证明策略及其有效性问题

何 锫

(长沙交通学院)

PROOF STRATEGIES AND VALIDITY

He Pei

(*Changsha Communications Institute*)

ABSTRACT

INCAPS (INteractive Computer-Aided Proving System) is a proof system of temporal logic. This paper outlines its proof strategies' (tactics, tacticals) variety and structure. The kernel of it is to discuss the validity of tactics and tacticals. After introducing several new concepts: proof strategies' level number, function tree and B function defined on it, we show that INCAPS' proof strategies are valid.

摘 要

INCAPS(INteractive Computer-Aided Proving System) 是一个面向时序逻辑的交互式计算机辅助证明系统。本文简要介绍了其证明策略(tactics, tacticals) 的类别、结构，并在引入证明策略的层次数、函数树及树上B 函数等新概念前提下，深入探讨和证明了INCAPS 证明策略的有效性问题。

§1. 前言

证明一个定理通常有两种思维方式——向前推演和向后推演。如果从证明过程的树形描述看，前者的思绪行向是由叶节点(前提) 逼向根节点(结论)，后者则由根节点(目标)出发寻找相应的叶节点(子目标)。虽然许多场合下人们习惯于向前推演，但也有另一些场合尤其求解一些相当困难的问题时，我们还需借助于向后推演。

1989 年11 月24 日收到，1990 年3 月26 日定稿。本工作所属的INCAPS 系统得到了国家自然科学基金资助。

INCAPS(INteractive Computer-Aided Proving System) 是一个面向时序逻辑的交互式计算机辅助证明系统。在INCAPS 中研制证明策略还受到另一个重要因素的影响，这便是LCF(Logic for Computable Functions) 系统[1, 2, 3, 4, 5]。LCF 是关于可计算函数的逻辑系统，它包括五个部分: 元语言ML、核心逻辑PPλ(Polymorphic Predicate λ-calculus)、目标分解函数、优化机制及一些用于维护理论体系的命令。LCF 的特点之一是支持向后推演，这主要通过一组称为tactics 和tacticals 的证明策略实现的。

当然，在INCAPS 中研制开发证明策略有其内在的特性，文[6] 对它们进行了详细讨论。INCAPS 的核心逻辑是FOTL(First Order Temporal Logic), LCF 则为PPλ。正是这种逻辑的差异，使得INCAPS 与LCF 不仅在应用领域而且在实现技术上都有很大的不同。关于FOTL 读者可参见[7, 8, 9]。该系统与Kröger 公理系统[10] 是等价的。

§2. INCAPS 的证明策略: tactics tacticals

tactics 是向后推演的核心成分，它们大致分为三类: a) 规则依赖类; b) 拥有定理参数类; c) 简单分解类。所谓规则依赖是指这类tactics(函数) 的定义直接依赖着逻辑系统的具体推理规则。它们一般表征了规则的逆运算。如设有规则 $\frac{A\ \ A \to B}{B}$, 则关联于它的a 型tactic 把B 分解为/A 和A → B。后两种tactics 的设置可用于简化证明的书写。这三类tactics 统称基本tactics。基本tactics 均有效(研制过程保证)。

所以引入b 型tactics 还有一种特殊考虑。我们知道，a 型tactics 直接依赖于FOTL，它们通常仅具一个形参，而这个形参位置又只能填入待证目标。这就产生了一些新的问题: 如何将已证得的定理用于求解未知的目标? 如何引入关于某个问题所固有的公理假设? 因此，我们必须研制b 型tactics, 使得它们除拥有目标形参外，还拥有一个定理形参。INCAPS 中属于这类的tactics 有: Accept, Contr-tac, Asume-tac, Check-Assume-tac, Strip-Assume-tac 及Strip-tac。

尽管tactics 有类的差异，甚至对它们的理解还可拓广到包含复合tactics 在内，但它们之间也存在着统一的基础，那便是结构一致。下面是用ML 函数型程序语言[11] 给出的tactic 描述。

```
type
  validation=thm list → thm;
  goal=v of form list * from |
        h of form list * form;
  tactic=goal → (goal list * validation);
```

以上thm 表示定理类型，thm list 为表类型，其元素为定理。其它类此。

tacticals 是一些关于tactics 的操作算子，它可将若干简单tactics 集成为一个复合tactic。如对于这样一个tactical THEN, 当应用到tac_1, tac_2 两个tactic 上时，我们可得一复合tactic(THEN tac_1 tac_2)。INCAPS 系统的tacticals 有两类，它们分别是基本tacticals 和表tacticals。属前类的有THEN, ORELSE, Repeat 及THENL。属后类的有Every 和First。以下是用类ML 语言给出的THEN 的定义。

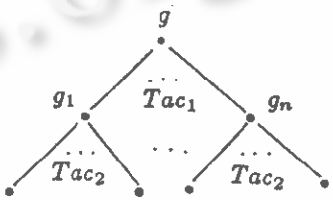
```
function THEN tac1 tac2 g
  = Let val ([g1, g2, ···, gm], v)=tac1 g
      in
```

```
Let val (G_i, v_i)=tac_2 g_i
      (for 0 < i < m+1)
            in
   (G_1 @ G_2 @ ··· @ G_m, V(v_1, ··· v_m))
 end
end;
```

这里@ 是表的拼接算子。$V(v_1,\cdots,v_m)$ 表示一个由诸$v_i(0<i<m+1)$ 按一定顺序复合而得的函数。

引入tacticals 的目的在于：简化证明的书写；节约计算的机时。不妨看下面图b 示例(约定：图a 是一种关于tactic 分解目标g 的表示法，其意义是Tac 应用到g 上可产生n 个子目标$g_i(1\le i\le n)$。由于Tac 此时能正常分解目标g, 故也说Tac 应用到g 成功)：

图a



图b

显然仅仅采用单步执行tactic 方式实现图b 形式的分解方案是相当繁琐的，不仅需反复调用目标管理系统(负责记录、组织、管理子目标) 诸命令，而且tactics 的书写工作达n+1 次之多。因此寻求一种简洁表达形式，研制能将若干简单tactics 集于一体的复合算子tactical 的任务便迫在眉睫。THEN tac_1 tac_2 g 就是引入tacticals 后对图b 分解方案的一种成功描述。

§3. INCAPS 证明策略的有效性问题

前面已简要介绍了INCAPS 证明策略的分类、结构及引入各类证明策略的理由，这里我们将深入探讨证明策略的有效性。这是一个非常重要的问题，虽然R.Milner、D.A. Schmidt 和L.C.Paulson 在讨论LCF 证明策略时对此分别在文[1]、[12] 及论著[5] 中予以了相当重视，但却未给出证明。本节试图对INCAPS 证明策略的有效性给出一种可能的证明方法。

定义1(可达)　给定一个定理$F_1\vdash W_1$ 和一个目标$F\ |?\!-\ W$, 如果它们间满足：$W_1=W$ 和$F_1\subseteq F$, 我们就说定理$F_1\vdash W_1$ 可达目标$F|?\!-\ W$, 并且记为：$F|?\!-\ W\leftarrow\!\!\!\!-\!\!\!\!-\ F_1\vdash W_1$。这里F、$F_1$ 均为公式的集合，而W、W_1 则为公式。

定义2(有效)　设T: $g\mapsto([g_1, g_2, \cdots, g_n], v)$ 是(复合)tactic, $g_i \xleftarrow{a} th_i(1\le i\le n)$, 那么如果$v|th_1, th_2, \cdots, th_n|$ 可达g, 则说T 有效。特别地n=0 时，要求v| | 可达g。

当然[5] 还提出了一个与有效性完全不同的概念——tactic 的保守性，但此处不予考虑。正如文中所说“一个证明策略必须是有效的，但不必保守的。”因为保守性要求结

论蕴涵前提，这在很多情形过于苛刻。如对应于下面一条平凡的规则就会有一个非保守的证明策略。

规则：$\dfrac{\Gamma \vdash A}{\Gamma \vdash A \vee B}$

显然，在tactical 模型中做向后推演，我们期望使用有效的tactics，不过遇上非有效tactics 的机率也是很高的。考察这样一个tactic:

$$T:\ F|?\!-\ A \wedge B \longmapsto ([F|?\!-\ A],\ \lambda t.\wedge I(t,\ B \vdash B))$$

$$\left\{\text{这里}\wedge I\text{表示规则}:\ \frac{\begin{array}{c}F_1 \vdash A\\ F_2 \vdash B\end{array}}{F_1 + F_2 \vdash A \wedge B}\right\}$$

它是非有效tactic，一旦经tacticals 与其它tactics 进行复合就会产生成千上万的非有效tactics。因此作为证明器，无论旨在处理古典逻辑还是时序逻辑，这个问题必须得到有效的控制和克服。

定理1　INCAPS 中基本tactics 及由THEN、ORELSE、THENL 和Repeat 等tacticals 复合而成的tactics 都是有效的。

在证明它之前，我们还得引入以下几个概念：(复合)tactic 的层次数、函数树及其上的B 函数。后者有助于INCAPS 中复合tactic 的validation 求值。

定义3(层次数)　设Tac 为(复合)tactic，则|Tac| 为：

a. 0，若Tac 为基本tactics；

b. k+1，若Tac 为THEN Tac_1 Tac_2 或

　ORELSE Tac_1 Tac_2 且$\max\{|Tac_1|, |Tac_2|\} = k$；

c. k+1，若Tac 为THENL Tac_0 $[Tac_1, \cdots, Tac_n]$

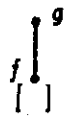
　且$\max\{|Tac_0|, |Tac_1|, \cdots, |Tac_n|\} = k$。

(以上|Tac| 表示Tac 的层次数)

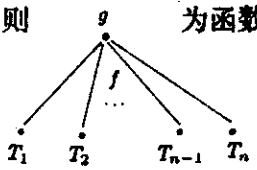
定义4 (函数树)

a. 名为g 的孤根为函数树，记为：$\bullet g$

b. 下面形式的图为函数树。其中g 为名，f 为函数。



c. 设$T_i(1 \le i \le n)$ 为函数树，则



为函数树。其中g 为名，f 为函数。

d. 一棵函数树当且仅当由以上步骤的有限次实施而得。

定义5 (B 函数)　B 是函数树上的一个特殊函数，其定义为:

$$BT=\begin{cases} Sg & T=\bullet g \\ Cf[\,] & T=\overset{g}{\underset{f}{\bullet}} \\ Rf[BT_1,\cdots,BT_n] & T=\overset{g}{\underset{T_1}{\overset{f}{\bullet}}}\cdots\overset{}{T_n} \end{cases}$$

其中S、C 和R 是下面一些函数:

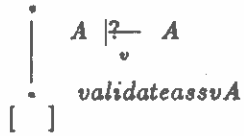
S: $Sg[\cdots,th,\cdots]=th$, 如果$g\xleftarrow{a}th$(第一个), 否则匹配失败。

C: $Cf[\,]thl=f[\,]$

R: $Rf[v_1,v_2,\cdots,v_n]thl=f[v_1thl,v_2thl,\cdots,v_nthl]$

引理1　任何(复合)tactic 应用到目标g 若能成功，则该应用的运行过程对应于一棵函数树。

该引理的证明较为简单，在此从略。如对应于tac-assv　$A\ \underset{v}{|?\!-}\ A=([\,],\ \text{validate assv}\ A)$, 我们有这样一棵函数树:



值得注意的一个事实是:

$$\begin{aligned} &B\left(\begin{array}{l} \bullet A\ \underset{v}{|?\!-}\ A \\ \quad\big|\ \text{validate assv}\ A \\ \bullet\ [\,] \end{array}\right)thl \\ &=C(\text{validate assv}\ A)[\,]thl \\ &=\text{validate assv}\ A[\,] \\ &=\text{assv}\ A(\text{可达目标}A\ \underset{v}{|?\!-}\ A, \text{参见}[6,7,8,9]) \end{aligned}$$

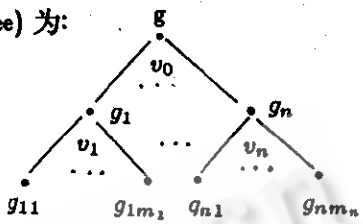
以上结果由validate 函数定义所得。这一结果也说明了tac-assv 是一有效的tactic。

至此，我们讨论复合tactic 应用于目标g 时复合validation 的取值问题。

定义6　设Tac 为tactic, 它应用于目标g 后产生子目标$[g_1, g_2, \cdots, g_n]$, 相应的函数树为Tree, 则复合validation 可取B Tree。

引理2 若Tac_1, Tac_2有效, $\mathrm{Tac}_1\ g=([g_1,\cdots,g_n],v_0)$, $\mathrm{Tac}_2\ g_i=([g_{i_1},\cdots,g_{im_i}],v_i)(1\le i\le n)$, 则复合tactic THEN Tac_1 Tac_2: $g\longmapsto([g_{11},\cdots,g_{1m_1},\cdots,g_{n1},\cdots,g_{nm_n}],v)$ 有效。这里v由定义6确定。

证明: 相应于THEN Tac_1 Tac_2 g的函数树(记为Tree)为:



依题设知:

$$
\begin{aligned}
V &= B\ \mathrm{Tree}\\
&= Rv_0[B\bullet\ g_1,\cdots,B\qquad g_n\ ]\\
&\quad\ v_1\qquad\qquad\ v_n\\
&\quad\ g_{11}\quad g_{1m_1}\quad g_{n1}\quad g_{nm_n}\\
&= Rv_0[Rv_1[B\bullet g_{11},\cdots,B\bullet g_1m_1]\\
&\qquad\qquad\vdots\\
&\qquad\qquad Rv_n[B\bullet g_{n1},\cdots,B\bullet g_{nm_n}]]\\
&= Rv_0[Rv_1[Sg_{11},\cdots,Sg_1m_1],\\
&\qquad\qquad\vdots\\
&\qquad\qquad Rv_n[Sg_{n1},\cdots,Sg_{nm_n}]]
\end{aligned}
$$

设$g_{ij}\xleftarrow{a}th_{ij}\quad(1\le i\le n,\ 1\le j\le m_i)$

记$L=[th_{11},\cdots,th_{1m_1},\cdots,th_{n1},\cdots,th_{nm_n}]$

则

$$
\begin{aligned}
vL &= Rv_0[Rv_1[Sg_{11},\cdots,Sg_{1m_1}]\\
&\qquad\qquad\vdots\\
&\qquad\qquad Rv_n[Sg_{n1},\cdots,Sg_{nm_n}]]L\\
&= v_0[Rv_1[Sg_{11},\cdots,Sg_{1m_1}]L\\
&\qquad\qquad\vdots\\
&\qquad\qquad Rv_n[Sg_{n1},\cdots,Sg_{nm_n}]L]\\
&= v_0[v_1[Sg_{11}L,\cdots,Sg_{1m_1}L]\\
&\qquad\qquad\vdots\\
&\qquad\qquad v_n[Sg_{n1}L,\cdots,Sg_{nm_n}L]]
\end{aligned}
$$

显然$g_{ij} \xleftarrow{a} Sg_{ij}L$, 不失一般性令$Sg_{ij}L = th_{ij}$

则: $vL = v_0[v_1[th_{11}, \cdots, th_{1m_1}]$

$\vdots$

$v_n[th_{n1}, \cdots, th_{nm_n}]]$

因为Tac_1, Tac_2 有效, $g_{ij} \xleftarrow{a} th_{ij}$

所以$g_i \xleftarrow{a} v_i[th_{i1}, \cdots, th_{im_i}]$ $(1 \le i \le n)$

所以$g \xleftarrow{a} v_0[v_1[th_{11}, \cdots, th_{1m_1}]$

$\vdots$

$v_n[th_{n1}, \cdots, th_{nm_n}]]$。证毕。

引理3　若Tac, $\mathrm{Tac}_i (1 \le i \le n)$ 有效, $\mathrm{Tac}\ g = ([g_1, \cdots, g_n], v_0)$, $\mathrm{Tac}_i\ g_i = ([g_{i1}, \cdots, g_{im_i}], v_i)$, 则复合tactic THENL $\mathrm{Tac}[\mathrm{Tac}_1, \cdots, \mathrm{Tac}_n]$: $g \longmapsto ([g_{11}, \cdots, g_{1m_1}, \cdots, g_{n1}, \cdots, g_{nm_n}], v)$ 有效。这里v 由定义6 确定。

证明: 类似引理2。证毕。

引理4　INCAPS 中任何tactic T(基本的或经由THEN, ORELSE, THENL 复合而得)及目标g, 若$T\ g = ([g_1, \cdots, g_n], v)$, v 由定义6 所确定, 则当$g_i \xleftarrow{a} th_i$ $(1 \le i \le n)$, 有$g \xleftarrow{a} v[th_1, th_2, \cdots, th_n]$。

证明: 施归纳于证明策略的层次。

1) 基始: T 为基本tactic— — 层次数为0, 证明是平凡的。

2) 归纳: 设对一切层次数小于k 的tactics 及任何目标g, 引理成立。现设法证明对层次数为k 的tactics 引理也成立。这个证明可分三种情况。

a. 设k 层tactic 为THEN Tac_1 Tac_2, 则由归纳假设及引理2, 引理成立。

b. 若k 层tactic 为THENL $\mathrm{Tac}[\mathrm{Tac}_1, \cdots, \mathrm{Tac}_n]$, 则由归纳假设及引理3, 引理成立。

c. 若k 层tactic 为ORELSE Tac_1 Tac_2, 则证明是平凡的。

3) 结论: 由归纳基始1) 和归纳步骤2), 引理得证。证毕。

下面我们来讨论很早就给出的定理1 的证明。

证明: 只要证得由Repeat 复合而得的tactic Repeat Tac 有效便可。

由附录及题设Repeat Tac 终止知, Repeat Tac 此时实质为一个由基本策略通过有限个THEN, ORELSE 和THENL(可能来自Tac) 按一定规律复合而得的tactic。据引理4, 此时Repeat Tac 有效。证毕。

推论1　INCAPS 中由First, Every 复合而得的tactics 是有效的。

证明: 略。

推论2　INCAPS 中tactics 均有效。

证明: INCAPS 中tactics 有基本的和复合的(经THEN, ORELSE, Repeat, First, Every 和THENL 复合而得) 两类。它们都具有效性。这只由定理1 及推论1 便知。证毕。

致谢: 本文的写作得到了唐稚松教授、黎仁蔚同志的指导和支持, 在此一并致谢。

附录 Tacticals

THENL tac [tac_1, ⋯, tac_m]g =
  let val ([g_1, g_2, ⋯, g_m], v)=tac g
                in
    let val (G'_i , v'_i) = tac_i g_i　　(for $1 \le i \le m$)
                in
    (G'_1@G'_2 @ ⋯ @G'_m, v')
    end
  end;

这里v' 由定义6 确定。任何一个tactic 分解目标失败则THENL 失败。

ORELSE tac_1 tac_2 g= tac_1 g handle?⟹ tac_2 g;

First [ ] g=failure |

First [tac_1, tac_2, ⋯, tac_n] g=
  (ORELSE tac_1 (First [tac_2, ⋯, tac_n]))g;

Repeat tac g=
  (ORELSE (THEN tac (Repeat tac))
          All-tac)g;

Every [ ]g=All-tac g |

Every [tac_2, ⋯, tac_m] g=
  (THEN tac_1 Every [tac_2, ⋯, tac_m]) g;

以上tacticals 是用类ML 语言书写的。其中All-tac 是如下策略： All-tac: g⟼ ([g], λ[t]. t)。

参考文献

[1] R.Milner, LCF: A Way of Doing Proofs with A Machine, Math. Foundations of Computer Science 1979, Lecture Notes in Computer Science, Vol. 74.

[2] L.C.Paulson, Interactive Theorem Proving with Cambridge LCF, Techn. Rept. No. 80, University of Cambridge (1985).

[3] M.Gorden, R.Milner and others, A Metalanguage for Interactive Proof in LCF, Conference Record of the Fifth Annual ACM Symposium on Principles of Programming Language.

[4] L.C.Paulson, Lessons Learned from LCF: A Survey of Natural Deduction Proofs, Computer Journal, Vol. 28, No. 5, 1985.

[5] L.C.Paulson, Logic and Computation, Cambridge University Press, 1987, Cambridge.

[6] 何锦，面向时序逻辑的证明系统：INCAPS，硕士论文，中国科学院软件研究所 1989。

[7] 黎仁蔚，INCAPS：一个交互式计算机辅助定理证明系统，计算机学报，12:12(1989)。

[8] R.W.Li, N System: A Natural Temporal Deduction System, Chinese Science Bulletin, 34:5, 1989.

[9] R.W.Li, A Natural Deduction System of Temporal Logic, JCST, 3:3, 1988.

[10] Kröger, Temporal Logic of Programs, Springer-Verlag, 1987.

[11] R.Milner, The Standard ML Core Language, Polymorphism, 2:2(1985).

[12] D.A.Schmidt, A Programming Notation for Tactical Reasoning, Seventh Conference on Automated Deduction, Springer LNCS 170, Pages 445-459(1984).

[13] C.S.Tang (ed.) and XYZ Group, The Temporal Logic Language XYZ/E (Its Syntax and Explanation) Tech. Rep. No. IS-CAS-XYZ-90-1, The Institute of Software Academia Sinica, Beijings, 1990.