データベース・システム 64－5
（１９８８．３．１５）

非正規形演繹データベースの問合せ評価法

木山 稔 [+]　　横田 一正 [++]
[+] ＮＴＴ 情報通信処理研究所　　[++] 新世代コンピュータ技術開発機構

非正規形モデルを始め構造データを扱うモデルと演繹データベースの２つを統合する試みが注目をされ始めている。ICOTで開発が進められている知識ベースシステムKappaは，非正規形演繹データベースを実現することを目指しており，CRLはその論理プログラミング言語である。
非正規形演繹データベースを実現するには，言語と共に評価アルゴリズムが重要である。本稿では，CRLの非再帰問合せを項を表現するグラフをベースに非正規関係代数表現に変換する方法，及び，再帰問合せをマジック集合法を拡張したアルゴリズムを用いて最適化する方法について述べる。

Query Evaluation Algorithms in Deductive Databases over Nested Relations

Minoru KIYAMA[+]　　Kazumasa YOKOTA[++]

[+] NTT Communications and Information Processing Laboratories
1-2356, Take, Yokosuka-shi, Kanagawa-ken 238-03, Japan
[++] Institute for New Generation Computer Technology
1-4-28, Mita 1-Chome, Minato-ku, Tokyo 108, Japan

One of main directions of database research is extensions of relational model, such as nested relational model, and another one is the use of deductive rules. It is very important to integrate these two approaches. Kappa, which is under development at ICOT, aimes at a deductive database system based on nested relations, using a logic programming language called CRL. In this paper, we propose evaluation algorithms of queries written in CRL. Non-recursive queries are reduced into the expressions of nested relational algebra using graph representations of terms. Recursive ones are optimized using magic sets algorithm.

1．はじめに

非正規形モデルを始め構造データを扱うモデルがここ数年来注目されている。一方，データベースを論理の一部とみなし，プログラムとデータを統一的に扱う演繹データベースもデータベース分野の主要な研究対象となっている。これらの，データベース研究の2大潮流を統合することは，極めて重要であり，強力な枠組みを提供することが可能になる[Yokota 87][Bancilhon 86a][Beeri 87a][Maier 86]。ICOTで開発している知識ベースシステムKappaは，非正規形演繹データベースをその一部として実現することをめざしており、CRLはそのプログラミング言語である[Yokota 87]。

非正規形演繹データベースを実現するには，言語と共に評価アルゴリズムが重要である。再帰を含まない問合せでは，非正規形の関係論理，関係代数とからんで，幾つかの問合せ変換法が研究されている[Abiteboul 87][木山 87][中野 87][Scholl 86]。また，ホーン節を対象とした再帰問合せに関しては，ボトムアップ評価法やその最適化手法であるマジック集合法等の様々なアルゴリズムが提案されている[Bancilhon 86b]。本稿では，CRLの非再帰問合せを非正規関係代数表現に変換する方法，及び，再帰問合せをマジック集合法を拡張したアルゴリズムを用いて最適化する方法について述べる。

2章で，CRLについて簡単に説明したあと，3章では再帰を含まない場合，4章では再帰を含む場合の問合せの評価アルゴリズムについて述べる。

2．非正規関係上の論理プログラミング言語ＣＲＬ

CRLについて，例をベースに簡単に説明する。なお，厳密な定式化は，[Yokota 87]を参照されたい。

2．1 基本要素

CRLの基本要素は，属性と値の組であり，これを項(nested attribute value pair term) と呼ぶ。項はホーン節の述語に対応するものであるが，述語と比較して次の利点を持つ。

(a)項を階層化することにより，タプルの入れ子構造を自然に表現可能
(b)引数の位置，個数固定からの解放。スキーマ変更時におけるプログラムのデータ独立
(c)全引数の指定は不要。部分情報(不完全な情報)を自然に表現可能

[例1] 項

(1)原始的な項（属性／値）

```
parent/a
```

(2)*連結子による項の結合。タプルを表現

```
parent/a*children/c
```

(3)集合値の表現

```
parent/a*children/{c1,c2}
```

(4)項の階層化(部分項を持つ項)による入れ子構造の表現

```
parent/(father/a*mother/b)*children/c
```

(5)項の階層化と集合構成子による完全な入れ子構造の表現

```
parent/a*children/{name/c1*hobbies/h1,name/c2*hobbies/h2}
```

なお，(3)のchildrenのように単純値を集合とする属性を繰返し単純属性，(5)のchildren のようにタプルの集合を値として持つ属性を繰返しグループ属性と呼ぶ。

2．2 モデル

モデルは，部分タグ木(partially tagged tree)をベースとして与える。部分タグ木は，属性のドメインを表現する木と属性から値への関数からなり，その最大の特徴は，ネスト，アンネスト演算によって意味をかえないことにある。つまり，利用者は，属性が集合値か単純値かということは全く意識することなく問合せを記述可能である。

[例2] 部分タグ木

(1)parent/a*children/{c1,c2}
部分タグ木=({⟨⟩,paren,children},{{(parent,a),(children,c1)},{(parent,a),(children,c2)}})

(2)parent/a*children/c1.
parent/a*children/c2.
部分タグ木=(1)と同じ

CRLプログラムのモデルは,部分タグ木の集合で与えられる。つまり,部分タグ木はホーン節のエルブラン領域の基礎原始式に相当する。上記のモデルの与え方により,CRLには最小モデルが存在するという,ホーン節の良質な性質が継承される。また,集合値同志の単一化は,両者の積集合が空でない場合に可能とする。

ホーン節の手続き的意味は,SLD導出で与えられるが,CRLにおいてもSLD導出がほぼ使える。異なる点は,前述のネスト,アンネスト演算によって意味をかえないという点を実現するために,集合値を含む問合せに対しては,特別な導出を持つ。例えば,

```
:-parent/X*children/{c1,c3}.
parent/a*children/{c1,c2}.
parent/a*children/{c2,c3}.
parent/b*children/{c3,c4}.
```

の解代入は,通常のSLD導出では,X=a,X=a,X=bであるが,parent/b*children/{c1,c3}のモデルは,プログラムのモデルに含まれない。そこで,CRLではレストゴールと呼ばれるものを生成して,X=bが解代入にならないようにチエックする。第3番目の事実とゴールから ,parent/b*children/{c1}というレストゴールを生成するが,このゴールからはもはや導出節をつくれないので,手続きは失敗し,X=bは解代入にはならない。

なお、CRLでは,繰返しグループ属性を単一化に解決すべき問題があるため,いまのところ未サポートであるが, ここでは, 「集合構成子{}に含まれる任意の項に変数が存在してはならない」という制限付で許している。但し, 厳密にモデル, 手続き的意味が与えられているわけではなく, あくまでもデータベースとからめた直観的解釈だけにとどめる。

3. 非再帰問合せの評価アルゴリズム

Kappaのデータベース層は,非正規形を内部構造として持つデータベース管理システムである。従って, CRLの事実をKappaデータベースとして格納することを考えたとき CRLのプログラムからKappaデータベースへの問合せを作成することが必要になる。つまり, CRLで与えられたtuple-at-timeの手続き的意味と等価なset-at-timeの手続き的意味をデータベース操作で実現しなければならない。

現在Kappaでは, 原始コマンドレベルまでしか実現されていないが, 上記の実現には関係代数レベルが適当と考え,まず,非正規関係代数を定義することから始める。なお, 以後例題として以下のスキーマ[Schek 86]を用いる。

DEPT(D,DN,AE(AN,AJD),TE(TN,TJD,C(CN,Y)))　　PERSON(PN,PA)

3.1 非正規関係代数

一般に,関係代数の仕様は関係の格納方式,処理方式に深く依存したものになると考える。 [Jaescheke 82]では,選択演算子に,集合の包含関係,所属関係を導入しているが,繰返し単純属性だけを対象にしている。繰返しグループ属性を扱えるものに[中野 87] [Schek 86]等がある。[中野 87]では,関係の格納に正規形を用いているので,非正規形関係をアンネストしてから繰返しグループ値に対する操作を行う。一方,Kappaデータベースでは非正規形をそのままの形で内部構造に持つことから, ネスト関係代数[Schek 86]と親和性が良いと考える。ネスト関係代数は,繰返しグループ値に対する操作を代数表現の入れ子によって表す。なお、Kappaでは、より簡単で、Kappaに適した非正規関係代数を考えており、その変換法も検討中である。

(1)ネスト関係代数[Schek 86]

非正規形は関係の入れ子構造をもつので, 内側の関係に対して演算を施こす必要があり, 関係代数表現の入れ子は自然な拡張である。

正規形関係代数では,選択演算の条件式内や射影演算の射影リストには属性名しか現われなかった。ネスト関係代数ではこの部分に関係代数表現を記述できるよう拡張されており, 繰返しグループ属性に対する演算を記述する。 以下では, 入れ子構造が導入された選択演算と射影演算について簡単に説明する。

(2)選択演算及び射影演算

選択演算では、選択条件式中に選択演算および射影演算を記述することができる。関係代数表現の評価は内側から行なわれる。つまり,関係代数表現の内側の演算は,各タプルの繰返し属性値 (関係)に対して施され,その結果は外側の選択演算の条件式に定数として渡され,全体の代数表現が評価される。射影演算では、射影リスト中に選択, 射影, さらに2項演算子も記述することができる。評価はやはり ,内側から外側へ進む。

[例3]入れ子構造を持つ選択演算： 課名(DN)がreseachで運営部門(AE)の職種(AJD)にtranslationを含むもの。

σ[DN=research]σ[σ[AJD=translation]AE$\neq\phi$]DEPT

[例4]入れ子構造を持つ射影演算： 課番号(D)とその課に属する技術者のID(TN)と職種(TJD)を得よ。

Π[D,Π[TN,TJD]TE]DEPT

3.2 変換アルゴリズム

3.1.1 特徴

変換アルゴリズムはグラフを用いて定式化する。ホーン節を対象とする演繹データベースでは，まず，プログラムからルール／ゴール木を作成してから，次に，ボトムアップに関係代数表現を構築していく[Ullman 85]。一方，CRLは項レベルに木構造を持つことから，上記の方法が項の評価に適用可能と考えた。また，この方法はプログラム節の評価，さらにはプログラムの評価に自然に拡張可能である。

アルゴリズムは，まず，属性及び値をノードとするグラフを作成する。次に，リーフからルートに向い，ボトムアップに関係代数表現を構築していく。関係代数表現は，ノードを1段上がる毎に徐々に作成される。つまり，ノードの種別とそれへの（子ノードからの）入力によって出力が決まり，さらにそれが親ノードへの入力となり，上位に伝わる。そして，ルートノードの出力が最終的な変換結果となる。アルゴリズムの記述は，ノードの種別とそのノードへの入力によって，出力がどのように決まるかを規定することによって行う。

以下では，まず，一つの項を関係代数表現に変換するアルゴリズムを述べ，次にそれをプログラムまで拡張する。

3.1.2 項の変換

(1)項のグラフ表現

項は次のような，ノードとリンクからなるグラフである。

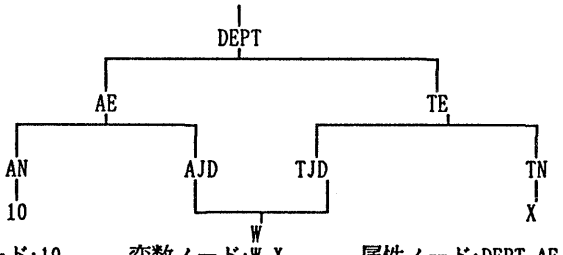
ノード：項内に現われる属性名及び値，　　リンク：項とその部分項間(項間の階層関係を表す)

但し，項内に同一の変数が複数ある場合は，それらは一つのノードとする。このため，グラフは木にならない。また，以下のようにノードを分類する。

定数ノード，変数ノード，属性ノード：それぞれ，定数，変数，属性名からなるノード
ルートノード：グラフの最上位
リーフノード：グラフの最下位(定数ノード，もしくは変数ノード)
タグノード：複数の親を持つ変数ノード

[例5]項のグラフ表現

DEPT/(AE/(AN/10*AJD/W)*TE/(TN/X*TJD/W))



定数ノード:10　　変数ノード:W,X　　属性ノード:DEPT,AE,AN,AJD,TE,TJD,TN
ルートノード:DEPT　リーフノード:10,W,X　タグノード:W

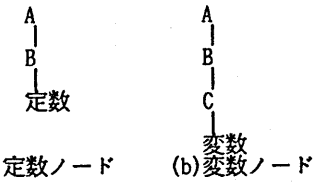
(2)タグノードを含まない場合の変換

タグノードを含まないとき，グラフは木になる。従って，複数項間の関係はいっさいないので，選択，射影演算だけから関係代数表現を得ることが可能である。

ボトムアップに関係代数表現を作成するので，まず，リーフノードに着目する。明らかに，定数ノードは選択演算に，変数ノードは射影演算に関連する。問題は，木の段数が深くなったときに，どのように関係代数表現を作成していくかである。まず，定数ノードを考える(図1(a))。定数ノードは定数そのものを出力するものとし，それは親ノードBへの入力となる。属性ノードBは，選択条件式にあたる"B=定数"を出力する。さらに，ノードAは，これを受け取り，"σ[B=定数]A$\neq\phi$"を出力する。これも選択条件式であるが，繰返しグループ属性に対応するために，一方の値が選択演算を表す表現になっている。この様にして，木を1段上るにつれ，選択演算の入れ子が深くなる。

一方，変数ノードの場合(図1(b))，まず，自分自身はϕを出力とする。ϕを受け取った親ノードは，定数を入力とした時と異なり，属性名"C"そのものを出力とする。これは，射影リストに相当するものである。さらに，ノードBは，これを受け取りΠ[C]Bを，ノードAはΠ[Π[C]B]Aを出力する。選択演算の時と同様に，木を上るにつれ，射影演算の入れ子が増す。

選択，射影演算が複数存在する場合，また混在する場合の処理も含め，タグノードがないグラフの関係代数表現を得るための，各ノードの動作を表1に示す。表1の動作は関係代数表現を得る上で、必要にして十分である。



(a)定数ノード　(b)変数ノード

図1　木の深さと関係代数表現

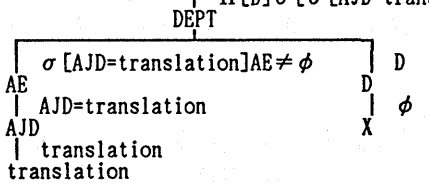
表1　ノードの動作（タグノードなし）　sf(selection formula)　pl(projection lists)

	ノード種別	入力	出力
1	定数ノード	なし	定数
2	変数ノード	なし	ϕ
3	属性ノード	定数	属性名=定数(=⟩sf)
4	属性ノード	ϕ	属性名(=⟩pl)
5	属性ノード	sf	σ[sf]属性名$\neq\phi$(=⟩sf)
6	属性ノード	pl	Π[pl]属性名(=⟩pl)
7	属性ノード	sf1, sf2	σ[sf1$\wedge$sf2]属性名$\neq\phi$(=⟩sf)
8	属性ノード	pl1, pl2	Π[pl1, pl2]属性名(=⟩pl)
9	属性ノード	sf, pl	Π[pl]σ[sf]属性名(=⟩pl)

[例6]タグノードがない場合の項の変換

DEPT/(AE/(AJD/translation)*D/X).

Π[D]σ[σ[AJD=translation]AE$\neq\phi$]DEPT



(3)タグノードを含む場合の変換

タグノードは複数の項間に関連があることを示しているので，結合演算に対応することが容易に推測できる。ボトムアップに関係代数表現を作成するため，一旦タグノードから分岐して別々に作られた代数表現は，再び出会うことによってはじめて完全なものと成る。

まず，別々に作られた表現を統合するための目印として，各タグノードに対してタグを設ける（各変数に対し，その小文字をタグとする)。タグノードからは，対応するタグを出力する。タグは，定数と全く同じ扱いをされ，表1に従い関係代数表現が作られる。そして，同一のタグを持つ入力がみつかったときに，関係代数表現を統合するために，マージノードを動的に生成する。マージノードは，同一のタグを持つノードとその親ノードの間に挿入される。なお，同一タグをもたないリンクはマージノードを経由しない。

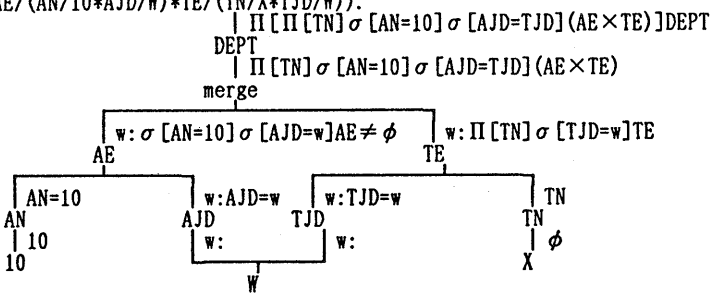
マージノードの入力と出力の関係は表2の通りである。基本的に，2つの表現から結合演算を作りだす操作を行う。タグを定数としてもつ2つの選択条件式を結合して，タグを除いた一つの条件式にし，それと共に，選択演算の対象となっていた2つの関係の直積を新たな演算対象とする。さらに，選択演算がネストしていたり，射影演算と共に現われたりするため幾つかのバリエーションが生じる。なお、表2は、演算子が数個の簡単な場合であり、さらに複雑な一般形が存在するがここでは省略する。

表2　マージノードの動作

	入力	出力
1	属性名1=タグ 属性名2=タグ	属性名1=属性名2
2	σ[属性名1=タグ]属性名2$\neq\phi$ σ[属性名3=タグ]属性名4$\neq\phi$	σ[属性名1=属性名3]（属性名2×属性名4)$\neq\phi$
3	属性名1=タグ σ[属性名2=タグ]属性名3$\neq\phi$	σ[属性名1=属性名2]（属性名1×属性名3)$\neq\phi$
4	属性名1=タグ Π[pl]σ[属性名2=タグ]属性名3	Π[pl]σ[属性名1=属性名2]（属性名1×属性名3)
5	σ[属性名3=タグ]属性名4$\neq\phi$ Π[pl]σ[属性名1=タグ]属性名2	Π[pl]σ[属性名1=属性名3]（属性名2×属性名4)
6	Π[pl1]σ[属性名1=タグ]属性名2 Π[pl2]σ[属性名3=タグ]属性名4	Π[pl1, pl2]σ[属性名1=属性名3]（属性名2×属性名4)

[例7]タグノードを含む場合の変換

DEPT/(AE/(AN/10*AJD/W)*TE/(TN/X*TJD/W)).



3.1.3 複数項の変換

問合せには,項がカンマで区切られて,複数現われることがある。このような場合には,以下のようにして単一の項に置き換えてから前述のアルゴリズムを適用する。まず,項の連結子カンマを*に置き換え,次に,それら全体がDBという属性名を持つ項の部分項となるようにする。例えば,DEPT/(AN/X*AE/AJD/W),PERSON/(PN/W*PA/Y).はDB/(DEPT/(AN/X*AE/AJD/W)*PERSON/(PN/W*PA/Y)).のように変換する。

ノードDBは特殊なノードであり,マージノードとほぼ同じ働きをする。マージノードでは結合を最外側から行ったが,DBノードでは内側から行う。これは,ノードDBの出力がさらに他のノードの入力になることがないために可能である。つまり,出力された関係代数表現が選択条件式や射影リスト内で使われることがないので,両入力の最外側の属性名が結合後も最外側である必要はない。上記の例はDBノードでつぎのように変換される。

Π[AN]σ[σ[AJD=w]AE≠φ]DEPT
Π[PA]σ[PN=w]PERSON
====⟩Π[AN,Π[PA]σ[AJD=AN](AE×PERSON)] DEPT

また,:-a/{c1,c2}*b/Xのように集合値に関する選択は,これを単純にΠ[b](σ[a={c1,c2}]ab)と変換してもうまくゆかない。なぜなら,求めるXはc1,c2のどちらの組ともモデル中になければならないからである。CRLのSLD導出ではこの点を回避するためにレストゴールとマーク付代入を導入していた。データベース検索では問合せを分解して(つまり,静的にあらかじめレストゴールを作成する):-a/c1*b/X,a/c2*b/X.のように変換して実行することが提案されている[Yokota 87]が,このような場合も本処理で扱うことができる。

3.1.4 プログラムの変換

プログラムへの拡張は以下のようなグラフを作ることによって行う。

まず,プログラムと問合せから束縛を伝播しながらルール/ゴール木を作る。グラフのリーフにあるルールノードは項を表すグラフに置き換え,リーフ以外のルールノードは,子ノードからの入力の和集合を出力するノードとする。ゴールノードはDBノードに置き換える。このようにして作成したグラフを前記のアルゴリズムを使い、ボトムアップに評価していくことにより、プログラムの変換が可能である。

4. 再帰問合せの評価アルゴリズム

ホーン節を対象とした演繹データベースの再帰問合せ評価法には,様々な方法が提案されている[Bacilhon86a]。CRLの再帰問合せの評価には,ホーン節で提案されたアルゴリズムを拡張して使用するのが,最も着実なアプローチと考えた。まず,Kappaデータベースとの親和性から,ボトムアップ評価法が適しているのは明らかである。単純(naive)なボトムアップ評価法では,次の問題点を持つことが指摘されている。

(a)解の生成に関与しない事実を中間結果として生成する
(b)同じ中間結果を繰返し生成する

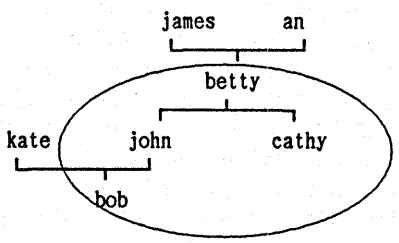
(a)について,多くの最適化手法が提案されているが,ここではマジック集合法[Bancilhon86c][Berri 87b]をベースとしてCRLの最適化手法を考える。しかし,マジック集合法がCRLに最適なアルゴリズムと考えたわけではなく,既存のアルゴリズムをCRLに適用したときに発生する問題点を明確にするためのひとつの材料として用いたにすぎない。今後,マジック集合法以外の最適化手法に関しても,CRLへの拡張と性能,適用範囲を明確化していく必要がある。

ホーン節を対象としたマジック集合法について説明する。問合せに定数が含まれている(束縛されている)とき,ボトムアップ評価法を適用して,最後に選択演算を施す方法では,途中で結果の生成に関係しない事実を取り出すことになり,効率が悪い。そこで,マジック集合法は問合せの束縛情報を伝播し,あらかじめ結果に関

与する事実を絞り込む。絞り込まれた事実をマジック集合と呼び，それを求めるために用意された述語をマジック述語と呼ぶ。以下に，[Bancilhon 86c]で提案されたオリジナルなアルゴリズムを使った変換例を示す。

[例8] ホーン節によるプログラム[Yokota87]

```
:-ancestor(betty, Y).
ancestor(X, Y):-parent(X, Y).
ancestor(X, Y):-parent(X, Z), ancestor(Z, Y).
parent(james, betty).
parent(an, betty).
parent(betty, john).
parent(betty, cathy).
parent(john, bob).
parent(kate, bob).
```



マジック集合

[変換後]

```
:-ancestor^bf(betty, Y).
ancestor^bf(X, Y):-parent(X, Y).
ancestor^bf(X, Y):-magic.ancestor^bf(X), parent(X, Z), ancestor^bf(Z, Y).
magic.ancestor^bf(Z):-parent(X, Z), magic.ancestor^bf(X).
magic.ancestor^bf(betty).
```

図2　ホーン節によるプログラムのマッジク集合

4．1 繰返し単純属性への適用

繰返し単純属性に対するマジック集合法を考察する。前述の例題をCRLで記述すると，以下のようになる。

[例9]CRLによるプログラム(ホーン節によるプログラム例と同じ意味を持つ)

```
:-ad/(a/betty*d/Y).
ad/(a/X*d/Y):-pc/(p/X*c/Y).
ad/(a/X*d/Y):-pc/(p/X*c/Z), ad/(a/Z*b/Y).
pc/(p/{james, an}*c/betty).
pc/(p/betty*c/{john, cathy}).
pc/(p/{john, kate}*c/bob).
```

問合せの束縛が:-ad/(a/betty*d/Y). のように単純値の場合はホーン節の場合と同じアルゴリズムを適用することが可能である。但し、束縛情報は，述語にベクター表示されるのではなく，各項に与えられる。しかし，束縛が:-ad/(a/{betty, kate}*d/Y)のように集合値の場合には，最適化を行っているとはいえない。つまり，マジック述語の初期値が magic.ad/(a/{betty, kate}). になるため，これをそのまま評価するとbettyとkateのどちらかに関与する事実がすべて得られてしまう（図3(a)）。ところが，求める解は，bettyとkateの共通の先祖であるため，magic.ad/(a/{betty}). やmagic.ad/(a/{kate}). を初期値とした方が関与する事実をより小さく絞り込むことが可能である。そこで，図3(b)のようにマジック述語を作成し，bettyと kateのどちらにも関与する事実をマジック集合とすることが考えられる。

```
(a):-ad/(a^b/{betty, kate}*d^f/Y).
 ad/(a^b/X*d^f/Y):-magic.ad/(a^b/X), pc/(p/X*c/Y).
 ad/(a^b/X*d^f/Y):-magic.ad/(a^b/X), pc/(p/X*c/Z), ad/(a^b/Z*d^f/Y).
 magic.ad/(a^b/Z):-pc/(p/X*c/Z), magic.ad/(a^b/X).
 magic.ad/(a^b/{betty, kate}).
(b)magic.ad/(a^b/X):-magic.ad1/(a^b/X), magic.ad2(a^b/X).
 magic.ad1/(a^b/Z):-pc/(p/X*c/Z), magic.ad1/(a^b/X).
 magic.ad2/(a^b/Z):-pc/(p/X*c/Z), magic.ad2/(a^b/X).
 magic.ad1/(a^b/betty).
 magic.ad2/(a^b/kate).
```

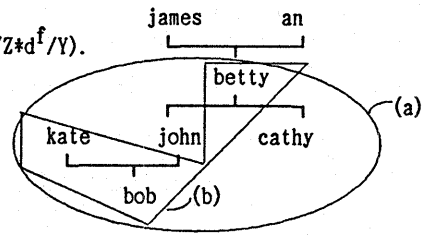


図3　繰り返し単純属性に適用した場合のマジック集合

このようなことが起こる理由は，CRLのレストゴールに関係する。つまり，マジック集合を求める動作は，一見，ボトムアップ的にみえるが，実はトップダウンの動きをシュミレートしているにすぎない。CRLのSLD導出では，問合せに集合値を含む場合レストゴールの生成という特殊な動きをする。ところが，ホーン節と同じアルゴリズムでマジック述語を生成すると，このレストゴールに相当する部分が欠落してしまう。従って，なんらかの方

法でその部分を補ってやる必要が生じ,それが上述のような変換となって現われる。直観的にいえば、図3(b)は:-ad/(a/{betty,kate}*d/Y).を:-ad/(a/betty*d/Y),ad/(a/kate*d/Y).のように分解し、それぞれの項に対してホーン節のアルゴリズムを適用し、マジック集合を求めている。

4．2 繰り返しグループ属性への適用

繰り返しグループ属性で特徴的なことは、項の階層構造がプログラム中に出現することである。このことにより、マジック集合法を適用しようとした場合に、階層構造を持つ項の束縛情報をどのように管理するかが重要になる。しかし、これは、部分項の束縛情報を正確に伝達するよう、項の構造もルール間で伝播させる（即ち、問合せの定数部分を変数に置き変えたものとの単一化になる）ことにより解決できると思われる。むしろ問題なのはボトムアップ評価法そのものにある。つまり、再帰述語において、ボディ部の述語とそれと単一化可能なヘッド部の述語の項の構造が異なる。項の構造が異なるとは,a/Xとa/m/Yのように単一化可能であるが,形式が異なることを指す。このような状態は,再帰がない場合は,あらかじめ単一化を行うことにより回避することができた。しかし,再帰述語では,各呼出しで引数の構造が異なるので,むやみに単一化はできない。構造の違いは関係代数表現を作成することを困難にする。即ち、再帰述語の項の構造の違いにより関係代数表現がことなるため、実行中に動的に関係代数表現を変化させることが必要である。処理方式としては、まず、基本となる関係代数表現を求めておき、構造の変化を検知したときに、それを代数表現に変換し、最初の関係代数表現に代入する。なお、詳細については現在検討中である。

5.おわりに

非正規形モデルと演繹データベースを統合することにより,極めて強力な枠組みを得ることが可能である。本稿では,非正規形演繹データベース実現するための,問合せ評価アルゴリズムについて検討した。再帰を含まないCRLの問合せでは,項を表現するグラフをベースに，非正規関係代数表現に変換するアルゴリズムを定式化した。また,再帰問合せについては，マジック集合法を拡張することにより,比較的容易に最適化が可能であることを示した。

[謝辞]
本研究の機会を与えてくださったICOT第四研究室の内田俊一室長,NTT知識処理研究部の村上国男部長,吉田清主幹研究員,中野良平主幹研究員に深謝致します。また,ICOT第四研究室，及び，Kappaグループの諸氏には大変有益な助言を頂きました。

[参考文献]
[Abiteboul 87]Abiteboul,S. and Beeri,C.,"On the Power of Languages for the Manipulation of Complex Objects," draft,1987.
[Bancilhon 86a]Bancilhon,F.,"A Calculus for Complex Objects," PODS,1986.
[Bancilhon 86b]Bancilhon,F. and Ramakrishnan,R.,"An Amateur's Introduction to Recursive Query Processing Strategies," SIGMOD,1986.
[Bancilhon 86c]Bancilhon,F.,"Magic Sets and Other Strange Ways to Implement Logic Programs," PODS,1986.
[Bancilhon 87]Bancilhon,F. and et al.,"FAD, a Powerful and Simple Database Language," VLDB, 1987.
[Beeri 87a]Beeri,C.,"Sets and Negation in Logic Database Language(LDL1)," PODS,1987.
[Beeri 87b]Beeri,C. and Ramakrishnan,R.,"On the Power of Magic," PODS,1987.
[Jaeschke 82]Jaeschke,G. and Schek,H.-J.,"Remarks on the Algebra of Non First Normal Form Relations," PODS,1982.
[木山 87]木山,中野,"非正規形ﾋﾞｭｰに基づく検索の正規形操作への変換法," ﾃﾞｰﾀﾍﾞｰｽｼｽﾃﾑ研究会,87-DB-58,1987.
[Maier 86]Maier,D.,"A Logic for Objects,"Proc. of the Workshop on Foundation of Deductive Database and Logic Programming,1986.
[中野 87]中野,"非正規関係論理/非正規関係代数変換法," データベースシステム研究会,87-DB-61,1987.
[Schek 86]Schek,H.-J. and Scholl,M.,"The Relational Model with Relation-Valued Attributes," Information Systems,1986.
[Scholl 86]Scholl,M.,"Theoretical Foundation of Algebraic Otimization Utilizing Unnormalized Relations," ICDT,1986.
[Ullman 85]Ullman,J.D,"Implementation of Logic Query Languages for Databases," ACM transaction on Database Systems,Vol.10,No3,1985.
[Yokota 87]Yokota,K.,"Deducive Approach for Nested Relations,"ICOT TR,to appear.