Java言語に基づく照合フィルタのアプレット化について

5G－4　水口 卓也　新谷 虎松
名古屋工業大学知能情報システム学科

1 はじめに

今日，インターネット利用者が増加し，WWW を使った情報発信が活発に行われている．このような背景から，インターネットを利用したアプリケーションに期待が集まっている．インターネットを介した人工知能システムの研究も盛んに行われている．人工知能システムを実装する際に，プロダクションシステムがよく用いられる．インターネットを介したプロダクションシステムには，(1) インターネットを介したメッセージ通信が行える,(2) プラットホームに依存しない, を満たすのが望ましい．本研究では，(1),(2) の条件を満たす一般的な環境である Java 言語上にプロダクションシステムを実装した．Java 言語上に実装したことにより，人工知能システムをアプレット [1] として，一般的に利用できる．人工知能アプリケーションを動作させる際に，実行速度は無視できない問題である．人工知能アプリケーションは，できるだけ高速に動作することが望ましい．本論文では本研究で実装したプロダクションシステムである抹茶システムの推論の高速化アプローチとそのアプレット化について述べる．

2 抹茶システム

本研究では，インターネットを介してアプレットとしてダウンロードでき，複数のプラットフォームで動作するプロダクションシステムとして抹茶システムを実装した．抹茶システムでは，インターネットを介したメッセージ通信を行うための機構もそなえている [2]．抹茶システムでは，ルールコンパイラにより，ルールプログラムを Java 言語のコードに変換する．ルールコンパイラの生成した Java 言語のコードをコンパイルすることによりプロダクションシステムを実現する．このプロダクションシステムはアプレットとして動作させることができる．抹茶システムでは，ルールを Java の計算機構とあらかじめ用意されている Java のクラス群をそのまま利用することによって推論の高速化を行った．

Implementing MatchingFilter Using Applet on Java language
Takuya Mizuguchi, Toramatsu Shintani
Nagoya Institute of Technology, Dept. of Intelligence and Computer Science, Gokiso, Showa-ku, Nagoya, 466, JAPAN

3 照合フィルタのアプレット化

プロダクションシステムにおいて，高速性を保証することは重要である．照合フィルタは，ワーキングメモリの変化分に着目し，無駄な照合の繰り返しを極力避けることでプロダクションシステムを高速化するものである．照合フィルタの例として，RETE アルゴリズムや TREAT アルゴリズムなどがある．RETE アルゴリズムや TREAT アルゴリズムは，ルールの条件部 (LHS: Left Hands Size と呼ぶ) からある種のネットワークを構成し，照合の途中過程を記憶することで照合フィルタの機能を実現している．このような方法は，照合過程の記憶のためにメモリを消費し，照合過程の記憶のための特別な機構が必要である．本システムのように，インターネットを介してシステムをダウンロードする場合には，特別な機構のダウンロードに時間がかかり，ネットワークの負荷にもなるため，このような方法は好ましくない．LHS フィルタ [3] は，Prolog のインデキシング機能，論理変数束縛機構を利用して，照合を高速に行う照合フィルタである．LHS フィルタは RETE などのような照合過程の高速化のための機構を持たず，Prolog のプログラム自体がプロダクションシステムを構成している．

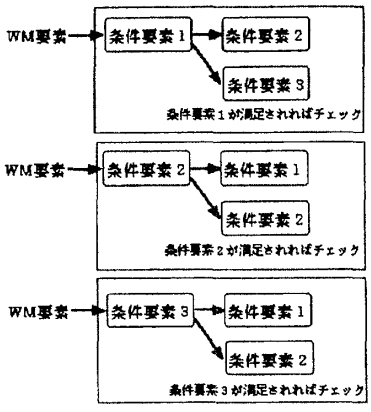

図 1: LHS モジュールの構成

抹茶システムでは，照合フィルタを LHS モジュールを用いて実現した．LHS モジュールは LHS フィルタ Java 言語上に実装したもので，Java 言語のプログラム自体がプロダクションシステムを構築する．このた

```
ルール
    if  g( s = t ) &          もし，WMにg(s = t)とg(v=1 )が存在する
        g( v = 1)             ときにthen以下を実行するルール
    then
            ...
生成されたJavaコード
    if(wme instanceof goal) {
      goal wme0 = (goal)wme,
      if(wme0 s equals("t") ){
        for(int i=0,i<wm goallist size(),i++){
          goal wme1 = (goal)(wm goallist elementAt(i)),
          if( wme1 v == 1) {
                        ...
          }
        }
      }
      else return(false),
```

図 2: ルールのコンパイル例

め，必要最低限の機構をダウンロードするだけですむ．ルールが条件要素 1，条件要素 2，条件要素 3 の 3 つから構成されている時，LHS モジュールは図 1のように 3 つ作成される．図 1の長方形で囲まれている部分が LHS モジュールである．それぞれの LHS モジュールはワーキングメモリ (WM と表記する) の変化分である WM 要素を受け取ってそれぞれの条件要素を満足するかを調べる．例えば条件要素 1 によって構成された LHS モジュールは，受け取った WM 要素が条件要素 1 を満足するか調べる．もし満足すれば，ルールの他の条件要素 (条件要素 2，条件要素 3) を満たすかを調べる．もしルールのすべての条件要素が満足されれば，インスタンシエーションを作成し，競合集合に追加する．LHS の条件要素ごとにモジュールを用意する理由として，WM の変化分がルールの LHS の条件要素のどれか一つでも満足させることが出来た時に，はじめて全ての条件要素が満足されるかを調べるために高速化がはかれることが挙げられる．

LHS モジュールはルールの条件と WM との照合に特別な機構を用いない．LHS モジュールは Java 言語の制御構文 (if 文，for 文) をそのまま用いて照合を行う．LHS モジュールは Java のプログラムとして実装されているため，アプレットとして利用できる．ルールの条件とそれからルールコンパイラによって生成される LHS モジュールのプログラム例を図 2に示す．

4 実験・評価

LHS フィルタのような言語自体の計算機構を用いて実装された照合フィルタは，RETE や TREAT などと同程度の推論の高速性を実現している [3]．ここでは特に RETE,TREAT との性能比較は行わず，Java 言語

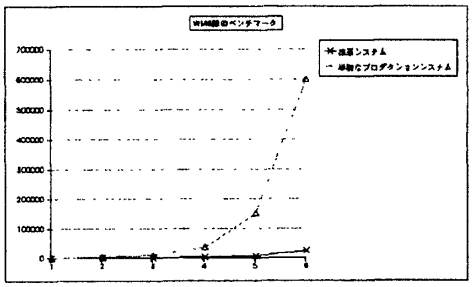

図 3: ハノイの塔の実行結果

で実装された単純なプロダクションシステムと性能比較を行った．このプロダクションシステムは，Java 言語の機構の上にプロダクションインタプリタを実装したものである．性能比較実験は全て PowerPC603 を搭載した Macintosh Performa 6210 上で行った．

実験には，典型的なベンチマークテスト用ルールであるハノイの塔 (5 ルール) を使用した．実験結果を図 3に示す．図 3において，横軸が円盤の枚数，縦軸が推論時間を表わしている．単位はそれぞれ，枚とミリ秒である．直線に×で表わしたものが抹茶システムの推論結果，破線に△で表わしたものが Java 言語を用いて実装された単純なプロダクションシステムの推論結果を表わしている．

実験の結果から，抹茶システムは，Java 言語の制御構造をそのまま利用した照合フィルタを使った高速に動作するプロダクションシステムであると言える．

5 おわりに

本研究では，Java 言語上にプロダクションシステムを実装した．Java 言語上に実装したことにより，構築した人工知能システムをアプレットとして一般にダウンロードして利用することができる．本論文では，本システムの推論高速化の手法と，そのアプレット化について述べ，実験により推論の高速性を実際に示した．

参考文献

[1] "The Java(TM) Language Specification," Sun Microsystems Computer Corporation, 1995.

[2] 水口卓也, 新谷虎松, "エージェントによるインターネットサービスの提供," 1996 年度電気関係学会東海支部連合大会講演論文集, pp.115-118, 1996.

[3] 新谷虎松,"Prolog におけるプロダクション照合フィルタの高速化," 情報処理学会論文誌, 第 32 巻, 第 1 号, pp.20-31, 1980.