

TM-1180

KL1 を用いた照合過程の高速化について

新谷 虎松 (富士通)

May, 1992

© 1992, ICOT

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

KL1を用いた照合過程の高速化について

新谷虎松

(株)富士通研究所 国際情報社会科学研究所

1 はじめに

筆者等は、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[2]を実現している[3]。本照合フィルタをLHSフィルタと呼ぶ。LHSフィルタは、並列論理型言語であるKL1を用いることにより、効果的に並列化／高速化することが可能である。本論文では、KL1に基づくLHSフィルタの高速化方式について論じる。ここでは、ひとつのPEにおける照合過程の最適化を指向することにより、LHSフィルタを高速化する。

2 LHSフィルタ

LHSフィルタは、ルールのLHS（条件部）をコンパイルすることにより、生成される。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。ルールで用いられる変数は、そのままLHSにおける変数として表現される。図1は、LHS節の生成の概略を表している。LHS節の数はルールの条件要素の数だけ生成される。図1において P_i は条件要素のパターンを表している。

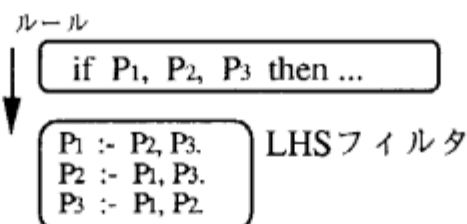


図1. LHSフィルタの概略

WMの変化は（例えば、新たなWM要素をWMに付加すること）、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、インスタンスーション（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、変数束縛の情報が節の本体からLHS節のヘッドへ後向きに伝播される。

前向き推論システムでは、普通、照合過程で複数のインスタンスーションが生成される。この複数のインスタンスーションには、ひとつのルールから変数束縛の組み合わせで複数のインスタンスーションが得られる場合と、複数のルールごとに得られる複数のインスタンスーションが含まれる。Prologを用いたインプリメンテーションでは、このどちらの場合も、単純にPrologのrepeat-fail機能によるバックトラッキングを素直に用いてLHS節に対するqueryとして複数のインスタンスーションを求めることができる。

3 KL1に基づくLHSフィルタ

3.1 解決すべき問題点

KL1では、Prologで利用したrepeat-fail機能によるバックトラッキングやヘッドユニフィケーションに基づく変数束縛チェック機能はなく、KL1に基づくLHSフィルタの実装においてこれら諸機能にかわる新たなLHS節の枠組みが必要となる。特に、Prologにおけるバックトラック機能は、インスタンス化の複数解を求めるための重要な機能となっている。さらに、KL1では、Prologのようにオブジェクトレベルの変数とメタレベルの変数を同じKL1の変数で表現することができない。ルールで記述された変数をユニファイ（もしくは、管理）する新たに特別な枠組みを導入する必要がある。ここでは、変数環境をベクタ表現をすることにより実現する[1]。

LHS節の本体は、WM要素が前もってassertされていることが前提となり、ヘッドユニフィケーション機能やretract機能を用いてWM要素のチェックおよび更新を実現している。KL1では、このようなassert-retract機能に基づくグローバルなデータベース機能はなくWM（および競合集合）をシステムの引数として（例えば、ベクタメモリなどを利用して）持って回る必要がある。これにより、Prologで採用したようなインデキシングによる効率的な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。

3.1 実現方式

図2は、LHSフィルタ節の実現方式の概略を示している。図2において、楕円はルールの条件要素のルールパターンを表している。ここで、3種類のルールパターンが色の濃淡の違いで例示されている。図2上は、Prologに基づく実現方式であり、WMがLHSフィルタ節とは独立に参照され、変数束縛のチェックがLHS節の本体でバックトラッキングによりチェックされる。一方、図2下は、KL1に基づく実現方式を表している。ここでは、WMがLHS節の引き数として実現され、変数束縛のチェックは、生成-テスト法によるパイプライン並列を用いてインスタンス化の検索が実現される。このインスタンス化は、ひとつのルールから得られる複数のインスタンス化に相当する。

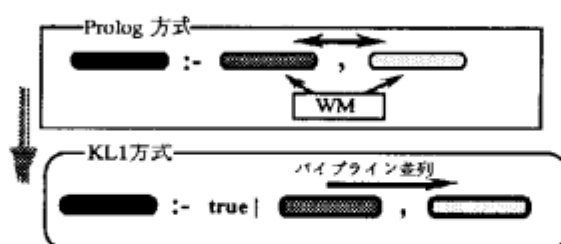


図2. LHSフィルタの実現方式

KL1に基づく実現方式において、全てのルールを対象にした複数のインスタンス化を得るためには、特別な枠組みが必要である。図3は、このような枠組みを示している。図3では、ルールr1およびr2により得られるLHS節が示されている。ここで、

矢印で示された節は、同じパターン（つまり、黒パターン）を表している。このパターンとマッチするWM要素が生じれば、これらLHS節の両方から結果を得る必要がある（ここでは、OR並列が実行される）。ところが、KL1の実行メカニズムでは、一方のコミットされたLHS節のみが実行される。そこで、図3下で示すように（AND並列により）各々のLHS節の実行結果を集めるマージ節を新たに導入する。本マージ節を照合マージ節と呼ぶ。照合マージ節により、全てのルールを対象にした複数のインスタンスエーションが得られる。照合マージ節の特長は、本体のゴール（つまり、LHS節の呼び出し）を独立的に行なうことができ(PE間の通信を極力避ける)、効果的にKL1の並列性を引き出すことができることである。

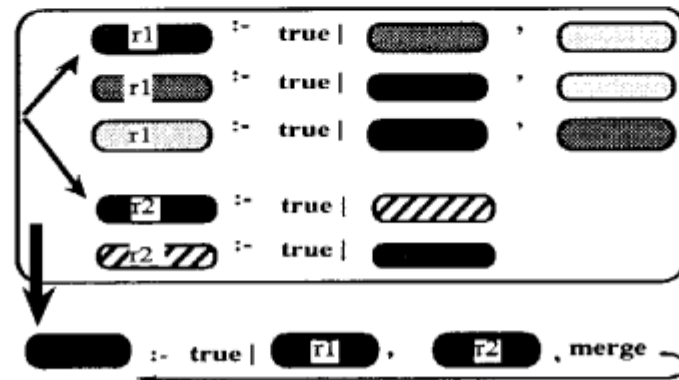


図3. 照合マージ節の概略

3.2 最適化方式

照合マージにおいて、LHS節がインスタンスエーションの生成に関連して完全に独立であれば（つまり、LHS節のヘッドパターンが表すWM要素がユニークに決定されること）、排他的にLHS節を実行でき、また実行結果をマージすることなしに独立的にインスタンスエーションを求めることができる。

LHS節呼び出しの排他性は、LHS節のヘッドのパターン（LHS節パターンと呼ぶ）に着目してそれらパターン間の包含関係を求めることにより、チェックできる。パターンの包含関係は、ユニフィケーションを用いて得られる。例えば、ユニフィケーション $unify(X, Y, Z)$ により得られる Z が X と等しくなれば、パターン X は Y に含まれる ($X \subseteq Y$) と定義する。もし、 X と Y がユニファイできなければ X と Y は独立している（排他的）とする。排他的なLHS節パターンを持ったLHS節は、呼び出しが排他的に実行可能である。

もし、照合マージ節におけるLHS節パターンが排他的である場合は、インスタンスエーションの探索に関連して無駄なLHS節の呼び出しが行なわれることになる。このような無駄なLHS節の呼び出しを回避するために、照合マージ節を効果的に分割可能である。図4は、LHS節パターン $r1, r2, r3, r4$ の包含関係（ここでは、 $r1, r3, r4$ は排他的であり、 $r2 \subseteq r1, r2 \subseteq r3$ ）に着目して、図4上で示された照合マージ節が図4下で示される照合マージ節に分割された例を示している。分割により、無駄なLHS節の呼び出しを回避できる。

最良の場合（つまり，全てのLHS節のヘッドパターンが排他的な場合），照合マージ節が不必要になる．

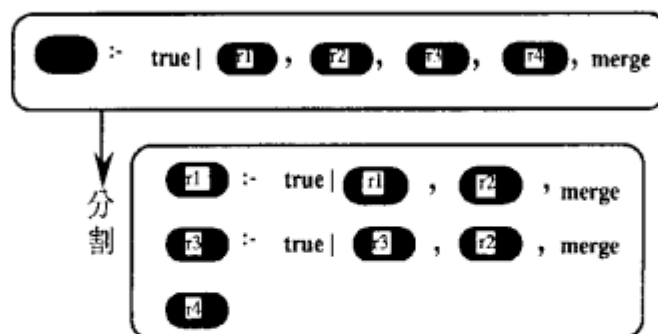


図4．照合マージ節の分割

4 評価

図5のグラフにおいて，新LHSフィルタは，旧LHSフィルタの照合マージ節を分割したものである．グラフにおいて，縦軸はルールひとつ当たりの平均実行時間(CPU時間(ms ec))である．横軸はルール数を示している．本性能評価で用いたテストルールは，WMの要素を次々に付加して行くものである．本テストルールは，その実行においてjoin演算も含まれ，通常のルールプログラミングで良く用いられる形態のサブセットとなっている．図5で示す例では，新LHSフィルタは，ルール数にはほとんど依存しない理想的な実行速度を達成している．

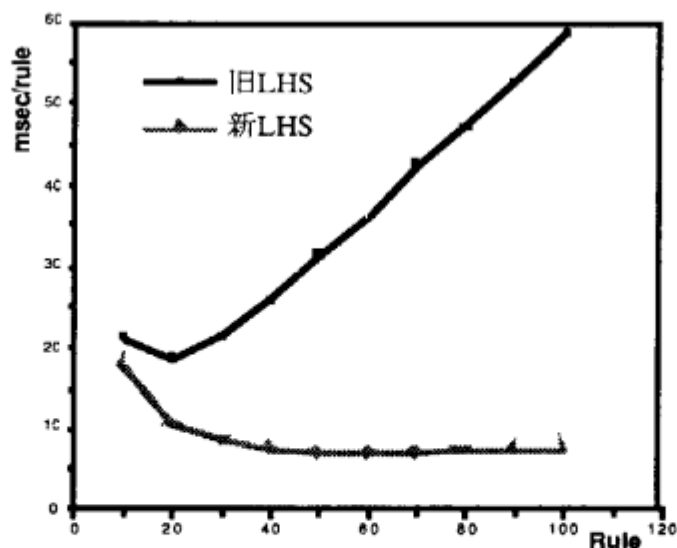


図5．LHSフィルタの評価

5 おわりに

今後の課題は、複数のプロセッサを効果的に用いるための実際的な負荷分散方式を取り入れることによる、照合過程の効率化／高速化である。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なわれたものである。

参考文献

- [1]越村, 藤田, 長谷川: KL1上のユニフィケーションプログラムとその評価, ICOT TM-975, 1990.
- [2]McDermott, J., A. Newell, J. Moore: The Efficiency of Certain Production Implementations, in Pattern Directed Inference Systems, Academic Press: pp. 155-176, 1978.
- [3]新谷: 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, vol. 32, No.1, pp. 20-31, 1991.