ICOT Technical Memorandum: TM-0500

TM-0500

演繹データベースにおける問合せ変換

宮崎収兄，羽生田博美，横田一正，伊藤英則

May, 1988

ICOT

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

(03) 456-3191～5
Telex ICOT J32964

Institute for New Generation Computer Technology

演繹データベースにおける問合せ変換

QUERY TRANSFORMATIONS IN DEDUCTIVE DATABASES

宮崎 収兄*，　羽生田 博美*，　横田 一正**，　伊藤 英則**

Nobuyoshi MIYAZAKI+，　Hiromi HANIUDA+，　Kazumasa YOKOTA++，　Hidenori ITOH++

*：沖電気工業(株)，　**：(財)新世代コンピュータ技術開発機構

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

概要

再帰問合せに関して，問合せを等価な別の形に変換し効率化を図る方式がいくつか提案されている．本稿では，これら問合せ変換の論理的基礎を与える．まず，等価変換の原理について議論し，モデル等価変換と目標等価変換について議論する．モデル等価変換は論理プログラム変換にも用いられている．これらの変換の原理について議論する．次いで問合せ変換の基礎となるホーン節変換について議論する．ホーン節変換は，外延データベースを保存する目標等価変換として特徴付けられ，本質的には関係データベースシステムにおいて最適化に用いられる関係問合せ変換の一般化である．一階述語論理のそれぞれ異なる原理に基いたホーン節変換，HCT/P（部分評価），HCT/S（代入），HCT/R（制約子）について議論する．また，ホーン節変換と他の方式との関連についても述べる．

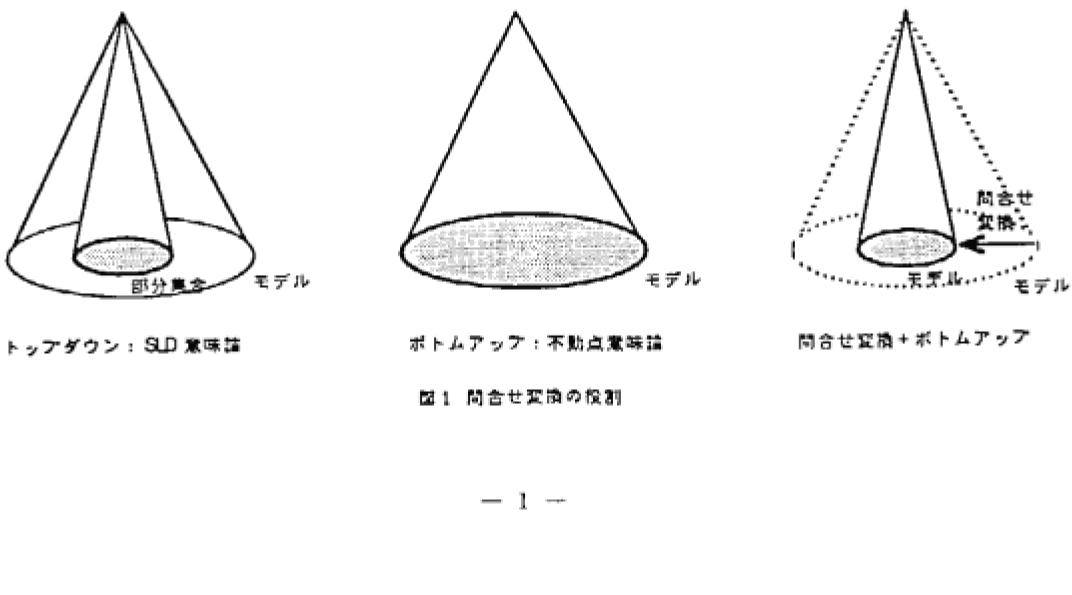
1.まえがき

演繹データベースにおける問合せ処理方式が多数提案されている [Bancilhon 86b]．これらの方式には，問合せを等価な別の形に変換するものがあるが，各方式で等価性の意味や効率性の基準や表現が異なっていたりするので，これら方式の比較や組み合せを容易に行うことはできない．

問合せ変換の主な役割は，後続処理の効率的な実行が可能になるように問合せの形を変換することである．演繹データベースの問合せ処理は2つに大別され，1つは SLD 意味論に基くトップダウン方式で，もう1つは不動点意味論に基くボトムアップ方式である．トップダウン方式では解を得る時，束縛伝播を行うことにより最小エルブラン・モデルの部分集合を計算する．ボトムアップ方式では，最小エルブラン・モデル全体の計算を行う．このため，ボトムアップ方式は，データベース演算をうまく利用するのには適しているが，そのままでは効率的ではない．問合せ変換では，解の等価性を保持したまま，最小エルブラン・モデルをより小さくするように節集合の変換を行う．問合せ変換をボトムアップ方式と組み合わせると，変換前より小さな最小エルブラン・モデルの計算を行えば解を求めることができる．したがって，この組み合わせを行った方式は何を計算するかという点ではトップダウン方式とほぼ同様になる．この様子を図1に示す．

関係問合せの等価変換は問合せ処理における最も重要な原理の1つとして用いられており [Ullman 82]，演繹データベースは関係データベースの自然な拡張である．したがって，演繹データベースにおける問合せ変換は，再帰問合せに対する処理を新たに考える必要があるが，関係データベースにおける問合せ変換の拡張である．

一方，演繹データベースにおける問合せ変換と同様に確定節集合の



トップダウン：SLD 意味論　　ボトムアップ：不動点意味論　　問合せ変換＋ボトムアップ

図1 問合せ変換の役割

変換を行うものとして，論理プログラミング分野におけるプログラム変換がある［古川 87］．演繹データベースと論理プログラムには主に次のような差異がある．

(1) 演繹データベースでは事実節の数が多いが，論理プログラムでは規則節の数が多い．
(2) 演繹データベースでは単一の解と同時に解集合も重要である．
(3) 通常，問合せ変換の入力は目標と内包データベースであり外延データベースは変換に用いられない．プログラム変換の入力は目的に依存する．
(4) 問合せ変換は自動変換である必要がある．プログラム変換では必ずしも自動であることは要求されない．

本稿では，問合せ変換の論理的基礎を与えることを試みる．等価変換の原理とそれを用いた変換アルゴリズムをいくつか提案する．また，他方式とこれらの変換の比較についても述べる．２章では，等価変換の基本原理を論じる．３章では，等価変換の特別な場合であるホーン節変換を論じその例を示す．

２ 等価変換

本章では，問合せ変換の原理を論じる．本章の命題の証明は簡単なので省略した．まず，演繹データベースとその問合せを定義する．［野口 87］にあるような論理プログラムの基礎的事項については既知とする．

［定義 2.0.1］記号を定義する．

<h, B>：頭を h，本体中の基本式集合を B とする節．本稿では'節'は確定節を意味する．尚，簡単のため，表記上混乱のない限り基本式の引数を省略する．

M(DDB)：DDB の最小エルブラン・モデル．M(DDB) は DDB の最小不動点でもある．

$C1 \Rightarrow C2$：節 C2 が 節 C1 の論理的帰結であることを意味する．

sym(X)：基本式 X の述語記号，または節 X の頭部の述語記号，または節集合 X 中の節の頭部の述語記号の集合．

$C_H(p,S)$，$C_B(p,S)$：述語記号 p をそれぞれ頭と本体に持つような，節集合 S の部分集合

ϕ，－：空集合と集合差を意味する．□

［定義 2.0.2］演繹データベース (DDB) は確定節の集合である．外延データベース (EDB) は具象単位節の集合であり，内包データベース (IDB) はそれ以外の節集合である．したがって，DDB = IDB∪EDB．本稿では（一般性を失うことなく）sym(IDB)∩sym(EDB) = ϕ と仮定する．□

［定義 2.0.3］問合せは，目標と呼ばれる基本式で記述される．g' を目標 g の各変数を具象項で置き換えた式とする．この時，問合せの解は DDB の論理的帰結であるような g' の集合 G = {g'|DDB⇒g'} である．□

［定義 2.2.4］DDB の依存グラフは，(N, E) で与えられる．ここで，N は DDB の述語記号集合，E は (p, q) の形の枝集合である．ただし，(p, q) は述語記号 p が頭に在り，q が本体に在る節が DDB に存在することを意味する．依存グラフで r から p への径路が存在する時，p は r から到達可能であるといい，$r \rightarrow^{*} p$ と書く．節 F の頭の述語記号が p から到達可能な時 F は p から到達可能であるといい $p \rightarrow^{*} F$ と書く．また，到達可能ではない時それぞれ，$r \not\rightarrow^{*} p$，$r \not\rightarrow^{*} F$ と書く．□

［定義 2.0.5］DDB の成分（以下成分）とは，DDB の依存グラフの強連結成分である．すなわち，$p \rightarrow^{*} q$ かつ $q \rightarrow^{*} p$ の時 p と q が同一成分に属する．この関係は同値関係なので，DDB の述語記号集合は成分に分解できる．また，DDB の節集合はその頭の述語記号により成分分解できる． N と M を DDB の成分とする．M の元から N の元への径路が存在する時 N は M より下であるといい，また M は N より上であるという． "下"関係は半順序である．□

［定義 2.0.6］ある成分に関して，上の成分に属する節の本体に現れる基本式を，その成分のエントリという．□

［定義 2.0.7］式 B, C に関して C = θB なる代入 θ が存在する時，B は C より一般的である（C は B より一般的でない）といい，C ≪ B と書く．C ≪ B かつ B ≪ C の時，B と C は等価であるという．□

［定義 2.0.8］B と C を節，C' を C の部分式とする． C' ≪ B の時 B は C を包摂するといい，B ⊆ C と書く．明らかに B ⊆ C ならば，{B}⇒C．□

２．１ モデル等価変換

本節では，問合せ変換において最も基本的な概念として，モデルを保存する変換，モデル等価変換について述べる．

［定義 2.1.1］M(DDB) = M(DDB') の時，DDB と DDB' はモデル等価であるといい， DDB ≡ DDB' と書く．□

≡ は対称で推移的かつ反射的なので同値関係である．この定義は，プログラム変換の基本原理として用いられている［古川 87］．

[定義 2.1.2] 確定節集合 DDB から確定節集合 DDB' への変換 f が，任意の与えられた DDB に対し DDB' ≡ DDB の時，f はモデル等価変換 (MET) であるという．□

[定義 2.1.3] S⊂DDB，任意の F∈S に対して，(DDB−S)⇒F とする． DDB から DDB−S への変換を冗長削除という．□

[命題 2.1.1] 冗長削除は MET である．□

通常，ある論理式が証明可能かどうかを判定することは困難であり，冗長削除を実際に行うのは困難である．[Sagiv 87] では，Datalog（関数の無い確定節）に対する，冗長削除アルゴリズムを提案している．[Kitakami 84] ではモデル推論に基いた冗長削除アルゴリズムを提案している．トートロジ節<p, {p}>の削除は冗長削除の一例である．

[定義 2.1.4] F∈DDB とする．{F}⇒F' かつ {F'}⇒F なる節が与えられた時，DDBから(DDB−{F})∪{F'}への変換を等価節置換という．□

[命題 2.1.2] 等価節置換は，MET である．□

本命題に基いた変換が [Sagiv 87] で提案されている．この他，[Tamaki 84] で提案された展開/たたみ込み (unfoid/fold) 変換のような METがある．ここでは，これらの変換については論じないが，関連する問題を後の章で述べる．

2.2 目標等価変換

モデル等価変換の概念は，問合せ変換でもプログラム変換でも重要な概念であるが， MET では最小エルブラン・モデル自身は変化しないため，問合せ処理の効率化には限界がある．したがって，問合せ変換ではもっと条件の弱い変換が必要である．

[定義 2.2.1] 目標 g に関する，演繹データベース DDB と DDB' の解をそれぞれ G と G' とする．G = G' の時，DDB と DDB' は g に関し目標等価であるといい，DDB ≡g DDB' と書く．□

[定義 2.2.2] f を変換とし DDB' = f(g, DDB) とする．任意の与えられた g と DDB について，DDB' ≡g DDB の時，f は目標等価変換 (GET) であるという．□

[定義 2.2.3] f を変換とし DDB' = f(g, DDB) とし G' と G をそれぞれ DDB' と DDB の g に関する解であるとする．G'⊃G の時 f は完全，G' ⊂ G の時 f は健全であるという．□

健全かつ完全な変換は GET である．

[命題 2.2.1] MET は任意の目標に対し GET である．□

[定義 2.2.4] GET f，h の結合 k = f∘h を以下のように定義する．
k(g, DDB) = h(g, f(g, DDB))．□

[命題 2.2.2] f と h を GET とする．≡g の推移性により，f∘h は GET である．□

[定義 2.2.5] g を目標，F∈DDB とし，g⇝F とする．DDB から DDB−{F} への変換を節削除という．
p を DDB 中に存在しないか，g⇝p なる述語記号，S を p を頭に持つ節の集合とする．DDB から DDB∪S への変換を節追加という．□

[命題 2.2.3] 節削除は GET である．□

[命題 2.2.4] 節追加は GET である．□

[定義 2.2.6] S を節集合，E⊂DDB，全ての F∈S に対して DDB ⇒ F とする．DDB から (DDB−E)∪S への変換を節置換という．
E を節集合，S⊂DDB，DDB' = (DDB−S)∪E とする．全ての F∈S に対して DDB'⇒F と仮定する．この時，DDB から DDB' への変換を逆節置換という．□

[命題 2.2.5] 節置換は健全であり，逆節置換は完全である．□

節置換は，最小エルブラン・モデルを縮小する変換であり，次節で述べる種々の変換の基本原理である．

3 ホーン節変換

演繹データベースでは，EDB が大きく，EDB 中の節は関係データベース技術による効率的処理が可能である．したがって，問合せ変換で EDB 中の節を変換することは通常得策ではない．このことから，次のような概念を導入することができる．

[定義 3.0.1] f が GET でかつ変換 h が存在し，次のように表せる時，f はホーン節変換 (HCT) であるという．任意の DDB に対して f(g, DDB) = h(g, IDB)∪EDB．□

HCT は EDB を変化させない GET である．なお h をホーン節変換と呼ぶこともある．

目標と内包データベースを合わせたものは，関係データベースでの

問合せに相当する．したがって，HCT は本質的に関係データベースにおける問合せ変換［Ullman 82］の一般化であり，HCT の定義を演繹データベースにおける問合せ変換の定義と考えることができる． HCT の主な目的は，これに引き続く処理とは独立に，モデルをより小さなモデルへと変換することである．したがって，HCT の最適化はモデルの大きさを判定基準とする．

以下では，前章までに述べた原理を応用し各種の HCT を論じる．

3.1 部分評価によるホーン節変換

［定義 3.1.1］節 $C \in DDB$ を 述語記号 p を本体に持つ節とする．C' を C と $F \in C_b(p, DDB)$ に対して p で（最大単一化置換による）導出を行った導出形とする．C' が存在するならば $\{C\} \cup C_b(p, DDB) \Rightarrow C'$ は明らかである．このような C' の集合を S とすると，DDB から $(DDB - (\{C\} - (\{C\} \cap C_b(p, DDB)))) \cup S$ への変換を導出による節置換という．□

導出による節置換は，［Tamaki 84］の展開と同じ概念である．

［命題 3.1.1］導出による節置換は MET である．
［証明］この変換は命題 2.2.5 により任意の目標に対して健全である．完全性は DDB と DDB' の SLD 木を比較すれば容易に証明できる．■

$sym(IDB) \cap sym(EDB) = \phi$ なので $p \in sym(IDB)$ ならば $C_b(p, DDB) = C_b(p, IDB)$ となることから次の定義を導入する．

［定義 3.1.2］導出による節置換に基いた変換 res(p, IDB) を次のように定義する．

```
res(p, IDB);
    R := φ;
    for every pair C_b∈C_b(p, IDB) and C_h∈C_h(p, IDB) do;
        R := R∪{resolvents of C_b and C_h on p};
    end;
    IDB' := IDB−(C_b(p, IDB)−C_h(p, IDB))∪R;
    return IDB';
end.□
```

［定義 3.1.3］部分評価手続き p_eval(g, IDB, T) を，T を与えられた述語記号集合として次のように定義する．

```
p_eval(g, IDB, T);
    repeat:
        select p s.t. p∈(sym(IDB)−T−sym(g));
        IDB := res(p, IDB);
    until no such p;
    IDB' := IDB−{C|g↛*C};
    return IDB';
end.□
```

［命題 3.1.2］p_eval(g, IDB, T) が停止するならば HCT である．
［証明］p_eval は導出による節置換と節削除の結合である．したがって，命題 3.1.1，2.2.2，2.2.3 により GET である． ■

集合 T を適当に与えることにより［宮崎 88a］，HCT を得ることができ，部分評価による HCT（HCT/P）と呼ぶ．HCT/P は，いくつかの述語記号を削除することで元のモデルを縮小している．

問合せは HCP/P により簡略化可能である．例えば，非再帰問合せは目標の述語記号以外は EDB の述語記号しか含まない形に変換できる．また，問合せを自己再帰節（同一の述語記号を頭と本体に持つ述語）と非再帰節しか含まない形に変換することも可能である．

HCT/P の概念は［Miyazaki 86］で提案され，［宮崎 88a］で一般化された．HCT/P の実現は［阿比留 87］と［Itoh 86b］に報告されている．また，HCT/P の複数問合せの最適化への応用は［Sakama 87］に報告されている．関係代数における同様の変換は［Ceri 86］に報告されている．

たたみ込み［Tamaki 84］を用いた HCT も考えることができるが，たたみ込みをどのような条件で行うかを決定するのが困難なため，たたみ込みに基く自動変換を定義するのは困難である．

3.2 代入によるホーン節変換（HCT/S）

［定義 3.2.1］節 C の頭の変数への具象代入の結果を C'とする．明らかに $\{C\} \Rightarrow C'$ である． DDB から $(DDB - \{C\}) \cup \{C'\}$ への変換を具象代入による節置換という．□

この変換は，命題 2.2.5 により健全であるので，変換の完全性を検討することにより HCT が得られる．

［定義 3.2.2］DDB の成分を具象代入による節置換により変換する手続き Comp/S(p, Comp) を次のように定義する．

```
Comp/S(p, Comp);
    I := {p};
    Comp' := Comp;
    repeat;
        C := a clause in Comp whose head symbol is in I;
        C' := a clause obtained from C by ground substitution
                    using ground terms in an elment of I;
```

```
        Comp' := Comp' − {C} ∪ {C'};
        I := I ∪ {(partially)grounded atoms in the body of C'};
    until no change;
    return Comp';
end.□
```

[命題 3.2.1] Comp' := Comp/S(p,Comp) とする．Comp' の各節の頭が，Comp' 中の節の本体に在る同じ述語記号の基本式全てに対してより一般的ならば，Comp $\equiv_p$ Comp' である．ここで，Comp $\equiv_p$ Comp' は Comp より下の節集合を E とした時 Comp∪E $\equiv_p$ Comp'∪E であることを意味する．

[証明] 健全性は命題 2.2.5 より明らか．完全性は命題の条件が DDB の SLD 木において各副目標が Comp' 中の節の頭より一般的でないことを保証していることからいえる．■

[定義 3.2.3] 代入による IDBの変換手続きHCT/S(g, IDB)を定義する．

```
HCT/S(g,IDB);
    Comp := a component of IDB that indludes C_h(g,IDB);
    Comp':= Comp/S(g,Comp);
    if not Comp' ≡g Comp then Comp' := Comp;
    IDB':= Comp';
    for every directly lower component Lcomp of Comp do;
        for every entry p of Lcomp
                that appears in Comp' do;
            IDB':= IDB' ∪ HCT/S
                    (p,Lcomp ∪ (lower components of Lcomp));
        end;
    end;
    return IDB';
end.□
```

HCT/S の結果は他の節よりより一般的でない節を含む場合があるが，このような冗長な節は，HCT/S の実行中あるいは実行後に，容易に削除することができる．

[命題 3.2.2] HCT/S は HCT である．

[証明] 命題 3.2.1 より明らか．■

HCT/S が関係代数での非再帰問合せや推移閉包における選択の分配 [Aho 79] の概念の拡張であることは明らかである．関係代数における同様の方法が [Ceri 86, Miyazaki 86] にあるが，その適用範囲は簡単な問合せに制限されている．HCT/S と同様の効果が，規則/目標グラフを用いても得られる[Kifer 86]． HCT/S は DDB を Datalog に制限せず，また出力が確定節集合なので他の任意の方法と組み合わせることが可能であり，最も一般的である．

3.3 制約子によるホーン節変換 (HCT/R)

本節では，包摂による HCT について述べる．

[定義 3.3.1] S⊂sym(DDB) とする．sym(r)∈S の時，DDB 中に存在しない述語記号 sym(r*) を sym(r) に対応させる．基本式 r* の引数の数は r のものと等しい． r* を頭に持つ節を DDB に追加する変換は命題 2.2.4 により GET である．

この準備のもとに DDB 中の全ての節を対象として，節（C とする）⟨r,B⟩∈DDB，ただし sym(r)∈S，を節（C' とする）⟨r,{r*}∪B⟩ で置き換える変換を考える．明らかに C ⊆ C'．したがって，この変換は節置換であり，任意の目標に対して健全である．この変換を制約子による節置換と呼ぶ．また r，sym(r)，C，r*，sym(r*)，C'，S をそれぞれ，被制約基本式，被制約述語記号，被制約節，制約基本式，制約述語記号（あるいは制約子），制約節，被制約述語記号集合と呼ぶ．□

制約子による節置換が完全な時，この変換は GET である．この変換が HCT の時，制約子による HCT (HCT/R) と呼ぶ．

[命題 3.3.1] HCT/R f，ただし DDB'= f(g,DDB)，の制約節を ⟨r*,B⟩ とする．DDB' で ⟨r*,B⟩ を ⟨r*,B'⟩，ただし B' は B の部分式，で置き換えた結果を DDB" とする．この時，DDB $\equiv_g$ DDB" である．

[証明] G，G'，G" をそれぞれ目標 g に関する DDB，DDB'，DDB" の解とする．DDB から DDB" への変換は節置換なので， G⊃G"．DDB' から DDB" への変換は逆節置換なので G"⊃G'．G = G' なので G = G"．■

[定義 3.3.2] g を目標，S を被制約述語記号集合，sym(g)∈S と仮定する．g と同じ引数の単位節 g*を頭部に持つ単位節を初期(制約)節と呼ぶ．□

[定義 3.3.3] sym(r)を被制約述語記号，被制約節を C'=⟨p,{r}∪R⟩ とする．R' を R の部分式とすると，r に対して節 C* = ⟨r*,B'⟩，ただし r* の引数は対応する r と同じ，を候補等価制約節（CERC）と呼ぶ．C' 中に sym(r) に対応する基本式が複数在る場合は，各基本式に対して CERC を定義する．□

[定義 3.3.4] 被制約節 C' の本体に被制約基本式が n 個在るとする．C' の各被制約基本式に対し１つずつ CERC を選んで得られる節集合を N とする．N の元は n 個である． C*∈ N について次のような関

係'→'を考える． $sym(C^*) = p^*$ で，被制約基本式 r が C^* の本体にある時，$sym(p) \rightarrow sym(r)$ と定義する．ここで，本体に複数回現れる述語記号は（適当な番号付け等を行って）異なる記号として扱う．C^* の本体に m 個の被制約基本式が在るとすると，C^* より→関係が m 個得られる．N の元の全てに対して →関係に対応する有向グラフを考え，これらグラフの和を考える．このグラフに閉路がない時，N は非循環であるという．また $R(N)$ をこのような N 全ての和集合とした時，全べての N が非循環ならば，$R(N)$ が非循環であるという．□

［命題 3.3.2］定義 3.3.1，3.3.4 によって演繹データベース DDB から得られる被制約節と初期節および非循環な $R(N)$ の元から構成される節集合を DDB' とする．この時，DDB $\equiv_g$ DDB'．
［証明のスケッチ］健全性は命題 2.2.5 より明らか．完全性は g を根とする DDB と DDB' の SLD木を以下のように比較して得られる．もし，DDB の各節点に対応する DDB' の節点が存在するなら，DDB の各成功点が DDB' に存在することになり変換は完全である．これは，DDB' に関して制約基本式を根とする部分木が，少なくとも1つの成功枝を持つことを数学的帰納法により証明することで示すことができる．非循環性の条件はSLD反駁における計算規則を定めるのに用いる．詳細な証明は［宮崎 88b］を参照．■

上記 HCT/R は概念的には単純であるが，ボトムアップの計算途中でエルブラン・モデル全体の値を取るような引数を含むため実現が容易ではない．このような引数を省くために次の定義を導入する．

［定義 3.3.5］定義 3.3.1 を次のように変更した変換を部分制約子による節置換と呼ぶ．
(1) b/f を r の引数の数と同じ長さの b と f の列とし，r^* を修飾して得られる基本式を $r^{*-b/f}$ とする．$r^{*-b/f}$ の引数の数は修飾中の b の数と等しい．この基本式の述語記号を部分制約子と呼ぶ．
(2) 被制約節は，$\langle r, \{r^*\} \cup R\rangle$ の代りに $\langle r, \{r^{*-b/f}\} \cup R\rangle$ となる．ここで，$r^{*-b/f}$ の引数は頭の r の b に対応する位置の引数と同じである．□

命題 3.3.1 は，この変換についても成立する．

［定義 3.3.6］部分制約子による節置換における初期節は定義 3.3.2 の g^* のかわりに部分制約基本式 $g^{*-b/f}$ を用いる．□

［定義 3.3.7］部分制約子による節置換における CERC は，定義 3.3.3 の r^* を $r^{*-b/f}$ で置き換えて得られる．ただし，$sym(r^*)$ に対応して $sym(r^{*-b/f})$ は複数考えられるので，その内1つを選ぶものとする．□

［定義 3.3.8］部分制約子による節置換における CERC 集合は，定義 3.3.3 を定義 3.3.7 で置き換える以外は定義 3.3.4 と同じである．非循環性の定義も r^* を $r^{*-b/f}$ で置き換えて得られる．□

［命題 3.3.3］f を部分制約子による節置換とし，DDB' = f(g, DDB) とする．制約節は，初期制約節と非循環な CERC 集合により与えられる．もし，sym({初期節} ∪ R(N)) が被制約節の本体中に現れる部分制約子全ての集合の部分集合ならば DDB' $\equiv_g$ DDB．□

本命題の証明は，命題 3.3.2 と同様である．本命題の条件は，命題 3.3.2 より多いが，これは1つの被制約述語記号に対応して複数の部分制約子が考えられるためである．

命題 3.3.3 は，部分制約子による節置換が GET であるための条件を与える．このような変換の最適化について議論しよう．EDB が大きい時，EDB の節を変換する費用は大きいので，一般の GET ではなく HCT/R を考える．

任意の EDB に対し，最小エルブラン・モデルを最小化するような HCT/R を得ることは困難であり，そのような変換方法は存在しないかも知れない．したがって，束縛条件を可能な限り保存するような束縛伝播アルゴリズムを用いた HCT を以下に示す．

［定義 3.3.9］被制約節 C = <r, B> の本体に非制約述語記号 p が存在するとする．B' を B から p を削除した式とし，C' = <p*, B'>，とするp* は C の本体の p と同じ引数を持つ．C' の束縛伝播記号（BPS）を次のように再帰的に定義する．
(a) B' 中の部分制約基本式中の変数は BPS である．
(b) 定数は BPS である．
(c) B' 中のある基本式のある引数中に BPS があり，（その引数が関数で）その引数中に変数が存在しなければ，その基本式の変数は全て BPS である．
(d) (a, b, c) 以外に BPS は無い．□

次に非循環 CERC 集合を生成するアルゴリズムを示す．このアルゴリズムは本質的には［Sacca 87］における束縛グラフを用いたアルゴリズムの拡張である．また，このアルゴリズムは［Beeri 87］の最適全 SIP を求めることに対応している．ただし，［Beeri 87］では最適 SIP を求める方法は述べられていない．

［定義 3.3.10］被制約節 C = <r, B> から CERC を生成する手続き genCERC(C) を次のように定義する．

```
genCERC(C);        /* C = <r,B>    */
    R := φ;
    for every restricted atom p in B do;
```

```
        B' := B－{p};
        C' := <p*,B'>;  /*  p* は p と同じ引数とする */
        decide BPS in C' according to definition 3.3.9;
        B" := B'－{atoms that do not contain BPSs};
        decide atom p*-b/f
            according to positions of BPSs in p* in F;
        R := R∪{<p*-b/f,B">};
    end;
    if R is not acyclic then R := makeAcyclic(R);
    return S;
end.□
```

手続き makeAcyclic(R) は CERC 集合中の節の本体から一部の基本式を削除し非循環な CERC 集合を出力する．どの基本式を削除すべきかは，通常 EDB に依存する．

この準備のもとで束縛伝播アルゴリズム（BPA）を示す．BPA は g から到達可能な sym(IDB) を全て被制約述語記号集合とするような変換を行う．

〔定義 3.3.11〕手続き <u>BPA(g, IDB)</u>．

```
BPA(g, IDB);
    Trestrictor := {g*-b/f}; /* g*-b/f is an initial clause */
    Trestricted := φ;
    do for every p*∈sym(Trestrictor);
        R := {restricted clauses having p* in their bodies
            generated according to definition 3.3.5;
        Trestricted := Trestricted∪R;
        do for every (new) C∈Trestricted do;
            Trestrictor := Trestrictor∪genCERC(C);
        end;
    end;
    return Trestricted∪Trestrictor;
end.□
```

定義と命題 3.3.3 から次の命題は明らかである．

〔命題 3.3.4〕BPA は HCT である．□

BPA は定数による束縛を可能な限り保存し，どのような IDB と EDB に対しても十分に小さな最小エルブラン・モデルを得られるという意味で最適な HCT/R である．genCERC により非循環な CERC 集合は多くの問合せに対して一意に定まる．しかし，makeAcyclic が必要な時，可能な ECRC 集合の中からどれが最適かを決定するためには，通常 EDB の情報が必要である．この決定に用いられる情報としては，関係の大きさや関係の索引の位置等がある．

HCT/R は，相互再帰を含む任意の問合せに対しても有効である．HCT/R または BPA と同様の結果を得るアルゴリズムもいくつか提案されている．例えば，マジック集合（MG）〔Bancilhon 86, Sacca 86, Sacca 87〕や一般化マジック集合（GMG）〔Beeri 87〕がある．これらの方法が HCT/R の条件を満足していることは容易に分かる．MG が用いられるのは線形や入れ子型等の特別な場合に場合に限定されるが，その結果は BPA と同じかほぼ同じである．GMG はもっと広い範囲の問合せに適用できる．GMG と HCT/R（BPA）の主な違いの一部を以下に示す．

(a) シンタックス的には GMG は制約述語記号と被制約述語記号の両方が修飾される．HCT/R では制約述語記号しか修飾されない．被制約述語記号の修飾も定義の変更で可能だが，等価変換に本質的に必要ではない．

(b) GMG はトップダウン処理の手続き的セマンティックスにより定式化されているのに対し，HCT/R は宣言的セマンティックスで定式化されている．したがって，変換と実行アルゴリズムの分離がより完全である．例えば，GMG では DDB の節の形を多少制限するが，HCT/R では制限は無い．このように，命題 3.3.3 の HCT/R の条件は今まで知られている条件では最も弱い条件である．

3.4 その他の変換

前節までにモデルの大きさを縮小する変換について述べた．本節では，これら変換に後続する処理において行われる冗長な演算を省くような変換について述べる．

関係データベースにおける最適化の方法の1つに，共通な部分表現を見つけて，その部分を1回で処理する方法がある〔Ullman 82〕．演繹データベースでは，これは共通表現を新しい基本式で置き換えることに対応し，次の定義を導入する．

〔定義 3.4.1〕{B1, ... ,Bn} を {C1, ... ,Cn}⊂DDB の各節 Ci それぞれの本体の集合とする．Bi' を Bi の部分式とした時，お互いに等価な式 Bi' の集合 {B1',.., Bn'} が存在するとする．sym(pi) を DDB 中に存在しないと仮定して C = <pi,Bi'> とする．pi の引数は Bi' 中の変数全てとする．θi を代入とし，Bi = θi Bi かつ pi = θi pi とする．Ci' を Ci の本体中の Bi' を pi で置き換えた節とする．DDB から DDB－{C1, ..., Cn}∪{C, C1', ..., Cn'} への変換を<u>共通表現による逆節置換</u>という．□

〔命題 3.4.1〕共通表現による逆節置換は GET である．

〔証明〕Ci は C と Ci' の導出節なので，この変換は逆節置換であり，命題 2.2.5 により完全である．sym(pi) を頭に持つ節は C だけなの

で，この変換は健全である．この結果は，目標とは独立である．■

共通表現による逆節置換は 3.1 節の導出による節置換の逆変換であり，この変換を用いた HCT を考えることができる．例えば，HCT/R において genCERC での入力の被制約節と，それから生成される制約節には共通な部分式が多いので HCT/R の出力に対して共通表現による逆節置換を適用できる．この考えに基いた方法として，ミニマジック [Sacca 87] と一般化補助マジック [Beeri 87] とがある．

以上は，変換に続く処理のアルゴリズムとは独立な方法であるが，後続処理のアルゴリズムに依存した方式として，数え上げ/逆数え上げ [Banchilhon 86a]，一般化数え上げ [Beeri 87, Sacca 86a]，マジック数え上げ [Sacca 86b] がある．これらの方式は，等価性や等価変換の定義に対する修正，例えば等価性は，引数の追加を許すために，解集合の射影を対象とするよう一般化を行うことにより，HCT と同一の枠組みで定式化することができると考えられる。

4 結論

本論では，問合せ変換の原理を論じ，モデル等価変換と目標等価変換等を定義した．等価変換の特別な場合であるホーン節変換（HCT）を議論し，HCT/P，HCT/S，HCT/R 等の方法を示した．HCT により，解を保存したまま最小エルブラン・モデルを小さくできるので，特にボトムアップによる処理の効率化が図れる．従来の方法とこれら HCT との比較を論じ，問合せの簡略化，選択の分配，マジック集合，共通表現による方法等が HCT の特別な場合であることを示した．

HCT は確定節集合から確定節集合への変換なので，問合せ最適化で HCT どうしや他の方法を組み合わせることが可能である．例えば，最初に HCT/P で問合せを簡略化し， HCT/S の適用可能性を検査する．HCT/S で最適化した結果が得られないなら HCT/R を用いる等が考えられる．

本稿では，データベースを確定節に限定したが，節の本体に負リテラルを許した層状（stratified）データベースにおける問合せ変換を考えることもできる．この時，HCT/P と HCT/S の拡張は，これらの変換で層構造が保存されるので容易に行うことができる．しかし，HCT/R では一般には層構造が保存されないので，本稿の議論の直接的な拡張ではうまくいかない場合がある．層状データベースにおける HCT については別に報告する予定である．

問合せ変換は FGCS プロジェクトにおける KBM PHI システム [Itoh 86a] における問合せコンパイルの一部として使用する．現在，HCT/P と HCT/S の実現は完了し HCT/R の実現を行っている．

《参考文献》

[Aho 79] Aho, A. V. and Ullman, J. D., Universality of Data Retrieval Languages, 6th ACM POPL, 1979.

[Bancilhon 86a] Bancilhon, F., et. al., Magic Sets and Other Strange Ways to Implement Logic Programs, 5th ACM PODS, 1986.

[Bancilhon 87b] Bancilhon, F. and Ramakrishnan, R., An Amateur's Introduction to Recursive Query Processing Strategies, ACM SIGMOD, 1986.

[Beeri 87] Beeri, C. and Ramakrishman, R., On the Power of Magic, ACM PODS, 1987.

[Ceri 86] Ceri, S., et. al., Translation and Optimization of Logic Queries: The Algebraic Approach, 12th VLDB, 1986.

[Itoh 86a] Itoh, H., Research and Development on Knowledge Base Systems at ICOT, 12th VLDB, 1986.

[Itoh 86b] Itoh, H. et. al., A Deductive Database System Written in Guarded Horn Clauses, ICOT TR-214, 1986.

[Kifer 86] Kifer, M. and Lozinskii, E. L., Filterling Data Flow in Deductive Databases, ICDT, 1986.

[Kitakami 84] Kitakami, H., et. al., A Methodology for Implementation of a Knowledge Aquisition System, Intl. SLP, 1984.

[Miyazaki 86] Miyazaki, N., Yokota, H. and Itoh, H., Compiling Horn Clause Queries in Deductive Databases: A Horn Clause Transformation Approach, ICOT TR-183, 1986.

[Sacca 86a] Sacca, D. and Zaniolo, C., On the Implementation of a Simple Class of Logic Queries for Databvases, ACM PODS, 1986.

[Sacca 86b] Sacca, D. and Zaniolo, C., The Generalized Counting Method for Recursive Logic Queries, ICDT, 1986.

[Sacca 87] Sacca, D. and Zaniolo, C., Implementation of Recursive Queries for a Data Language Based on Pure Horn Logic, ICLP, 1987.

[Sagiv 87] Sagiv, Y., Optimizing Datalog Programs, ACM PODS, 1987.

[Sakama 87] Sakama, C. and Itoh, H., Partial Evaluation of Queries in Deductive Databases, WS on Partial Evaluation and Mixed Computation, 1987.

[Tamaki 84] Tamaki, H. and Sato, T., Unfold/Fold Transformations of Logic Programs, 2nd ICLP, 1984.

[Ullman 82] Ullman, J. D., Principles of Database Systems, (2nd edition) Chapter 8, Computer Science Press, 1982.

[阿比留 87] 阿比留 他, KBMS PHI: 演繹データベース実験システム PHI/K2, 情報処理学会第34回全国大会, 1987.

[野口 87] 野口，滝沢，知識工学基礎論，オーム社，1987.

[宮崎 88a] 宮崎 他，ホーン節変換:演繹データベースにおける部分評価の応用，情報処理学会論文誌 29-1, 1988.

[宮崎 88b] 宮崎，伊藤，演繹データベースにおける制約子付最小不動点，ICOT TR-257, 1987, revised 1988.

[古川 87] 古川，溝口，プログラム変換，共立出版，1987.