# Prolog プログラムの実行過程の可視化*

5E－2

安達 由洋† 今木 孝哲†
† 東洋大学工学部

## 1 はじめに

Prolog は，ユニフィケーション，自動的バックトラッキング，などの特徴的機能を持つ高レベルで生産性の高い優れたプログラミング言語である．しかしながら，C や PASCAL などの手続き型言語にはないこれらの機能が Prolog の学習を難しくしている．

図を用いたプログラムの可視化は，プログラム仕様の把握やデバッグの支援に非常に効果的である．そこで我々は，Prolog プログラムを木構造図として可視化するための属性グラフ文法を定義し，この文法に基づいて実現した Prolog 可視化システムを報告した [3].

本論文では，Prolog 可視化システムにメタ・インタプリタ技法を用いて新たに追加したビジュアル・トレース機能と実行過程の変数の変化の表示機能について述べる．ビジュアル・トレース機能は，論理外述語 assert, retract や abolish の呼出し (call) によるプログラムの動的な変化に対応してプログラム図を動的に変化させ，プログラムの実行過程をリアルタイムにプログラム図上に反映する．また，変数の代入過程の表示機能は，ゴール実行による変数の変化をリアルタイムに表示する．これらの機能によりプログラムの動作解析やデバッグがより容易になる．

## 2 Prolog プログラム図の属性グラフ文法

Prolog プログラム図 (Prochart) の生成規則を定式化した Prolog プログラム図対応属性グラフ文法 (Prolog-AGG) を定義した．この文法は，プログラム図の生成を形式的に定義するプロダクションとレイアウト情報を属性として計算する意味規則から成り，全部で 21 の規則がある．この規則の一つである開始記号の書換え規則を Fig.1 に示す．

## 3 Prolog-AGG に基づく Prochart システム

本システムは，Prolog-AGG に基づいて Prolog から Prochart の内部表現へ変換するトランスレータと内部表現から Prochart を描画するビューア [3]，そして新たに実現したビジュアル・トレーサから構成される (Fig.2).

*Visualization of Prolog Program Execution
†Yoshihiro Adachi, Takanori Imaki, Faculty of Engineering, Toyo University

**Production**

[ prolog_program ]$_0$ ::= "prolog_program" [ goal_list ]$_1$ [ ]$_2$

C = {

**Semantic Rules**

```
x(1) = RootX                          x(2) = x(1) + w(1) + GapX
y(1) = RootY                          y(2) = y(1)
id(1) = 0                             id(2) = 1
head_list(1) = { }                    head_list(2) = head_list(1)
w(1) = get_width(["prolog_program"])
h(1) = MinH
cell(1) = "head"
string(1) = get_string(["prolog_program"])
line(1) = get_line(id(1), id(2))
nc(0) = nc(2) + 1
subtree_w(0) = w(1) + GapX + subtree_w(2)
subtree_h(0) = max(h(1), subtree_h(2))
```

Fig. 1. 開始記号の書き換え規則

以下に，ビジュアル・トレーサの機能について説明する．

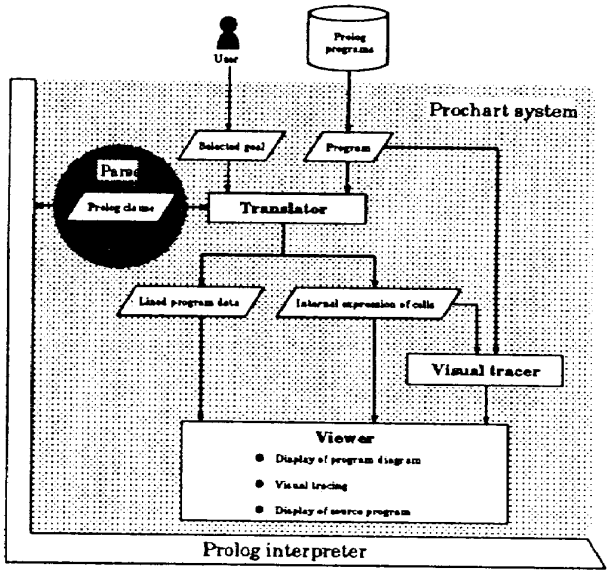


Fig. 2. Prochart システムの構成図

### 3.1 ビジュアル・トレース機能

プログラムの実行状況は，ゴールを実行した結果の真偽に応じて対応するセルの色を変化させることで可視化している (Fig.3)．このセルの色の変化の流れから，ユーザは即座に実行過程を理解することができる．

本トレース機能は，知識プログラミングでは特に重要となる assert, retract, abolish などの論理外述語呼出しによるプログラムの動的変化にも対応している．例によ

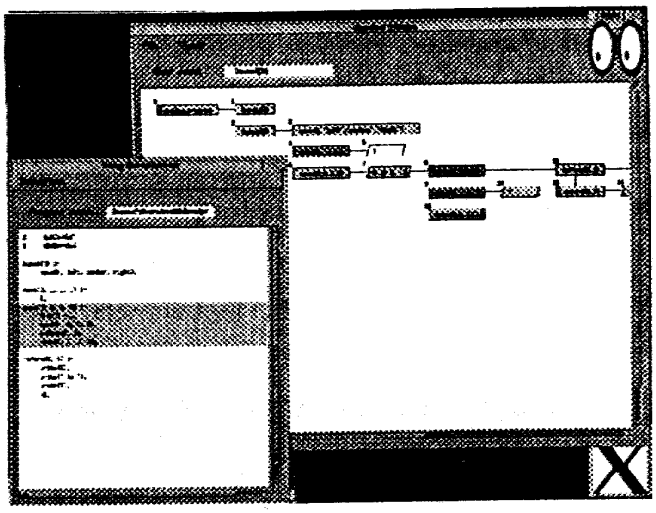

Fig. 3. ビジュアル・トレース画面

り，assertz の呼出しによる Prochart の動的変化を説明する (Fig.4).

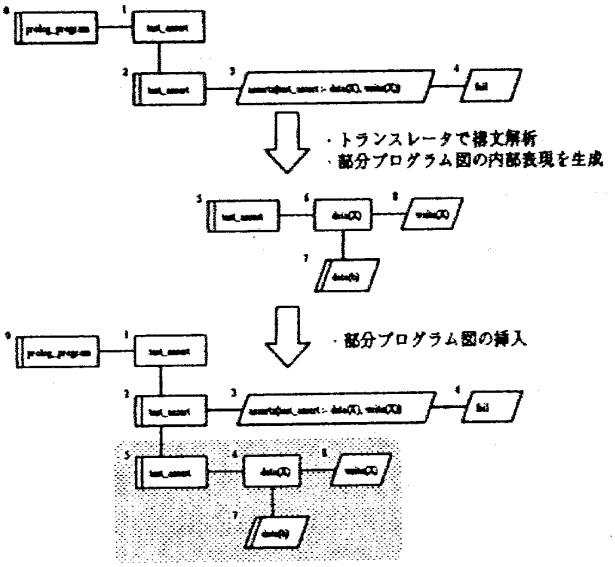


Fig. 4. assertz 呼出しによる Prochart の変化

ビジュアル・トレーサの実行がセル 3 に到達すると，assertz で挿入する節の頭部と Prochart の呼出しセルの内部表現をパターンマッチする．パターンマッチが成功した呼出しセル 1 の位置から挿入する座標を決定する．そして，挿入する節をトランスレータで構文解析して，その内部表現をもとに部分プログラム図を挿入する．これにより Prolog の論理外述語呼出しに応じて Prochart もリアルタイムに変化してその実行過程を可視化する．

### 3.2　変数変化の表示機能

メタ・インタプリタ技法を用いて，ゴール実行過程での変数の変化のリアルタイムな表示機能を実現した．これはメタ・インタプリタで各ゴールの変数とその内容を管理し，ゴール呼出し毎にセルの ID ，ゴールの変数名，変数の内容を一覧にしてテキスト表示するものである (Fig.5)．ゴール呼出しの後に，別のゴール呼出しで値が代入されるような状況も表示することができる．

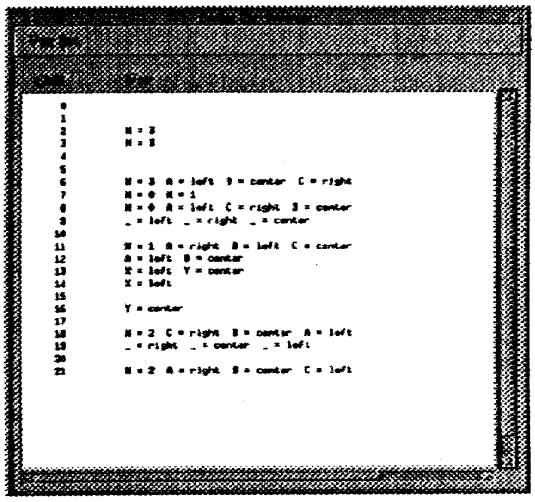

Fig. 5. 変数変化の表示画面

## 4　おわりに

論理外述語 assert，retract, abolish の呼出しによるプログラムの動的変化も可視化するビジュアル・トレース機能と変数の変化をリアルタイムに表示する機能を持ったメタ・インタプリタを実現し，Prochart システムと統合した．

有名な Transparent Prolog Machine(TPM) [1] をはじめ，これまで報告されている Prolog 可視化システムは pure Prolog を扱うものがほとんどであり，assert, retract, abolish の呼出しによる図の動的な変化をリアルタイムに可視化するシステムは本論文のものがはじめてであると思われる．

今後は，大規模なプログラムに対する Prochart の表示方法（例えば，粗い (coarse) 表示と細かい (fine) 表示）やエディタとの連携機能などを検討し，より高度な Prolog プログラミング支援環境へと発展させる．

### 参考文献

[1] Mike Brayshow and Marc Eisenstadt : A practical graphical tracer for Prolog, Int. J.Man-Machine Studies, 35, pp.597-631(1991)

[2] Y.Adachi, K.Anzai, K.Anzai, K.Tsuchida and T.Yaku : Hierarchical Program Diagram Editor Based on Attribute Graph Grammar, Proc. IEEE COMPSAC'96, pp.205-213(Aug. 1996)

[3] 安達，今木: Prolog プログラムの属性グラフ文法に基づいた可視化，情報処理学会研究報告，Vol.96, NO.84, pp.1-8(1996)