証明木パターン記述言語としての Tactics

1U−2

南 俊朗
(株) 富士通研究所 ネットメディア研究センター
minami@flab.fujitsu.co.jp

1 はじめに

過去，様々な対話型の論証／証明支援システム [2, 3, 5, 7, 8] が開発されてきた．それらのシステムにおいては，利用者との対話により，一歩一歩進める証明を基本としつつも，容易に構成できるひとかたまりの証明を一度に行うことで証明を (半) 自動化する機能を持っている．この機能は Tactics と呼ばれる．

LCF[2] を始め多くの論証支援システムに採り入れられている Tactics は与えられたゴールを部分ゴールに分解する後向き導出を自動化するために用いられる．我々の開発した汎用論証支援システム EUODHILOS-II[6, 7] における Tactics は，後向き導出だけでなく証明された結果から新しい結果を導く前向き導出の両方を混合して指定することができる．このような違いはあるものの，Tactics が導出過程を手続きとして記述する，いわば，Tactics を関数的に捉える点は類似している．

本論文では，これらとは異なるアプローチとして証明木のパターン表現記述を宣言的 Tactics としてとらえる方式を提案し，手続的 Tactics との違いの概要を示す．本方式は，具体的には，導出規則を論理式の間の多対一の対応関係として捉え，これら関係の合成によって証明木全体の構成を表現する．Tactics をこのように宣言的に捉えることで，様々な論証スタイルへ柔軟に対応できる．また，これは，人間の把握する証明の全体構造を自然に表現するのに適しており，人間の状況把握を促進し，進め方を発想する能力に依存する対話的な論証支援システムの Tactics 記述言語として，手続的記述と比べてより適しているものと考えられる．

2 手続的な Tactics 記述

手続的 Tactics と宣言的パターン表現 Tactics を比較するため，本稿では，次のような通常の命題論理 [1] における $\neg A \vee B \to (A \to B)$ の証明例を用いる．

$$
\cfrac{\cfrac{\cfrac{[-A|B]^1 \quad \cfrac{[A]^2 \quad [-A]^3}{B}\,\text{-E} \quad [B]^4}{B}\,|\text{E}_{3,4}}{A \to B}\,\to\text{I}_2}{(-A|B) \to (A \to B)}\,\to\text{I}_1
$$

Tactics as Pattern Description Language for Proof Trees
Toshiro Minami
Netmedia Laboratory, Fujitsu Laboratories Ltd.
2-2-1 Momochihama, Sawara, Fukuoka 814-8588 Japan

ここでは論理記号 ¬ は − で，∨ は | で表現されている．導出規則として以下のような自然演繹法 [9] スタイルの推論規則が用いられている．

$$
\cfrac{\begin{matrix}[X]\\ \vdots \\ Y\end{matrix}}{X \to Y}\to I \qquad \cfrac{X|Y \quad \begin{matrix}[X]\\ \vdots \\ Z\end{matrix} \quad \begin{matrix}[Y]\\ \vdots \\ Z\end{matrix}}{Z}\,|E \qquad \cfrac{X \quad -X}{Y}\,-E
$$

これを証明するための Tactics の一例を EUODHILOS-II における表記法により次に示す．

```
(then (repeat (backward_tac "->I"))
      (backward_tac "|E")
      (backward_tac "-E"))
```

本 Tactics の表現している手続きは $\to I$ の規則を適用できる限り逆向き適用し，その後順に $|E$,$-E$ を適用するというものである．この例に見られるように，手続的 Tactics は導出を通常のプログラム言語における命令実行と解釈し，実行の順序を記載したものと類似した表現形態となっている．

3 宣言的 Tactics 記述による証明木パターンの表現

通常のプログラム言語においても，Prolog などの，いわゆる宣言的言語が存在する．本節では，Tactics の宣言的記述へ向けて基礎的な考察を進める．

本記述法は次の方針に従って検討を進める．導出規則を関係演算とみなすことで，部分証明も一種の導出規則として扱うことが可能となる．

まず，必要な概念を定める．論理系における表現を (論理) 式 (Formula) と呼び，式全体の集合を $\mathcal{F}$ と表す．

導出規則 R は一般に次の形式で表される．

$$
\cfrac{\begin{matrix}[A_{11}][A_{12}]\ldots\\ \vdots \\ P_1\end{matrix} \quad \cdots \quad \begin{matrix}[A_{n1}][A_{n2}]\ldots\\ \vdots \\ P_n\end{matrix}}{C}\,Cond
$$

ここに，結論式 C, 前提式 $P_1,\ldots,P_n$, 仮定式 $A_{11},\ldots,A_{n1},\ldots$ は $\mathcal{F}$ の要素である．仮定の除去 (Discharge) は随時行われるものとし，簡単のため適用条件 $Cond$ のない場合に絞って話を進める．

以上の前提の下，上記導出規則 R は以下のように前提式 P の集合と結論式 C の間の多対一の関係：

$\{P_1, \ldots, P_n\}RC$
と捉えることができる．なお，前提式の個数 n を規則 R の Arity と呼ぶ．

規則 R に対して次の定義により規則の適用関係を定める．（ここに，$\alpha, \beta, \ldots$ は論理式の集合とする．）

$$\alpha[R]\beta \Leftrightarrow \exists\alpha', \gamma, C \text{ s.t. } \alpha' \cap \gamma = \emptyset, \alpha = \alpha' \cup \gamma, \\ \alpha' RC, \text{ そして } \beta = \{C\} \cup \gamma$$

規則 R と S の合成を以下の性質により定める．なお，Δ は $\alpha[\Delta]\beta \Leftrightarrow \alpha = \beta$ で特徴づけられる規則とする．

- $\alpha[RS]\beta \Leftrightarrow \exists\gamma$ s.t. $\alpha[R]\gamma$ かつ $\gamma[S]\beta$
- $\alpha[R;S]\beta \Leftrightarrow \alpha[R]\beta$ または $\alpha[S]\beta$
- $R* = \Delta; R; RR; RRR; \cdots$

例： 規則 R, S が以下のように定義されるものとする：

$$\frac{A \quad B}{C}\,R \qquad \frac{C \quad D}{E}\,S$$

合成結果 RS は以下の導出規則と同値になる．

$$\frac{A \quad B \quad D}{E}\,RS$$

本稿では詳しく述べないがメタ変数[5]を含む式はメタ変数への代入として得られる式全体からなる集合，すなわち式のパターンを表現するものと解釈できる．このような式の集合は以下に示すように Arity 1 の規則 (Filtering Rule) とみなすことができる．F を論理式の集合とするとき．

$$\alpha[F]\beta \Leftrightarrow \alpha = F\beta \text{ または } \beta = F\alpha$$

ここに，$F\alpha = \{F \cap A | A \in \alpha\}$ である．β に関しても同様である．

本記述言語を用いると上記証明例の Tactics は次のように表すことができる．

```
(-E)(|E)(->I)*
```

4 まとめ

本稿は，関係表現による証明木パターンの記述を Tactics として用いる方式の基本的な考え方を提案した．本方式を用いることで初心者にとっても証明の構造を直観的に見通し良く記述することが可能である．

証明のスタイルには Mizar[10] に見られる宣言的なものと，HOL[3] に見られる手続的なものとに大別される[4]．本稿で述べた証明木パターン記述 Tactics は手続的な方式を基本としつつも宣言的な利点も取り入れようとする試みであるといえる．

宣言型言語の強みは，関係操作モデルに基づく構成の自由さである．前向き，後向き，それらの混合と，人間の行う論証は様々なスタイルの組み合わせである．それを支援するための Tactics も，様々なスタイルでの適用に対応する必要がある．そのような柔軟な論証スタイルに対応した推論及び Tactics の新しいモデルの更なる追求，そして本記述方式に対応した Tactics 記述言語の設計および処理系の実装を進め，様々なスタイルでの論証実験を通して人間が記述に用いるための言語系としての，そして処理系としての完成度を上げることが重要な課題である．

参考文献

[1] Andrews, P.B.: An Introduction to Mathematical Logic and Type Theory: To Truth through Proof. Academic Press (1986).

[2] Gordon, M. J., et al.: Edinburgh LCF, LNCS, Vol. 78, pp.221-270, Springer-Verlag (1979).

[3] Gordon, M.J.: HOL – A Proof Generating System for Higher-Order Logic. In VLSI Specification, Verification and Synthesis, pp.73-128, Kluwer Academic Publishers (1988).

[4] Harrison, J.: Proof Style, URL: http://www.cl.cam.ac.uk/Research/Reports/TR410-jrh-proof-style.dvi.gz (1997).

[5] 南俊朗，大橋恭子，沢村一，大谷武：汎用論証支援システム EUODHILOS 図解マニュアル，Research Report IIAS-RR-92-18J，(株) 富士通研究所国際研 (1992).

[6] Ohtani, T., Sawamura. H., and Minami T.: EUODHILOS-II on top of GNU Epoch. In A. Bundy, editor, Automated Deduction – CADE-12, LNAI, Vol.814, pp.816-820, Springer-Verlag (1994).

[7] 大谷武，沢村一，南俊朗：汎用論証支援システム EUODHILOS-II の設計と実装：情報処理学会誌 Vol.38 No.1, pp.9-22 (1997).

[8] Paulson, L.C.: The Foundation of a Generic Theorem Prover, Journal of Automated Reasoning, Vol.5, pp.363-397. Kluwer Academic Pbulishers (1989).

[9] Prawitz, D.: Natural Deduction – A Proof-Theoretical Study. Acta Universitatis Stockholmiensis, Stockholm Studies in Philosophy, 3rd ed., Stockholm, Almqvist & Wiksell (1965).

[10] Trybulec. A. and Blair, H.: Computer Assisted Reasoning with MIZAR, IJCAI'85, pp.26-28 (1985).