形。

Skolem 标准形： $\forall x_1 \forall x_2 \cdots \forall x_n M$

$M$ ：子句的合取式，称为Skolem标准形的母式。

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

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

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

(7) **略去全称量词**。由于母式的变量均受全称量词的约束，因此可省略掉全称量词。

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

# 谓词逻辑的归结原理和过程

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

谓词公式化为子句集的步骤：

(8) **把母式用子句集表示**。把母式中每一个合取元称为一个子句，省去合取联结词，这样就可把母式写成集合的形式表示，每一个元素就是一个子句。

$$\longleftrightarrow \{(\neg P(x, f(x)), Q(x, g(x))), (\neg P(x, f(x)), \neg R(x, g(x)))\}$$

(9) **子句变量标准化**。对某些变量重新命名，使任意两个子句不会有相同的变量出现。这是因为在使用子句集进行证明推理的过程中，有时需要例化某一个全称量词约束的变量，该步骤可以使公式尽量保持其一般化形式，增加了应用过程的灵活性。

$$\longleftrightarrow \{(\neg P(x, f(x)), Q(x, g(x))), (\neg P(y, f(y)), \neg R(y, g(y)))\}$$

# 谓词逻辑的归结原理和过程

✱ 例1 将下列谓词公式化为不含存在量词的前束型。

$$\exists x \forall y (\forall z (P(z) \wedge \neg Q(x, z)) \rightarrow R(x, y, f(a)))$$

# 谓词逻辑的归结原理和过程

✿ 例1 将下列谓词公式化为不含存在量词的前束型。

$$\exists x \forall y (\forall z (P(z) \wedge \neg Q(x, z)) \rightarrow R(x, y, f(a)))$$

- (1) 消去存在量词

$$\forall y (\forall z (P(z) \wedge \neg Q(b, z)) \rightarrow R(b, y, f(a)))$$

- (2) 消去蕴涵符号

$$\forall y (\neg \forall z (P(z) \wedge \neg Q(b, z)) \vee R(b, y, f(a)))$$

$$\forall y (\exists z (\neg P(z) \vee Q(b, z)) \vee R(b, y, f(a)))$$

- (3) 设 $z$ 的Skolem函数是 $g(y)$ , 则

$$\forall y (\neg P(g(y)) \vee Q(b, g(y)) \vee R(b, y, f(a)))$$

# 谓词逻辑的归结原理和过程

✱ 例2 将下列谓词公式化为子句集。

$$\forall x\{[\neg P(x) \vee \neg Q(x)] \rightarrow \exists y[S(x, y) \wedge Q(x)]\} \wedge \forall x[P(x) \vee B(x)]$$

# 谓词逻辑的归结原理和过程

✱ 例2 将下列谓词公式化为子句集。

$$\forall x\{[\neg P(x) \vee \neg Q(x)] \rightarrow \exists y[S(x, y) \wedge Q(x)]\} \wedge \forall x[P(x) \vee B(x)]$$

- (1) 消去蕴涵符号

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

- (2) 把否定符号移到每个谓词前面

$$\forall x\{[P(x) \wedge Q(x)] \vee \exists y[S(x, y) \wedge Q(x)]\} \wedge \forall x[P(x) \vee B(x)]$$

- (3) 变量标准化

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

- (4) 消去存在量词，设y的Skolem函数是 $f(x)$ ，则

$$\forall x\{[P(x) \wedge Q(x)] \vee [S(x, f(x)) \wedge Q(x)]\} \wedge \forall w[P(w) \vee B(w)]$$

# 谓词逻辑的归结原理和过程

✱ 例2 将下列谓词公式化为子句集（续）。

## (5) 化为前束型

$$\forall x \forall w \{ \{ [P(x) \wedge Q(x)] \vee [S(x, f(x)) \wedge Q(x)] \} \wedge [P(w) \vee B(w)] \}$$

## (6) 化为标准形

$$\forall x \forall w \{ \{ [Q(x) \wedge P(x)] \vee [Q(x) \wedge S(x, f(x))] \} \wedge [P(w) \vee B(w)] \}$$

$$\forall x \forall w \{ Q(x) \wedge [P(x) \vee S(x, f(x))] \wedge [P(w) \vee B(w)] \}$$

## (7) 略去全称量词

$$Q(x) \wedge [P(x) \vee S(x, f(x))] \wedge [P(w) \vee B(w)]$$

## (8) 消去合取词，把母式用子句集表示

$$\{Q(x), (P(x), S(x, f(x))), (P(w), B(w))\}$$

## (9) 子句变量标准化 $\{Q(x), (P(y), S(y, f(y))), (P(w), B(w))\}$

# 谓词逻辑的归结原理和过程

- 在证明定理的演绎过程中，经常要对量化的表达式进行匹配操作，因而需要对项作变量置换使表达式一致起来。

## 归结过程：

- ◆ 若 $S$ 中两个子句间有相同互补文字的谓词，但它们的项不同，则必须找出对应的不一致项；
- ◆ 进行变量置换，使它们的对应项一致；
- ◆ 求归结式看能否推导出空子句。



# 谓词逻辑的归结原理和过程

## 合一 (Unify) :

在谓词逻辑的归结过程中，寻找项之间合适的变量置换使表达式一致，这个过程称为合一。

- 一个表达式的项可以是常量符号、变量符号或函数式。
- 表达式的例 (instance) 是指在表达式中用置换项置换变量后而得到的一个特定的表达式。
- 用  $\sigma = \{v_1/t_1, v_2/t_2, \dots, v_n/t_n\}$  来表示任一**置换** (教材表示相反)。  
 $v_i/t_i$  是指表达式中的变量  $v_i$  以项  $t_i$  来替换，且不允许  $v_i$  用与  $v_i$  有关的项  $t_i$  (但是  $t_i$  中可以包含其它变量) 作置换。  
为了便于理解，后续记  $\sigma = \{v_1 = t_1, v_2 = t_2, \dots, v_n = t_n\}$ 。
- 用  $\sigma$  对表达式  $E$  作置换后的例简记为  $E\sigma$ 。

# 谓词逻辑的归结原理和过程

## 合一 (Unify) :

- 例如,  $P(x, g(y, z))\{x = y, y = f(a)\} \Rightarrow P(y, g(f(a), z))$
- 注意: 置换是同时进行的, 而不是先后进行的。
- 可以对表达式多次置换, 如用  $\theta$  和  $\sigma$  依次对  $E$  进行置换, 记为  $(E\theta)\sigma$ 。其结果等价于先将这两个置换合成 (组合) 为一个置换, 即  $\theta\sigma$ , 再用合成置换对  $E$  进行置换, 即  $E(\theta\sigma)$ 。

# 谓词逻辑的归结原理和过程

## 合一 (Unify) :

$$\theta = \{x_1 = s_1, x_2 = s_2, \dots, x_m = s_m\}, \sigma = \{y_1 = t_1, y_2 = t_2, \dots, y_k = t_k\}$$

- 合成置换 $\theta\sigma$ 的组成：1)  $\theta$ 的置换对，只是 $\theta$ 的项被 $\sigma$ 作了置换；2)  $\sigma$ 中与 $\theta$ 变量不同的那些变量对。
- 合成置换 $\theta\sigma$ 的步骤：
  - 1. Get  $S = \{x_1 = s_1\sigma, x_2 = s_2\sigma, \dots, x_m = s_m\sigma, y_1 = t_1, y_2 = t_2, \dots, y_k = t_k\}$
  - 2. Delete any equation  $y_i = s_i$  where  $y_i$  is equal to one of the  $x_j$  in  $\theta$
  - 3. Delete any identities, i.e., equations of the form  $v = v$

# 谓词逻辑的归结原理和过程

## 合一 (Unify) :

- 令  $\theta = \{x = f(y), y = z\}$ ,  $\sigma = \{x = a, y = b, z = y\}$ 
  - 1. Get  $S = \{x = f(b), y = y, x = a, y = b, z = y\}$
  - 2. Delete  $x = a$ ; Delete  $y = b$
  - 3. Delete  $y = y$

$$\theta\sigma = S = \{x = f(b), z = y\}$$

这样的合成法可使  $(E\theta)\sigma = E(\theta\sigma)$ , 即可结合。  
但置换是不可交换的, 即  $\theta\sigma \neq \sigma\theta$ 。

空置换  $\epsilon = \{\}$  也是一个置换, 且  $\theta\epsilon = \theta$ 。

# 谓词逻辑的归结原理和过程

## 合一 (Unifier) :

- A unifier (合一) of two formulas  $f$  and  $g$  is a substitution  $\sigma$  that makes  $f$  and  $g$  syntactically identical.
- Note that not all formulas can be unified – substitutions only affect variables.
- e.g.,  $P(f(x), a)$  and  $P(y, f(w))$  cannot be unified, as there is no way of making  $a = f(w)$  with a substitution.

# 谓词逻辑的归结原理和过程

## 最一般的合一项 (Most General Unifier) :

A substitution  $\sigma$  of two formulas  $f$  and  $g$  is a Most General Unifier (MGU) if

- $\sigma$  is a unifier.
- For every other unifier  $\theta$  of  $f$  and  $g$  there must exist a third substitution  $\lambda$  such that  $\theta = \sigma\lambda$ .

This says that every other unifier is “more specialized” than  $\sigma$ .

The MGU of a pair of formulas  $f$  and  $g$  is unique up to renaming.

# 谓词逻辑的归结原理和过程

## MGU示例:

- $P(f(x), z)$  and  $P(y, a)$
- $\sigma = \{y = f(a), x = a, z = a\}$  is a unifier, but not an MGU
- $\theta = \{y = f(x), z = a\}$  is an MGU
- $\sigma = \theta\lambda$ , where  $\lambda = \{x = a\}$

# 谓词逻辑的归结原理和过程

## 计算MGU:

- The MGU is the “least specialized” way of making atomic formulas with variables match.
- We can compute MGUs.
- Intuitively we line up the two formulas and find the first sub-expression where they disagree.
- The pair of subexpressions where they first disagree is called the disagreement set. 差异集
- The algorithm works by successively fixing disagreement sets until the two formulas become syntactically identical.



# 谓词逻辑的归结原理和过程

## 计算MGU:

Given two atomic formulas  $f$  and  $g$

- ①  $\sigma = \{\}$ ;  $S = \{f, g\}$
- ② If  $S$  contains an identical pair of formulas, stop and return  $\sigma$  as the MGU of  $f$  and  $g$ .
- ③ Else find the disagreement set  $D = \{e_1, e_2\}$  of  $S$
- ④ If  $e_1 = V$  a variable, and  $e_2 = t$  a term not containing  $V$  (or vice-versa) then let  $\sigma = \sigma\{V = t\}$ ;  $S = S\{V = t\}$ ; Goto 2
- ⑤ Else stop,  $f$  and  $g$  cannot be unified.

Note: to update  $\sigma$ , we must compose  $\sigma$  with  $\{V = t\}$ .  
A common error is to just add  $V = t$  to  $\sigma$ .

# 谓词逻辑的归结原理和过程

计算MGU:

示例: (P. 67 例2. 18)

练习:

- ①  $P(f(a), g(x))$  and  $P(y, y)$
- ②  $P(a, x, h(g(z)))$  and  $P(z, h(y), h(y))$
- ③  $P(x, x)$  and  $P(y, f(y))$

# 谓词逻辑的归结原理和过程

## 归结原理和过程：

From the two clauses  $\{\rho_1\} \cup c_1$  and  $\{\neg\rho_2\} \cup c_2$ , where there exists a MGU  $\sigma$  for  $\rho_1$  and  $\rho_2$ , infer the clause  $(c_1 \cup c_2)\sigma$

**Theorem.**  $S \vdash ()$  iff  $S$  is unsatisfiable

1.  $(P(x), Q(g(x)))$
2.  $(R(a), Q(z), \neg P(a))$
3.  $R[1a, 2c]\{X=a\} (Q(g(a)), R(a), Q(z))$

- “R” means resolution step.
- “1a” means the 1st (a-th) literal in the first clause:  $P(x)$ .
- “2c” means the 3rd (c-th) literal in the second clause:  $\neg P(a)$ .
- 1a and 2c are the “clashing” literals.
- $\{X = a\}$  is the MGU applied.

# 示例1

已知：

- (1) 会朗读的人是识字的，
- (2) 海豚都不识字，
- (3) 有些海豚是很机灵的。

证明：有些很机灵的东西不会朗读。

解：把问题用谓词逻辑描述如下，

已知：

- (1)  $\forall x (R(x) \rightarrow L(x))$
- (2)  $\forall x (D(x) \rightarrow \neg L(x))$
- (3)  $\exists x (D(x) \wedge I(x))$

求证：  $\exists x (I(x) \wedge \neg R(x))$

# 示例1

- 前提化简，待证结论取反并化成子句形，求得子句集：

1.  $(\neg R(x), L(x))$
2.  $(\neg D(y), \neg L(y))$
3.  $D(a)$
4.  $I(a)$
5.  $(\neg I(z), R(z))$

一个可行的证明过程：

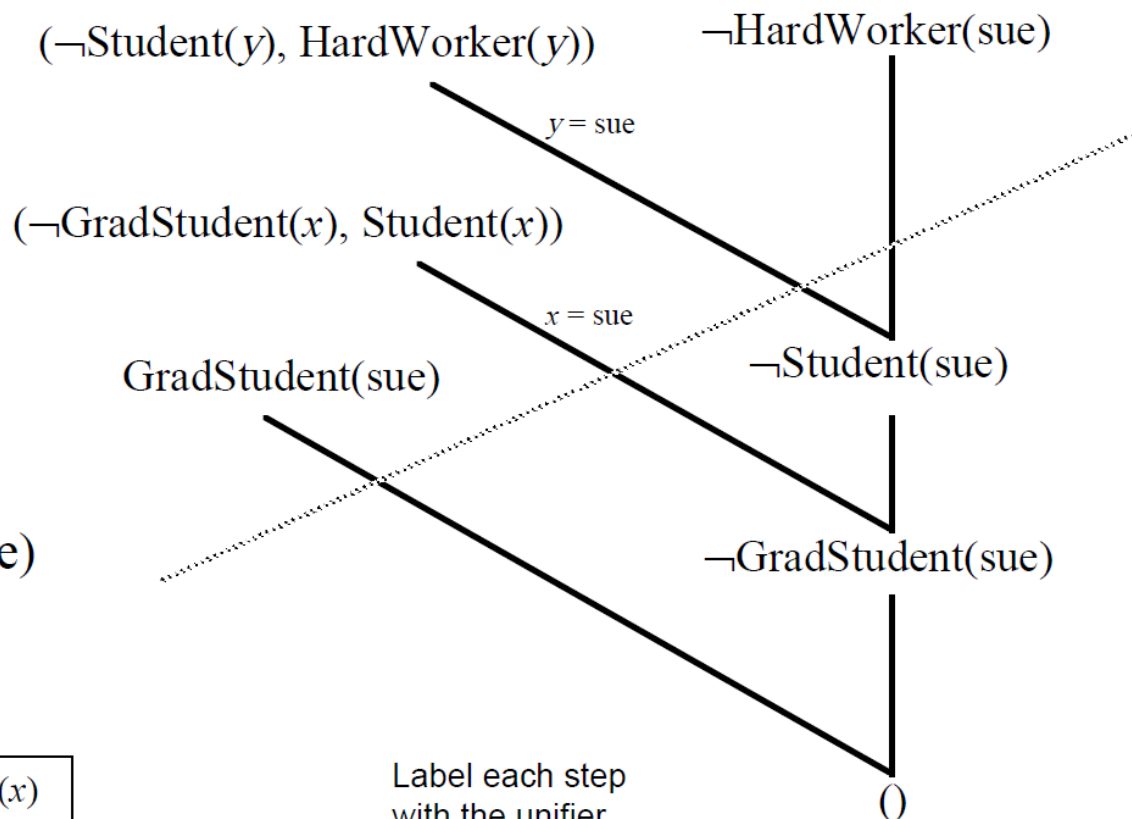
6.  $R[4, 5] \{z = a\} R(a)$
7.  $R[1, 6] \{x = a\} L(a)$
8.  $R[2, 7] \{y = a\} \neg D(a)$
9.  $R[3, 8] ()$

# 示例2

?  
KB  $\models$  HardWorker(sue)

KB

$\forall x \text{ GradStudent}(x) \rightarrow \text{Student}(x)$ $\forall x \text{ Student}(x) \rightarrow \text{HardWorker}(x)$ $\text{GradStudent}(\text{sue})$
---



Label each step  
with the unifier

Point to relevant  
literals in clauses

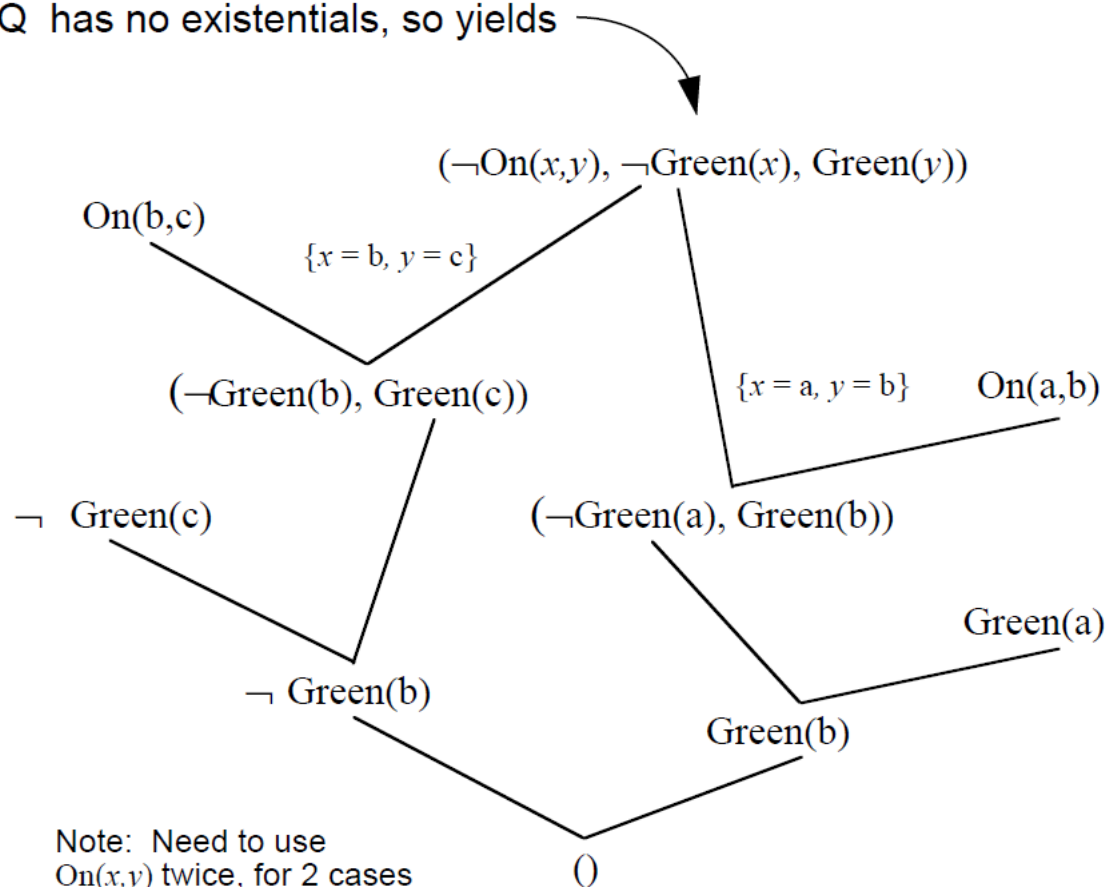
# 示例3

KB = {On(a,b), On(b,c), Green(a),  $\neg$ Green(c)}

already in CNF

Query =  $\exists x \exists y [\text{On}(x,y) \wedge \text{Green}(x) \wedge \neg \text{Green}(y)]$

Note:  $\neg Q$  has no existentials, so yields



# 练习

Prove that  $\exists y \forall x P(x, y) \models \forall x \exists y P(x, y)$

- $\exists y \forall x P(x, y) \Rightarrow 1. P(x, a)$
- $\neg \forall x \exists y P(x, y) \Leftrightarrow \exists x \forall y \neg P(x, y) \Rightarrow 2. \neg P(b, y)$
- $R[1,2]\{x = b, y = a\}()$

Exercises: Prove

- $\forall x P(x) \vee \forall x Q(x) \models \forall x (P(x) \vee Q(x))$
- $\exists x (P(x) \wedge Q(x)) \models \exists x P(x) \wedge \exists x Q(x)$



# 不可判定问题

We use 1 for  $\text{succ}(0)$ , 2 for  $\text{succ}(\text{succ}(0))$ , ...

KB:

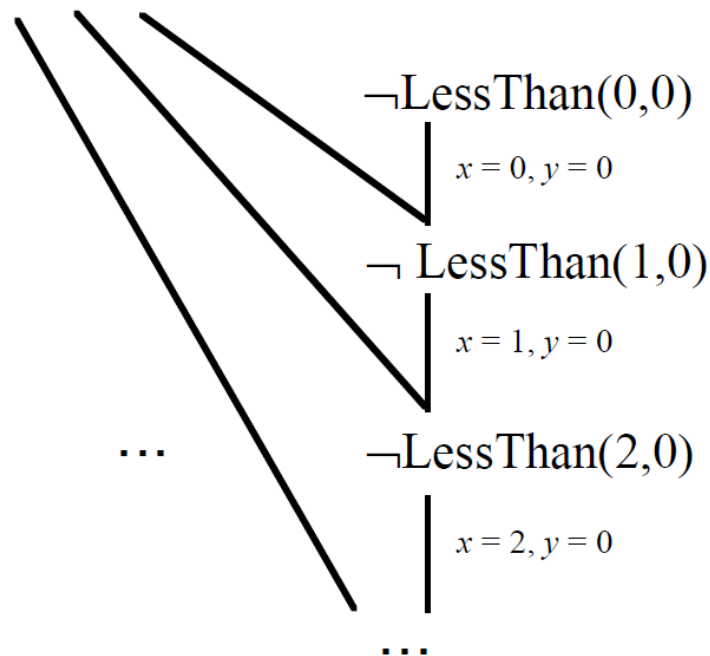
$\text{LessThan}(\text{succ}(x), y) \rightarrow \text{LessThan}(x, y)$

Query:

$\text{LessThan}(0, 0)$

Should fail since  $\text{KB} \not\models Q$

$(\text{LessThan}(x, y), \neg \text{LessThan}(\text{succ}(x), y))$



Infinite branch of resolvents

对于谓词逻辑，若子句集不可满足，则必存在一个从该子句集到空子句的推导；若从子句集存在一个到空子句的推导，则该子句集是不可满足的。

如果没有归结出空子句，则既不能说  $S$  不可满足，也不能说  $S$  是可满足的。

# 不可判定问题

- 可判定的问题：如果存在一个算法或过程，该算法用于求解该类问题时，可在有限步内停止，并给出正确的解答。
- 如果不存在这样的算法或过程则称这类问题是**不可判定的**。例如，There can be no procedure to decide if a set of clauses is satisfiable.

**Theorem.**  $S \vdash ()$  iff  $S$  is unsatisfiable

However, there is no procedure to check if  $S \vdash ()$ , because

When  $S$  is satisfiable, the search for  $()$  may not terminate

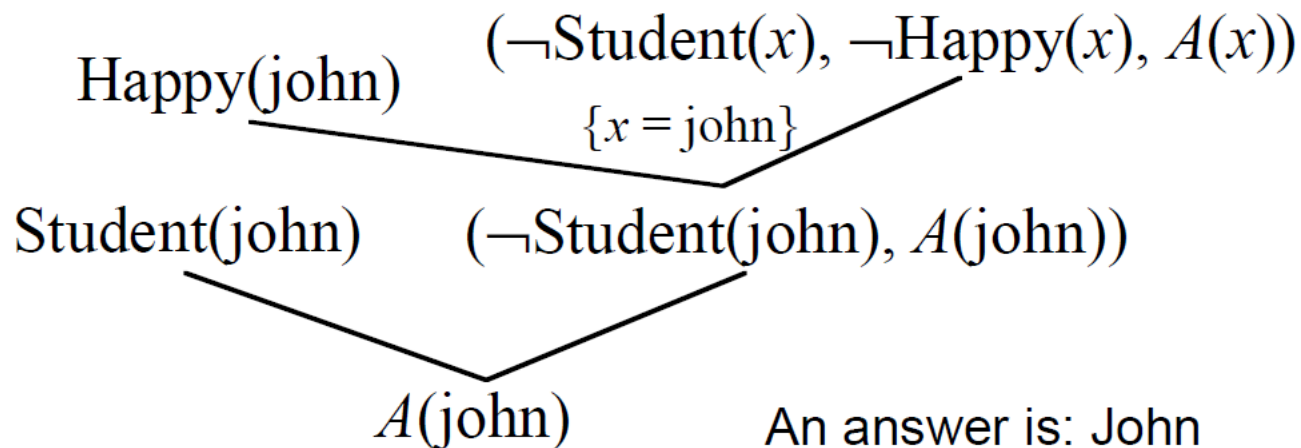
# 应用归结原理求解问题

- Replace query  $\exists x P(x)$  by  $\exists x [P(x) \wedge \neg answer(x)]$
- Instead of deriving  $()$ , derive any clause containing just the answer predicate
- 应用归结原理求解问题的步骤：
  - (1) 已知前提  $F$  用谓词公式表示，并化为子句集  $S$ ；
  - (2) 把待求解的问题  $P$  用谓词公式表示，并否定 $P$ ，再与  $answer$  构成析取式  $(\neg P \vee answer)$ ；
  - (3) 把  $(\neg P \vee answer)$  化为子句集，并入到子句集  $S$ 中，得到子句集 $S'$ ；
  - (4) 对  $S'$  应用归结原理进行归结；
  - (5) 若得到归结式 $answer$ ，则答案就在 $answer$ 中。

## 示例1

KB: Student(john)  
Student(jane)  
Happy(john)

Q:  $\exists x[\text{Student}(x) \wedge \text{Happy}(x)]$



# 示例2

KB:

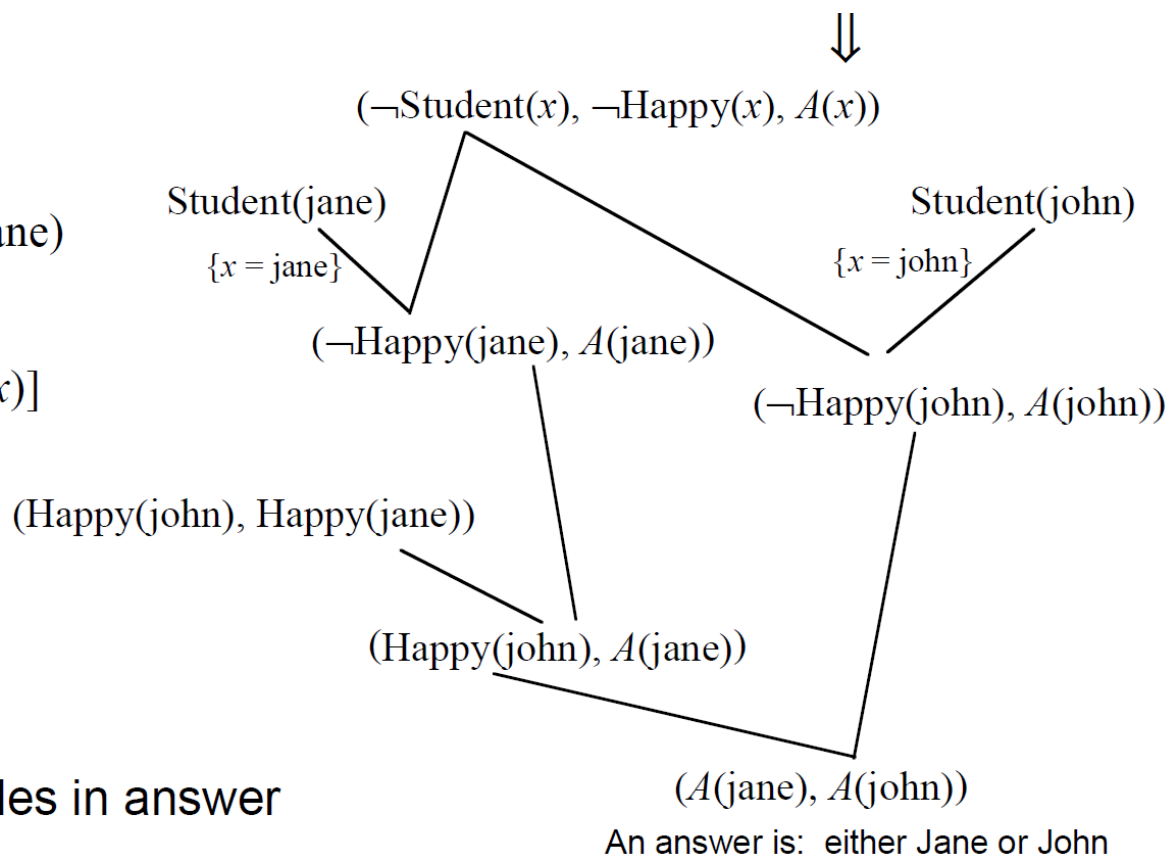
Student(john)

Student(jane)

Happy(john)  $\vee$  Happy(jane)

Query:

$\exists x[\text{Student}(x) \wedge \text{Happy}(x)]$



Note: can have variables in answer

# 练习

- Whoever can read is literate.
- Dolphins are not literate.
- Flipper is an intelligent dolphin.
- Who is intelligent but cannot read.

Use predicates:  $R(x)$ ,  $L(x)$ ,  $D(x)$ ,  $I(x)$