
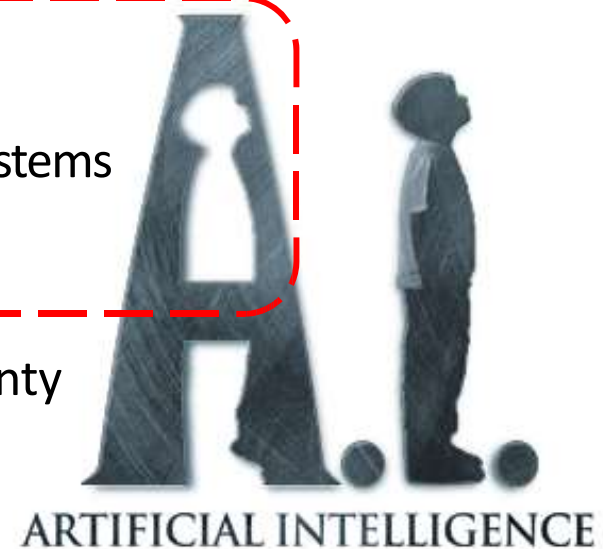




Ch.3 Searching & Reasoning Tech 第三章 搜索推理技术 II

- 
- 5. 消解原理 Resolution Principle
 - 6. 规则演绎系统 Rule-based Deduction Systems
 - 7. 产生式系统 Production System
 - 8. 不确定性推理 Reasoning with Uncertainty
 - 9. 非单调推理 Nonmonotonic Reasoning





Outline

What is reasoning?

- 从一个或几个已知的判断(前提)推出新判断(结论)的过程

一阶逻辑推理

- 置换与合一
- 消解
- 基于规则的演绎方法

产生式系统





什么是推理?

Ability and Process of **making decision** based on **facts and knowledge**.

■ 机制/原理 mechanism

How to do reasoning theoretically?

■ 控制策略 control strategy

How to realize the mechanism?



Mechanisms of Reasoning

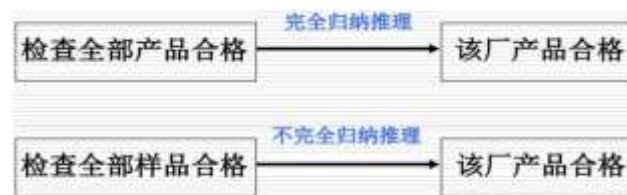
按照 逻辑基础 分类：

Deduction (演绎) vs. Induction (归纳)

一般 → 个体

- 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)

个体 → 一般



按照 知识的确定性 分类：

Reasoning under certainty (确定) vs. uncertainty (不确定)

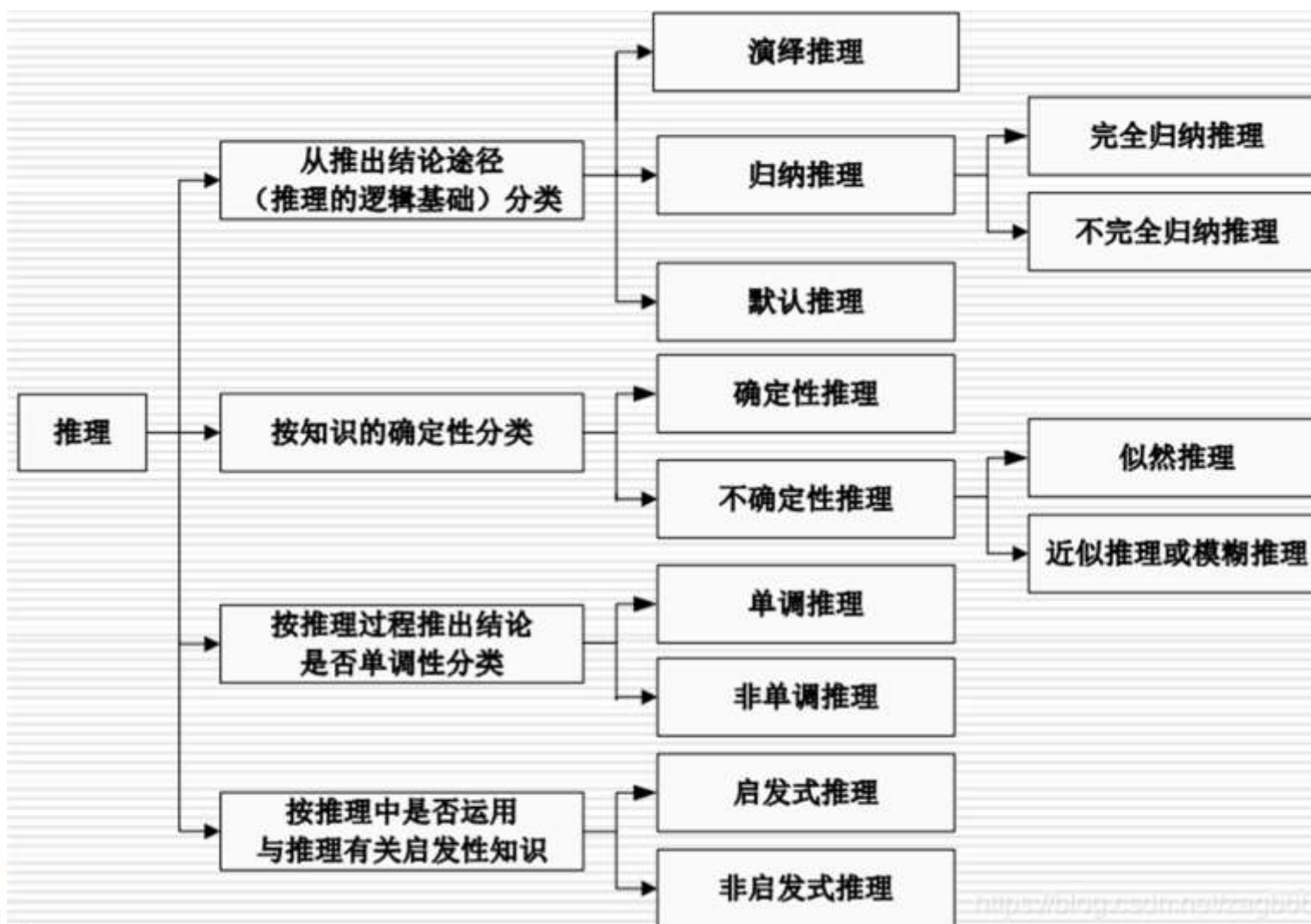
按照 推理过程的单调性 分类：

Monotonic (单调) vs. Non-monotonic (非单调)





Mechanisms of Reasoning





Control Strategies in Reasoning

推理方向

事实 \rightarrow 结论 (正向推理, 数据驱动)

事实 \leftarrow 结论 (逆向推理, 目标驱动)

事实 \leftrightarrow 结论 (双向推理、混合推理)

冲突消解

事实与知识的匹配

搜索



Outline

- What is reasoning? —
- 阶逻辑推理
 - 置换与合一
 - 消解** (Resolution) 基
 - 于规则的演绎方法
- 产生式系统 (Production-rule System)



推理是如何进行的?

- 推理过程多种多样

- 例1:

- 如果今天不下雨，我就去你家
- 今天没有下雨

- 例2:

- 小王说他下午或者去图书馆或者在家休息
- 小王没去图书馆

- 计算机如何选择?



消解原理（归结原理）

❁ 美国数学家鲁滨逊

提出消解原理（1965年）

基本的出发点：要证明一个命题为真都可以通过证明其否命题为假来得到。

将多样的推理规则**简化**为一个——
消解。



鲁滨逊



什么叫消解

例1:

小王说他下午或者去图书馆或者在家休息

小王没去图书馆

R——小王下午去图书馆

S——小王下午在家休息

$$\left. \begin{array}{l} R \vee S \\ \sim R \end{array} \right\} \Rightarrow S$$

例2:

如果今天不下雨，我就去你家

今天没有下雨

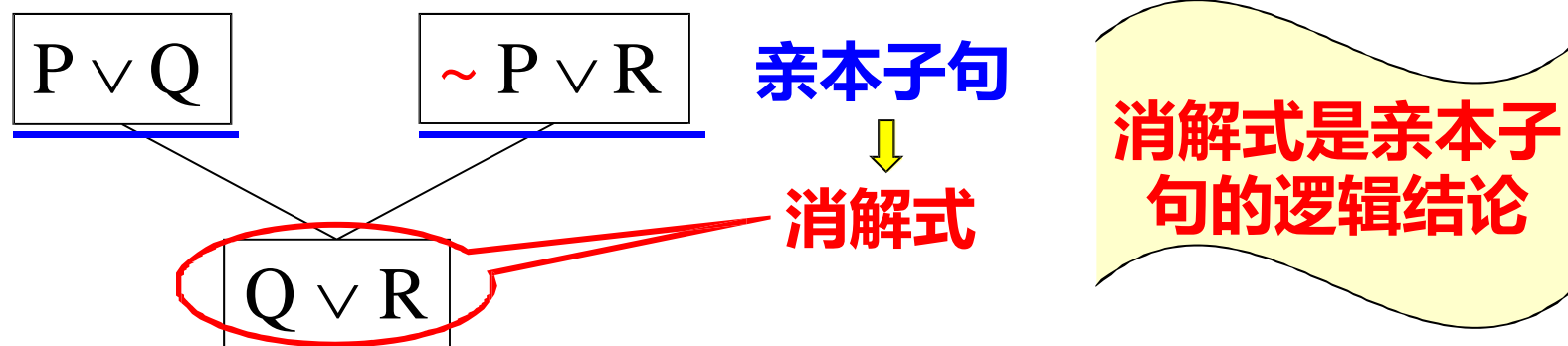
$$\sim P \rightarrow Q \Leftrightarrow P \vee Q$$

$$\sim P$$





什么叫消解



注意：

- 消解只能在仅含否定和析取连接词的公式（子句）间进行
- 必须先



含变量的消解

例：苏格拉底论断

凡人都会死. $(\forall x) (\text{Man}(x) \Rightarrow \text{Mortal}(x))$

苏格拉底是人. $\text{Man}(\text{Socrates})$

如何得到结论：苏格拉底会死. $\text{Mortal}(\text{Socrates})$

要完成消解还面临几个问题

❏ “ \forall ”和“ \Rightarrow ”必须去掉

● $\text{Man}(x) \Rightarrow \text{Mortal}(x) \Leftrightarrow \sim \text{Man}(x) \vee \text{Mortal}(x)$

● “ \forall ”怎么办?

化为子句集

❏ 如果能去掉“ \forall ”， $\sim \text{Man}(x)$ 和 $\text{Man}(\text{Socrates})$ 也不能构成互补对，形式不一样，怎么办?

置换与合一





最一般合一者(Most general unifier, mgu)

- 给定公式集 S , 设 S 的最一般合一者为 θ , 则对 S 的所有其他合一者 φ , 都存在一个置换 σ 使得

$$S \varphi = (S \theta) \sigma$$

- 关键点: create minimal instantiation changes!

表达式 $F = \{P(x, f(y), B), P(x, f(B), B)\}$

合一者 $s = \{A/x, B/y\}$

$Fs = \{P(A, f(B), B)\}$

$g = \{B/y\}$

$Fg = \{P(x, f(B), B)\}$

mgu

$$Fs = Fgs'$$





合一算法(The unification algorithm)

❁ 差异集 (*disagreement set*)

- ❁ 通过设置指针从表达式的最左边开始逐一比较，直到发现不一致，由两表达式**不一致的项**所构成的集合称为**差异集**，或**分歧集**

❁ e.g.

- ❁ $S = \{p(f(x), h(y), a), p(f(x), z, a)\}$

- ❁ 差异集: $D_0 = \{h(y), z\}$.



• 合一算法

1. Put $k=0$, $F_k=F$ and $\sigma_0=\varepsilon$. (ε represent empty)
2. 如果 F_k 是一个单项式, 那么停止, σ_k 是 F 的 mgu .
3. 否则, 求出 F_k 的差异集 D_k
 - 如果在 D_k 中存在元素 v 和 t , 使 v 是一个未出现在 t 中的变量, 那么执行 $\sigma_{k+1} = \sigma_k \{t/v\}$, $F_{k+1}=F_k\{t/v\}$.
 - 增加 k 值, 返回第2步 increment and go to 2.
 - 否则, 停止, F 是不可合一的.

注意:

t/v 表示 t 替换原始表达式中变量 v .



MGU example

练习：求 $F = \{ P(a, x, f(g(y))) , P(z, h(z, u), f(u)) \}$ 的 mgu .

k=0: $F_0 = F$; $\sigma_0 = \varepsilon$

$$D_0 = \{a, z\}, \sigma_1 = \sigma_0 \cdot \{a/z\}$$

$$F_1 = F_0\{a/z\} = \{ P(a, x, f(g(y))) , P(a, h(a, u), f(u)) \}$$

k=1: $D_1 = \{x, h(a, u)\}, \sigma_2 = \sigma_1 \cdot \{h(a, u)/x\} = \{a/z, h(a, u)/x\}$

$$F_2 = F_1\{h(a, u)/x\} = \{ P(a, h(a, u), f(g(y))) , P(a, h(a, u), f(u)) \}$$

k=2: $D_2 = \{g(y), u\}, \sigma_3 = \sigma_2 \cdot \{g(y)/u\} = \{a/z, h(a, g(y))/x, g(y)/u\}$

$$F_3 = F_2\{g(y)/u\} = \{ P(a, h(a, g(y)), f(g(y))) , \\ (P(a, h(a, g(y)), f(g(y))) \} = \{ P(a, h(a, g(y)), f(g(y))) \}$$

mgu



Question

练习1: 求 $F = \{ P(a, x, f(g(y))) , P(z, h(z, u), f(u)) \}$ 的 mgu .

$$\sigma_3 = \{a/z, h(a, g(y))/x, g(y)/u\}$$

- 练习2: 求 $P(x, y, f(g(y)))$ 和 $P(a, h(x, u), f(z))$ 的 mgu .
- 练习3: 求 $W = \{Q(f(a), g(x)), Q(y, y)\}$ 的 mgu .
- 练习4: 求 $W = \{P(f(y), y), P(x, a)\}$ 的 mgu .

Answer

- $mgu = \{a/x, h(a, u)/y, g(h(a, u))/z\}$

-



$$mgu = \{P(f(a), a)\}$$



3.5 Resolution Principle 消解原理

基本概念

- 文字(Literal)

- 一个原子公式和原子公式的否定

- 比如: P , $\sim P$, $Q(x, y)$, $\sim R(f(x, y), z)$

- 子句(Clause)

- 文字的析取组成

- 比如: $P(x) \vee Q(x)$, $\sim R(x, f(y)) \vee S(x, g(x))$

- 空子句(Empty clause, NIL)

- unsatisfiable



合取范式(Conjunctive Normal Form,CNF)

- 子句的合取形式

子句集(Clause Set)

- $\{ P(x) \vee Q(x), \sim R(x, f(y)) \vee S(x, g(x)) \}$

- Equals: $(P(x) \vee Q(x)) \wedge (\sim R(x, f(y)) \vee S(x, g(x)))$

反驳(Refutation)

- 对谓词演算公式进行分解和化简, 消去一些符号, 以求得导出子句, 又称归结。



3.5.1 Conversion to Clause Form 子句集的求取

- 任一谓词演算公式可以化成一个子句集。其变换过程由九个步骤组成。
- 例子：将下列谓词演算公式化为一个子句集

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

$$(1) (\forall x) \{ \sim P(x) \vee [(\forall y) [\sim P(y) \vee P(f(x, y))] \wedge \sim (\forall y) [\sim Q(x, y) \vee P(y)]] \}$$

Start:

(1) 消去蕴涵符号

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



$$(2) (\forall x) \{ \sim P(x) \vee \{ (\forall y) [\sim P(y) \vee P(f(x, y))] \wedge (\exists y) [Q(x, y) \wedge \sim P(y)] \} \}$$

- (2) 使用DeMorgan's laws, **减少否定符号的辖域范围**, 每个否定符号只用到一个谓词符号上。

$$(3) (\forall x) \{ \sim P(x) \vee \{ (\forall y) [\sim P(y) \vee P(f(x, y))] \wedge (\exists w) [Q(x, w) \wedge \sim P(w)] \} \}$$

- (3) **对变量标准化**: 保证每个量词有其自己唯一的哑元。
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。



(4) $(\forall x) \{ \sim P(x) \vee \{ (\forall y) [\sim P(y) \vee P(f(x, y))] \}$
 $\wedge [Q(x, g(x)) \wedge \sim (g(x))] \}$
式中, $w = g(x)$ 为一Skolem函数。

(4)消去存在量词

两种情况

存在量词 “ \exists ” 在全称量词 “ \forall ” 的辖域内

- E.g. $\forall x \exists y \text{Height}(x, y)$
- 所存在的 y 可能依赖于 x 值
- 令这种依赖关系明显地由函数 $f(x)$ 定义, 把每个 y 值映射到存在的那个 x , 该函数称之为 *Skolem function*.
- 消除“ \exists ”, 改写为 $\forall x \text{Height}(x, f(x))$
- E.g. $\forall x \forall y \exists z P(x, y, z) \Rightarrow \forall x \forall y P(x, y, f(x, y))$

存在量词 “ \exists ” 不在全称量词 “ \forall ” 的辖域内

- E.g. $\exists x \forall y P(x, y)$
- 用不含变量的Skolem函数即**常量**, $\exists x \forall y P(x, y)$ becomes $\forall y P(A, y)$





$$(4) (\forall x) \{ \sim P(x) \vee \{ (\forall y) [\sim P(y) \vee P(f(x, y))] \} \\ \wedge [Q(x, g(x)) \wedge \sim (g(x))] \} \}$$

(5) 化为前束形

到这一步，已经没有任何存在量词 \exists ，但每个全称量词 \forall 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。

prenex form :

前束形={前缀} {母式}

全称量词串 无量词公式

$$(5) (\forall x)(\forall y) \{ \sim P(x) \vee \{ [\sim P(y) \vee P(f(x, y))] \\ \wedge [Q(x, g(x)) \wedge \sim P(g(x))] \} \}$$



$$(6) (\forall x)(\forall y) \{ [\sim P(x) \vee \sim P(y) \vee P(f(x, y))] \wedge [\sim P(x) \vee Q(x, g(x))] \wedge [\sim P(x) \vee \sim (g(x))] \}$$

(6) Convert the matrix into a conjunctive normal form;

把母式化为合取范式。 可以反复应用分配律：

$A \vee (B \wedge C)$ 等价于 $(A \vee B) \wedge (A \vee C)$

$$(7) \{ [\sim P(x) \vee \sim P(y) \vee P(f(x, y))] \wedge [\sim P(x) \vee Q(x, g(x))] \wedge [\sim P(x) \vee \sim (g(x))] \}$$

(7) Remove the prefix, eliminate universal quantifiers

消去全称量词



$$(8) \{ \sim P(x) \vee \sim P(y) \vee P(f(x,y)), \\ \sim P(x) \vee Q(x,g(x)), \\ \sim P(x) \vee \sim P(g(x)) \}$$

(8) 消去连词符号 \wedge ;

用 $\{A \vee B\}$ 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**。任一个只由文字的析取构成的合式公式叫做一个子句。

(9) , 使一个变量符号不出现在一个以
上的子句中.

$$(9) \{ \sim P(x_1) \vee \sim P(y) \vee P[f(x_1,y)] , \\ \sim P(x_2) \vee Q[x_2,g(x_2)] , \\ \sim P(x_3) \vee \sim P[g(x_3)] \}$$



Question

转换为子句集:

"Everyone who loves all animals is loved by someone."

$$\forall x \{ [\forall y (Animal(y) \Rightarrow Loves(x,y))] \Rightarrow \exists y Loves(y,x) \}$$

Answer

1. Eliminate implications 消去蕴含符合

$$\forall x (\sim \forall y (\sim \text{Animal}(y) \vee \text{Loves}(x, y)) \vee \exists y \text{Loves}(y, x))$$

2. Move ~ inwards 减小否定辖域范围

$$\forall x (\exists y \sim (\sim \text{Animal}(y) \vee \text{Loves}(x, y)) \vee \exists y \text{Loves}(y, x))$$

$$\forall x (\exists y (\text{Animal}(y) \wedge \sim \text{Loves}(x, y)) \vee \exists y \text{Loves}(y, x))$$

3. Standardize variables 变量标准化

$$\forall x (\exists y (\text{Animal}(y) \wedge \sim \text{Loves}(x, y)) \vee \exists z \text{Loves}(z, x))$$

4. Skolemize 消去存在量词

$$\forall x ((\text{Animal}(f(x)) \wedge \sim \text{Loves}(x, f(x))) \vee \text{Loves}(g(x), x))$$

5. Convert to CNF 化为合取范式

$$\forall x ((\text{Animal}(f(x)) \vee \text{Loves}(g(x), x)) \wedge (\sim \text{Loves}(x, f(x)) \vee \text{Loves}(g(x), x)))$$

6. Drop universal quantifiers 消去全称量词

$$(\text{Animal}(f(x)) \vee \text{Loves}(g(x), x)) \wedge (\sim \text{Loves}(x, f(x)) \vee \text{Loves}(g(x), x))$$

7. The clauses set is: 消去连词符合变子句集

$$\{\text{Animal}(f(x)) \vee \text{Loves}(g(x), x), \sim \text{Loves}(x, f(x)) \vee \text{Loves}(g(x), x)\}$$

8. Standardize variables 变量标准化

$$(1) \text{Animal}(f(x1)) \vee \text{Loves}(g(x1), x1)$$

$$(2) \sim \text{Loves}(x2, f(x2)) \vee \text{Loves}(g(x2), x2)$$



3.5.2 Resolution Inference Rules

消解推理规则

消解式的定义(Resolvent Definition)

- 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 具有最一般合一者 σ , 那么通过消解可以从这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。



Proof of Resolution Rule

消解:

$$\begin{array}{l} C_1 = L \vee \mathcal{C}_1 \\ C_2 = \neg L \vee \mathcal{C}_2 \end{array} \quad \left. \vphantom{\begin{array}{l} C_1 \\ C_2 \end{array}} \right\} \rightarrow C_{12} = \mathcal{C}_1 \vee \mathcal{C}_2$$

Proof:

$$\mathcal{C}_1 \vee L \Leftrightarrow \sim \mathcal{C}_1 \Rightarrow L \quad \sim L \vee \mathcal{C}_2 \Leftrightarrow L \Rightarrow \mathcal{C}_2$$

$$\therefore C_1 \wedge C_2 = (\sim \mathcal{C}_1 \Rightarrow L) \wedge (L \Rightarrow \mathcal{C}_2)$$

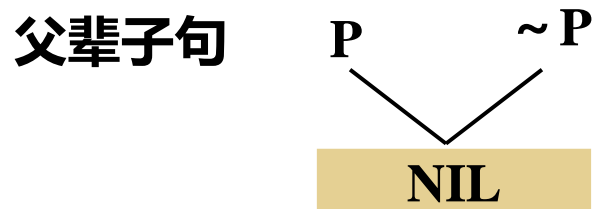
$$\lceil (\sim \mathcal{C}_1 \Rightarrow L) \wedge (L \Rightarrow \mathcal{C}_2) \rceil \Rightarrow [\sim \mathcal{C}_1 \Rightarrow \mathcal{C}_2]$$

$$\sim \mathcal{C}_1 \Rightarrow \mathcal{C}_2 \Leftrightarrow \mathcal{C}_1 \vee \mathcal{C}_2 = C_{12} \quad \therefore C_1 \wedge C_2 \Rightarrow C_{12}$$

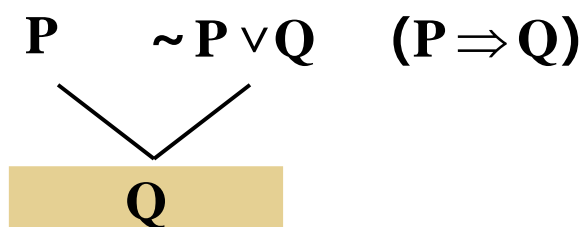


Resolvent Examples 消解式例子

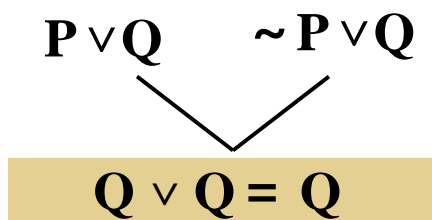
(a) 空子句



(b) 假言推理 (Modus ponens)



(c) 合并





(d)重言式(Tautologie)

Note: Each time only one pair of literal can be eliminate

$$\begin{array}{c} P \vee Q \quad \sim P \vee \sim Q \\ \swarrow \quad \searrow \\ Q \vee \sim Q \end{array}$$

$$\begin{array}{c} P \vee Q \quad \sim P \vee \sim Q \\ \swarrow \quad \searrow \\ P \vee \sim P \end{array}$$

(e) Chain 链式 (三段论)

$$\begin{array}{c} \sim P \vee Q \quad \sim Q \vee R \\ \swarrow \quad \searrow \\ \sim P \vee R \end{array}$$



3.5.3 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。

Example

$$\begin{array}{ccc} P[x, f(y)] \vee Q(x) \vee R[f(a), y] & & \sim P[f(f(a)), z] \vee R(z, w) \\ & \searrow \quad \swarrow & \\ & \sigma = \{f(f(a))/x, f(y)/z\} & \\ & Q[f(f(a))] \vee R(f(a), y) \vee R(f(y), w) & \end{array}$$



3.5.4 消解反演求解过程

消解反演 (Resolution Refutation)

给出一个公式集 $\{S\}$ 和目标公式 L , 通过反证或反演来求证目标公式 L 。

resolution refutation.

1. Convert S to clause form
2. Convert the negation of L ($\sim L$) to clause form
3. Combine the clauses resulting from steps 1 and 2 into a single set T .
4. Iteratively apply resolution to the clauses in T and add the results to T either until there are no more resolvents that can be added or until the empty clause NIL is produced



3.5.4 消解反演求解过程

消解反演

给出一个公式集 $\{S\}$ 和目标公式 L , 通过反证或反演来求证目标公式 L

- ❖ 否定 L , 得 $\sim L$;
- ❖ 把 $\sim L$ 添加到 S 中去;
- ❖ 把新产生的集合 $T = \{ \sim L, S \}$ 化成子句集;
- ❖ 应用消解原理, 力图推导出一个表示矛盾的空子句
- ❖



例 1

事实公式:

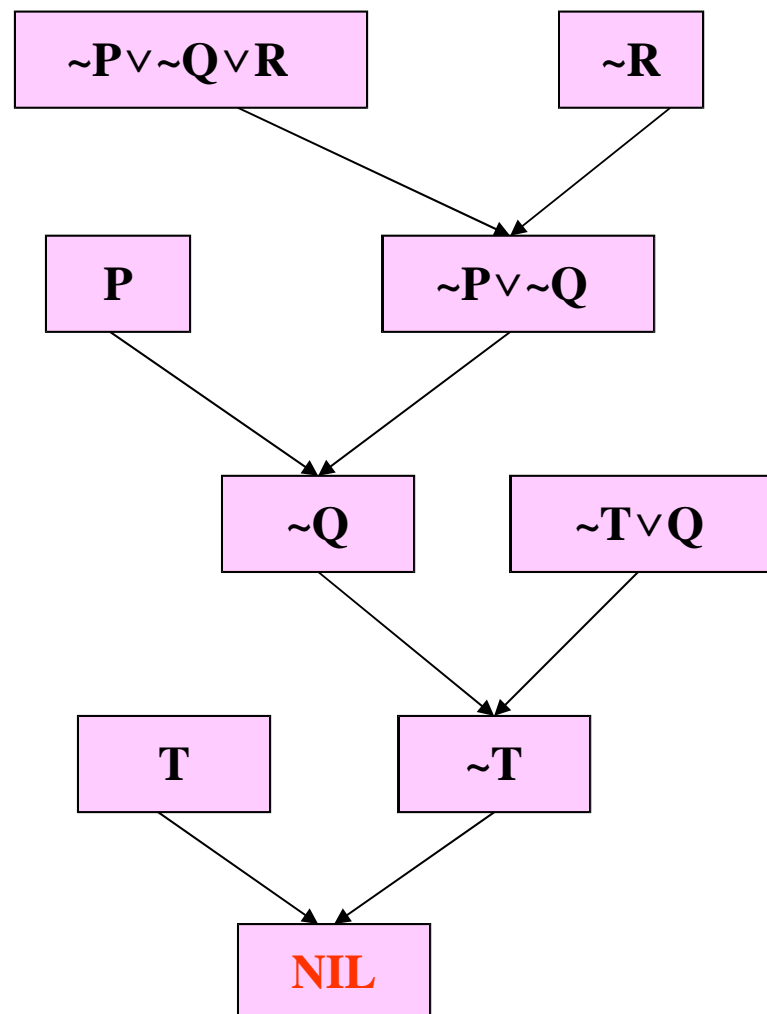
$\{P, (P \wedge Q) \Rightarrow R,$
 $(S \vee T) \Rightarrow Q, T\},$

证明结论R.

否定结论, 将公式化为子句,
得子句集:

$\{P, \sim P \vee \sim Q \vee R,$
 $\sim S \vee Q, \sim T \vee Q, T,$
 $\sim R\}$

消解反演树





例 2: Happy student

☼ “Happy Student”: Everyone who pass the computer test and win the prize is happy. Everyone who wish study or is lucky can pass all tests. Zhang doesn’t study, but he is lucky. Every lucky person can win the prize.

☼ Prove: Zhang is happy



求解Solution

Step1: 问题的一阶谓词逻辑表示

Facts or Knowledge:

$$(\forall x) (\text{Pass}(x, \text{computer}) \wedge \text{Win}(x, \text{prize})) \Rightarrow \text{Happy}(x)$$

$$(\forall x) (\forall y) (\text{Study}(x) \vee \text{Lucky}(x) \Rightarrow \text{Pass}(x, y))$$

$$\sim \text{Study}(\text{zhang}) \wedge \text{Lucky}(\text{zhang})$$

$$(\forall x) (\text{Lucky}(x) \Rightarrow \text{Win}(x, \text{prize}))$$

否定的结论: $\sim \text{Happy}(\text{zhang})$

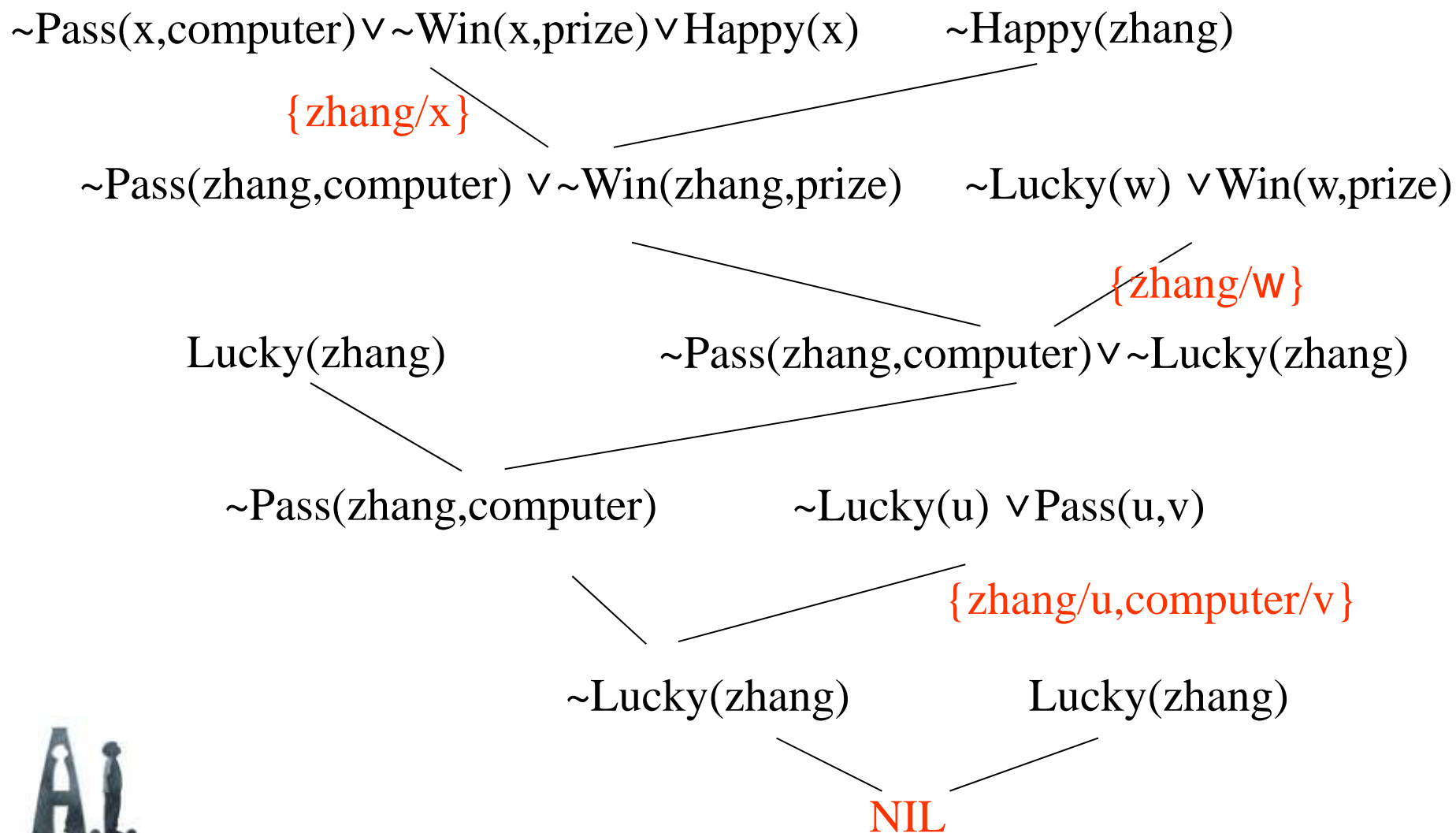


Step2: 将公式化为子句

- 1 $\sim \text{Pass}(x, \text{computer}) \vee \sim \text{Win}(x, \text{prize}) \vee \text{Happy}(x)$
- 2 $\sim \text{Study}(y) \vee \text{Pass}(y, z)$
- 3 $\sim \text{Lucky}(u) \vee \text{Pass}(u, v)$
- 4 $\sim \text{Study}(\text{zhang})$
- 5 $\text{Lucky}(\text{zhang})$
- 6 $\sim \text{Lucky}(w) \vee \text{Win}(w, \text{prize})$
- 7 $\sim \text{Happy}(\text{zhang})$



Step3: 消解反演求空子句





例3 储蓄问题

前提：每个储蓄钱的人都获得利息。

结论：如果没有利息，那么就没有人去储蓄钱

证明：

(1) 规定原子公式：

$S(x,y)$ 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 ”

(2) 用谓词公式表示前提和结论：

前提： $(\forall x)[(\exists y)(S(x,y)) \wedge M(y)] \Rightarrow [(\exists y)(I(y) \wedge E(x,y))]$

结论： $[\sim (\exists x)I(x)] \Rightarrow [(\forall x)(\forall y)(M(y) \Rightarrow \sim S(x,y))]$



(3) 化为子句形

把前提化为子句形：

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

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

把结论的否定化为子句形：

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

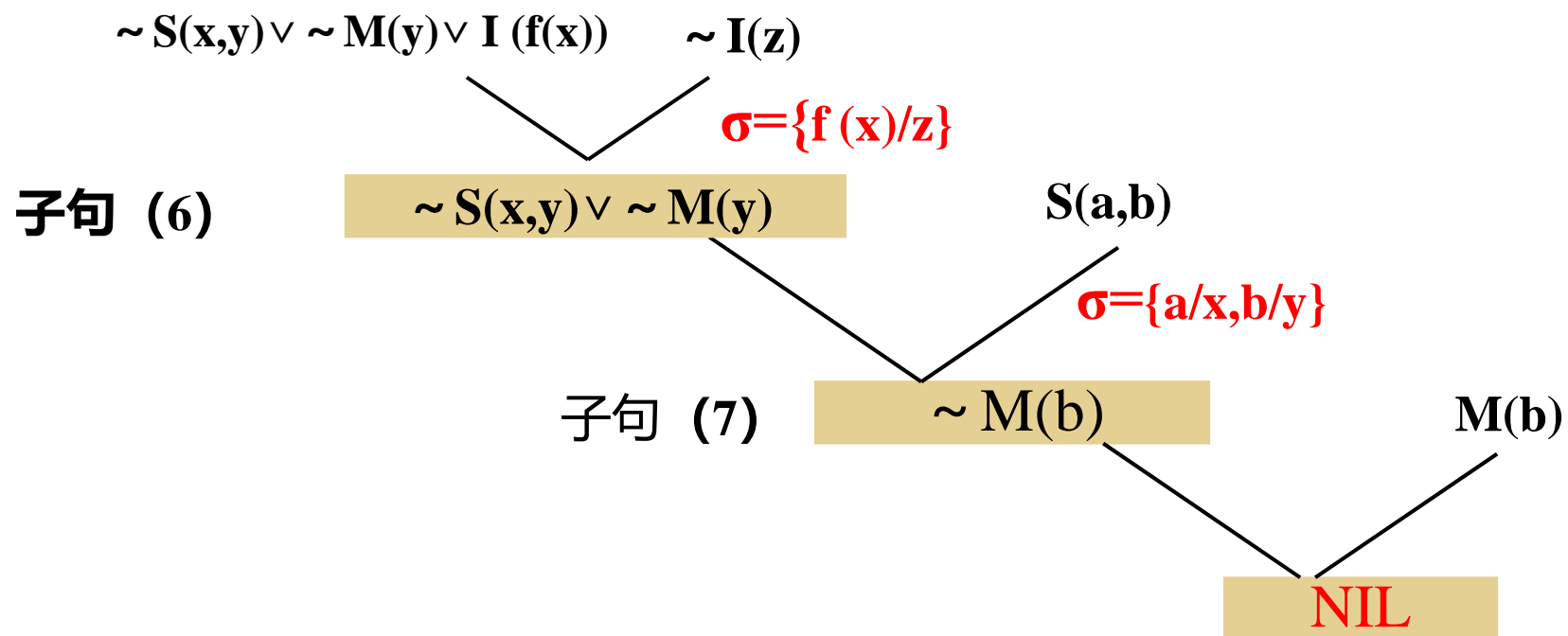
$$4) S(a,b)$$

$$5) M(b)$$





(4) 消解反演求空子句 (NIL)





反演求解

- 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。
- 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。
- 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**。



例 1

Given: “Zhang and Li are classmates; If x and y are classmates, then the classroom of x is the one of y ; Now Zhang is at 302 classroom.”

Question: “Now which classroom is Li at?”

Answer:

Define the predicates:

$C(x, y)$ x and y are classmates;

$At(x, u)$ x is at u classroom.

Represent the given facts as *wffs*:

$C(zhang, li)$

$(\forall x) (\forall y) (\forall u) (C(x, y) \wedge At(x, u) \Rightarrow At(y, u))$

$At(zhang, 302)$

Represent the Question's negation as *wffs*:

$\sim(\exists v) At(li, v)$ 即求 $v = ? ? ?$



谓词公式表示命题和事实 *wffs* :

● $C(\text{zhang}, \text{li})$

● $\sim C(x, y) \vee \sim At(x, u) \vee At(y, u)$

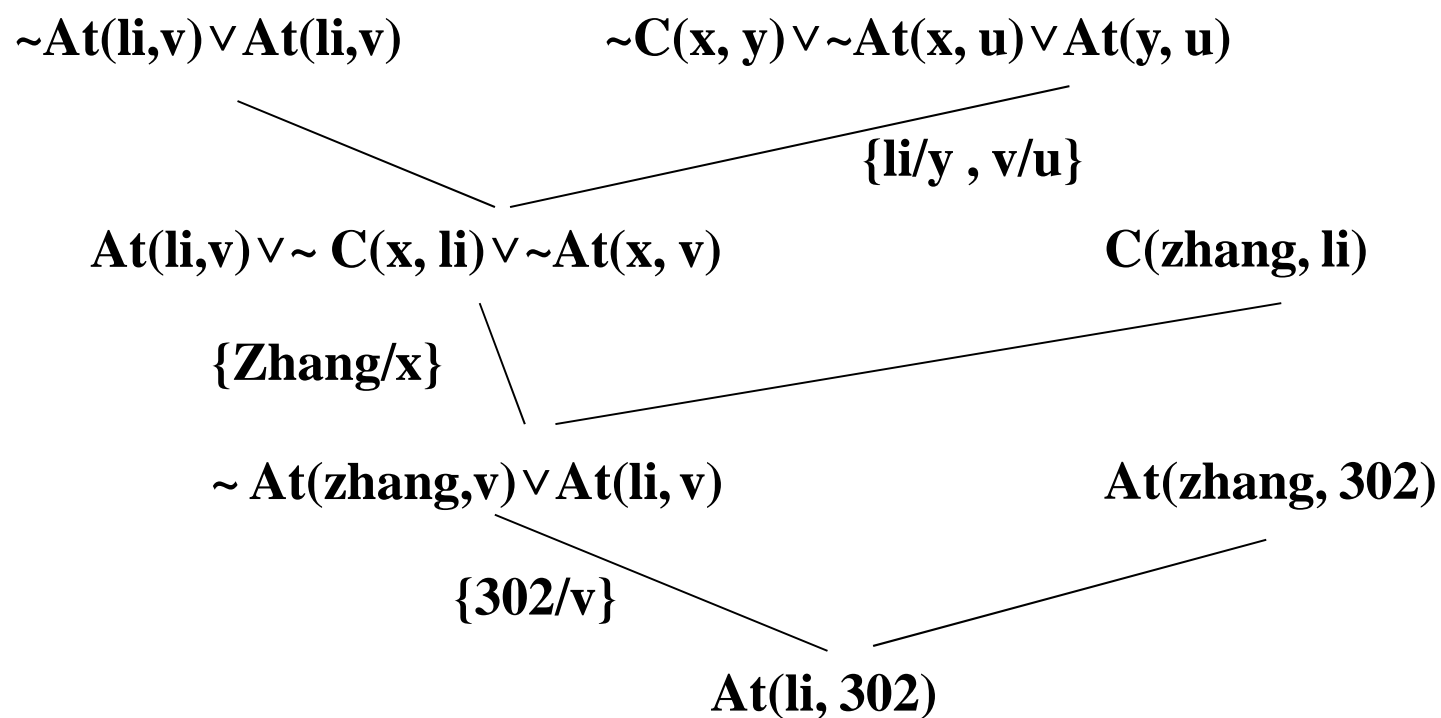
● $At(\text{zhang}, 302)$

转换目标否定子句和添加目标否定之否定:

$\sim At(\text{li}, v) \vee At(\text{li}, v)$



反演求解



用从根部求得答案



例 2

“如果无论John到哪里去，Fido也就去那里，那么如果John在学校里， Fido在哪里呢？”

解答：(1) 用谓词公式表示命题和事实：

事实S: $(\forall x)[AT(JOHN, x) \Rightarrow AT(FIDO, x)]$

$AT(JOHN, SCHOOL)$

目标: $(\exists x) AT(FIDO, x)$ 求x

(2) 用反演求解过程(1):

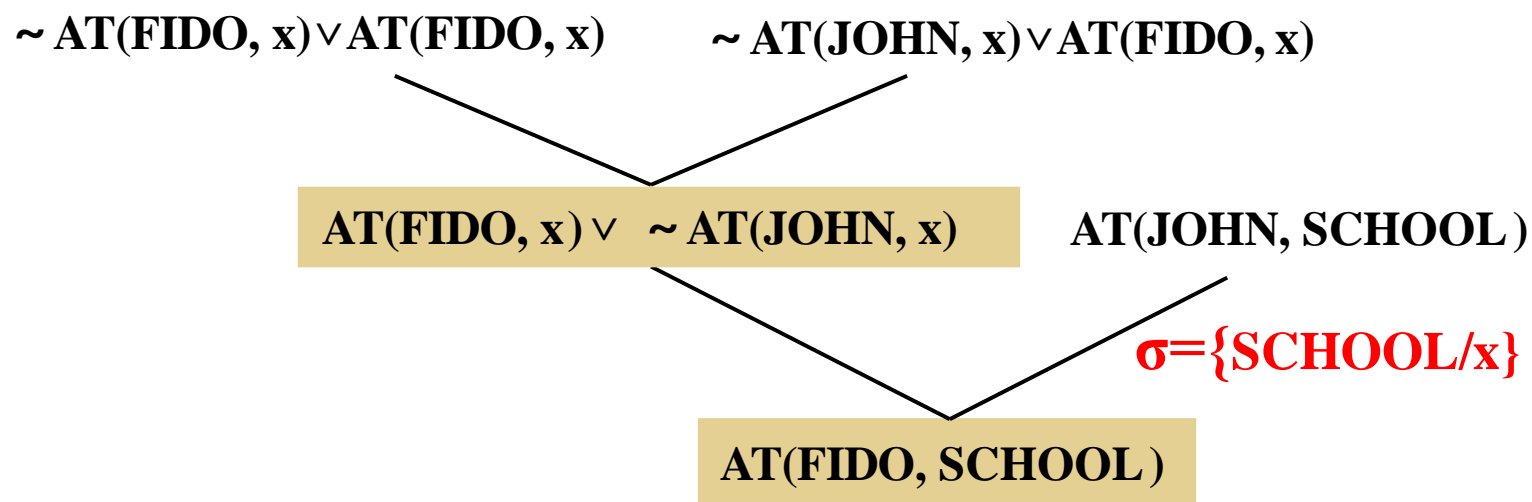
目标否定子句: $\sim AT(FIDO, x)$

将其添加到目标否定之否定的子句中:

$\sim AT(FIDO, x) \vee AT(FIDO, x)$



(3) 用反演求解过程(2) :



(4) 用从根部求得答案 $\text{AT}(\text{FIDO}, \text{SCHOOL})$



Homework 1 (40分)

✚ 教材p.115 习题3-15 (子句求取)



Homework 2 (30分)

❖ 假设已知下列事实：

- 1) 小李 (Li) 喜欢容易的 (Easy) 课程 (Course) 。 2
-) 小李不喜欢难的 (Difficult) 课程。
- 3 工程类 (Eng) 课程都是难的。
- 4 物理类 (Phy) 课程都是容易的。
- 5 小吴 (Wu) 喜欢所有小李不喜欢的课程。
- 6 Phy200是物理类课程。
- 7 Eng300是工程类课程。

❖ 请用消解反演法回答下列问题：

- 1 小李喜欢什么课程？
- 2 小吴喜欢Eng300课程吗？



Homework 3 (30分)

❁ 某公司招聘工作人员，A，B，C三人应试，经面试后公司表示：

- (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

问：公司一定录取谁？