相干命题逻辑自然推理系统 NR 的自动证明*

郭远华，曾振柄
(华东师范大学 上海市高可信计算重点实验室，上海 200062)

摘　要：给出了相干命题逻辑自然推理系统 NR 的自动证明算法。首先将待证命题公式 A 的子公式组成一个初始集合 P，对其中的元素采用系统 NR 的推理规则得到新的命题公式加入 P，当得到秩为 0 的 A 时命题得证；然后对 A 的证明树进行整理即得到演绎序列。对系统 NR 的大部分定理证明取得了良好的效果，算法生成的演绎序列清晰可读，接近手工推理。

关键词：相干命题；自然推理；自动证明；可读证明
中图分类号：TP181　　文献标志码：A　　文章编号：1001-3695(2009)10-3639-03
doi:10.3969/j.issn.1001-3695.2009.10.010

Automated reasoning for natural deduction system NR of relevance propositional logic

GUO Yuan-hua, ZENG Zhen-bing
(Shanghai Key Laboratory of Trustworthy Computing, East China Normal University, Shanghai 200062, China)

Abstract: This paper presented an automated reasoning algorithm for natural deduction system (NR) of relevance propositional logic. Sub-formulas of formula A composed an initial set P, and added the new formulas produced by applying deducing rules of system NR among elements of P to P. Proved proposition A if A was produced and its rank was zero. Then arranged the reasoning tree of A and achieved the deduction sequence. This algorithm is effective for most theorems in system NR. The deduction sequences created by the algorithm are readable and similar to human prooves.

Key words: relevance proposition; natural deduction; automated reasoning; readable proof

自动推理是人工智能研究的基础工作，可以分为基于代数的几何定理证明与基于逻辑的自动推理两个范畴。我国学者吴文俊建立的“吴方法”和张景中提出的面积法、消点法在几何定理证明方面取得了瞩目的成就。在几何定理机器可读证明中，有前推搜索法[1]、后推搜索法[2]、双向搜索法、反证法等。基于逻辑的自动推理证明方法有多种，如归结法[3]、语义 tableau 法[4]、公理系统证明法、自然推理证明法、相继式演算证明法等。不同的方法对于不同的逻辑系统的处理能力各有优劣。

自然推理的发端始于波兰数学家、逻辑学家雅斯科夫斯基和德国数学天才、逻辑学家甘岑。自然推理的核心[5]是引入假设然后撤除假设的做法。弗雷格、罗素和希尔伯特等人发展起来的公理系统，其证明过程是从系统的公理出发，根据少量的推论规则推出一系列定理。与之不同，自然推理系统可以不借助公理，通过引入假设(包括前提)作为推演的出发点，运用推演规则推出另外一些命题，并通过撤除假设使这些被推出的命题独立于该假设。当除前提以外的假设都被撤除，所给推论的有效性便被证明；如果没有前提，所证结果便是本系统的一个定理。加拿大人工智能专家、逻辑学家 Pelletier 指出[6]：“自然演绎是当代哲学家们最为熟悉的一种逻辑，甚至它是许多当代哲学家们关于逻辑所知道的一切。”相干命题逻辑[7]的主要代表是系统 R，初始符号和形成规则类似于经典逻辑，其中连接词“→”解释为相干蕴涵。与 R 相对应的自然推理系统是 NR，13 条推理规则分别是：假设规则、蕴涵引入规则→I、蕴涵消去规则→E、重复规则、重述规则、析取引入规则∨I、析取消去规则∨E、合取引入规则∧I、合取消去规则∧E、分配规则、否定引入规则～I、换质位规则、双重否定消去规则～～E。

本文将自然推理系统中的推理序列竖着排列，并在序列中每一个公式 A_i 的左方划上(若干条)竖线，竖线的根数就是该公式的秩，记为 $R(A_i)$。例如，$|A_1$ 表示为 1，$\|A_2$ 表示 $R(A_2)$ 为 2。若 A_i 的左方没有竖线，则意味着 $R(A_i)$ 为 0。A_i 对应的相干标记集记为 $I(A_i)$，系统 NR 中的定理 A 记为 $\vdash_{NR} A$。对系统 NR 的详细描述参见文献[7]。

1　换质位规则的扩展

文献[7]对换质位规则的描述为：若 $A_1, A_2, \cdots, A_k (k \geqslant 3)$ 是一个长度为 k 的推理，A_i 和 $A_j (1 \leqslant i < j < k)$ 是其中位于 A_k 之前的某两个公式。A_k 的形式为 $\sim A_i$，$I(A_i) = a$，A_j 是用假设规则引入的，$I(A_j) = \{n\} (n \geqslant 1)$，$I(A_k) = b$ 且 $n \in b$，$R(A_i) = n-1$，$R(A_j) = R(A_k) = n$，对每一个 $A_m (m = i+1, i+2, \cdots, j-1)$，都有 $R(A_m) \geqslant n-1$，对每一个 $A_t (t = j+1, j+2, \cdots, k-1)$，都有 $R(A_t) \geqslant n$，则 $A_1, A_2, \cdots, A_k, A_{k+1}$ 可构成一个长度为 $k+1$ 的推理。这里的 A_{k+1} 是 $\sim A_j$，$I(A_{k+1}) = a \cup b - \{n\}$，$R(A_{k+1}) = n-1$。这时称用换质位规则由 A_i、A_j 和 A_k 推出了 A_{k+1}，如图 1 所示。

收稿日期：2009-02-08；修回日期：2009-04-03　基金项目：国家自然科学基金重点资助项目(90718041)

作者简介：郭远华(1978-)，男，湖北天门人，博士研究生，主要研究方向为计算机自动推理(gyhua2003@126.com)；曾振柄(1963-)，男，甘肃皋兰人，教授，博导，主要研究方向为计算机自动推理.

上述规则要求 A_i 在假设 A_j 之前，并且 $n\notin a$；扩展的换质位规则还包含的一种情形是 A_i 在 A_j 和 A_k 之间且 $n\in a$，如图 2 所示。

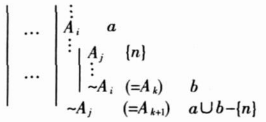

图 1　换质位规则情形 1

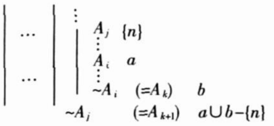

图 2　换质位规则情形 2

对扩展的换质位规则的正确性证明，只需要证明第二种情况的正确性，即若 $A\rightarrow B$ 且 $A\rightarrow\sim B$，则 $\sim A$。

证明：

a) $A\rightarrow B$	(已知条件)
b) $(A\rightarrow B)\rightarrow((A\rightarrow\sim B)\rightarrow\sim A)$	(定理)
c) $(A\rightarrow\sim B)\rightarrow\sim A$	a)b) MP
d) $A\rightarrow\sim B$	(已知条件)
e) $\sim A$	c)d) MP

2　主要数据结构

用数字代表命题公式中的各连接词、变元、公式有助于程序设计的简洁。文中公式 A、B、C、D、…、Z 分别对应的数字编号为 97、98、99、100、…、122，连接词 ～、∧、∨、→ 对应的数字编号为 200、195、190、185。公式 $A\rightarrow A\vee B$ 的数字序列为 97、185、97、190、98。

如图 3 和 4 所示，公式 $A\rightarrow A\vee B$ 和 $B\rightarrow A\vee B$ 有相同的右子公式。为了减小冗余，将公式写成二叉树的形式，用数组 ExpList 存储所有树中的所有节点。ExpList 的元素为 TreeNode 结构体：

```
struct TreeNode{
    int data;
    int lch;
    int rch;
}
```

其中：data 是节点字符的数字编号；lch、rch 是 ExpList 中的下标，分别表示节点的左右孩子，值为 －1 表示没有对应的孩子。若 $A\rightarrow A\vee B$ 和 $B\rightarrow A\vee B$ 是首先读取的两个公式，读入后 ExpList 如表 1 所示。

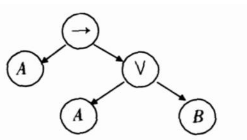

图 3　$A\rightarrow A\vee B$ 的二叉树

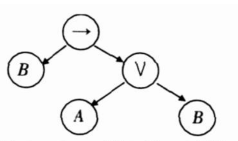

图 4　$B\rightarrow A\vee B$ 的二叉树

表 1　ExpList 的元素

节点字段	ExpList 的元素				
	0	1	2	3	4
lch	－1	－1	0	0	1
data	97	98	190	185	185
rch	－1	－1	1	2	2

表 1 中并排横列的 0、1、2、3、4 是 ExpList 的下标，每个下标下面竖列的三个数字是该下标对应的节点的 lch、data 和 rch。这样，ExpList[i] 就对应了一棵以它为根节点的二叉树，对树中序遍历就得到了下标 i 代表的逻辑公式，也就是说数组下标与逻辑公式一一对应。前述两公式在表 1 中的下标分别是 3、4，它们公共的子公式对应的下标是 2，中序遍历以 ExpList[3] 为根节点的二叉树就得到 $A\rightarrow A\vee B$ 的数字串形式。随着读入的逻辑式的增加，ExpList 动态增长，且读入次序不同，它的内容也不同。为了使得每个公式惟一对应一个下标，在向 ExpList 添加元素时要确保其中没有相同的元素。

ProofList 是证明序列数组，其元素为 ProofNode 结构。

```
struct ProofNode{
    int data;
    int lch;
    int mid;
    int rch;
    int rank;
    int rule;
    CUIntArray* pSet;
}
```

其中：data 为数组 ExpList 的下标，表示 ExpList[data] 对应的公式；lch、mid、rch 为数组 ProofList 的下标，分别表示公式的左中右孩子，初始值 －1 表示无对应孩子，若公式由规则 2、11、12 之一产生，则左孩子指向假设孩子公式；rank 为公式的秩；rule 为产生公式的规则，前述 13 条规则分别编号为 1～13；pSet 为指向 $I(A_i)$ 的指针。在下文中，用 $rule(A_i)$ 表示 A_i 的产生规则。

3　算法

3.1　求证

求证命题公式 A 的思路是：A 的子公式组成集合 P，对 P 中的元素使用推理规则产生新公式加入 P，直到得到 A 且 $I(A)$ 为空。求证过程伪代码如下：

建立动态数组 ProofList，数组每个元素代表一个命题公式；

遍历待证命题公式 A 的二叉树，将遍历到的子节点 A_i 加入 ProofList 作为推理假设，$I(A_i)$ 的元素为 A_i 在 ProofList 中的下标；

设置 curIndex 为 0，topIndex 为 ProofList 中当前元素数量（即添加的假设数）；

while(curIndex ≤ topIndex){

ProofList[0]，…，ProofList[curIndex] 对应的公式组成集合 P，ProofList[curIndex] 与公式 ProofList[j]（j ≤ curIndex）依次根据 10 条推理规则（假设、重复、重述规则除外，假设规则前面已经添加，重复规则、重述规则在得到初始证明序列后处理）产生 n 个新公式 A_{i1}，A_{i2}，…，A_{ij}，…，A_{in} 及对应的 $I(A_{i1})$，$I(A_{i2})$，…，$I(A_{ij})$，…，$I(A_{in})$，topIndex 增加 n。若 $A=A_{ij}$ 且 $I(A_{ij})$ 为空，则命题得证，否则 curIndex 增加 1。

}

若 A 得证，则得到以 A 为根节点的证明树。

3.2　整理

对得证的公式 A，对其证明树递归后序遍历得到初始的证明序列数组 ShowList，其元素也为 ProofNode。遍历的同时，设置每一个公式的 rank 值。首先设置 A 的 rank 为 0，A_i 是 A_j 的父节点，若 $rule(A_i)$ 为 2、11 或 12，则 $R(A_j)=R(A_i)+1$，否则 $R(A_j)=R(A_i)$。

序列需要作以下六步的调整：

a）换质位规则的第一种情形会产生图 5 的序列。其中 A_{i-1} 是对 A_i 的引用，需要将 $\sim A_j$ 的中孩子 A_{i-1} 删除整理成图 1 的形式以符合人们的阅读习惯。对 $\sim A_j$ 中孩子的重定位在序列 A_0，…，A_k，A_{k+1}，…，A_{j-1}（$0\leqslant R(A_{k+1})-R(A_k)\leqslant 1$）中由后向前地查找。

以下各步按照秩由高到低顺序逐层处理。

b）调整同一命题重复嵌套假设，如式 $((A\rightarrow A)\rightarrow(A\rightarrow B))\rightarrow(A\rightarrow B\vee C)$ 的初始证明序列，如图 6 所示。将两个以同一假设初始的嵌套子证明序列调整为并列，第二个 B 采用重

复规则引用第一个，如图 7所示。

c)处理重述规则。对于证明序列中 rank=k的公式 A_i 在序列 $A_0, \cdots, A_j, A_{j+1}, \cdots, A_{i-1}$ $(0 \leqslant R(A_{j+1}) - R(A_j) \leqslant 1)$ 中由后向前查找与之相同的公式 A_j，若有则标记 A_i 的 rule为 5(重述规则)，若 $R(A_j) < k-1$ 还要在前述序列中查找第一个 $R(A_k) = k-1$，并在其后添加公式 A_i。

d)处理重复规则。若 A_i 由蕴涵引入规则→I或换质位规则第一种情形推导且其左右孩子是同一公式则标记 ShowList[i−1] (A_i 的右孩子)的 rule为 4(重复规则)。见后文例 2。

e)对于同一竖线右边有多个相同公式的情况，如图 8所示，保留其中第一个和 rule为 4的命题，删除其他的。

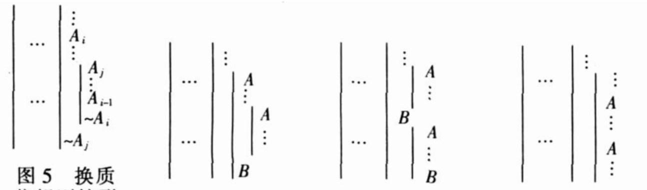

图 5 换质位规则情形 1产生的初始序列　图 6 同一假设嵌套　图 7 同一假设并列　图 8 同一秩序列上相同的公式

经过上述处理，ShowList中公式的左中右孩子需要重定位。对 A_i 的孩子重定位方法是由后向前在序列 $A_0, \cdots, A_j, A_{j+1}, \cdots, A_{i-1}$ $(0 \leqslant R(A_{j+1}) - R(A_j) \leqslant 1)$ 中查找左中右孩子。

f)调整 $I(A_i)$ 的元素。求证部分的伪代码说明若初始假设公式在 ProofList中的下标组成集合 S，则 $I(A_i) \in S$。在证明序列中各假设的 $I(A_i)$ 都不同，不符合阅读习惯，需要调整 $I(A_i)$。这个过程的伪代码如下：

```
定义整数数组 nArray，添加 -1；
for(ShowList中每一个的元素 ShowList[i])
{
  if(ShowList[i].rule==1)
  将 I(Ai)元素添入 nArray
  if(ShowList[i].rule==2, 11, 12)
  删除 nArray的尾元素；
  查找 ShowList[i].pSet(即 I(Ai))中的每一个元素在 nArray中
对应的下标，这些下标构成新的集合 I'(Ai)并用 I'(Ai)取代 I(Ai)；
}
```

向 nArray中添加 −1是为了随后查找到的下标从 1开始。

4 实例

例 1 $\vdash_{NR} (A \to B) \to \sim(A \wedge \sim B)$

证明

a)	$A \to B$	{1}	假设
b)	$A \wedge \sim B$	{2}	假设
c)	A	{2}	b合取消去
d)	$A \to B$	{1}	a重述
e)	B	{1, 2}	c, d蕴涵消去
f)	$\sim B$	{2}	b合取消去
g)	$\sim(A \wedge \sim B)$	{1}	b, e, f换质位
h)	$(A \to B) \to \sim(A \wedge \sim B)$		a, g蕴涵引入

例 2 $\vdash_{NR} A \to \sim\sim A$

证明

a)	A	{1}	假设
b)	$\sim A$	{2}	假设
c)	$\sim A$	{2}	b重复
d)	$\sim\sim A$	{1}	a, b, c换质位
e)	$A \to \sim\sim A$		a, d蕴涵引入

例 3 $\vdash_{NR} (A \to B) \wedge (C \to D) \to (A \vee C \to B \vee D)$

证明

a)	$(A \to B) \wedge (C \to D)$	{1}	假设
b)	$A \vee C$	{2}	假设
c)	$(A \to B) \wedge (C \to D)$	{1}	a重述
d)	A	{3}	假设
e)	$(A \to B) \wedge (C \to D)$	{1}	c重述
f)	$A \to B$	{1}	e合取消去
g)	B	{1, 3}	d, f蕴涵消去
h)	$B \vee D$	{1, 3}	g析取引入
i)	$A \to B \vee D$	{1}	d, h蕴涵引入
j)	C	{3}	假设
k)	$(A \to B) \wedge (C \to D)$	{1}	c重述
l)	$C \to D$	{1}	k合取消去
m)	D	{1, 3}	j, l蕴涵消去
n)	$B \vee D$	{1, 3}	m析取引入
o)	$C \to B \vee D$	{1}	j, n蕴涵引入
p)	$B \vee D$	{1, 2}	b, i, o析取消去
q)	$A \vee C \to B \vee D$	{1}	b, p蕴涵引入
r)	$(A \to B) \wedge (C \to D) \to (A \vee C \to B \vee D)$		a, q蕴涵引入

5 结束语

本文算法是系统 NR的 13条推理规则的代码实现，具有正确性。算法初始假设公式集合只包含公式的子公式，其完备性和终止性不能保证，如公式 $A \vee \sim A$ 的证明需以 $\sim(A \vee \sim A)$ 为假设之一，可以通过限制推理深度和时间确保算法的终止。在初始假设数量为 n，证明树的深度为 h的情况下，算法复杂度为 n^{h+1}。自然推理法在前推过程中会产生大量中间公式，对于复杂的命题公式会出现公式膨胀。与归结法、Tableau等方法相比，总的来说本文算法在效率上不占优势，其特点在于能够清晰地给出演绎序列，更接近手工推理，在对可读演绎序列有要求的场合，如教学演示有一定的应用空间。

参考文献：

[1] 郭四稳，李传中. 自动推理中反证法的研究[J]. 计算机应用与软件，2007，28(8)：41-43.
[2] 徐茜. 双向推理系统在初等几何自动解题中的实现[J]. 计算机应用研究，2004，21(11)：232-234.
[3] 王国俊. 数理逻辑引论与归结原理[M]. 北京：科学出版社，2006.
[4] 刘全，孙吉贵. 基于语义 Tableau的一阶逻辑自动定理证明[J]. 计算机工程与应用，2005，41(23)：22-24.
[5] 陈晓平. 自然演绎与逻辑教材[J]. 哲学动态，2006(增刊)：89-95.
[6] PELLETIER F J. A brief history of natural deduction[J]. History and Philosophy of Logic，1999，20(1)：1-31.
[7] 冯棉. 相干与衍推逻辑[M]. 上海：上海人民出版社，1993.