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†

宮崎収兄†† 羽生田博美†† 伊藤英則†††

演繹データベースにおいて再帰型の問合せの処理方式がいくつか提案されているが，多くの方法ではその処理性能が問合せ中の述語数に依存する．したがって，問合せから一部の述語を除去し簡略化してから処理を行うことにより効率化を図ることができる．本論文では，このような簡略化を行うホーン節変換を，論理型プログラムの部分評価の概念に基づき提案する．部分評価は，論理型プログラムの実行時に一部の述語の評価を保留するアルゴリズムであり，プログラムの部分計算法の一種である．部分評価の結果は，再びホーン節で表現される．部分評価で保留する述語の与えかたを変化させることにより各種のホーン節変換を定義できる．また，これらのホーン節変換は“not”述語を含む問合せの簡略化にも適用できる．

1. まえがき

演繹データベースの問合せは，ホーン節を用いて表せる．その問合せ処理方式は，数多く提案されているが[2]，大部分の方式では，問合せ中の述語数に性能が依存する．したがって，問合せ中の一部の述語を除去し問合せを簡略化することにより処理性能を向上することができる．

基本アルゴリズムの中に，述語の除去機能が組み込まれている問合せ処理方式もある[6]が，適用範囲が簡単な問合せに限定されている．また，問合せを関係代数に変換してから簡略化する方法[4]も提案されているが，関係代数に基づいた方法にしか適用できないという限界がある．

問合せの簡略化をホーン節形式のまま行えば，任意の問合せの前処理として適用できる．このような処理をホーン節変換と呼ぶ．ホーン節変換は，問合せのコンパイル時，特にプリコンパイル時に適用すると効果が大きいと考えられる．

本論文では，部分評価を用いたホーン節変換を提案し，その性質を論じる．また，論理型言語処理系を用いたホーン節変換の実現方法について論じる．なお，本論文では，特に明記しない限り術語は文献[15]に従う．

2. 問合せ処理の問題点とホーン節変換

演繹データベース（DDB）は，ホーン節（より正確には，確定節）の集合であり，具象単位節（事実節）からなる外延データベース（EDB）と，それ以外の節（規則節）からなる内包データベース（IDB）から構成される．すなわち，

DDB≡確定節集合
＝IDB∪EDB.

DDB に対する問合せは，$X_1, \cdots, X_n$ を自由変数とする一階の述語論理式 $\phi(X_1, \cdots, X_n)$ で記述される．この問合せの解は，c_i を定数または具象項，$\vdash$ を証明可能性を表すとして，

DDB$\vdash \phi(c_1, \cdots, c_n)$

であるような，$c_1, \cdots, c_n$ の組の集合である．

一般に，閉じた論理式 f と g が与えられた時，一階述語論理の演繹定理および帰謬法定理[15]から，

DDB$\cup \{f\} \vdash g \Leftrightarrow$ DDB$\vdash g \leftarrow f$

である．したがって，問合せ中に規則節またはその集合を記述することと，その節または節集合をデータベースに入れることとは，実質的に等価である．このため，本論文では問合せの形式を制限し，規則節はすべて IDB に含まれると考え，問合せは1つの述語からなる基本式 g で記述するものとする．また，誤解の恐れが無い場合，g と IDB の全体を問合せし，g を目標と呼ぶ．

データベースでは，解の個々の元ではなく，解集合を問題とする．このため，事実節で同じ述語記号を持つ節集合を実関係に，規則節の頭部に現れる述語を仮想関係に対応させ，関係データベースにおける関係代数演算に類似した方法で，これらの関係を計算することにより問合せ処理を行う方式が多い[2]．

次のような問合せの処理を考えよう．

例1 [問合せ Q 1]
目標：$p(X, Y)$
IDB：$p(X, Y) \leftarrow p1(X, Y)$
$p(X, Y) \leftarrow q(X, Z),\ p2(Z, Y)$

† Horn Clause Transformation: An Application of Partial Evaluation to Deductive Databases by NOBUYOSHI MIYAZAKI, HIROMI HANIUDA (Systems Laboratory, Oki Electric Industry Co., Ltd.) and HIDENORI ITOH (Research Center, Institute for New Generation Computer Technology).

†† 沖電気工業(株)研究開発本部総合システム研究所
††† (財)新世代コンピュータ技術開発機構研究所

$q(X, Y) \leftarrow p(X, Y)$

EDB には，$p1$ と $p2$ に対応する関係が存在すると仮定する．この時，次のような問題がある．

（1） p と q が相互に依存しているので，適用できない問合せ処理方式がある[2]．

（2） 適用できる方式でも，2つの仮想関係 p と q の計算を行うため処理効率が良くない場合が多い．

この問合せは，q を消去した次の問合せ Q2 と等価であることが容易に分かる．

[問合せ Q2]

目標：$p(X, Y)$

IDB：$p(X, Y) \leftarrow p1(X, Y)$

$p(X, Y) \leftarrow p(X, Z),\ p2(Z, Y)$

Q1 は処理できない方式でも，Q2 のように簡略化すれば処理できる[2]．また，Q2 では，仮想関係が p だけなので，Q1 よりも効率の良い計算ができる[4),9)]．

一般の問合せが与えられた時，実際の問合せ処理を行う前に，例1の Q1 から Q2 への変換のような，問合せの簡略化を行っておくことにより，問合せ処理の適用範囲を広げたり，問合せ処理の効率を向上させることが可能である．

目標 g と DDB を与えた時，その解 Ans を出力する演繹データベース処理系 eval を考えよう．すなわち，

Ans :=eval(g, DDB).

演繹データベース IDB∪EDB と目標 g が与えられた時，これらを入力とし，次に示すような性質を持つ変換 hct をホーン節変換と呼ぶ．

IDB′ :=hct(g, IDB) かつ

eval(g, IDB′∪EDB)=eval(g, IDB∪EDB).

すなわち，hct は IDB 部分を変換し，g に対して元の DDB によるものと同じ解を与える IDB′∪EDB を生成する変換である．

ホーン節変換を用いた問合せ処理は，次のように2段階に分けて行われる．

IDB′ :=hct(g, IDB);

Ans :=eval(g, IDB′∪EDB).

以下では，部分評価を用いたホーン節変換で問合せを簡略化することにより，その後の問合せ処理の効率化を図る方法を議論する．この効率化は，通常のプログラムで部分計算を行うことでプログラムを特殊化し，実行時の高速化を図る[16)]のと類似しており，Prolog における部分評価とその応用も提案されている[14)]．

部分評価は，プログラム変換で用いられる展開(unfolding)[13)]とも密接な関係がある．本論文では，部分評価を問合せ構造の簡略化に応用した変換について論じる．

3. 部分評価

本章では，ホーン節変換を定義するための準備として，部分評価処理系 p_eval の概念を導入する．

3.1 諸定義

本論に入る前に，さらにいくつかの基本概念を定義する．なお，議論を簡単化するため，IDB 中の規則節の頭部の述語記号には，EDB の述語と同じものは無いと仮定する（述語記号の付け替え等で，簡単にこの条件を満足できるので，この仮定を導入しても一般性を失わない）．また，同じ述語記号の述語は引数の数が等しいと仮定する．

IDB 中の述語記号が同じ述語を1つの節点に対応させ，節の頭部が本体中の述語に依存すると考えて，有向グラフを考える（p が q に依存する場合，$p \rightarrow q$ とする）．これを，IDB の依存グラフと呼ぶ．

依存グラフ中で，ある述語 p から径路が存在する述語を，p から到達可能な述語と呼ぶ．到達可能な節も同様に考えることができる．さらに，自分自身に到達可能な述語を，再帰述語，自分自身への枝（長さ1の径路）が存在する述語を，自己再帰述語と呼ぶ．また，IDB 中の節の頭部の述語を中間述語と言う．

依存グラフの強連結成分は，互いに到達可能な節点の集合とこれらの節点間の径路である[4)]．IDB 中の成分とは，対応する依存グラフ中の強連結成分に対応する述語集合を言う．同一成分に属するという関係は同値関係である．

成分中の述語の定義節のすべてに関して，本体部に含まれる同じ成分の述語がたかだか1つの時，問合せの成分は線形であると言う．再帰述語を含む成分を再帰成分と言い，問合せ中の全成分が線形の時，問合せは線形であると言う．

問合せの処理は DDB 上で行うが，この処理は DDB 中で g から到達可能な節を対象としても結果は同じである．ホーン節変換は，概念的には DDB 中から g から到達可能な節集合を取り出す操作も含むが，以下では議論を簡単化するため，DDB は，g から到達可能な節のみを含むものと仮定する．

以下では，EDB 中の述語記号の集合を E で，IDB 中の比較述語記号（=, <等）の集合を C で表す．

3.2 部分評価処理系

一階述語論理における証明は，導出原理に基づいて

行われる．導出は，単一化可能な基本式 a_1 と a_2 とを，正リテラルと負リテラルとしてそれぞれ持つ2つの節，c_1 と c_2 があった時，これらの節をもとに a_1 と a_2 を消し，新しい節 c を生成する．

証明の方法として，Prolog 等で用いている SLD 導出等多くの方法が提案されているが，これらの方法間の差異は，導出原理をいかに適用するかにある[15]．

部分評価は，一部の述語については導出を行うが，あらかじめ指定された述語については導出を行わないような処理である．

まず，与えられた述語記号 p について，すべての可能な導出を行う処理系 res(p, IDB),

IDB′ :=res(p, IDB)

を考える．P(IDB) を IDB 中の述語記号の集合，$B(p)$ を述語記号 p を本体に持つ IDB 中の節の集合，$H(p)$ を頭部の述語記号が p である IDB 中の節の集合として，res(p, IDB) は次のように定義される．

res(p, IDB);
  for every pair $b_i \in B(p)$ and $h_j \in H(p)$ do
    b'_{ij} :=
      result of resolution of b_i and h_j on p;
  /*b_i の本体に複数回 p が現れる時は，その各々に対して導出を行う */
  IDB′ :=(IDB−($B(p)$−$H(p)$))∪ {b'_{ij}}
end.

$H(p)$ 中の節の本体に p が無かった場合，IDB′ の各節の本体に p は現れなくなる．IDB′ の生成時に，$H(p)$ を残すのは，本体に p が残っている時，p を頭部に持つ節がその後の p の評価に必要なためである．この結果，res(p, IDB) は IDB の節の頭部の述語を目標とする問合せに対して等価変換となり，結果は有限回の演算で一意に定まる．

以上の準備の下に，非評価述語記号の集合 T を与えた時，T の元については導出を行わないような部分評価処理系 p_eval

IDB′ :=p_eval(g, IDB, T)

を，sym(g) を述語 g の述語記号を表すとして，次のように構成する．ただし，ここでは IDB の変換を問題とするので，$E \cup C \subseteq T$ とする．

p_eval(g, IDB, T);
  repeat
    select p s. t.
    $p \in (P(\text{IDB}) - T - \{\text{sym}(g)\})$;
    IDB :=res(p, IDB)
  until(no such p);
  D :=(IDB 中で，頭部の述語記号が
      $T \cup \{\text{sym}(g)\}$ の元でない節の集合);
  IDB′ :=IDB−D
end.

p_eval の定義から，IDB′ が有限ならば，P(IDB′) の元は目標の述語記号または T の元である．p_eval が停止し，有限の IDB′ を出力する限り，p_eval は目標 g に関する等価変換になっている．すなわち，g の具像基本式 g' に関して

IDB∪EDB⊢g' ⇔ IDB′∪EDB⊢g'.

次章では，この p_eval を用いたホーン節変換について述べる．

4. ホーン節変換

4.1 ホーン節変換の構成的定義

出力が有限になるような p_eval の適用を考えると，p_eval は g に関する等価変換なので，2章で述べたホーン節変換の性質を満たす．以下では，このような変換を行うために，非評価述語記号集合 T を選び，ホーン節変換を構成する．

まず，前章と同じく $E \cup C \subseteq T$ とする．

p_eval 中で res(p, IDB) を実行する時，結果の節の本体に p が残ることがあれば p_eval は停止しない．これは，何度 res(p, IDB) を実行しても，本体中の p を消去できないからである．このようなことが起こるのは，p が再帰述語である場合のみである．したがって，全再帰述語の述語記号を T の元とすれば，p_eval は停止する．

IDB 中の全再帰述語の述語記号を T の元とすることにより得られるホーン節変換を，タイプ1ホーン節変換と呼ぶ．

[タイプ1ホーン節変換 (hct 1)]

R=｛再帰述記号｝として，
  IDB′ :=hct 1(g, IDB)
    =p_eval(g, IDB, $E \cup C \cup R$).

hct 1 の結果は有限で一意に定まり，IDB′ の元となる節の本体には，$E \cup C \cup R$ の元に対応する述語のみが現れる．

hct1 により，本体に現れる非再帰型の中間述語はすべて除去できる．しかし，依存グラフ中に長い閉路がある場合，この閉路中の述語はすべて再帰述語になるので，hct1 では簡略化が行えない．したがって，前章例1の Q1 のような簡単な線形問合せでも簡略化ができ

ず，単純な問合せにのみ適用できる効率の良い処理アルゴリズム[1),4),9)]が適用できない．そこで，このような問合せを簡略化できる，もっと強い変換を考えよう．

問合せの依存グラフ中で，各閉路に関して，その閉路中の述語のうち少なくとも1つの述語記号を非評価述語記号集合の元としよう．このように非評価述語記号集合を定めた場合，p_eval が停止することは容易に確かめられる．各閉路から少なくとも1つ非評価述語を選ぶ時，最も容易な方法は，次に導入する目標依存再帰述語である．

問合せを依存グラフで表した時，1つの閉路と，目標からこの閉路に至る1つの径路に関して，閉路を構成する述語のうち，最も目標からの距離が短いものを目標依存再帰述語と呼ぶ．目標から1つの閉路への径路が複数存在する時は，各々の径路について目標依存再帰述語が存在すると考える．

GR を目標依存再帰述語の述語記号の集合として，次のホーン節変換を導入できる．

[タイプ2ホーン節変換 (hct 2)]

IDB′ :=hct 2(g, IDB)
=p_eval(g, IDB, $EUCUGR$).

目標依存再帰述語の定義から，hct 2 は必ず停止する．目標依存再帰述語を求める方法は後述する．先に提案した hct[9] は，hct 2 の一種である．

明らかに {目標依存再帰述語} ⊆ {再帰述語} なので，hct 2 は hct 1 よりもさらに問合せを簡略化できる，より強い変換である．例1の Q1 に hct 2 を適用すると，Q2 を得る．

hct 1，hct 2 共に，変換アルゴリズムは必ず停止し，しかも結果が一意に定まる．しかし，hct 2 は，部分評価ではこれ以上述語を消去できないという意味での，最も強い変換ではない．hct 2 の結果残った中間述語は，大部分が自己再帰になるが，**図1**の問合せのように例外的に自己再帰でない述語が残る場合がある．

一般に，自己再帰述語は，部分評価の方法では消去できないが，非自己再帰述語は消去できる．したがって，結果に現れる中間述語がすべて自己再帰になるような変換を考えることができる．これをタイプ3ホーン節変換と呼ぶ．

[タイプ3ホーン節変換 (hct 3)]

hct 3(g, IDB);
g'←g なる節を IDB に追加する;
repeat
R := {IDB 中の再帰述語記号};

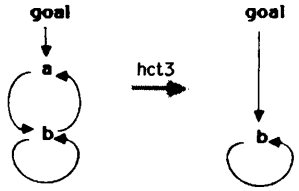

図1 タイプ3ホーン節変換の例
Fig. 1 An example of type-3 Horn clause transformation.

R := R − {R 中の非自己再帰述語記号 1つ};
IDB := p_eval(g', IDC, $RUEUC$)
until (R に非自己再帰述語記号が存在しない);
g'←gという節が存在すればこれを消去する;
各節の g' を g に変更する;
IDB′ := IDB
end.

hct 3 では g が自己再帰でない場合，g を対象とした導出を行う必要が生じることがあるため，いったん別の述語に目標を変更し，処理後元に戻す操作を行っている．

hct 3 の結果，IDB 中の中間述語はすべて自己再帰になるので，部分評価ではこれ以上の簡略化は行えない．この意味で，hct 3 は最も強い変換である．hct 2 では簡略化できないが，hct 3 では簡略化可能な例を図1に示す．規則節集合は省略し，依存グラフを示した．

4.2 ホーン節変換の性質

ホーン節変換では，一般に，問合せの形から変換後の形を推定することは困難である．しかし，非再帰中間述語が消去される，すなわち，非再帰述語からなる中間的成分が消去される，という性質と共に次の2つの性質がある．

[性質1]

問合せの成分が線形で，成分中の各述語に関して再帰述語が現れる定義節がたかだか1つならば，この成分は，hct 2 (したがって，hct 3) によって1つの述語からなる成分に縮退する．

この性質が成り立つのは，上の条件を満たす問合せの成分の閉路は1つなので，述語の消去順序に係わらず閉路の数は変化しないため，最終的に1つの自己再帰述語の閉路が残るからである．

線形問合せでも，1つの述語を本体とする複数の再帰的な節であるときは縮退できない場合がある．また，hct 3 に関して次の性質がある．

[性質2]

問合せの連結成分が，1つの述語に縮退可能なた

めの十分条件は，その依存グラフから1つの述語を削除した時，その成分に閉路がなくなるような述語が存在することである．

この性質が成立するのは，このような述語以外の述語を順次 hct 3 により削除することにより，最終的に1つの述語に縮退させることができるからである．なお，IDB の節中に定数も関数も存在しない場合，関係代数の縮退アルゴリズム[4]と同様にこの条件は必要十分条件になる．

hct 3 は最も強い変換であるが，結果が消去する述語の選び方に依存し，必ずしも一意には定まらないという問題がある．例えば例1の Q1 に hct 3 を適用した時，q から消去すれば Q2 を得るが，p から消去した場合は等価だが異なった IDB′ を得る．また，変換後の述語数も一意には定まらない．このため，hct 3 では消去する述語を選択する戦略が重要となる．通常は，多くの閉路に含まれる述語をできるだけ残すようにするのが良いと考えられる．これらの事項を考慮したアルゴリズムの改良は，今後の検討課題である．

4.3 否定の扱い

論理式をホーン節に限定した Prolog のような処理系では，“否定”の概念は，厳密には拡張を行わないと扱えない．Prolog ではこの問題を，閉世界仮説の下での失敗による否定 (negation as failure) として，“not” 述語を用いて扱っている[5),10)]．すなわち，p が有限回の導出で証明できない時，p の否定が証明されたと見なして扱っている．

このような，“否定”をホーン節に導入した時，関係データベースにおいて関係代数で記述できるものは，すべてホーン節で記述できる．したがって，“否定”を加えれば演繹データベースは，関係データベースの拡張になる．

ホーン節変換では，“否定”を “not” および “not” 中に現れる述語の述語記号をすべて T の元にすることにより扱うことができる．

5. ホーン節変換の実現方式

ホーン節変換の実現を，3章の p_eval のアルゴリズムで実現する方法もあるが，Prolog のような論理型言語処理系のアルゴリズムに部分評価の概念を導入して実現することができる．また，Prolog のような深さ優先探索ではなく，幅優先探索によっても実現できる[9)]．本章では，Prolog をもとにしたホーン節変換の実現方式を述べる．

5.1 部分評価処理系の実現

Prolog は，目標節の反駁を導出により行う処理系である．演繹データベースの問合せの基本式を目標節の本体に置き，DDB を Prolog のプログラムとして，Prolog で全解探索を行えば問合せの処理を行うことができる．この時，Prolog の処理は，目標の述語を根とした SLD 木を作成しているものとみなすことができる．Prolog の処理を変更し，部分評価の概念を導入した Prolog_P を以下のように構成する．

(1) 非評価述語記号の集合 T を与えた時，T の元については導出を行わない．

(2) 目標節の述語については，T の元であっても，SLD 木の根の部分だけは導出を行う．これは，目標の述語を頭部に持つ節を IDB から取り出すためである．

このような処理系で目標 g の処理を行った場合，各径路で導出が成功して停止した時，導出節中には T の元だけが残っている．これを R とした時，この結果は，$g \leftarrow R$ なる節を与えるものと解釈できる．すなわち，R が真ならば g が真になる．したがって，Prolog_P は，

IDB′ := Prolog_P(g, IDB, T)

のように，部分評価の結果を節集合として与えるものと考えることができる．

Prolog_P の出力中の節の頭部は g である．T の元には，$E \cup C$ の元ではないもの，すなわち，EDB には含まれず，IDB で他の述語により定義されているものが含まれる．これらの節を追加しないと IDB′ は g に関して IDB と等価にはならない．Prolog に基づいた部分評価処理系 p_eval′，

IDB′ := p_eval′(g, IDB, T)

を次のように定義する．

```
p_eval'(g, IDB, T);
  IDB' := { };
  for every predicate p
      s. t. sym(p)∈((T−(E∪C))∪{sym(g)}) do
    Temp := Prolog_P(p, IDB, T);
    IDB' := IDB'∪Temp
  end
end.
```

p_eval′ の出力 IDB′ は，IDB′ が有限であれば問合せ g に対して元の IDB と等価である．

5.2 非評価述語の決定

ホーン節変換では，部分評価処理系と共に非評価述

語の決定が必要である．EDB の述語や比較述語の決定は容易である．したがって，問題はホーン節変換の型に応じた非評価述語記号集合 T の決定である．

まず，hct 1 ではすべての再帰述語を非評価述語としているが，再帰述語を求めることは容易に実現できる．

hct 2 では，目標依存再帰述語を非評価述語とする．目標依存再帰述語は，目標を根とした SLD 木の構成時，各径路上に最初に 2 度現れるという性質を持つ．したがって，目標依存再帰述語は，前述の Prolog-P を以下のように変形して判定できる．

(1) 各径路中で導出により消去した述語を記憶する．

(2) 導出過程で，各径路において既に導出に用いた述語が再び導出の対象になった時は，その述語の導出を行わないで，目標依存再帰述語と判定する．

目標依存再帰述語は，このような処理系を $T=E \cup C$ として実行することにより検出できる．なお，非評価述語の判定においては，述語中の定数や関数の扱いに注意しないと必要な節の一部を見逃す恐れがある[9]．

hct 3 は，定義自身に T の決定アルゴリズムを含んでいる．しかし，これをそのまま実行することは，効率的ではないと考えられる．効率的な実現方法としては，まず hct 2 を行い，hct 2 の結果に非自己再帰述語が残っている時のみ hct 3 を適用することが考えられる．

5.3 実 現 例

ホーン節変換のうち，hct 2 の実現を行い，その実現性と有効性の検討を行った．以下では，Prolog で記述した演繹データベース実験システム PHI/K[2] におけるホーン節変換の実現例[12]について述べる．

hct の実現は，Prolog 処理系の変更によっても行えるが，メタプログラミング技法[3]を用いた方が効率上の問題はあるが容易である．メタプログラミングは，Prolog 自身により Prolog 処理系を容易に実現できることを利用した方法であり，この処理系をメタプログラム処理系と呼ぶ．メタプログラム処理系の呼出しは，プログラム中で，あらかじめ決められた述語，例えば demo(IDB, Goal) のような述語を呼び出すことによって行う．demo(IDB, Goal) は，Goal を指定された IDB のプログラムに対して実行することを意味している．メタプログラム，すなわち IDB は，idb(Clause) の形で記述されている．

メタプログラム処理系は，容易に変更できるので，前節で述べた方式に基づいて hct 2 の実現を行った．hct 2 の処理は，次の 2 段階で行っている．

(1) 5.2 節の方法による非評価述語の決定．

(2) 5.1 節の方法による IDB の変換．

この結果，以下の事項が確認された．

(1) ホーン節変換が簡単な方法で実現でき，問合せ簡略化の効果がある．

(2) ホーン節変換の過程で，q より到達可能な節のみを IDB から取り出すことができる．

(3) 目標中の定数を選択条件と考え定数の伝播を行う，Aho-Ullman[1] や Kifer-Lozinskii[8] の最適化方式の適用可能性の判定を，ホーン節変換の応用により行える．

(4) hct では，E の元に対応しない非再帰述語は消去する．このため，関係データベースの問合せ最適化で用いられるような共通表現の検出時には，いったん消去した述語を復元することが必要になる場合がある．

これらの事項のうち，(1)はホーン節変換の実現性の確認である．(2)については，Prolog の性質から明らかである．

ホーン節変換において，問合せの目標や節に定数が含まれても，3 章から 4 章までの議論には影響しない．しかし，ホーン節変換の実現時には，目標中の定数を部分評価時にどう扱うかによって処理効率や結果の節集合の大きさが変化する．

一方，Aho-Ullman や Kiffer-Lozinskii の最適化は，目標中に定数が存在する時，目標の仮想関係全体を計算してから定数に対応する部分を取り出すのではなく，その定数に対応する部分を直接計算する方法である．これらの方法が適用できるか否かは，節中に定数を伝播した時に本体に現れる述語の一般性によって定まる．このため，ホーン節変換時に述語の一般性の判定を行えば，ホーン節変換自身の効率化と共にこれらの最適化技法の適用可能性が判定できる．実験システムでは，Aho-Ullman の最適化の適用性判定をホーン節変換を用いて行った．

共通表現の問題は，現在のホーン節変換の問題点となっている．これについては，非再帰述語を非評価述語にすることにより，原理的には解決できる．しかし，どの述語を非評価述語とすべきかは，今後の検討課題である．

ホーン節変換の実現例および適用例としては，このほかに並列論理型言語 GHC による実現[7]，1 つの問合せではなく問合せ集合の最適化への適用[11]がある．

6. むすび

演繹データベースの問合せを簡略化するホーン節変換を，部分評価の概念に基づき提案し，その実現方法を述べた．ホーン節変換により，述語数の削減による効率化や，最適な実行方式選択の容易化ができる．簡略化の結果は，元の問合せと同じくホーン節で表現されるので，従来の方式と異なって，任意の問合せ処理方式の前処理として適用できる．また，関数を含んだデータベースの変換も可能になる．

今後は，タイプ3ホーン節変換への改良の検討を行うと共に，全体システムへの組み込みの最適な方法を検討する予定である．

謝辞　本論文の作成にあたり，御意見御助言をいただいた，(財)新世代コンピュータ技術開発機構および沖電気工業(株)の KBMS PHI 研究の関係者，同社の木下哲男氏，京都大学の西尾章次郎博士，(株)富士通研究所の横田治夫氏に感謝する．

参考文献

1) Aho, A. V. and Ullman, J. D.: Universality of Data Retrieval Languages, *Proc. 6 th ACM POPL*, pp. 110-120 (1979).
2) Bancilhon, F.: An Amateur's Introduction to Recursive Query Processing Strategies, *Proc. of ACM SIGMOD*, pp. 16-52 (1986).
3) Bowen, K. A. and Kowalski, R. A.: Amalgamating Language and Metalanguage in Logic Programming, Clark, K. L. et al. (eds.), *Logic Programming*, Academic Press, London (1982).
4) Ceri, S., Gottlob, G. and Lavazza, L.: Translation and Optimization of Logic Queries: The Algebraic Approach, *Proc. of 12 th VLDB*, pp. 395-402 (1986).
5) Clark, K.L.: Negation as Failure, Gallaire, H. and Minker, J. (eds.), *Logic and Data Bases*, pp. 293-322, Plenum Press, New York (1978).
6) Henschen, L. and Naqvi, S.: On Compiling Queries in Recursive First-Order Databases, *JACM*, Vol. 31, No. 1, pp. 47-85 (1984).
7) Itoh, H., Takewaki, T., Miyazaki, N. and Mitomo, Y.: Deductive Database System Written in Guarded Horn Clauses, ICOT Technical Report TR-214, ICOT (1986).
8) Kifer, M. and Lozinskii, E. L.: A Framework for an Efficient Implementation of Deductive Databases, *Proc. IPSJ 6 th Advanced Database Symposium*, pp. 109-116 (1987).
9) Miyazaki, N., Yokota, H. and Itoh, H.: Compiling Horn Clause Queries in Deductive Databases: A Horn Clause Transformation Approach, ICOT Technical Report TR-183, ICOT (1986).
10) Reiter, R.: On Closed World Data Bases, Gallaire, H. and Minker, J. (eds.), *Logic and Data Bases*, pp. 55-76, Plenum Press, New York (1978).
11) Sakama, C. and Itoh, H.: Partial Evaluation of Queries in Deductive Databases, *Workshop on Partial Evaluation and Mixed Computation* (1987).
12) 阿比留ほか：KBMS PHI：演繹データベース実験システム PHI/K², 第 34 回情報処理学会全国大会論文集, pp. 1491-1492 (1987).
13) 佐藤，玉木：Prolog に於けるプログラム変換，*Proc. of Logic Programming Conf. '83, ICOT*, 6-1 (1983).
14) 竹内ほか：Prolog プログラムの部分計算とメタプログラムの特殊化への応用，*Proc. of Logic Programming Conf. '85, ICOT*, pp. 155-165 (1985).
15) 野口，滝沢：知識工学基礎論，オーム社，東京 (1986).
16) 二村：プログラムの部分計算法，電子通信学会誌，Vol. 66, No. 2, pp. 157-165 (1983).

(昭和 62 年 6 月 1 日受付)
(昭和 62 年 12 月 9 日採録)

宮崎　収兄（正会員）

昭和 48 年京都大学理学部卒業．同年沖電気工業(株)入社．昭和 54 年イリノイ大学大学院計算機科学科修士課程修了．昭和 57 年より 3 年間(財)新世代コンピュータ技術開発機構に勤務．現在，沖電気工業(株)総合システム研究所に勤務．データベースおよび知識情報処理システムの研究開発に従事．電子情報通信学会，人工知能学会，ACM 各会員．

羽生田博美（正会員）

昭和 57 年東京理科大学理学部応用数学科卒業．同年沖電気工業(株)入社．以来，データベース・マシン，知識情報処理システムの研究開発に従事．人工知能学会，ACM，各会員．

伊藤　英則（正会員）

昭和 44 年福井大学工学部卒業．昭和 49 年名古屋大学大学院工学系研究科博士課程電気・電子専攻修了．工学博士．同年 NTT 電気通信研究所入社．現在，(財)新世代コンピュータ技術開発機構に勤務．研究室長．これまでに，オートマトンと数理言語理論の研究と DIPS 大型計算機オペレーティングシステムの研究開発に従事．現在，人工知能，知識情報処理システムの研究開発に従事．電子情報通信学会，人工知能学会，各会員．