

KBMS PHI の問合せ処理におけるルール変換の評価

Evaluation of Rule Transformation in Query Processing in KBMS PHI

阿比留幸展⁺, 羽生田博美⁺⁺, 森田幸伯⁺⁺, 宮崎収兄⁺⁺

Yukihiko Abiru⁺, Hiromi Haniuda⁺⁺, Yukihiko Morita⁺⁺, Nobuyoshi Miyazaki⁺⁺

⁺:(財)新世代コンピュータ技術開発機構, ⁺⁺: 沖電気工業株式会社

⁺:Institute for New Generation Computer Technology, ⁺⁺:Oki Electric Industry Co., Ltd

We have developed an experimental knowledge base management system PHI as part of knowledge base research in the Fifth Generation Computer Systems Project. PHI is based on the concept of deductive database. This paper discusses the evaluation of its recursive query processing strategy.

The basis of query processing is the bottom-up approach, and optimization is performed by rule transformations called Horn Clause Transformations (HCTs). We have developed three types HCTs, HCT/P, HCT/S and HCT/R. We have estimated and measured performance of query processing using these HCTs and combinations of them. We also compare our results to Bancilhon's analysis. It is shown that the performance of query processing using the bottom-up approach can be greatly improved by these rule transformations.

1. はじめに

ICOT の中期プロジェクトの一環として、分散型知識ベース実験システム PHI を試作した [羽生 87]。本論文では、PHI の問い合わせ処理におけるルール変換の評価を記述する。

演繹データベースにおいて、問合せ処理の効率化の方式は数多く研究されている。本システムでは上昇評価法による問合せ処理を行っており、処理効率化のためにルール変換を用いている。本論文では本システムの実測値による評価を述べ、[Banc88] における理論的予測と比較、検討する。

2. PHI におけるルール変換

本システムにおける問い合わせ処理は、上昇評価法による最小不動点演算を関係データベースを用いて行うことを基本としている。この方式は、データが大量で、集合単位の処理を行う場合に適している。問合せ処理の効率化はルールを効率的なデータベース処理を行えるように変換することにより行っている。変換は [高橋 86] による 3 種の HCT (Horn Clause Transformation) を適宜組み合わせることにより行っている。各 HCT の概要を記述する。

HCT/P (Partial evaluation) は部分評価により中間述語を除去し、ルールの簡約化を図るものである。本変換により、処理対象となるルールが減少し、その分効率化が期待出来る。また、以下に示す HCT/S、HCT/R と組み合わせることにより、制約

の伝播が容易になる。HCT/P の例を図 1 に示す。

$$\begin{array}{ll} a(X,Y) :- p(X,Y). & \text{HCT/P} \quad a(X,Y) :- \text{edb}(p(X,Y)). \\ a(X,Y) :- p(X,Z), a(Z,Y). & \Rightarrow \quad a(X,Y) :- \text{edb}(p(X,Z)), a(Z,Y). \\ p(X,Y) :- \text{edb}(p(X,Y)). & \end{array}$$

図 1: HCT/P の例

HCT/S (Substitution) は、問合せの目標中の条件をルールに代入することにより検索空間の削減を図り、効率化を図るものである [阿比 88]。一般に再帰がないルールは、HCT/P が行われていれば定数代入が可能である。再帰があるルールに関しては、定数代入が可能な場合と可能でない場合がある。すなわち、SLD 導出において導出過程に出現する述語がすべて互いに変形 (variant) である時に定数代入が可能である。HCT/S の例を図 2 に示す。

$$\begin{array}{ll} a(X,Y) :- p(X,Y). & \text{HCT/S} \quad a(X,c) :- p(X,c). \\ a(X,Y) :- p(X,Z), a(Z,Y). & \Rightarrow \quad a(X,c) :- p(X,Z), a(Z,c). \\ ?-a(X,c). & \end{array}$$

図 2: HCT/S の例

HCT/R (Restrictor) は制約子と呼ぶ述語を元のルールに付加して、それにより最小不動点集合の削減を図るものである。HCT/R の例を図 3 に示す。

3. 理論的予測

本章では PBI における問合せ処理のコストの理論的予測を述べる。始めに扱うデータの考察を行い、次に各ルール、戦略ごとに処理コストを予測する。

本評価において扱うデータは、[Banc88] において扱っている木、逆木、シリンダとする (図 4)。なお、逆木は図の木をさかさまにしたものである。また、図のように最上段を深さ 0 とし、以下、 n 段目を深さ n と呼ぶ。木、逆木、シリンダについては、以下のような考察が出来る。尚、考察においては、高さを h 、幅を b とする。

• 全アーク数 (ダブル数)

木、逆木: $2^{h+1} - 2$

シリンダ: $2bh$

• 深さ n のある一つのノードから到達可能なノード数 (d_n):

木: $2^{h+1-n} - 1$

逆木: $h - n$

シリンダ: $2 + 3 + \dots + (h - n) + (h - n + 1)$

$= (h - n)(h - n + 3)/2$ (ただし、 $h < b$)

• 全パス数

木、逆木: $d_0 \cdot 1 + d_1 \cdot 2 + \dots + d_{h-1} \cdot 2^{h-1}$

$= \sum_{i=0}^{h-1} (2^{h+1-i} - 2) \cdot 2^i = (h - 1)2^{h+1} + 2$

シリンダ: $\sum_{i=0}^{h-1} b(h-i)(h-i+3)/2 = bh(h+1)(h+5)/6$

評価を行うルールは以下の 3 種類とする。edb($p(X,Y)$) 中の p を木、逆木、シリンダに変化させて評価する。

idb1:

$a(X,Y) \rightarrow p(X,Y).$

$a(X,Y) \rightarrow p(X,Z), a(Z,Y).$

$p(X,Y) \rightarrow edb(p(X,Y)).$

$query1(X) \rightarrow a(c,X).$

idb2:

a, p に関しては idb1 と同じ

$query2(X) \rightarrow a(X,c).$

idb3:

$a(X,Y) \rightarrow p(X,Y).$

$a(X,Y) \rightarrow a(X,Z), a(Z,Y).$

$p(X,Y) \rightarrow edb(p(X,Y)).$

$query3(X) \rightarrow a(c,X).$

以下、idb1 ~ idb3 に関して処理コストを計算する。処理コストの計算方法は、述語に対応する結果リレーションのダブル数の総和とする。すなわち、最小不動点演算における繰り返し演算でリレーション r を求める場合のコストは、繰り返しの回数を m 、 i 回目の演算で算出された r のダブル数を $r(i)$ とすると

$$r(1) + r(2) + \dots + r(m)$$

となる。問合せ中の制約条件に深さ n のノードとした。また、本章において HCT を組み合わせた戦略は、以下の 5 つとする:

(a) HCT を実行しない

(b) HCT/P, HCT/R を実行

(c) HCT/P, HCT/S を実行

(d) HCT/R のみを実行

(e) HCT/S のみを実行

これらをルールに対し適宜選択し、処理コストを予測する。

(1) idb1 のコスト

idb1 は HCT/S が適用出来ないため、HCT/R に対する評価を行う。ここでは (a) と (b) の処理コストを予測する。

(a) HCT を実行しなかった場合

この場合、中間リレーションをすべて求めてから、結果リレーションを求めることになる。本システムでは、線形ルールに関しては、Semi-Naive (SN) 方式 [Banc88] で繰り返し処理を行っている。この方式で実行すると、 k 回目の繰り返し演算では長さ k のパスが算出される。すなわち、長さ k のパスの数を l_k とすると、 $r(k) = l_k$ となる。したがって、コストは以下のようになる:

木: $h \cdot 2^{h+1} + 2^{h+1-n} - 1$

逆木: $h \cdot 2^{h+1} + h - n$

シリンダ: $2bh + bh(h+1)(h+5)/6 + (h-n)(h-n+3)/2$

(b) HCT/P, HCT/R を実行した場合

変換後は図 3 のようになる。ここで、 $a^* \cdot fb$ は、問い合わせのノードを頂点として、そこから到達可能な全ノード集合を求めている。また、 a は、ノード集合 $a^* \cdot fb$ から構成された元のグラフの部分グラフの全パス集合である。したがって、処理コストは以下のようになる (ただし $h - n$ を k とする):

木: $(2^{h+1} - 1) \cdot 2 + \{(k-1) \cdot 2^k + 2\} = (k+3)2^k$

逆木: $2k + k(k-1)/2 = k(k+3)$

シリンダ: $k(k+3)/2 \cdot 2 + \sum_{i=0}^{k-1} (i+1)(h-i)(h-i+3)/2$
 $= k(k+3) + k(k+1)(k+2)(k+7)/24$

(2) idb2 のコスト

idb2 は、HCT/S が適用可能なルールである。したがって、ここでは (b) ~ (e) までの戦略について考慮する。なお (a) は idb1 とほぼ同じコスト、処理時間となる。

(b) HCT/P, HCT/R 実行後のルールは、以下のものになる:

$a^* \cdot fb(c).$

$a(X,Y) \rightarrow a^* \cdot fb(Y), p(X,Y).$

$a(X,Y) \rightarrow a^* \cdot fb(Y), p(X,Z), a(Z,Y).$

$query2(X) \rightarrow a(X,c).$

したがって、 a は、第 2 引数が c のノードであるもののみを求めることになる。処理コストは以下のようになる:

木: $1 + 2n$

逆木: $1 + 2(2^{n+1} - 2)$

シリンダ: $1 + 2(n(n+1) - 1)$

(c) HCT/P, HCT/S 実行後のルールは、図 2 のようになる。

$a^* \cdot fb$ の 1 タプルを除いて、明らかに (b) の場合と、コストは同じである。

(d) HCT/R のみ実行した場合は、(b) の場合と比較すると $p^* \cdot fb$ と p の分だけコストが増える。これらは、それぞれ a

および query2 のコストと等価なため、(d) のコストは (b) の約 2~3 倍になる。

(e) HCT/S のみ実行した場合は、(c) に中間リレーション p のタプル数 $2^{h+1} - 2$ を加えたコストになる。

(3) idb3 のコスト

idb3 に関しては、idb1 と比較した観点で処理コストを予測する。idb1 の場合と比較すると、ルールが非線形である関係上本システムでは Semi Naive ではなく Naive 方式 (N) で最小不動点演算を実行している。したがって a を求めるときの処理コストは、

$$l_1 + (l_1 + l_2) + (l_1 + l_2 + l_3 + l_4) + (l_1 + \dots + l_h) + \dots + (l_1 + \dots + l_h)$$

となる。したがって、繰り返しの回数は $\lceil \log_2 h \rceil$ となる。このことから、(a),(b) 共に idb1 のコストを $c(idb1)$ とすると、idb3 のコストは約 $c(idb1)h/\log_2 h$ と考えられる。

4. 実測

本評価では問合せに対する応答時間を測定した。応答時間にはルール変換の時間、ルールの関係演算への変換時間および関係演算の実行時間を含む。高さ h は 6, 8, 10, 12 で、深さ n は、 $h/2$ とした。また 3 つのデータの高さ、タプル数を一定に近づけるため、シリンダにおいて $b = \lceil 2^h/h \rceil$ とした。

測定結果を図 5~8 に示す。図 5 は idb1、図 6, 7 は idb2、また図 8 は idb3 の測定結果を示している。

図において、縦軸は応答時間、横軸はデータのグラフ表現における高さ h である。なお、ルール変換の時間及び関係演算への変換時間は、全体の処理時間の中では無視できる時間であり(縦軸の数値で 0.3 程度)、大半は関係演算の実行時間である。また、HCT を行わなかった場合において、 $h = 10$ や、 $h = 12$ では、計測不能の部分があり、それらの値は図には記述されていない。

予測と異なった箇所は、図 5 と図 8 の戦略 (b) において、すべての形のデータに対して idb1 と idb3 の差がほとんど出なかったところである。これは、本システムにおいては、繰り返しの回数による負担が大きく、非線形の場合は、繰り返しの収束が速いためである。また、処理コストの計算以上に HCT/P の効果が大きい。これに関しては、次節の HCT の評価で考察する。

5. 評価

まず、[Banc88] との比較を行う。

HCT/R は [Banc88] の Generalized Magic Set (GMS) と処理コストが等価であり、HCT/S は Static Filtering (SF) と処理コストが等価である。また、HCT を行わない場合が、N, SN になる。本システムでは、HCT を行わない場合は、ルールの線形性により N, SN を使い分けている。[Banc88] ではタプルが 10^5 であり、本評価は最大 2^{13} であるため、結果にややギャップが出て

いる。しかし、実測のグラフを理論的に拡張してみると、大きな差にはならないことがわかる。また、idb2 において、われわれの予測や、[Banc88] では GMS と SF が全く等価と計算しているが、HCT の評価で述べるように HCT/S (SF) のほうが、少し効率的であった。

次に HCT の効果について述べる。

- HCT/R はどんな場合でも、処理効率が向上する。すなわち、100 タプル程度の小さなリレーションが対象の場合でも、変換のコストや、述語数が増えることによる join 演算等のコストが無視出来るぐらいに検索空間を小さくする効果が大きい。
- HCT/P は、HCT/S や、HCT/R と組み合わせた場合、制約条件を edb 述語にまで伝播させるため、効果が大きい。特に HCT/S においては、中間述語があると、制約条件がうまく伝播できないことが多いため、中間述語の削除の効果は大きい。また、本システムの関係演算の実行方式は、データベース中のリレーションに対しては索引を利用した演算を実行するのに対し、中間リレーションは索引を持たないので、中間リレーションが大きい場合に効率が低下する。したがって、HCT/P はルールによっては、結果リレーションの総和による処理コストの予測以上に効果がある場合がある。
- HCT/S がうまく行く場合は予測上は HCT/R と同じ処理コストであるが、HCT/R よりやや処理速度が速くなっている。これは新しい述語が導入され、かつその分 join 演算や中間述語のコストが増えるためである。したがって、HCT/S がうまく行く場合は HCT/S の方が効率的である。

6. おわりに

PHI におけるルール変換の処理コストを予測し、実測値をもとに評価した。今後は本システムをふまえて、ICOT プロジェクト後期のシステムを開発していく予定である。

参考文献

- [羽生 87] 羽生田, 他, “分散知識ベース管理システム PHI”, 情報処理学会 DB 研究会, 61-2, 1987
- [Banc88] F. Bancilhon and R. Ramakrishnan, “Performance Evaluation of Data Intensive Logic Programs”, in *Foundations of Deductive Databases and Logic Programming*, Morgan Kaufmann Publishers, Inc., 1988
- [宮崎 88] 宮崎, 他, “演繹データベースにおける問合せ変換”, 信学会研究会, DE-88-5, 1988
- [阿比 88] 阿比留, 他, “演繹データベースにおける制約伝播の実現方式”, 情報処理学会第 36 回全国大会, 3P-3, 1988

$a(X,Y) \rightarrow p(X,Y).$
 $a(X,Y) \rightarrow p(X,Z), a(Z,Y).$
 $?-a(c,Y).$
 HCT/R
 $\Rightarrow$
 $a^*bf(c).$
 $a(X,Y) \rightarrow a^*bf(X), p(X,Y).$
 $a(X,Y) \rightarrow a^*bf(X), p(X,Z), a(Z,Y).$
 $a^*bf(Z) \rightarrow a^*bf(X), p(X,Z).$

図3: HCT/R の例

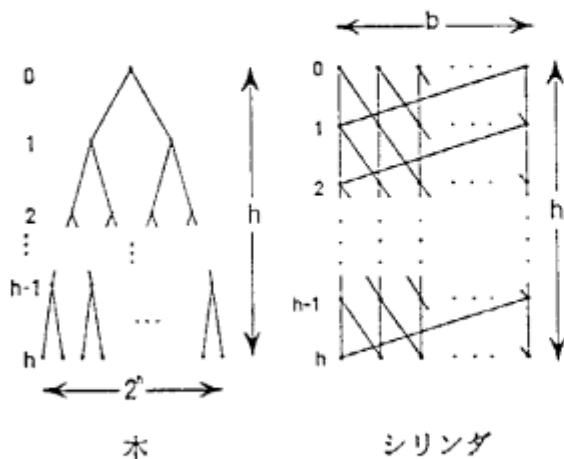


図4: データのグラフ表現

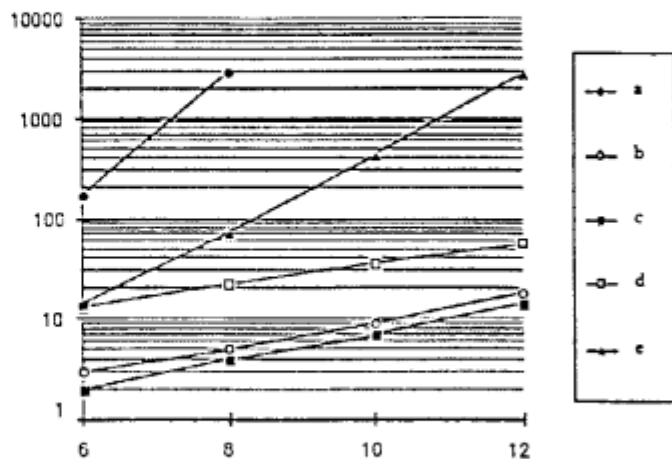


図7: idb2 のシリンダの結果

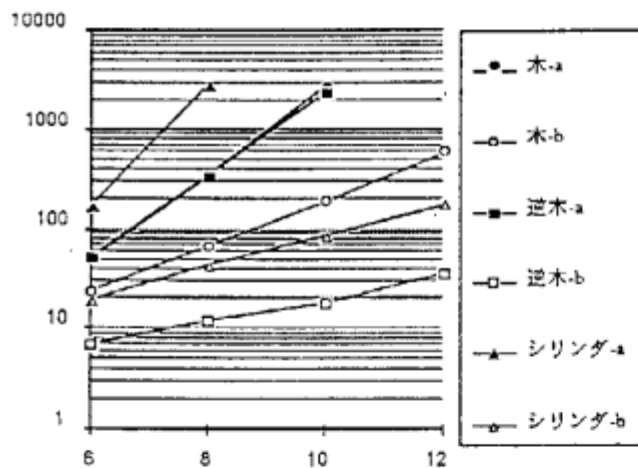


図5: idb1 の処理結果

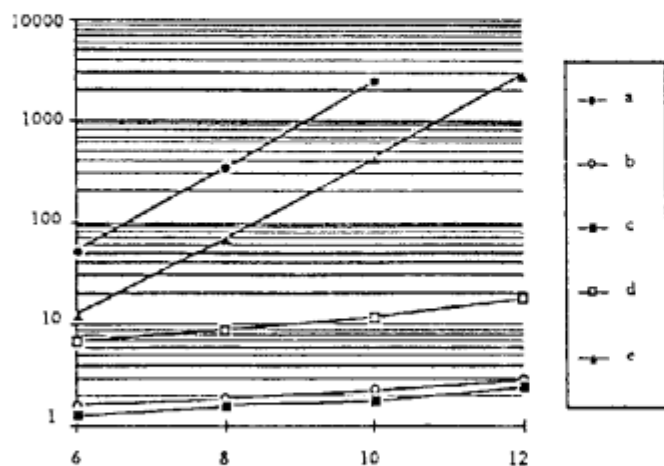


図6: idb2 の木の結果

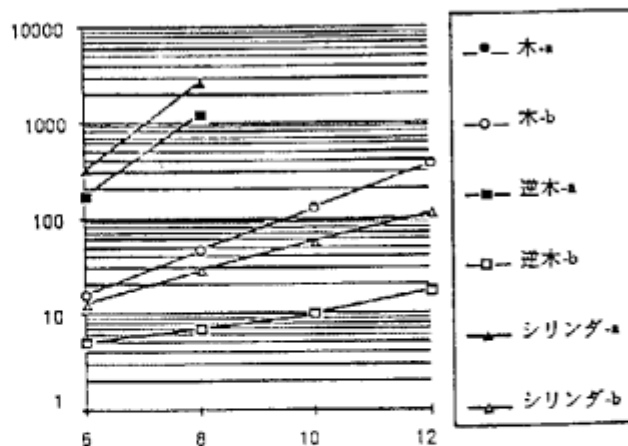


図8: idb3 の結果