人工智能的重要成果
归结原理

归结原理

- 几乎任何一本人工智能教材，都要讲“归结原理”
- 我们也介绍一下，主要是要靠你自己读一个晚上
- 不适合课堂上讲。我们之讲一下大体思路。考试时一定要考的。

- 搜索、归结原理是传统人工智能的主要成果
- 从归结原理发展出来了Prolog语言，当年日本“五代机”的机器语言。是人工智能衰落的标志

谓词演算及应用

- 是一种形式语言，具有严密的理论体系
- 是一种常用的知识表示方法
  - 例：
  - City（北京）
  - City（上海）
  - Age（张三，23）
  - $(\forall x)(\forall y)(\forall z)(F(x, y) \wedge F(y, z) \rightarrow GF(x, z))$

4.1 归结原理

- 归结原理是一种定理证明方法，1965年由Robinson提出，从理论上解决了定理证明问题。
- 归结原理的提出，对机器定理证明问题起到了推动作用。

子句集

- 无量词约束
- 元素只是文字的析取
- 否定符只作用于单个文字
- 元素间默认为合取
- 例：$\{\sim I(z) \vee R(z), I(A), \sim R(x) \vee L(x), \sim D(y)\}$

化子句集的方法

例：$(\exists z)(\forall x)(\exists y)\{[(P(x)\vee Q(x))\rightarrow R(y)]\vee U(z)\}$

1, 消蕴涵符

理论根据：$a\rightarrow b \Rightarrow \sim a\vee b$

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

2, 移动否定符

理论根据：$\sim(a\vee b) \Rightarrow \sim a\wedge \sim b$

$\sim(a\wedge b) \Rightarrow \sim a\vee \sim b$

$\sim(\exists x)P(x) \Rightarrow (\forall x)\sim P(x)$

【南开：不存在腐败，任意一个都不腐败】

$\sim(\forall x)P(x) \Rightarrow (\exists x)\sim P(x)$

【天津：并非所有人都腐败，存在一个不腐败的】

$(\exists z)(\forall x)(\exists y)\{[(\sim P(x)\wedge \sim Q(x))\vee R(y)]\vee U(z)\}$

化子句集的方法（续1）

3, 变量标准化

即：对于不同的约束，对应于不同的变量

$(\exists x)A(x) \vee (\exists x)B(x) \Rightarrow (\exists x)A(x) \vee (\exists y)B(y)$

4, 量词左移

$(\exists x)A(x) \vee (\exists y)B(y) \Rightarrow (\exists x)(\exists y)\{A(x) \vee B(y)\}$

5, 消存在量词 (skolem化）

原则：对于一个受存在量词约束的变量，如果他不受全程量词约束，则该变量用一个常量代替，如果他受全程量词约束，则该变量用一个函数代替。

$(\exists z)(\forall x)(\exists y)\{[(\sim P(x) \wedge \sim Q(x)) \vee R(y)] \vee U(z)\}$

$\Rightarrow (\forall x)\{[(\sim P(x) \wedge \sim Q(x)) \vee R(f(x))] \vee U(a)\}$

化子句集的方法（续2）

6, 化为合取范式

即$(a \vee b) \wedge (c \vee d) \wedge (e \vee f)$的形式

$(\forall x)\{[(\sim P(x) \wedge \sim Q(x)) \vee R(f(x))] \vee U(a)\}$

$\Rightarrow (\forall x)\{(\sim P(x) \wedge \sim Q(x)) \vee R(f(x)) \vee U(a)\}$

$\Rightarrow (\forall x)\{[\sim P(x) \vee R(f(x)) \vee U(a)] \wedge$

$[\sim Q(x)) \vee R(f(x)) \vee U(a)]\}$

7, 隐去全程量词

$\{[\sim P(x) \vee R(f(x)) \vee U(a)] \wedge [\sim Q(x)) \vee R(f(x)) \vee U(a)]\}$

化子句集的方法（续3）

8, 表示为子句集

{~P(x) ∨ R(f(x))∨U(a), ~Q(x))∨ R(f(x))∨U(a)}

9, 变量标准化（变量换名）

{~P(x1) ∨ R(f(x1))∨U(a), ~Q(x2))∨ R(f(x2))∨U(a)}

总括

- 只有六种符号： ∀， ∃； →， ∨， ∧， ~
- 简化成子句集
- 1.消掉蕴含
- 2.把否定的都压倒最后去
- 3. ∀∃，无关的变量取不同名字，从而统统移到最左边
- 4.去掉存在量词，无约束变常量，有约束变函数
- 5.把所有的存在量词都去掉
- 6. 第一层都变成“与”关系，一个个子句
  - 变成了“与”，就拆出来
- 第二层都变成“或”关系
- 非常像代数式的变换

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

归结原理

定理：

若S是合式公式F的子句集，则F永假的充要条件是S不可满足。

S不可满足：若nil$\in$S，则S不可满足。

使用归结原理证明定理的思路

目标的否定连同已知条件一起，化为子句集，并给出一种变换的方法，使得 $S \rightarrow S_1 \rightarrow S_2 \rightarrow \ldots \rightarrow S_n$同时保证当$S_n$不可满足时，有S不可满足。

4.2 归结方法（命题逻辑）

- 设子句：

  $C_1=L\vee C_1'$

  $C_2=(\sim L)\vee C_2'$

  则归结式C为：

  $C=C_1'\vee C_2'$

- 定理：

  子句集$S=\{C_1, C_2, \ldots, C_n\}$与子句集 $S_1=\{C, C_1, C_2, \ldots, C_n\}$的不可满足性是等价的。其中，C是$C_1$和$C_2$的归结式。

归结的例子

设公理集：

P,

$(P \wedge Q) \rightarrow R$,

$(S \vee T) \rightarrow Q$,

T

求证：R

- 化子句集：
- $(P \wedge Q) \rightarrow R$
- $\Rightarrow \sim(P \wedge Q) \vee R$
- $\Rightarrow \sim P \vee \sim Q \vee R$
- $(S \vee T) \rightarrow Q$
- $\Rightarrow \sim (S \vee T) \vee Q$
- $\Rightarrow (\sim S \wedge \sim T) \vee Q$
- $\Rightarrow (\sim S \vee Q) \wedge (\sim T \vee Q)$
- $\Rightarrow \{\sim S \vee Q, \sim T \vee Q\}$

子句集：

(1) P

(2) $\sim P \vee \sim Q \vee R$

(3) $\sim S \vee Q$

(4) $\sim T \vee Q$

(5) T

(6) $\sim R$（目标求反）

子句集：

(1) P

(2) ~P∨~Q∨R

(3) ~S∨Q

(4) ~T∨Q

(5) T

(6) ~R（目标求反）

归结：

(7) ~P∨~Q　　(2, 6)

(8) ~Q　　(1, 7)

(9) ~T　　(4, 8)

(10) nil　　(5, 9)

- 怎么理解、记住？
- $L \vee C_1'$，$(\sim L) \vee C_2'$ 归结式 $C_1' \vee C_2'$
- L有可能真，也可能假
- L为假时 C_1' 必为真；
- L为真时，$(\sim L)$ 必为假，C_2' 必为真
- 不管怎么样：$C_1' \vee C_2'$ 总能成立
- 但这样一变化，L没了。消掉了L

- 为什么总能证明出来?
- 如果各个式子都出现L，不出现~L，怎么办?
- 或者各个式子都出现~ L，不出现L，怎么办?
- 各个谓词都出现这种情况怎么能证明呢?

- 实际上不会出现的。想想，如果出现的都是L，令L为永真（相反的情况令L为永假）。各个式子都如此，各个子句都为真了。本来就推不出NIL
- 这不是一个严格的证明，但让你记住它

4.3 谓词逻辑的归结原理

- 问题：如何找归结对

  例：$P(x) \vee Q(y)$， $\sim P(f(y)) \vee R(y)$

  $P(A) \vee Q(y)$， $\sim P(f(y)) \vee R(y)$

- 基本概念
  - 置换

  $s=\{t_1/v_1, t_2/v_2, \ldots, t_n/v_n\}$

  对公式E实施置换s后得到的公式称为E的例，记作E_s。

  例：$s1=\{z/x, A/y\}$，则：

  $P[x, f(y), B]_s=P[z, f(A), B]$

- 合一

如果存在一个S置换，使得$\{E_i\}$中 $E_{1s}=E_{2s}=E_{3s}=\ldots=E_{ns}$，则称$\{E_i\}$是可合一的。S为$\{E_i\}$的合一者。

例：$\{P(x, f(y), B), P(z, f(B), B)\}$

置换$s=\{A/x, B/y, A/z\}$是一个合一者, 因为：

$P(x, f(y), B)_s= P(A, f(B), B)$

$P(z, f(B), B)_s= P(A, f(B), B)$

置换$s=\{z/x, B/y\}$和置换$s=\{x/z, B/y\}$也都是这两个谓词公式的合一者。

结论：合一者不唯一。

- 最一般合一者（mgu）

  置换最少，限制最少，产生的例最具一般性。

  如前面的例子：

  {P(x, f(y), B), P(z, f(B), B)}

  对于置换{A/x, B/y, A/z}，产生的例是：

  P(A, f(B), B)

  对于置换={z/x, B/y}，产生的例是：

  P(z, f(B), B)

- mgu也不是唯一的。

合一算法

例：{P(x, x, z), P(f(y), f(B), y)}

前缀表示：

(P x x z)

(P (f y) (f B) y)

置换：{(f y)/x}

(P (f y) (f y) z)

(P (f y) (f B) y)

置换：{B/y}, 并使得{(f B)/x}

(P (f B) (f B) z)

(P (f B) (f B) B)

置换：{B/z}

得到置换：{(f B)/x， B/y，B/z}

置换后的结果： (P (f B) (f B) B)

谓词逻辑的归结方法

- 对于子句$C_1 \vee L_1$和$C_2 \vee L_2$，如果L_1与$\sim L_2$可合一，且s是其合一者，则$(C_1 \vee C_2)_s$是其归结式。
- 例： $P(x) \vee Q(y)$，$\sim P(f(z)) \vee R(z)$

  $\Rightarrow Q(y) \vee R(z)$

归结举例

设公理集：

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

$(\forall x)(D(x)\rightarrow \sim L(x))$

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

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

化子句集：

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

$\Rightarrow (\forall x)(\sim R(x)\vee L(x))$

$\Rightarrow \sim R(x)\vee L(x)$ (1)

$$(\forall x)(D(x) \rightarrow \sim L(x))$$

$$\Rightarrow (\forall x)(\sim D(x) \vee \sim L(x))$$

$$\Rightarrow \sim D(x) \vee \sim L(x) \qquad (2)$$

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

$$\Rightarrow D(A) \wedge I(A)$$

$$\Rightarrow D(A) \qquad (3)$$

$$I(A) \qquad (4)$$

- 目标求反：

  $\sim(\exists x)(I(x)\wedge\sim R(x))$

$\Rightarrow (\forall x)\sim(I(x)\wedge\sim R(x))$

$\Rightarrow (\forall x)(\sim I(x)\vee R(x))$

$\Rightarrow \sim I(x)\vee R(x)$ (5)

换名后得字句集：

$\sim R(x_1)\vee L(x_1)$

$\sim D(x_2)\vee\sim L(x_2)$

$D(A)$

$I(A)$

$\sim I(x_5)\vee R(x_5)$

例题的归结树

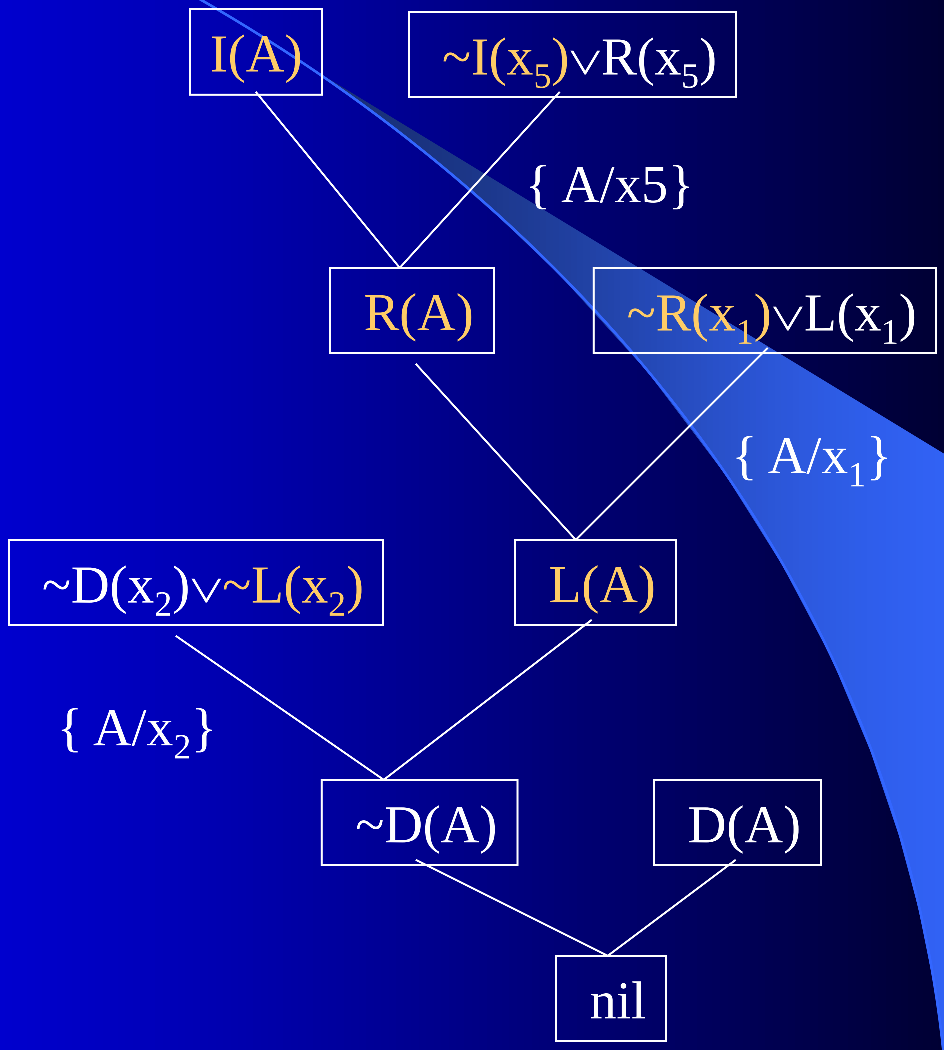
$\sim R(x_1) \vee L(x_1)$

$\sim D(x_2) \vee \sim L(x_2)$

$D(A)$

$I(A)$

$\sim I(x_5) \vee R(x_5)$



4.4 基于归结的问答系统

- 例：

  已知：$(\forall x)[AT(John, x) \rightarrow AT(Fido, x)]$

  $AT(John, School)$

  求证：$(\exists x)AT(Fido, x)$

  子句集：

  $\sim AT(John, x_1) \vee AT(Fido, x_1)$

  $AT(John, School)$

  $\sim AT(Fido, x_2)$

子句集： $\sim AT(John, x_1) \vee AT(Fido, x_1)$

$AT(John, School)$

$\sim AT(Fido, x_2)$

$AT(Fido, x_2) \vee$ $\boxed{\sim AT(Fido, x_2)}$ $\boxed{\sim AT(John, x_1) \vee AT(Fido, x_1)}$

$\{x2/x1\}$

$\boxed{AT(John, School)}$ $\boxed{\sim AT(John, x_2)}$ $\vee AT(Fido, x_2)$

$\{School/x_2\}$

$\boxed{nil}$ $AT(Fido, School)$

提取回答的过程

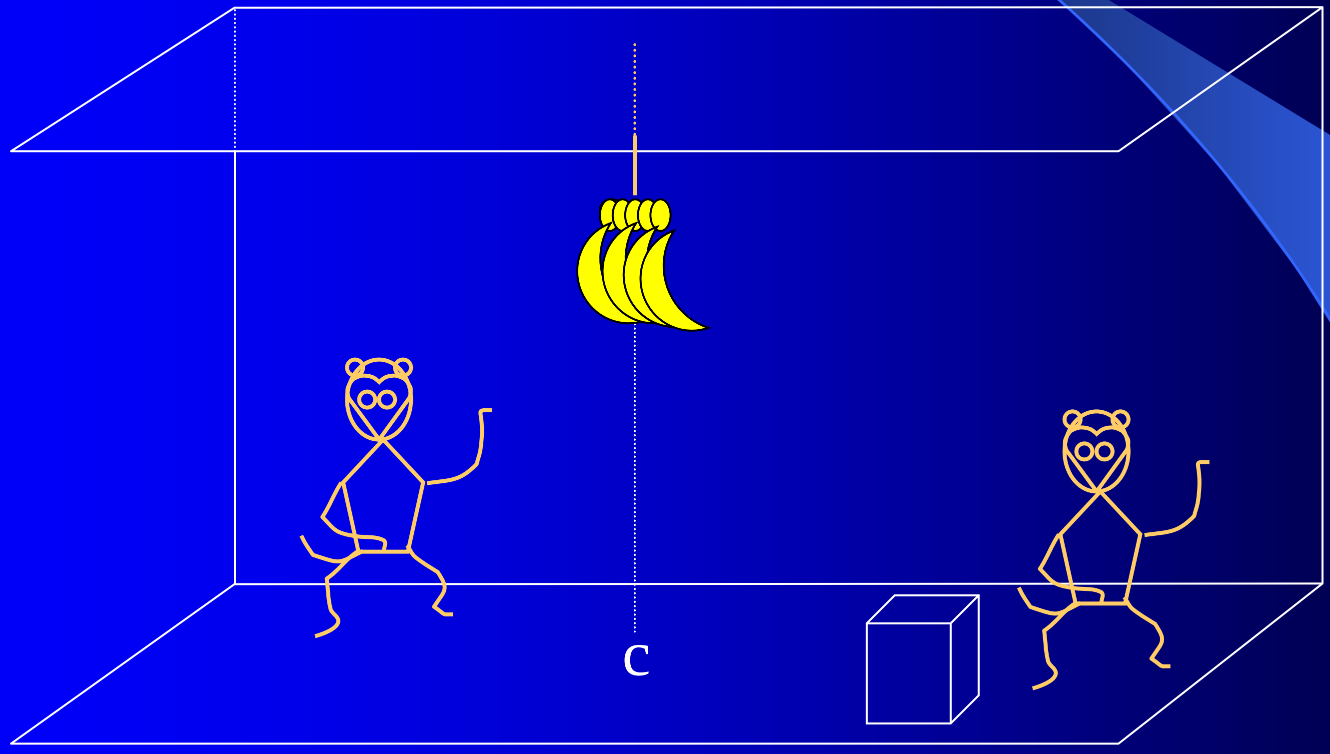
- 先进行归结，证明结论的正确性；
- 用重言式代替结论求反得到的子句；
- 按照证明过程，进行归结；
- 最后，在原来为空的地方，得到的就是提取的回答。
- 修改后的证明树称为修改证明树

- 还有一种办法，就是“或”上一个ANSWER（x）。寻求NIL变成寻求得到ANSWER（值）。这时候答案就是 x=值
- 理解成，只有 x=值 的时候，才推出矛盾。因为用的是反证法，这时才成立

- 这也是用推理获得一般性答案的一个方法
- 虽然现在不常用，但是很重要的
- 使得我们第一次觉得：通过机械的方法，还真能得到某种智能

- 我们学的人工智能原理，最深印象就是猴子摘到了香蕉

例：猴子摘香蕉问题



问题的表示

已知：

1, $\sim ON(s_0)$

2, $(\forall x)(\forall s)(\sim ON(s) \rightarrow AT(box, x, push(x, s)))$

3, $(\forall s)(ON(climb(s)))$

4, $(\forall s)((ON(s) \wedge AT(box, c, s)) \rightarrow HB(grasp(s)))$

5, $(\forall x)(\forall s)(AT(box, x, s) \rightarrow AT(box, x, climb(s)))$

求解：$(\exists s)HB(s)$

问题的子句集

1, $\sim ON(s_0)$

2, $ON(s_1) \vee AT(box, x_1, push(x_1, s_1))$

3, $ON(climb(s_2))$

4, $\sim ON(s_3) \vee \sim AT(box, c, s_3) \vee HB(grasp(s_3))$

5, $\sim AT(box, x_4, s_4) \vee AT(box, x_4, climb(s_4))$

6, $\sim HB(s_5)$

1, $\sim ON(s_0)$

2, $ON(s_1) \vee AT(box, x_1, push(x_1, s_1))$

3, $ON(climb(s_2))$

4, $\sim ON(s_3) \vee \sim AT(box, c, s_3) \vee HB(grasp(s_3))$

5, $\sim AT(box, x_4, s_4) \vee AT(box, x_4, climb(s_4))$

6, $\sim HB(s_5)$

$HB(s_5) \vee$ | $\sim HB(s_5)$ | $\sim ON(s_3) \vee \sim AT(box, c, s_3) \vee HB(grasp(s_3))$

$\{grasp(s_3)/s_5\}$

$HB(grasp(s_3)) \vee$ | $\sim ON(s_3) \vee \sim AT(box, c, s_3)$ | $ON(climb(s_2))$

$\{climb(s_2)/s_3\}$

$\sim AT(box, c, climb(s_2))$ | $\vee HB(grasp(climb(s_2)))$

$\sim ON(s_0)$ | $ON(s_1) \vee AT(box, x_1, push(x_1, s_1))$

$\{s_0/s_1\}$

$AT(box, x1, push(x1, s0))$

$\{x_4/x_1, push(x_4, s_0)/s_4\}$

$\sim AT(box, x_4, s_4) \vee AT(box, x_4, climb(s_4))$

$AT(box, x_4, climb(push(x_4, s_0)))$

$\{c/x_4, push(c, s_0)/s_2\}$

NIL | $HB(grasp(climb(push(c, s_0))))$

- 归结原理作为六十年代里程碑式的成果
- 大家要了解个大概
- 有用时再仔细读一读

- 基本原理与解线性方程有些类似
- 虽然有些枯燥，但归结原理是很巧妙的
- 记住了解线性方程，基本上就记住了归结原理