TM-0980

推論システムにおけるルール照合フィルタの並列化

新谷虎松（富士通）

December, 1990

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03)3456-3191～5
Telex ICOT J32964

Institute for New Generation Computer Technology

推論システムにおけるルール照合フィルタの並列化

新谷虎松
富士通（株）国際情報社会科学研究所

１．はじめに

OPS5で代表される前向き推論型プロダクションシステムの高速化技法として，RETEアルゴリズム[1]やTREATアルゴリズム[2]が良く知られている．これらアルゴリズムは，プロダクションシステムの推論メカニズムにおける照合過程を効率良く実行するためのものであり，McDermott[3]が提案した照合過程のためのルール照合フィルタの機能を実現している．実際のインプリメンテーションでは，一般に，ルール照合フィルタはルールコンパイラーを用いて，ルールの条件部からある種のネットワークを構成することにより実現される．最近では，推論をさらに高速化するために，これらアルゴリズムの並列化が研究されている．元来，OPS型プロダクションシステムの実行メカニズムにはかなり多くの並列性が内在していることが良く知られている[4]．例えば，D.Kalpらは，RETEアルゴリズムにおける内部メカニズムを並列化した並列化OPS5を試作している[5]．

筆者らは，Prolog上で論理型言語の利点を生かした高速なルール照合フィルタ（LHSフィルタと呼ぶ）を実現している[6]．LHSフィルタは，並列論理型言語（例えば，GHCやKL1）等を用いることにより，効果的に並列化することが可能である．本稿では，GHCの並列機能に基づくLHSフィルタの並列化について論じる．

２．ＬＨＳフィルタ

LHSフィルタの概略は図１のように示すことができる．図１は，図上で与えられたルール（プロダクションシステムKORE/IE[6]の表記である）のLHS（条件部）をコンパイルすることにより，図下で示すLHSフィルタが生成されることを示している．LHSフィルタは，LHS節と呼ばれるPrologのホーン節を用いて実現され，推論システムにおける照合過程の機能を提供する．大文字で始まるアトムはProlog論理変数を表す．ルールのLHSは，スロット記述を引数とするPrologの項で表現される．LHS節のヘッドの引数は，スロット値の並び，タイムタグ，及びインスタンシエーションから構成される．LHSにおける変数は，LHS節ではPrologの論理変数として表現される．スロット値の並びの順は，スロットと対応させるために予め定義される．LHS節のヘッドはWM要素の変化を利用するための受け口として利用される．

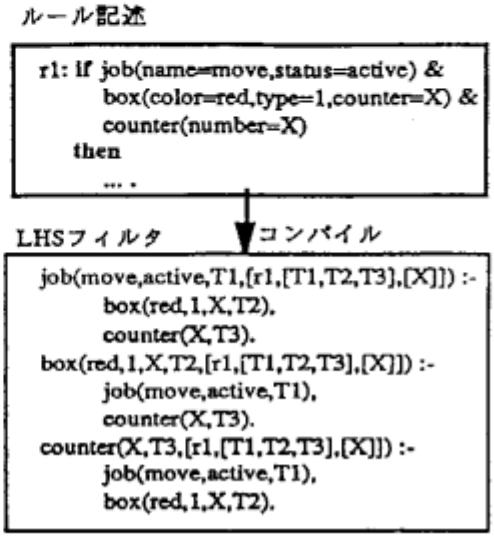

図１．PrologにおけるLHSフィルタの生成

３．ＬＨＳフィルタにおける並列性

3.1．ＬＨＳフィルタを用いた照合過

WM（ワーキングメモリ）を変化させるものとして，make, modify, removeコマンドがある．例えば，makeコマンドは新たなWM要素をWMに付加する．WMの変化は，LHS節へのProlog queryへと変換することにより，照合過程を行うきっかけとなる．queryは，LHS節のヘッドを呼び出す形式に生成され，インスタンシエーション（つまり，実行可能なルールとその変数束縛情報）を求めるために実行される．LHS節本体のゴールが成功すると，LHS節のヘッドの最後の引数へ論理変数束縛の情報が節の本体からヘッドへ後向きに伝播される．その結果，呼び出したqueryに対して具現化された（つまり，変数を含まない）インスタンシエーション情報が出力される．OPS型プロダクションシステムでは，普通，照合過程で複数のインスタンシエーションが生成される．Prologを用いたインプリメンテーションでは，Prologのrepeat-fail機能を素直に用いてLHS節に対するqueryの複数解を求める．

一方，並列型言語では，このようなrepeat-fail機能は

On Parallel Implementation of a Rule Matching Filter in Inference Systems
Toramatsu SHINTANI : (tora@iias.fujitsu.co.jp)
IIAS-SIS, FUJITSU LIMITED

なく，普通，再帰的定義を用いて実現する．つまり，一つのqueryに対して全てのLHS節について並列に可能性を調べることになる．また，図1のLHS節本体は，WM要素が前もってassertされていることが前提となり，WM要素のチェックをしている．GHCでは，このようなassert-retract機能はなくWMをシステムの引数として持って回る必要がある．Prolog上でのLHS節本体のゴール呼出とその実行はPrologのインデキシング機能により高速化した．また、ここでのゴール呼出はPrologの後戻り機能を利用することにより整合的な変数束縛を見つけている。

一方，GHC上のインプリメンテーションでは，GHCの並列論理型言語としての制約のため、Prologで用いたような(1)LHS節頭でのユニフィケーション機能の利用や(2)後戻り機能を用いることはできない。(1)のためには、節の頭で行なっていたユニフィケーションをGHC節の本体で行なう必要がある。(2)を実現するためには、変数の整合性をgenerate-and-test法を用いてチェックする。例えば、図1で示す第一番目のLHS節(jobに関した)の本体において、変数Xのチェックは第1ゴール(boxに関した)がそのgeneratorとして、第2ゴール(counterに関した）がそのテストして用いる。テストに成功したものが求める変数束縛の情報となる。これにより、LHS節本体のゴール呼出を効果的に並列化し、照合過程を効率化することができる。

3．2．認識-行動サイクルにおける並列性

LHSフィルタを用いた認識-行動サイクルは図2のように示せる．図2において，Prolog節は推論ステッパーであるrunコマンドの定義の概略を表している．認識-行動サイクルはrunコマンドを用いて実行される．引数Nは認識-行動サイクルの繰り返し数を指定するために用いられる．節の本体は，3つのゴールにより構成される．第1番目のゴールconflict_resolutionは，競合集合から予め指定された競合解消戦略を用いてひとつのインスタンシエーション"Instantiation"を選ぶ．第2番目のゴールrhsは選ばれたインスタンシエーションの情報を用いて，選ばれたルールのRHSを実行する．普通，RHSの実行はWMの変更を伴う．WMの変更は，LHSフィルタ対する照合過程に帰着され，競合集合を更新する．認識-行動サイクルは，繰り返し数Nから1を減じて，第3番目のゴールrunを呼び出すことにより再帰的に繰り返される．サイクルは，引数Nが0になるか，もしくは，第1ゴールでインスタンシエーションが得られない時（つまり，競合集合が空の時）に終了する．

図2で示すように，prologインプリメンテーションでは，節のハッシュインデキシング効果を積極的に利用するために，WMや競合集合をPrologの内部データベースを用いて実現している．ここでは，WMや競合集合の更新には，Prologのassert及びretract機能を利用している．

LHSフィルタを並列化するためには（つまり，GHC等で記述するためには），WMや競合集合を引数として持ち歩くことが必要である．これにより，インデキシングによる高速性は犠牲になるが，プログラムの並列性や拡張性は向上する．特筆すべきことは，ゴールrhsとゴールconflict_resolutionとのパイプライン並列によりゴールconflict_resolutionにおいて，競合解消が部分的に計算が可能になり効率化できることである。

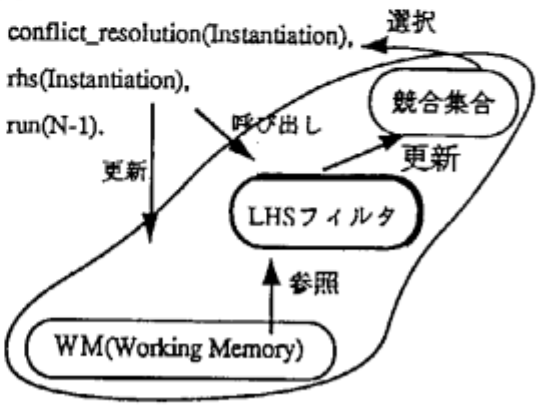

図2. LHSフィルタを用いた認識-行動サイクル

4．おわりに

並列論理型言語に基づくLHSフィルタでは，推論システムに内在する並列性を吟味・明確化することにより可能な限り並列化し，その効率化を図った．さらに，ルールを複数のモジュールに分割することにより，複数のプロセッサで並列に処理する必要がある．尚，本研究は第5世代コンピュータプロジェクトの一環として行われた．

参考文献

1)Forgy,C.L.: Rete:A Fast Algorithm for Many Pattern/ Many Object Pattern Match Problem, AI Vol.19, pp.17-37(1982)

2)Miranker,D.P.: TREAT:A Better Match Algorithm for AI Production System,AAI-87,pp42-47(1987)

3)McDermott,J. and et al. : The Efficiency of Certain Production Implementations, in PDIS, Academic Press, pp.155-176(1978)

4)Gupta,A. and et al.: Parallel algorithm and architectures for rule-based systems, Int. Sympo. on Computer Architecture,pp.28-37(1986)

5)Kalp,D. and et al.: Parallel OPS5 User's Manual, CMU-CS-88-187,(1988)

6)Shintani,T:A Fast Prolog-based Production System KORE/IE,Proc.of the Fifth Int. Conf. and Sympo. on Logic Programming,MIT Press,pp.26-41(1988).