ISSN1000-9825 Journal of Software 软件学报 2000,11(1):91~94

多模式下的反合一算法*

许锡春[1,2] 胡运发[1] 施伯乐[1]

[1](复旦大学计算机科学系 上海 200433)

[2](上海海运学院计算中心 上海 200135)

E-mail: xcxu@shmtu.edu.cn

摘要 证明了多模式环境下存在最小反合一,提出了一种反合一算法,并介绍了多模式下反合一的应用.

关键词 合一,反合一,多模式,算法,相似.

中图法分类号 TP301

合一操作计算的是最大下界,而与合一对应的反合一操作计算的是最小上界.合一操作在定理证明、自动推理等方面起到重要的作用,反合一同样也是一种非常有用的操作,可以应用于类似推理、程序重写、机器学习等方面.这些年来,对反合一的研究正在加强.反合一最早是由 Plotkin[1]和 Reynolds[2]分别提出来的.与合一一样,反合一也存在各种形式.最近,许多学者对各种形式的反合一进行了研究.Page[3]研究了基于各种约束形式的反合一,基于约束的反合一可以避免过分抽象(overgeneralization).Hasker[4]研究了二阶反合一,提出二阶反合一可以包含更多的信息,但不存在二阶最小反合一,并通过增加限制来保证得到明确的反合一.Pfenning[5]研究了基于类型的、以 λ 演算为基础的反合一,并提出类型也是一种约束,同样可以避免过分抽象.这些研究都有一个共同的特点:逻辑结构都是在单模式下,即对于谓词 P,其模式 $P(a_1,a_2,\ldots)$ 是唯一的.但是,在不同的应用中可以有多种模式,例如,在面向对象的系统中允许操作重载;在类似推理中,有时一个问题可以有多种描述形式.当然,也可以用不同的谓词来表示,但是这样做,一方面,其共性不能表示;另一方面,必使谓词数目成倍增加.

本文对多模式下的反合一进行了研究,证明了在多模式下存在着最小反合一,并提出了计算最小反合一的算法.

1 基本概念

我们把所有变量的集合记为 V,常量的集合记为 C,类型的集合记为 T.在此基础上,我们定义有关概念.

定义 1. 模式.谓词 P 的模式为 $P(x_1:t_1,x_2:t_2,\ldots,x_n:t_n)$,其中 x_i 为变量,t_i 为类型.

我们采用表达式即类型的思想,把每一谓词也看作类型,即谓词 P 也称为类型 P,并且模式也可以表示为 $P(t_1(x_1),t_2(x_2),\ldots,t_n(x_n))$,因此,模式中的参数也可以是另一模式,类似于关系数据库中的非第一范式.我们可以把()看作是一种操作.

定义 2. 多模式.谓词 P 存在多个模式,其表示可以有以下两种形式:

$$P(t_1(x_1),t_2(x_2),\ldots,t_n(x_n));$$

$$P(t_1(x_1),t_2(x_2),\ldots,[t_i(x_i)],\ldots,t_n(x_n)),$$

其中[ ]表示可选.

---

* 作者许锡春,1965年生,博士,副研究员,主要研究领域为数据库系统,知识库系统,基于案例的推理.胡运发,1946年生,教授,博士生导师,主要研究领域为数据库系统,知识库系统,人工智能.施伯乐,1935年生,教授,博士生导师,主要研究领域为数据库理论及应用,知识库系统.

本文通讯联系人:许锡春,上海 200135,上海海运学院计算中心

本文1998-09-25收到原稿,1999-02-01收到修改稿

定义 3. 导出模式. 假设 P 的模式为 $P(t_1(x_1),t_2(x_2),\ldots,[t_i(x_i)],\ldots,t_n(x_n))$,则 $P(t_1(x_1),t_2(x_2),\ldots,t_i(x_i),\ldots,t_n(x_n))$ 和 $P(t_1(x_1),t_2(x_2),\ldots,t_{i-1}(x_{i-1}),t_{i+1}(x_{i+1}),\ldots,t_n(x_n))$ 都是模式 $P(t_1(x_1),t_2(x_2),\ldots,[t_i(x_i)],\ldots,t_n(x_n))$ 的导出模式.

定义 4. 模式包含. 对于模式 $P_1(x_1,x_2,\ldots,x_n)$,$P_2(y_1,y_2,\ldots,y_m)$,若 P_1,P_2 相同,对任何 $i\in(1,n)$,存在 $j\in(1,m)$,$x_i=y_i$,且对于 $i_1<i_2$,有 $j_1<j_2$.

定义 5. 模式的合并. 若模式 $P_1(x_1,x_2,\ldots,x_n)$,$P_2(y_1,y_2,\ldots,y_m)$ 为 P 的导出模式,则 P_1,P_2 的合并模式 $P_0(z_1,z_2,\ldots,z_n)$ 为 P 的导出模式中包含模式 P_1,P_2 的最小模式.

定理 1. 模式的合并是唯一的.

定义 6. 实例. 对于模式 P 中各参数,用相同类型的项代入,即为 P 的实例. $\varnothing$ 表示空,只能代入可选的参数.

定义 7. 项. 项可以有以下几种形式:(1) $\varnothing$ 为项;(2) 常量为项;(3) 变量为项;(4) 模式 P 的实例为项.

2 反合一问题

反合一计算的是两个实例的最小上界. 为了比较大小,类似于合一运算,可以利用替换来定义一个实例序.

定义 8. 实例序. 对于两个项 t,s,若存在替换 σ,使得 $\sigma t=s$,则定义实例序为 t 大于 s,记为 $t\geqslant s$.

合一采用项来替换变量,得到共同的实例. 而与之相反,我们也可以用变量来代替项,得到共同的抽象,这就是反合一.

定义 9. 反合一. 对于两个项 t,s,若存在替换 σ,θ 及项 g,使得 $\sigma g=t,\theta g=s$,则 g 为 t,s 的反合一.

合一操作一般计算最大的共同实例(基于实例序),同样地,反合一操作一般计算最小反合一.

定义 10. 最小反合一. 若 g 为 t,s 的反合一,对于任何 t,s 的反合一 q,有 $g\leqslant q$,则称 g 为 t,s 的最小反合一.

定义 11. 多模式下的反合一. 对于谓词 P 的模式的两个实例 t,s,若存在替换 σ,θ 及 P 的模式的实例 g,使得 $\sigma g=t,\theta g=s$,则 g 为 t,s 在多模式下的反合一.

由于 P 也是一种类型,允许 g 为 $P(x)$,其中 x 为变量. 替换可以有以下几种形式:(1) $x\to\varnothing$,(2) $x\to Q(\ldots)$,(3) $x\to(t_1,t_2,\ldots,t_n)$.
但在最后一种情况下,要求 $P(t_1,t_2,\ldots,t_n)$ 为 P 的实例.

根据文献[1,4]的研究,在单模式下,有下面的结论.

定理 2. 存在最小的一阶反合一.

定理 3. 不存在最小的二阶反合一.

在多模式下,我们可以得出定理 4.

定理 4. 多模式下存在最小反合一.

证明:对于模式 P 的实例 A,B,可以有下列 3 种情况:

(1) A,B 为 P 的同一模式的实例.

(2) A,B 分别为 P 的两个导出模式的实例.

(3) A,B 为 P 的两个模式的实例(即除了(1),(2)之外的情况)

我们分别对这 3 种情况进行证明.

对于情况(1),根据定理 2,存在最小反合一.

对于情况(2),假设 P 的两个导出模式为 P_1,P_2. 首先可以得到 P_1,P_2 的并集 P_0,根据定理 1,P_0 是唯一的. 然后,把 A,B 映射为 P_0 的实例 A_0,B_0. 根据定理 2,只要证明 A 到 A_0 的映射是唯一的即可. 由于 P_0 是 P_1,P_2 的合并,因此,P_0 包含 P_1,对于 P_0 的属性 i,若其存在于 P_1 中,$A_{0i}=A_i$;若其不存在于 P_1 中,取 $A_{0i}=\varnothing$. 显然,这样构造的 A_0 是唯一的.

对于情况(3),假设 A,B 的形式为 $P(a_1,a_2,\ldots,a_n)$,$P(b_1,b_2,\ldots,b_m)$,n 和 m 可以不同. 根据定义 11,$P(x)$ 是 A,B 的反合一. 下面只要证明 $P(x)$ 为最小反合一即可.

设 $P(x)$ 不是最小反合一,则必定存在另一反合一 $P(y_1,y_2,\ldots,y_k)$,且 k 一定大于 1. 由于 A,B 为不同模式

的实例,因此 A,B 中必存在 i,使得 a_i,b_i 的类型不一致,在替换中,y_i 不能是 $y_i\to a_i,y_i\to b_i$,只能是 $y_i\to\varnothing,y_i\to b_i$(或 $y_i\to a_i,y_i\to\varnothing$).如果剩余部分能匹配(类型一致),则 B 的模式必包含 A 的模式,这与条件不符.如果剩余部分还是不能匹配,那么同样地,若存在反合一 $P(y_1,y_2,\ldots,y_k)$,则 A 的模式与 B 的模式必为另一模式的导出模式.因此不存在形为 $P(y_1,y_2,\ldots,y_k)$的反合一,即 $P(x)$为最小反合一. □

3 反合一算法

根据上节的介绍,在多模式下,存在最小反合一,其求解算法如下.

算法 1. 反合一算法 AU

输入:谓词 P 的模式及 P 的模式的两个实例 α_1,α_2

输出:α_1,α_2 的最小反合一 α

(1) 若 α_1,α_2 为谓词 P 的同一模式的实例,则

$$AU(\alpha_1,\alpha_2)=P(\ldots AU(x_i,y_i)\ldots);$$

(2) 若 α_1,α_2 为谓词 P 的一个模式的不同导出模式 P_1,P_2 的实例,

计算 P_1,P_2 的合并模式 P_0,

把 α_1,α_2 转化为 P_0 的实例 α_1^1,α_2^1,则

$$AU(\alpha_1,\alpha_2)=AU(\alpha_1^1,\alpha_2^1)=P(\ldots AU(x_i^1,y_i^1)\ldots);$$

(3) 若 α_1,α_2 为谓词 P 的不同模式 P_1,P_2 的实例,则

$$AU(\alpha_1,\alpha_2)=P(x),$$

其中 x 为一个新变量.

定理 5. 算法 AU 是正确的.

证明:可以从定理 4 的证明直接导出.

4 应 用

多模式下的反合一可应用于许多方面.下面介绍利用多模式下的反合一来计算两个英文句子的相似性.

对于 t,s 的反合一 g,也就是 t,s 的概括(抽象),g 与 t,s 越接近,说明 t,s 越相似,t 与 s 的相似性可以用替换 δ,θ 来刻画:

$$sim(t,s)=1-\frac{|\delta|+|\theta|}{|t|+|s|}.$$

我们采用文献[6]介绍的衍生句法中的深层结构来表示英文句子,采用模式描述短语规则.例如,句子 S 的一个模式为

$$S(NP(x_1),Aux(x_2),VP(x_3),[Adv-x(x_4)]).$$

例 1:"You and John are happy"与"I studied English yesterday"这两个句子可表示为

$S(NP(NP(Pron(You)),NP(N(John)),Aux(Tns(Pres)),VP(Cop(be),AP(Adj(happy)))),$

$S(NP(Pron(I)),Aux(Tns(past)),VP(V(study),NP(N(English))),Adv\text{-}t(NP(N(yesterday)))).$

反合一的结果为

$S(NP(x_1),Aux(Tns(x_2)),VP(x_3),Adv\text{-}t(x_4)),$

$\delta=\{x_1\to NP(Pron(You)),NP(N(John);x_2\to Pres;x_3\to Cop(be),AP(Adj(happy);x_4\to\varnothing\},$

$\theta=\{x_1\to Pron(I);x_2\to past;x_3\to V(study),NP(N(English));x_4\to NP(N(yesterday))\}.$

相似度为

$$sim(t,s)=1-\frac{12+11}{17+17}=1-\frac{23}{34}=\frac{11}{34}=0.324.$$

例 2:"They wrote letters yesterday"与"I studied English yesterday"这两个句子可以表示为

$S(NP(Pron(They)),Aux(Tns(past)),VP(V(write),NP(N(letter))),Adv\text{-}t(NP(N(yesterday))))$,

$S(NP(Pron(I)),Aux(Tns(past)),VP(V(study),NP(N(English))),Adv\text{-}t(NP(N(yesterday))))$.

反合一的结果为：

$S(NP(pron(x_1)),Aux(Tns(past)),VP(v(x_2),NP(N(x_3))),Adv\text{-}t(NP(N(yesterday))))$,

$\delta=\{x_1\to They;x_2\to write;x_3\to letter\}$,

$\theta=\{x_1\to I;x_2\to study;x_3\to English\}$.

相似度为

$$sim(t,s)=1-\frac{3+3}{17+17}=1-\frac{6}{34}=\frac{28}{34}=0.8235.$$

从直觉上看，第1组是完全不相似的，而第2组是很相似的，计算结果与我们的直觉是一致的.

5 结 论

反合一与合一一样，是一种非常有用的操作.本文针对多模式，对反合一问题进行了研究，证明了在多模式下存在最小反合一，并提出计算最小反合一的算法.本文还介绍了用多模式下的反合一算法作为案例的相似性计算的方法，这一方法正应用于目前正处于实现阶段的基于案例的英汉翻译系统中.

参考文献

1 Plotkin G D. A note on inductive generalization. In: Meltzer B, Michie D eds. Machine Intelligence 5. Edinburgh: Edinburgh University Press, 1970. 153～163

2 Reynolds J C. Transformational systems and the algebraic structure of atomic formulas. In: Meltzer B, Michie D eds. Machine Intelligence 5. Edinburgh: Edinburgh University Press, 1970. 135～151

3 Page C D. Anti-Unification in constraint logics: foundations and applications to learn ability in first-order logic, to speed-up learning, and to deduction [Ph. D. Thesis]. Urbana-Champaign, Illinois: University of Illinois at Urbana-Champaign, 1993

4 Hasker R W. The replay of program derivations [Ph. D. Thesis]. Urbana-Champaign, Illinois: University of Illinois at Urbana-Champaign, 1995

5 Pfenning F. Unification and anti-unification in the calculus of constructions. In: Kahu G ed. Proceedings of the 6th Annual IEEE Symposium on Logic in Computer Science. Los Alamitos, CA: IEEE Computer Society Press, 1991. 74～85

6 Lin Tian-ming, Li Gui-wan. Generative grammar of English. Guangzhou: Guangdong People's Press, 1983
(林天明，李桂婉.英语衍生句法.广州：广东人民出版社，1983)

Anti-Unification Algorithm Based on Multi-Pattern

XU Xi-chun[1,2]　HU Yun-fa[1]　SHI Bai-le[1]

[1](*Department of Computer Science　Fudan University　Shanghai*　200433)

[2](*Computing Center　Shanghai Maritime University　Shanghai*　200135)

Abstract　In this paper, the existence of the least anti-unification in multi-pattern is proved and an algorithm for the anti-unification is proposed. In addition, the application of the anti-unification in multi-pattern is also introduced.

Key words　Unification, anti-unification, multi-pattern, algorithm, similarity.