数理解析研究所講究録
第189巻 1973年 52-59

Theorem-proving の program

東教大 理 西村敏男
東教大 理 井出 修
九大 工 大矢建正
東教大 理 神居雅志

§1.

定理の証明を機械によって行わせる試みは，既にいくつかある。第8回プログラミング・シンポジウム（1967）で報告された西村，伊大知等[1]の研究もその一つである。述語論理の体系としては，LK を基礎とし，Definition の推論規則を加えたもので，Problem が Assumption（[1]では 'Theorem' と呼んでいる），Definition と共に計算機の入力として与えられたとき，もし，その Problem が Assumption 及び Defimtion から証明可能ならば，その証明を出力するものである。我々の試みは上記のものを発展させたもので，証明を得るための基本的な考え方は同一のものである。しかし，次のような新しい

概念や処理手順の導入によって，計算機の処理能力の増大を計った。

1) Function の導入

Predicate の argument は単に variable のみに限らないで，term の概念を入れ，function value も含むことにした．また，function symbol は constant と考えるのではなく，variable とし bound され得るものとした．

Assumption の existential quantifier によって bound された variable は，それに先行する universal quantifier によって bound された variables に depend した function value（Skolem function）に置き換え，existential quantifier を消去する．

2) Type の導入

各 variable（term）は一つの type に属するものとし，variable へ代入されるべき term は type が同じものに限ることにする。

Type declaration の入力により，variable の属する type は定められる．

3) Assumption 及び Lemma の reduction

Assumption または assumption に cut の推論規則を適用して得られた lemma（以下単に assumption ということにす

51

3) L_1, L_2 が

$$\vdash L_1 \to L_2$$

であるとき，$L_1 < L_2$ と書いて，L_1 は L_2 より "強い" と呼ぶことにする．$L_1 < L_2$ のとき，assumption の set から L_2 を除くことを "assumption の reduction" と呼ぶことにする．

4) 各 part の module 化

Theorem proving の program の主な部分は

a) 'DECOMPOSE' --- Problem を推論規則に従って分解する

b) 'CHECK' --- Upper most sequent が provable であるかどうかの check を行う．

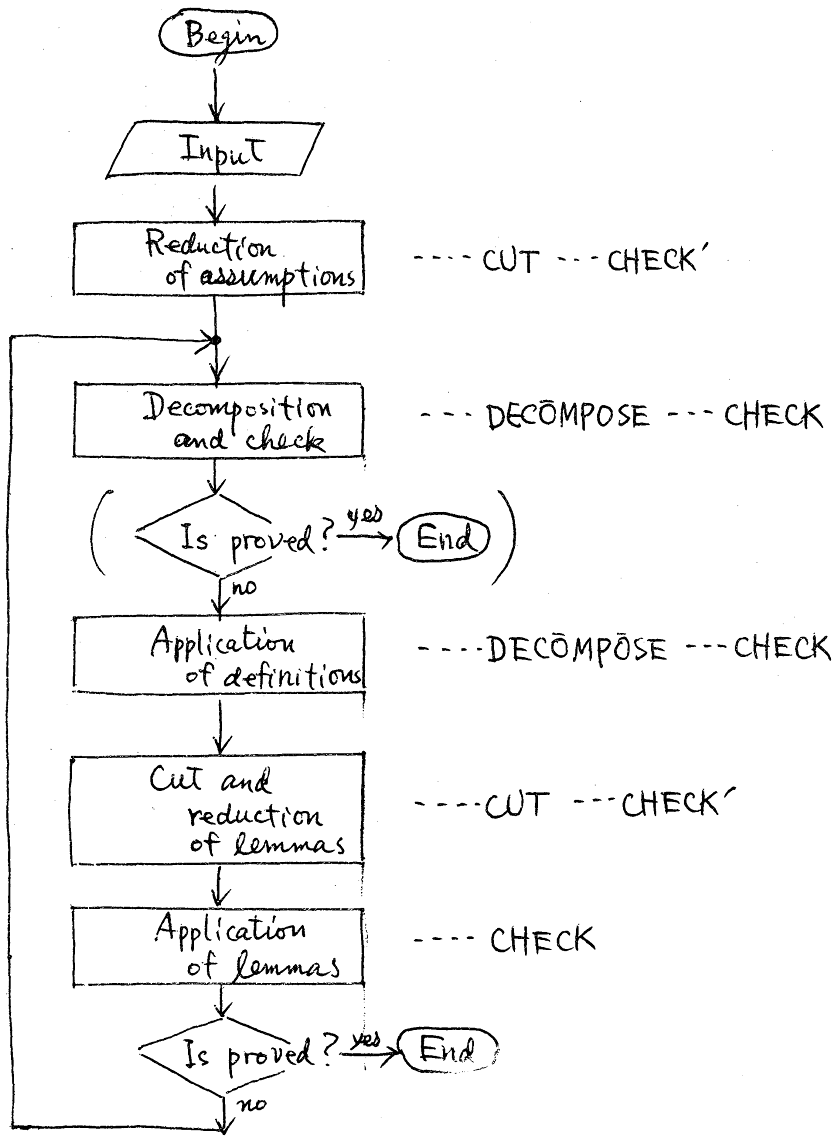
c) 'CUT' --- Assumption に cut の推論規則を適用して lemma を作り出し，reduction を行う．

と呼ばれる routine である．これらの routines 及び I/O-routines 等は module 化されており，processing arrangement を入力指令により control できるようにしている．

5) その他，計算機の機種による物理的制約，プログラミング・テクニック上の制約が緩められ，space-，time-economy についての配慮がなされた。

§2.

Algorithm の概略は次のようなものである。



1) Input

Data (Type declarations, Definitions, Assumptions, Problem) 及び processing control の入力を行う。

2) Reduction of assumptions

入力されたassumptionsを整理する. 即ち, single formula assumptionとcutできるassumptionにはcutを行い, 得られたlemmas及びassumptionsについてreductionを行う.

(以下の例においては, 次のような記号で記述する.

bound variable : u, v, w, x, y, z

free variable : u, v, w, x, y, z 以外の英小文字

apparent variable : ギリシャ文字

およびそれらに $'$ または添字をつけたもの

predicate symbol : 英大文字, および $=, \underset{A}{=}, \underset{B}{=}, \underset{C}{=}, \in, \underset{A}{\in}, \underset{B}{\in}, \underset{C}{\in}$)

例 Ass. 1 : $Q(x, y)$

Ass. 2 : $\neg P(x, y, z, u), \neg Q(u, v), R(w(u), w(v))$

Ass. 2 において, bound variable x, y を新しい bound variable x', y' で置き換え,

Ass 2' : $\neg P(x', y', z, u), \neg Q(u, v), R(w(u), w(v))$

とする. Ass. 2' の u に x を, v に y を代入して cut を行うと次の lemma を得る.

Lemma : $\neg P(x', y', z, x), R(w(x), w(y))$

Lemma は Ass. 2 より強い. したがって assumptions

は

$$\begin{cases} \text{Ass. 1} : Q(x, y) \\ \text{Lemma} : \neg P(x', y', z, x),\ R(w(x), w(y)) \end{cases}$$

に reduce される.

3) Decomposition and check

入力された problem を推論規則によって分解する. もしも, upper most sequent が得られたならば provable であるかどうかの check を行う. 即ち, もしこの sequent に属するすべての apparent variable にどのような term が代入されようとも, この sequent が provable であるならば, この sequent は ~~捨てられ~~ 処理を終ったものとして, 処理中の sequent 群から除かれる. ~~さ~~. またこの sequent に属するいくつかの apparent variable に適当な term が代入されたとき provable であるならば, それらの apparent variable と term の対は sequent と共に記録される.

例1. $\begin{cases} \text{Ass.} : x \in a \\ \text{Seq.} : \xi \in a,\ \neg \eta \in a \end{cases}$

$x \leftarrow \xi$
trivially provable from ass.

$\left(\xi \leftarrow \eta \right.$
provable from logical axiom $\left. \right)$

例2. $\begin{cases} \text{Ass.} : \neg x \underset{B}{=} y,\ z(x) \underset{C}{=} z(y) \\ \text{Seq.} : \neg \alpha \underset{B}{=} f(a),\ \neg \beta \underset{C}{=} g(b),\ \neg \gamma \underset{A}{=} \delta,\ g(\xi) \underset{C}{=} g(\eta) \end{cases}$

$\begin{cases} \beta \leftarrow g(\xi) \\ \eta \leftarrow b \end{cases}$
provable from logical axiom

$$\begin{cases} x \leftarrow \alpha \\ y \leftarrow f(a) \\ z \leftarrow g \\ \xi \leftarrow \alpha \\ \eta \leftarrow f(a) \end{cases}$$

provable from ass.

4) Application of Definition

~~4)~~ Upper most sequents を構成する(prime) formulaで, Definition が適用されうるものにはすべて適用する. 即ち, ある $\underbrace{\text{definition}}_{}$ (の左辺) にあらわれる bound variable に適当な termを代入したとき, その左辺が upper most sequent の formula と一致するならば, その右辺を sequent に付け加える. そして更に分解・check を行う.

5) Cut and reduction

Assumption と, 前回の cut で得られた lemma との間で cut の推論規則を適用できるものがあれば, 適用して新しい lemma を作り出す. 得られた lemmas は assumptions 及び既存の lemmas との間で reduction が行われる.

例 Ass 1: $\neg x_{11} = x_{12}$, $z_{11}(x_{11}) = z_{11}(x_{12})$

Ass. 2: $\neg x_{21} = x_{22}$, $\neg x_{22} = x_{23}$, $x_{21} = x_{23}$

1 回目 の cut

L 1: $\neg x_{111} = x_{112}$, $z_{111}(z_{112}(x_{111})) = z_{111}(z_{112}(x_{112}))$

reduced by Ass. 1 $\{$ Ass. 1_1 , Ass. 1_2 $\}$

L 2: $\neg x_{121} = x_{122}$, $\neg z_{121}(x_{122}) = x_{123}$, $z_{121}(x_{121}) = x_{123}$

$\{$ Ass. 1_2 , Ass. 2_1 $\}$

⋮

$$L\ 4:\ \neg x_{141} = x_{142},\ \neg x_{142} = x_{143},\ z_{141}(x_{141}) = z_{141}(x_{143})$$

$$\{Ass.1_1,\ Ass.2_3\}$$

⋮

2回目の cut

$$L\ 7:\ \neg x_{171} = x_{172},\ \neg x_{173} = x_{171},\ z_{171}(x_{173}) = z_{171}(x_{172})$$

reduced by L4. $\{Ass.1_2,\ L2_2\}$

6) Application of lemmas

Upper most sequents が新しく得られた lemma から provable かどうかの check を行う。

7) Is proved ?

Problem を分解して得られているすべての upper most sequent が，provable になるような apparent variables への terms の代入が可能かどうかを調べ，もし可能ならば，problem は assumptions 及び definitions から証明可能であったことになる。

文献

[1] 西村，伊大知　数学の証明を行なうプログラム
第8回プログラミング・シンポジウム報告集