Computer Science and Application 计算机科学与应用, 2011, 1, 51-56
doi:10.4236/csa.2011.12011 Published Online September 2011 (http://www.hanspub.org/journal/csa/)

The Study of Resolution Principle Based on Predicate Logic

Youyun Ao
School of Computer and Information, Anqing Teachers College, Anqing
Email: youyun.ao@gmail.com
Received: May 12th, 2011; revised: Jul. 11th, 2011; accepted: Jul. 25th, 2011.

Abstract: Resolution principle based on predication logic is an effective approach to implementing machine reasoning or automated reasoning, and applying it can prove theorems and extract answers to questions. This paper analyzes the theoretical basis of resolution principle based on predicate logic, and discusses some methods and procedures of applying resolution principle based on predicate logic to theorem proving and problem's answer extracting.

Keywords: Predicate Logic; Resolution Principle; Artificial Intelligence; Knowledge Representation

基于谓词逻辑的归结原理研究

敖友云
安庆师范学院，计算机与信息学院，安庆
Email: youyun.ao@gmail.com
收稿日期：2011 年 5 月 12 日；修回日期：2011 年 7 月 11 日；录用日期：2011 年 7 月 25 日

摘　要：基于谓词逻辑的归结原理是实现机器推理或自动推理的有效途径，利用它可以证明定理和提取问题答案。分析了基于谓词逻辑的归结原理的理论基础，并论述了将基于谓词逻辑的归结原理应用于定理证明和问题答案提取的方法及步骤。

关键词：谓词逻辑；归结原理；人工智能；知识表示

1. 引言

谓词逻辑是在命题逻辑的基础上发展而来的，通过引入量词，它比命题逻辑能更有效地表示和求证复杂问题。谓词逻辑采用形式化语言系统，通过一定的推理规则和控制策略，研究前提和结论之间的蕴涵关系。谓词逻辑具有严格的理论基础，可以保证推理过程和结论的正确性，同时它的形式化语言接近人类的自然语言，容易为人类所理解和接受[1]。

经典的演绎推理系统具有证明过程自然、易于理解、推理规则丰富、推理过程灵活等优点，但也存在推理过程中容易产生组合爆炸、证明方法难以判定等缺点[2,3]。Robinson[4]于 1965 年借助演绎推理反证法的思想提出了归结原理，它是一种形式单一、处理规则简单，可以在机器上实现的逻辑推理技术。归结原理的提出，为人类提供了一种简单易行的方法实现定理的证明和问题的求解，使定理证明可以在机器上机械现实，是自动推理的重大突破[5,6]。

本文从理论基础、求证问题的方法及步骤方面对基于谓词逻辑的归结原理进行了研究。论文的组织结构如下：第 2 节分析了基于谓词逻辑的归结原理的理论基础；第 3 节讨论了将基于谓词逻辑的归结原理应用于定理证明和问题答案提取的方法及步骤；最后对全文进行了总结。

2. 基于谓词逻辑的归结原理的理论基础

2.1. 归结原理的基本思想

根据谓词公式和其对应的子句集在不可满足的意义下是等价的，为了证明谓词公式是不可满足的，可

以转化为证明其对应的子句集是不满足的。Robinson归结原理的基本思想描述如下[4]：首先将待证明问题的结论的否定和问题的前提表示成谓词公式，然后将这些谓词公式转化为一个相应的子句集 S，接下来按一定的归结策略检验子句集 S 中是否含有空子句□，若含有空子句，则表明子句集 S 是不可满足的，问题得到证明；否则继续在子句集 S 中选择合适的子句对进行归结，并将归结式并入子句集 S 中，直至推出空子句或不能继续归结为止。

Robinson 归结原理中的子句(clause)是指任何文字的析取式，而空子句指的是不含任何文字的子句，记为□或 NIL，子句集 S 中各子句之间是合取关系，只要其中有一个子句是不可满足的，则整个子句集 S 就是不可满足的。因为空子句是不可满足的，所以在归结过程中若出现空子句，则整个子句集 S 就是不可满足的。归结原理的完备性是指子句集 S 是不可满足的，当且仅当存在一个从 S 到空子句的归结过程。

2.2. Skolem 函数

引入 Skolem 函数的目的是消去谓词公式中的存在量词。通常，谓词公式包含量词和母式两部分，在消去存在量词时要区分下列两种情况[7]：1) 若要消去的存在量词不位于任何一个全称量词的辖域内，则选用一个未曾出现在母式中的新常量去替代母式中受该存在量词约束的变量，然后删去该存在量词；2) 若要消去的存在量词位于一个或多个全称量词的辖域内，则选用一个未曾出现在母式中的新函数去替代母式中受该存在量词约束的变量，而该新函数的变量就是这些全称量词所约束的变量，然后删去该存在量词。例如，下列谓词公式：

$$(\exists x)(\forall y)(\forall z)(\exists u)(\exists v)\left(\neg P(x,y,z)\vee Q(x,u,v)\right)$$

消去存在量词后得到：

$$(\forall y)(\forall z)\left(\neg P(a,y,z)\vee Q\left(a,f(y,z),g(y,z)\right)\right)$$

其中 a 为常量，$x=a$，$u=f(y,z)$ 和 $v=g(y,z)$ 为 Skolem 函数。

通常，把消去存在量词时用到的新常量和新函数统称为 Skolem 函数，而把消去谓词公式中全部存在量词后得到的谓词公式称为 Skolem 范式。由于在消去存在量词时所用的 Skolem 函数可以不同，因此消去存在量词后得到的 Skolem 范式并不唯一。一般地，一个谓词公式与其 Skolem 范式并不等值，但若一个谓词公式是不可满足的，则其 Skolem 范式也一定是不可满足的，即引入 Skolem 函数并不影响原谓词公式的不可满足性。

2.3. 置换与合一

设 C_1 和 C_2 为两个不同的子句，L_1 和 L_2 分别为 C_1 和 C_2 中的两个文字，且 L_1 和 L_2 同属于一个谓词，不妨设 L_1 是原子，而 L_2 是原子的否定。若 L_1 和 $\neg L_2$ 中存在对应个体项不相同时，需要进行置换，解决匹配问题，才能使得推理继续进行。称 $\theta=\{t_1/x_1,\cdots,t_n/x_n\}$ 为有限集上的一个置换(substitution)[8],其中 $x_i\neq x_j\ (i\neq j)$，$i,j=1,2,\cdots,n$，t_i 为常量、变量或函数，t_i/x_i 表示用 t_i 替代 x_i，$t_i\neq x_i$，$i=1,2,\ldots,n$，且不允许 x_i 循环出现在另一个 t_j 中。例如，集合 $\{g(a)/x,f(x)/y\}$ 和 $\{a/x,c/y,f(b)/z\}$ 是两个置换，而集合 $\{g(y)/x,f(x)/y\}$ 不是一个置换。

置换可以进行合成运算，置换的合成也是置换。设有置换 $\theta=\{t_1/x_1,\cdots,t_n/x_n\}$，$\lambda=\{u_1/y_1,\cdots,u_m/y_m\}$，则 θ 与 λ 的合成(记为 $\theta\circ\lambda$)也是一个置换[9],通过下列方法得到：首先求出集合 $\{t_1\lambda/x_1,\cdots,t_n\lambda/x_n,u_1/y_1\cdots,u_m/y_m\}$，然后从中删去以下两种元素：当 $t_i\lambda=x_i$ 时，删去元素 $t_i\lambda/x_i\ (i=1,2,\cdots,n)$；当 $y_j\in\{x_1,x_2,\cdots,x_n\}$ 时，删去元素 $u_j/y_j\ (j=1,2,\ldots,m)$。例如，设置换 $\theta=\{f(y)/x,z/y\}$，$\lambda=\{a/x,b/y,y/z\}$，则置换的合成 $\theta\circ\lambda=\{f(b)/x,y/z\}$。

置换的目的是为了合一，而合一通过置换使得 L_1 和 $\neg L_2$ 保持一致。设 L_1 和 $\neg L_2$ 存在一个置换 θ，使得 $L_1\theta=\neg L_2\theta$，则称 $L_1,\neg L_2$ 是可合一的，θ 称为合一置换(unification)。若 $L_1,\neg L_2$ 有合一置换 σ，且对 $L_1,\neg L_2$ 的任意合一置换 θ 都存在置换 λ 使得 $\theta=\sigma\circ\lambda$，则称 σ 是 $L_1,\neg L_2$ 的最一般(最简单)合一置换，记作 mgu。两个公式的合一置换未必唯一，但它的最一般合一置换一定是唯一的。例如，置换 $\theta=\{c/x,b/y\}$ 是 $\{P(a,x,f(y)),P(a,x,f(b))\}$ 的一个合一置换，而它的最一般合一置换为 $\sigma=\{b/y\}$。

2.4. 二元归结

在合一置换的基础上可以对两个子句进行归结，

得到归结式。设 C_1 和 C_2 为两个不同的子句，L_1 和 L_2 分别为 C_1 和 C_2 中的两个文字且无公共变量。若 L_1 和 $\neg L_2$ 存在最一般合一置换 σ，则称 $C_{12}=\left(C_1\sigma-\{L_1\sigma\}\right)\cup\left(C_2\sigma-\{L_2\sigma\}\right)$ 为 C_1 和 C_2 的二元归结式(resolvent)，而称子句 C_1 和 C_2 为归结式 C_{12} 的亲本子句。文字 L_1 和 L_2 称为归结式上的文字[9]。例如，子句 $P(x)\vee Q(x)$ 和 $\neg Q(f(y))\vee R(x)$ 的二元归结式为 $P(f(y))\vee R(f(y))$，其中 $\sigma=\{f(y)/x\}$。

在对子句进行归结时若 L_1 和 L_2 具有公共变量，则需要先对 L_1 或 L_2 中的变量进行换名，才能通过合一置换对 C_1 和 C_2 进行归结。例如，$C_1=P(x)$，$C_2=\neg P(f(x))$，$S=\{C_1,C_2\}$ 是不可满足的。$L_1=P(x)$ 和 $L_2=\neg P(f(x))$ 分别 C_1 和 C_2 中的两个文字，若先对 L_2 进行换名为 $L_2=\neg P(f(y))$，则通过对 C_1 和 C_2 进行归结，得到空子句□，从而证明 S 是不可满足的。归结过程中得到的归结式 C_{12} 是其亲本子句 C_1 和 C_2 的逻辑结论。为了证明子句集 S 是不可满足的，只要对其中可归结的子句对进行归结，并把归结式并入子句集 S 中，或者用归结式替代它的亲本子句，然后将原问题转化为判定新子句集是不可满足的。

2.5. 提取子句

利用推理规则和运算法则可以将任一谓词公式转化为相应的子句，一般步骤如下[9]：

1) 消去蕴涵符号 $\rightarrow$ 和双向蕴涵符号 $\leftrightarrow$。反复使用联结词化归律，用 $\neg$，$\vee$，$\wedge$ 替代公式中的 $\rightarrow$ 和 $\leftrightarrow$。

2) 减少否定符号的辖域。反复使用双重否定律、摩根定律、量词转换律，将每个否定符号 $\neg$ 移到仅靠谓词的位置，使得每个否定符号最多只作用于一个谓词。

3) 变量标准化。在一个量词的辖域内，把谓词公式中受该量词约束的变量全部用另外一个未曾出现过的任意变量代替，使不同量词约束的变量具有不同的名字。

4) 化为前束范式。把所有的量词依顺序全部移到公式的前面。

5) 消去存在量词。利用 Skolem 函数消去谓词公式中的存在量词，将谓词公式化为 Skolem 范式。

6) 消去全称量词。消去全称量词，剩下谓词公式中的母式部分。

7) 把母式化为合取范式。反复使用分配律把母式化为合取范式。

8) 消去联结词 $\wedge$。在母式中消去所有合取联结词，把母式用子句的形式表示出来。

9) 更改变量名称。有时称为变量分离标准化，对子句中的某些变量重新命名，使得任意两个子句中不出现相同的变量名。例如，下列谓词公式：

$$(\forall x)\Big(P(x)\rightarrow\big((\forall y)\big(P(y)\rightarrow P(f(x,y))\big)\wedge\neg(\forall y)\big(Q(x,y)\rightarrow P(y)\big)\big)\Big)$$

可以转化为子句：$\neg P(x)\vee\neg P(y)\vee P(f(x,y))$

$\neg P(z)\vee Q(z,g(z))$

$\neg P(w)\vee\neg P(g(w))$

2.6. 归结策略

归结过程中使用的归结策略直接影响归结的效率和完备性。归结策略研究的是解决如何选择合适的子句对进行归结，以尽量减少不必要的归结的问题。归结策略大致可分为两类[9]：删除策略和限制策略。删除策略是指在归结过程中随着归结的不断进行，子句集中的子句越来越多，通过删除子句集中某些无效的子句，以节省存储空间和减少搜索子句对的时间，提高归结的效率。常用的删除策略包括重言式删除策略、单文字删除策略、纯文字删除策略和包孕删除策略。限制策略是指通过对参加归结的子句进行某些限制，以减少归结的盲目性，提高归结的效率。常用的限制策略包括线性归结策略、支持集策略、输入归结策略、单文字子句策略、祖先过滤策略、语义归结策略和宽度优先策略等。在实际应用中通常把几种归结策略结合起来使用，以提高归结的效率和完备性。

3. 基于谓词逻辑的归结原理的使用方法及步骤

3.1. 证明定理

利用归结原理证明定理的过程称为归结反演(resolution refutation)。归结反演的证明过程可以用步骤进行描述，也可以用归结反演证明树的形式来表达[10]。归结反演的证明步骤如下[4]：

1) 定义谓词并把待证明的问题的前提和结论分别表示成谓词公式 F 和 G，并否定结论公式 G 得到 $\neg G$；

2) 将谓词公式 F 和 $\neg G$ 转化为一个子句集 S；

3) 利用归结原理对子句集 S 中的子句进行归结，并把新得到的归结式并入子句集 S 中，如此反复进行，若子句集 S 中出现空子句□，则停止归结，此时 G 得证。

例如 4.1[7] 已知下列前提：

(1) 马库斯(Marcus)是一个人。

(2) 马库斯是一个庞贝人(Pompeian)。

(3) 所有的庞贝人都是罗马人(Romans)。

(4) 凯撒(Caesar)是统治者(ruler)。

(5) 所有的罗马人要么忠于凯撒要么憎恨凯撒。

(6) 每个人都忠于某人。

(7) 人们仅设法暗杀他们不忠的统治者。

(8) 马库斯设法暗杀凯撒。

求证结论：马库斯憎恨凯撒。

使用归结反演的证明步骤如下：

步骤 1：定义谓词：$man(x)$ 表示 x 是人；$Pompeian(x)$ 表示 x 是庞贝人；$Roman(x)$ 表示 x 是罗马人；$ruler(x)$ 表示 x 是统治者；$loyalto(x,y)$ 表示 x 忠于 y；$hate(x,y)$ 表示 x 憎恨 y；$tryassassinate(x,y)$ 表示 x 暗杀 y。用谓词公式表示前提：

(1) $man(Marcus)$

(2) $Pompeian(Marcus)$

(3) $(\forall x)(Pompeian(x)\rightarrow Roman(x))$

(4) $ruler(Caesar)$

(5) $(\forall x)(Roman(x)\rightarrow loyalto(x,Caesar)\vee hate(x,Caesar))$

(6) $(\forall x)(\exists y)loyalto(x,y)$

(7) $(\forall x)(\forall y)(man(x)\wedge ruler(y)\wedge tryassas\sin ate(x,y)\rightarrow \neg loytalto(x,y))$

(8) $tryassas\sin ate(Marcus,Caesar)$

用谓词公式 G 表示结论：$hate(Marcus,Caesar)$

并用谓词公式 $\neg G$ 表示结论的否定：

(9) $\neg hate(Marcus,Caesar)$

步骤 2：将上述前提谓词公式和结论谓词公式 G 的否定转化为子句集。

(1) $man(Marcus)$

(2) $Pompeian(Marcus)$

(3) $\neg Pompeian(x)\vee Roman(x)$

(4) $ruler(Caesar)$

(5) $\neg Roman(x)\vee loyalto(x,Caesar)\vee hate(x,Caesar)$

(6) $loyalto(x,f(x))$

(7) $\neg man(x)\vee\neg ruler(y)\vee\neg tryassas\sin ate(x,y)\vee\neg loytalto(x,y)$

(8) $tryassassinate(Marcus,Caesar)$

(9) $\neg hate(Marcus,Caesar)$

步骤 3：利用归结原理对对上述子句集进行归结推理。

(10) $\neg Roman(Marcus)\vee loyalto(Marcus,Caesar)$ (9)与(5) 归结，$\sigma=\{Marcus/x\}$

(11) $\neg Pompeian(Marcus)\vee loyalto(Marcus,Caesar)$ (10)与(3) 归结，$\sigma=\{Marcus/x\}$

(12) $loyalto(Marcus,Caesar)$ (11)与(2)归结

(13) $\neg man(Marcus)\vee\neg ruler(Caesar)\vee\neg tryassassinate(Marcus,Caesar)$ (12)与(7)归结，$\sigma=\{Marcus/x,Caesar/y\}$

(14) $\neg ruler(Caesar)\vee\neg tryassassinate(Marcus,Caesar)$ (13)与(1) 归结

(15) $\neg tryassassinate(Marcus,Caesar)$ (14) 与 (4) 归结

(16) □ (15)与(8)归结

上述归结推理使用了线性归结策略，即每次归结得到的归结式直接参加同另一个子句进行的下一次归结，线性归结策略具有完备性。

3.2. 提取问题答案

归结反演不仅可以用于定理证明，而且可以用来提取问题的答案，其思想与定理证明类似，但具体有不同的方法。

3.2.1. 保留合一信息法

在归结反演的过程中，通过保留使用的合一置换信息，以便在归结反演结束时提取问题的答案。

例如 3.2 如果艾伦(*Allen*)在哪里 *Fido* 就跟到哪里，那么艾伦在图书馆，*Fido* 会在哪里？

使用保留合一信息法提取问题答案的步骤如下：

步骤 1：定义谓词： $AT(x,y)$ 表示 x 在 y 处。

前提：(1) $(\forall x)(AT(Allen,y)\to AT(Fido,y))$

(2) $AT(Allen,library)$

结论 G：$(\exists Z)AT(Fido,Z)$

结论的否定 $\neg G$：(3) $(\forall Z)\neg AT(Fido,Z)$

步骤 2：将上述前提谓词公式和结论谓词公式 G 的否定转化为子句集。

(1) $\neg AT(Allen,y)\vee AT(Fido,y)$

(2) $AT(Allen,library)$

(3) $\neg AT(Fido,Z)$

步骤 3：利用归结原理对对上述子句集进行归结推理。

(4) $\neg AT(Allen,y)$ (3)与(1)归结，$\sigma=\{y/Z\}$

(5) □ (4)与(2)归结，$\sigma=\{library/y\}$

然后利用归结反演过程中保留的合一信息提取问题答案，其过程如下：

$AT(Fido,Z)\xrightarrow{\sigma=\{y/Z\}}AT(Fido,y)\xrightarrow{\sigma=\{library/y\}}AT(Fido,library)$，由语句 $AT(Fido,library)$ 便可得 *Fido* 在图书馆。

3.2.2. 目标公式否定的子句的重言式法

这种方法在两个方面对归结反演的证明过程进行了修改[10]：一个是并入子句集 S 的是目标公式否定的子句与目标公式的子句的析取式，而不是目标公式的否定的子句；另一个是，在求得目标语句时就停止归结，而不是要等到出现空子句。对于例 3.2 采用这种方法提取问题答案的步骤如下：

步骤 1：定义谓词： $AT(x,y)$ 表示 x 在 y 处。

前提：(1) $(\forall x)(AT(Allen,y)\to AT(Fido,y))$

(2) $AT(Allen,library)$

结论 G：$(\exists Z)AT(Fido,Z)$

结论的否定 $\neg G$：(3) $(\forall Z)\neg AT(Fido,Z)$

步骤 2：将上述前提谓词公式和结论谓词公式 G 的否定转化为子句集。

(1) $\neg AT(Allen,y)\vee AT(Fido,y)$

(2) $AT(Allen,library)$

(3) $\neg AT(Fido,Z)\vee AT(Fido,Z)$

步骤 3：利用归结原理对对上述子句集进行归结推理。

(4) $\neg AT(Allen,y)\vee AT(Fido,y)$ (3)与(1)归结，$\sigma=\{y/Z\}$

(5) $AT(Fido,library)$ (4)与(2)归结，$\sigma=\{library/y\}$

由语句 $AT(Fido,library)$ 可得 *Fido* 在图书馆。

3.2.3. 目标公式否定的子句与谓词 $ANSWER(Z)$ 的析取式法

这种方法也在两个方面对归结反演的证明过程进行了修改：一个是并入子句集 S 的是目标公式否定的子句与谓词 $ANSWER(Z)$ 的析取式，而不是目标公式的否定的子句；另一个是，在求得答案 Z 时就停止归结，而不是要等到出现空子句。对于例 3.2 采用这种方法提取问题答案的步骤如下：

步骤 1：定义谓词： $AT(x,y)$ 表示 x 在 y 处。

前提：(1) $(\forall x)(AT(Allen,y)\to AT(Fido,y))$

(2) $AT(Allen,library)$

结论 G：$(\exists Z)AT(Fido,Z)$

结论的否定 $\neg G$：(3) $(\forall Z)\neg AT(Fido,Z)$

步骤 2：将上述前提谓词公式和结论谓词公式 G 的否定转化为子句集。

(1) $\neg AT(Allen,y)\vee AT(Fido,y)$

(2) $AT(Allen,library)$

(3) $\neg AT(Fido,Z)\vee ANSWER(Z)$

步骤 3：利用归结原理对对上述子句集进行归结推理。

(4) $\neg AT(Allen,y)\vee ANSWER(Z)$ (3)与(1)归结，$\sigma=\{y/Z\}$

(5) $ANSWER(library)$ (4)与(2)归结，$\sigma=\{library/y\}$

所以本题的答案为 *Fido* 在图书馆。

4. 结束语

谓词逻辑是一种有效的知识表示方法和高级知识推理工具。它既有严格的理论基础，又能在机器上处理精确性知识，而它的表达方式和人类自然语言又非常接近。因此，用谓词逻辑表示知识容易为人们所理解和接受。Robinson 归结原理是一种可以在机器上机

械实现的逻辑推理技术，是实现自动推理证明的主要途径。本文分析了基于谓词逻辑的归结原理的基本思想、相关的推理技术及归结策略，并通过实例讨论了利用基于谓词逻辑的归结原理证明和求解问题的方法及步骤。研究基于谓词逻辑的归结原理在机器上的实现技术；利用基于谓词逻辑的归结原理证明和求解一系列实际生活或生产中的问题；进一步扩充基于谓词逻辑的归结原理并用于处理不确定性知识等都是有意义的研究方向。

参考文献 (References)

[1] C. L. Chang, R. C. Lee. Symbolic logic and mechanical theorem proving. New York: Academic Press, 1973.

[2] 卢延鑫. 经典逻辑在人工智能知识推理中的应用[J]. 软件导刊, 2008, 7(1): 22-24.

[3] 潘美芹, 丁志军, 王永丽. 谓词逻辑推理中证明方法的判定[J]. 山东科技大学学报(自然科学版), 2005, 24(4): 84-86.

[4] J. A. Robinson. A machine-oriented logic based on the resolution principle. Journal of the ACM, 1965, 12(1): 46-54.

[5] 肖启莉, 肖启敏. 归结原理及其应用[J]. 计算机与数字工程, 2007, 35(5): 183, 187.

[6] 杨球, 孙宝林. 归结原理及其在数学定理证明中的应用[J]. 武汉交通科技大学学报, 2000, 24(4): 417-420.

[7] E. Rich, K. Knight. Artificial intelligence (Second Edition). New York: McGraw Hill, 1991.

[8] 刘云霞, 王迤冉, 宋玉杰. 谓词逻辑描述下的归结推理方法[J]. 周口师范学院学报, 2003, 20(2): 63-66.

[9] 王文杰, 史忠植. 人工智能原理辅导与练习[M]. 北京: 清华大学出版社, 2007.

[10] D. Luckham, N. J. Nilsson. Extracting information from resolution proof trees. Artificial Intelligence, 1971, 2(1): 27-54.

