

手続知識を管理する演繹データベース

3B-4

への代数的一方式

大森 匡 田中 英彦

東京大学工学部

1. はじめに

本稿では、演繹データベースシステム(以下DDB)において大量のルール節がディスクにある環境下のDBMS(以下ルールDB)について述べる。ここでは大量にあるルール節(以下ルール)を「メタ関係」と呼ぶ集合データで管理し、質問を「単一化を伴う関係代数(RAU)」と呼ぶ集合演算で記述、実行する。単一化を関係代数に組み込む概念はICOT-Delta2によってRBU演算として初めて提案された²。RBUはProlog処理系の記述を目的とするが、我々は大規模ルールDBの効率的な管理と質問処理を目的とする。以下、ルールDBに必要な諸機能を述べ、我々のRAUがこの機能を満たすことを述べる。

2. 大規模ルールデータベース管理システム

2.1 ルールDBの例

DDBなのでルールは関係データベース(以下RDB)のview定義であり、再帰述語は許さないと仮定する。故にルールのBody部は全て関係代数プログラム(以下RAP)にコンパイル可能である。この仮定の下でルールが大規模化し、ルール・ファクトがディスクにあり、特にファクト集合がRDBに管理されている環境を考える。ルールDBの例を図1に示す。今、各ルール節のHead部の述語をそのルール節の「種類」と呼ぶ。図1では二種類のルールEC, TGが管理されている。ルールがディスクに置く程大規模になる原因は

① 一種類のルールが大量 ② ルールの種類が大量の二つである。①の例として図1の種類「EC」のルールは、イーサネットのどの地域(Area)にどの型(Termtyp)の端末をつなぐかでBody部のRAPが全く異なるルール節を持つとすると、Areaが100個、Termtypが100個なら10000個の節を持ち大規模化する。②では1000種類のルールがある場合がある。今、このルールDBへの質問Q1, Q2を次のように与える。

Q1 「グループID "kelly" の端末がイーサネットに接続しており、かつ、グラフィック能力をもっているための条件を求めよ。但し、接続条件は "kb1" が、グラフィック能力は "kb2" が知っているルールを使うこと。」

Q2 「Q1の条件を満たす端末のユーザIDを求めよ。」
これを、図1のメタ言語による操作述語ruledb.demoを用いて記述すると下の様に書ける。

q1(Termtyp, Area, Uid, Cond) :-
ruledb("kb1", EC(Termtyp, "kelly", Area, Uid), Cond1),
ruledb("kb2", TG(Termtyp, "kelly", Area), Cond2),
Cond = (Cond1, Cond2).

q2(Uid) :- q1(Termtyp, Area, Uid, Cond),
demo(RDB, Cond).

管理データ
 EC(Termtyp, Gid, Area, Uid) :- Cond.
 地域<Area>のイーサネットにグループID<Gid>, ユーザID<Uid>, 端末型<Termtyp>の端末が接続する条件
 TG(Termtyp, Gid, Area) :- Cond.
 地域<Area>, グループID<Gid>, 端末型<Termtyp>の端末がグラフィック能力を持つ条件
 操作述語
 ruledb(KB, Head, Body).
 知識ベース<KB>がルール<Head>:-<Body>を持つ。
 demo(KB, Goal).
 理論<KB>において<Goal>が定理。特に、<KB>=
 RDBはファクト集合を管理している関係データベース

2.2 ルールDBの必要機能

2.1の質問Q1, Q2が表すように、ルールDBは次の機能を高速に支援しなければならない。

- (1) ルールの高速検索 — Q1.
 - (2) ルールを実行してファクトを高速に検索する — Q2
- (2)はさらに次の四つの必要機能に分類される。
- ① ルールの実行は別のルールを呼び出し、ディスク上のルールにランダムアクセスを起こす。これを回避する機能。
 - ② 評価戦略の変更⁵

例. 今, A, B — 条件A, Bを表すルールの集合

「<ルール集合>: ルールを満たすファクト集合として、条件A & Bを満たすファクト I (A & B) は、ルールをどこで実行するかで三通りの評価戦略が可能。

- i) AかつBのルールを求めて実行しファクトを求める。
- ii) ファクトIAを求めてから、これに対応するBのルールIA & Bを出して実行する。
- iii) ファクトIA, IBを求めてその共通部分を求める。

状況に応じてこれら評価戦略を変更できることが必要。

③ ファクトDBアクセスプラン(以下プラン)の最適化

例. $p(X) :- r1(X, Y), r2(X)$. に対しゴール節: $- p(l)$ が与えられたとする。但し, $r1, r2$ はファクト節集合でRDB上にある。このとき $\sigma_{X=l}(r1) \bowtie \sigma_{X=l}(r2)$ を始めからすべきで、 $\{X | p(X) \text{ is true}\}$ を求めてから $X=l$ でselection」とすべきでない。

④ 共通項共有⁴

例. ルール $p(X) :- RAP1, RAP0$. }
 $p(X) :- RAP2, RAP0$ } があって、ゴール節: $- p(X)$ ならRAP0は共有して一回の実行ですむ。

3. 代数的方式

3.1 メタ関係と単一化を伴う関係代数(RAU)

以下、ルールのBody部は予めRAPにコンパイルされているとする。こうするとルールDBへのランダムアクセスは回避でき、上の必要機能(2) — ①が満たされるが、ルールの大規模性の内、多種類の大規模性が一種類の大規模性に交換される。故にこの一種類の大規模性を効率良く処理しなければならない。我々は、代数的方式、即ち「大量に存在するものは集合データとして管理し、集合操作で質問を記述・実行する。」アプローチを取る。

今、一種類の条件A, Bを表すルールの集合を各々A, Bとし、次の代数演算を定義する。

- $A \cap B$: 条件A & Bを表すルール集合
- $\sigma_F A$: Aの内、条件Fを満たすルール集合
- $\pi_C A$: 属性Cに関するAのルール集合
- ΓA : ルールAを満たすファクト集合

さらに構造体を扱うため次のS, Pの演算子を用意する。

- S pattern A: Aの各元からpatternに応じて構造体を作る。
- P pattern A: Aの各元からpatternに応じて構造体を展開する。

ルール節を表すメタ関係と各演算子の例を図2に示す。一つのメタ関係は、一つのメタ述語を表すメタ言語上のファクト節集合を表形式で表したものである。図2の演算子 σ_F, π_C, Γ はhorn節に限れば上述の演算子定義を満たす。これらの演算子を「単一化を伴う関係代数(Relational Algebra extended with Unification: RAU)」と呼ぶ。RAUによって2節の質問Q1, Q2は図3のように質問木でかける。

図1. ルールDBの例。ruledb.demo 述語はメタ言語上の述語

この<メタ関係,RAU>の組は一つの代数系を成す。一般に代数系A=<集合D, D上で定義された演算子>が理論Tで記述された質問を正しく処理するためには、「Tの定理はAの演算子列で計算可能で、かつ、Aで計算されるTの整式は定理である。」ことが必要である。これを代数的等価性と呼ぶ。Tをsafeな関係論理、Aを関係代数にすれば代数的等価性が成立する。RAUの場合に代数的等価性を保つ質問クラスについては紙面の都合上別の機会に譲る。

• metarelation

R[A, B, Cond1]: a set of rule r(A,B) :- Cond1.

T[A, Cond2]: a set of t(A) :- Cond2.

$$\begin{array}{c} \text{L table} \rightarrow \text{R} \\ \text{representation} \end{array} \begin{array}{c} \text{A} \quad \text{B} \quad \text{Cond1} \\ f(X) \quad a \quad \text{rel1}(X) \end{array} \quad \begin{array}{c} \text{T} \quad \text{A} \quad \text{Cond2} \\ X \quad \text{rel2}(X) \end{array}$$

• S_[1,f(2,3)] ($\frac{1 \quad 2 \quad 3}{a \quad b \quad c} = \frac{1 \quad 2}{a \quad f(b,c)}$) 1,2,... is Attribute

• P_[1,f(2,3)] ($\frac{1 \quad 2}{a \quad f(b,c)} = \frac{1 \quad 2 \quad 3}{a \quad b \quad c}$) ID operand relation

• $\sigma_{1=f(X)}^u T[1,2] = \frac{A \quad \text{Cond3}}{f(X) \quad \text{rel2}(f(X))}$
 This is a set of rule q(A) :- Cond3.
 q(A) is defined as q(A) :- t(A) & A = f(X).
 A ^u B means "A unifies with B".

• $R \bowtie T = S_{[1,3 \& 5]} \sigma_{1=4}^u (R[1,2,3] \times T[4,5])$
 $= \frac{A \quad \text{Cond3}}{f(X) \quad \text{rel1}(X) \& \text{rel2}(f(X))}$ Cartesian Product

• $\prod_{[1,3]} R[1,2,3] = S_{[1,3]} R[1,2,3]$
 This is a set of rule q(A) :- Cond3.
 q(A) is defined as q(A) :- r(A,B) & t(A).

• $\prod_{[1,3]} R[1,2,3] = S_{[1,3]} R[1,2,3]$
 $= \frac{A \quad \text{Cond4}}{f(X) \quad \text{rel1}(X)}$ This is a set of rule q(A) :- Cond4. q(A) is defined as q(A) :- r(A,B).

• $I_2 T[1,2] = \frac{1 \quad 2}{a \quad \text{rel2}(a)}$ where $\frac{\text{rel2} \quad 1}{a}$
 This is a set of fact satisfying rule t.

図2. メタ関係と演算子の例

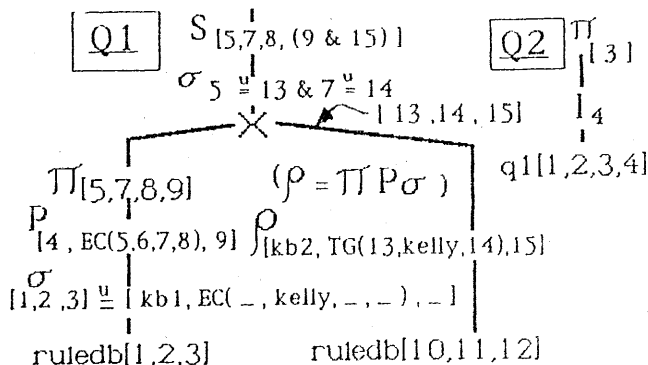


図3. 単一化を伴う関係代数による質問木

3.2 従来のDDB 実装法のルールDBとしての欠点

従来のDDB 実装法としてTopdown 法とBottom up 法とを取る。Topdown 法は、質問をProlog処理系でRAP に対応するゴール節に変換後、関係DBで実行する³。

Bottomup法はルール節集合を予め関係代数のネットワークにコンパイルして全てRAP で実行する⁴。2.2 の必要機能の内、Bottomup法はBody部をRAP ネットワークに予めコンパイルす

ることで(2) - ①を満たしている。さらに、コンパイル時に予め大きな単位で共通項を共有しており(2) - ④も満たす。しかし、共有項が大きすぎるためRAP ネットワークの枝刈り等の最適化は殆ど不可能である。そのため参照する殆ど全てのファクトDBページにアクセスし、(2) - ③は満たさない(図4)。ルールの検索はできず、固定評価戦略のため(1)、(2) - ②は満たさない。

対してTopdown 法はルールの検索は可能だが一回にルール一個を処理するので2.1 の質問Q1ではO(n²) (nはEC, TG各種類のルール節数)となり(1)は不十分。(2) - ③は実行する必要最小限のプランのみを求めないので満たす。

(2) - ①, ②はできない。④は、個々のプランを独立して実行するので満たさない。

- i) :- p(1) . } で p(X) :- r1(X), r2(X) .
- :- p(2) . } なら
- $\sigma_{X=1}(r1) \bowtie \sigma_{X=1}(r2)$ と
- $\sigma_{X=2}(r1) \bowtie \sigma_{X=2}(r2)$ をすべき。
- p(X)を共有すると{X | p(X)}を求めてから
- X = 1 or 2でselectionをするので無駄。
- ii) :- p(X), q(X) . } RAP1 \bowtie $\sigma_{X=3}(RAP3)$ と
- p(g(3,X)) :- RAP1. } RAP2 \bowtie $\sigma_{Y=5}(RAP3)$ の方が
- p(g(X,5)) :- RAP2. } {X | q(X)}を共有して
- q(g(X,Y)) :- RAP3. } RAP3を直接実行するより速い。

図4. 共通項共有によるファクトDBアクセスへの悪影響

3.3 RAUのルールDBとしての利点

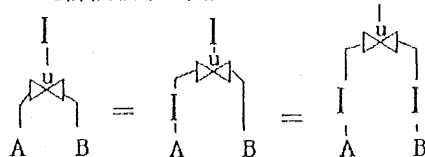
RAUによるルールDBは2章の各必要機能を満たす。

(1) RAUの内、最も負荷の重い単一化を伴う自然結合演算 $A \bowtie B = \prod \sigma_{X=Y}(A \times B)$ は集合演算アルゴリズムと専用ハードウェアによって高速に支援可能である¹。

(2) - ① 各ルールのBody部を予めRAP コンパイルしたのでBottom up 法同様満たす。

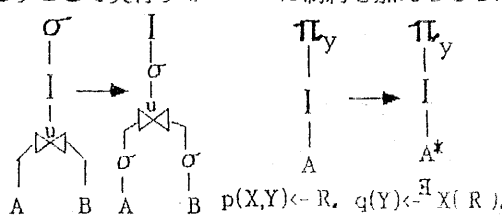
(2) - ② 1演算子を含む一つの質問木は一つの評価戦略を示している。RAUには演算子間の可換則 $I_c \cdot I_c = I_c$, $I_c(A \bowtie B) = I_c A \bowtie I_c B = I_c(A \bowtie B)$ が成立し、質問木を変換できる。故に、オペラントのメタ関係間の統計情報に基づいて評価戦略を変更できる。

例.



(2) - ③ Topdown 法同様、実行すべき必要最小限のルール集合を求めることが可能なので、各ファクトDBアクセスプランは最適化される。さらにRAU 質問木はsigma, piを葉ノードに落とすことで実行するルールに制約を加えることが可能。

例.



(2) - ④ ファクトDBアクセスプランは集合単位で実行されるので、その中で共通項を検出し共有することは可能。

4. おわりに

本稿では大規模ルールDBのために単一化を伴う関係代数(RAU)によるDBMSを提案した。RAUの形式的な定義、代数的等価性を持つ質問クラス、このクラスでのRAU 質問木最適化戦略については別途発表の予定である⁶。RAUの実行アルゴリズムは既に発表している¹。現在、(2) - ④の共通項共有は行っていない。このアルゴリズムは研究例も多く、RAUへの実装は今後の課題である。

- (参考文献) (1) 大森, 他, 情報処理全国大会33回5L-6
 (2) Yokota, et. al, VLDB-86 (3) Bocca, SIGMOD86
 (4) Tsur, VLDB86 (5) Ullman, TDBS, vol.10, No.3
 (6) 大森, 電子通信学会データ工学研究会3月発表予定