データベースシステム 106－14
（１９９６．１．２５）

内包表記に基づくオブジェクト指向データベースの問合せ処理

石川 佳治　　高橋 賢一　　植村 俊亮

奈良先端科学技術大学院大学 情報科学研究科

本稿では，内包表記 (comprehension) と呼ばれる問合せの記法に基づく，オブジェクト指向データベースの問合せ処理について論ずる．内包表記は汎用性と高い表現能力を持ちながら最適化の可能性も有している．本稿では，モノイド内包表記 (monoid comprehension) と呼ばれる内包表記を採用し，オブジェクトデータベース標準 ODMG-93 の問合せ言語 OQL を想定した問合せの最適化を試みる．特に，集合値検索 (set-based retrieval) に帰着する問合せを効率的に支援することを考え，集合値索引 (set-based index) を効率的に活用するための変換規則について議論する．

Query Processing Strategies for Object-Oriented Databases Based on Comprehension Syntax

Yoshiharu Ishikawa　　Ken'ichi Takahashi　　Shunsuke Uemura

Graduate Institute of Information Science, Nara Institute of Science and Technology

In this paper, we discuss *comprehension*-based query processing strategies for object-oriented database systems. While comprehensions are general and powerful query notations, they also have many optimization chances. In this paper, we employ *monoid comprehensions* as the intermediate query language for OQL of ODMG-93 Standard, and propose several optimizing techniques. Especially, we discuss query optimization strategies for queries resulting *set-based retrieval*. For such queries, we propose some transformation rules to efficiently use *set-based indexes*.

1 はじめに

オブジェクト指向データベースにおける問合せ処理は, 集合, バッグ (bag), リストなどの複数のコレクションが問合せ中に混在することや, 入れ子問合せ (nested query) を扱う必要があることなどから複雑となる. このような問題に対処するため, これまで関数型言語の分野で研究されてきた内包表記 (comprehension) [Wad87] の手法を問合せ最適化における中間的なターゲット言語として用いるアプローチが提案されている [BLST94, Feg94, FM95, PS94, Tri89]. 内包表記は高い表現力を持ちながら, 取り扱いの容易さを併せ持っている.

内包表記で表現された問合せの最適化についてはこれまでいくつかの研究がなされているが [PS94, Tri89], 論理的な最適化が中心であり, 索引の有無や実際の問合せ処理アルゴリズムなどの物理レベルの最適化まで考慮に入れた研究は少ない.

本研究では, 対象としてオブジェクトデータベース標準 ODMG-93 [CAD+95] の問合せ言語 OQL を選び, 内包表記に基づく問合せの最適化について議論する. 本稿では特に, 集合値索引 [IKO93, 石川 95b, IK95] を用いた問合せの効率化について述べる.

2 節では, 内包表記の概念と, そのデータベースにおける問合せ処理への応用について述べ, 次いで, 本研究で最適化の対象とするモノイド内包表記について説明を行なう. 3 節では集合値索引の概念とシグネチャファイルに基づく集合値索引の手法について述べる. 4 節ではまず, 問合せ処理の概要について述べ, 次いで内包表記に基づく問合せ処理における一般的な変換規則について述べる. 5 節では内包表記に基づく問合せから集合値検索条件を抽出し集合値索引を用いた問合せに変換するための規則と, その処理例を示す. 6 節ではまとめを行ない今後の課題について述べる.

2 内包表記とその問合せ処理への応用

2.1 内包表記とは

近年, データベースプログラミング言語の研究分野において, **内包表記** (complehension) に基づく問合せ処理や, その表現能力に関する議論が盛んになっている [BLST94, PS94, Tri89]. 内包表記はもともと関数型言語の分野で提案されたものであり, 単に内包表記といった場合にはリスト内包表記を指すことが多い. **リスト内包表記** (list comprehension) とは, 集合の内包表記 (例: $\{square\, x \mid x \in A \wedge odd\, x\}$) に対応する概念であり, リスト上の繰り返しを含んだ操作を洗練した形で記述するために提案されたものである [Wad87].

リスト内包表記は

$$[E \mid Q_1, Q_2, \ldots, Q_n]$$

という形式で表される. Eは式であり, $Q_1, Q_2, \ldots, Q_n$ は Eの変数に束縛と制約を与える 0 個以上の**限定子** (qualifier) の並びである. 限定子には $v \leftarrow L$ という形式の**生成式** (generator) と, 真偽値を返す条件式あるいは述語であるフィルタ式 (filter) の二種類がある. 生成式 $v \leftarrow L$ はリスト L からその要素を一つずつ取り出し変数 v に束縛し, その変数束縛はそれ以降の限定子の並びにおいて有効となる. フィルタ式は変数に対して制約を与え, 評価結果が真となった変数束縛のみがそれ以降の限定子の並びに送られる. このように, リスト内包表記では限定子の並びが左から順に評価されるという特徴があり, すべてのフィルタ式を満たした変数束縛のそれぞれについて式 Eが評価され, 結果がリストとして返される. リスト内包表記の例を以下に示す.

$$[square\, x \mid x \leftarrow A;\ odd\, x;\ x < 100]$$

内包表記では生成式の組合せなどにより任意の入れ子操作を表現できるため, 入れ子構造を持つデータに対する処理を自然に記述できる. また, リスト内包表記はλ計算 (lambda calculus) に直接的に変換することができるため [Wad87], 関数型言語における糖衣構文 (syntax sugar) としてとらえることもできる.

リスト内包表記が高い表現能力を有すことから, これをデータベースの問合せの表現のために用いることが試みられてきた [BLST94, PS94, Tri89]. リスト内包表記は最適化の可能性も有しており, リレーショナルデータベースにおける代表的な問合せ最適化のストラテジィに対応して内包表記の変換規則が存在することが示されている [Tri89].

しかし, オブジェクト指向データベースなどの複合データを扱うデータベースにおいては, 入れ子構造が扱えるだけでなく, 集合, バッグ (マルチセット), リストなどのさまざまなコレクションデータを扱える必要がある. そのため, 複数のコレクションデータを統一して扱えるように内包表記の言語を拡張した提案がなされている [Feg94, FM95]. このようなアプローチには, 複数のコレクション型に対し共通の記法を与えることにより, 最適化ための変換規則が複雑化しないという利点も存在する.

そのほかに, 内包表記に継承, オブジェクト識別性 (object identity), 経路式 (path expression) などのオブジェクト指向の概念を導入し, 直接的にオブジェクト指向データベースの実装を試みた研究も存在する [CT94].

2.2 モノイド内包表記

本稿の議論の対象とするのは, Fegaras らにより提案されたモノイド内包表記 (monoid comprehension) である [Feg94, FM95]. 彼らはこれをモノイドカルキュラス (monoid calculus) とも呼んでいる. [Feg94, FM95] では, モノイド内包表記と等価な代数系であるモノイド代数 (monoid algebra) が定義されており, モノイド内包表記からモノイド代数への変換規則, モノイド代数表現の正規化, モノイド代数の最適化などが論じられている.

モノイド内包表記/モノイド代数の特徴の一つは, 複数のコレクションデータが混在する問合せにおいて異なるコレクション間の変換を自動的に行なうことにある. OQL [CAD+95] などのオブジェクト指向データベースの問合せ言語においては複数のコレクションデータが混在して出現するため, コレクションの自動的な変換は大きな利点となる. これは, すべてのコレクション型を数学的なモノイド (monoid, 単位半群) としてとらえたことによる. たとえば, 集合 (set) については, 空集合 ($\emptyset$) を単位元とし, 和集合演算 ($\cup$) を結合律を満たす演算と考えることで, (set, $\emptyset$, $\cup$) がモノイドとなる. 異なるコレクションの間の変換は, モノイド間の準同型写像 (homomorphism) として形式化されている.

モノイド内包表記の基本形は以下のようになる.

$$\mathcal{M}\{E \mid Q_1, Q_2, \ldots, Q_n\}$$

$\mathcal{M}$ は, このモノイド内包表記表現を評価した結果の型 (モノイド型 (monoid type) と呼ばれる) を表す. モノイド型としては, たとえば個々のコレクション型に対応した, set, bag, list などがある. Eが式であり, $Q_1, Q_2, \ldots, Q_n$ が0個以上の限定子 (生成式またはフィルタ式) の並びであることはリスト内包表記と同様であるが, 生成式 $v \leftarrow u$ において u はリストである必要はなく, あるモノイド型のコレクションであればよい. 以下にモノイド内包表記の例を示す.

$$\mathrm{set}\{x.b, y.c \mid x \leftarrow S, y \leftarrow B, x.a = y.a\}$$

これは, 集合 Sとバッグ Bの要素を属性 a で結合 (join) し, 射影した結果を集合として返す問合せを示している.

モノイド型には, 上で述べたような通常のコレクション型に対応するものだけでなく, 存在限量 ($\exists$, existential quantification) や全称限量 ($\forall$, universal quantification) に対応する some, all などがある. たとえば some の場合, 真理値 $false$ を単位元とし, 結合律を満たす演算として論理和$\vee$を選ぶことで (some, $false$, $\vee$) がモノイドとなる. some に関するモノイド内包表記の例を示す.

$$\mathrm{some}\{a = x \mid x \leftarrow S\}$$

これは条件式 $a \in S$と等価である.

3 集合値索引について

入れ子状のコレクションデータを扱う OQL などの問合せ言語の処理では, 集合の包含関係などの集合値検索条件に基づく問合せ処理が必要となる. そのような問合せの効率化のため, 本研究では集合値索引 (set-based index) [IKO93, 石川 95b, IK95] を効率的に活用する最適化の手法について論じる.

集合値索引の簡単な活用例を示すため, 以下のようなモノイド内包表記の問合せを考える.

$$set\{d.name \mid d \leftarrow Dept,\ d.orders \supseteq \{pen, ink\}\}$$

この問合せは, ペンとインクを少なくとも注文している部署の名前の集合を求める問合せを表している. $\{pen, ink\}$ のように, 問合せで指定された集合を問合せ集合 (query set, Q) と呼び, 比較対象のデータベース中の集合をターゲット集合 (target set, T) と呼ぶことにする. 上の問合せはターゲット集合が問合せ集合を含むかどうかについての問合せであるから, この種の問合せを $T \supseteq Q$ (*has-subset*) と表す. メンバシップに関する問合せ $T \ni q$ (has-member) は $T \supseteq Q$ の特殊な場合である. 先の問合せにおいて orders 属性上に集合値索引 Iが存在すると, 問合せは以下のように変換できる.

$$set\{d.name \mid d \leftarrow SetIdx(I, has\text{-}subset, \{pen, ink\})\}$$

$SetIdx(\ldots)$ は, 集合値検索条件を満たす部署のオブジェクト識別子からなるコレクションを表している.

集合値索引を実現する手法として, 本稿では集合シグネチャファイル (set-based signature file) [IKO93, 石川 95b, IK95] を想定する. 集合シグネチャファイルは, 従来テキスト検索の分野で用いられてきたシグネチャファイルの手法を集合値検索に適用したものである. 集合シグネチャファイルの特長として, 上に示したような $T \supseteq Q$ の問合せだけでなく, 包含関係の向きが逆である場合 ($T \subseteq Q$, *is-subset*), 集合が交わりを持つかどうか ($T \sqcap Q$, *has-intersection*) [1], 集合が等しいかどうか ($T \equiv Q$, *is-equal*) など, 多くの種類の集合値検索を支援できることが挙げられる.

一方, 集合シグネチャファイルの構成手法から生じる欠点として, その検索結果の中に検索条件に誤ってマッチしたもの (フォルスドロップ (false drop) と呼ばれる) が余分に含まれる可能性があることが挙げられる. そのため, 集合シグネチャを用いた問合せ処理では, フォルスドロップをふるい落とすための処理であるフォルスドロップレゾリューション (false drop resolution) が不可欠である. 上で示した $SetIdx(\ldots)$ を集合シグネチャファイルにより実装した例を次に示す.

[1] $T \sqcap Q$ は $T \cap Q \neq \emptyset$ を意味する.

$set\{d \mid d \leftarrow SSigFile(I, has\text{-}subset, \{pen, ink\}),$
$d.orders \supseteq \{pen, ink\}\}$

$d.orders \supseteq \{pen, ink\}$ がフォルスドロップレゾリューションを表している. この条件は最初に与えられた問合せの条件と同じであるが, 集合シグネチャファイルにより十分に絞り込まれたデータを対象とした処理であるため, その処理コストは小さいと考えられる.

[石川 95a] では, 集合シグネチャファイルの物理構造のモノイド代数による記述を試み, 物理的なファイル構造を考慮した問合せ処理について考察した. また, [IK94, IK95] では, 集合値属性を持つ入れ子オブジェクト (nested object) に対する集合値索引について議論し, 集合シグネチャファイルを含む 4 種類の索引手法について比較検討を行なっている. 集合値検索手法としては, 他に RD-Tree [HNP95] が存在するが, 具体的な性能評価は行なわれていない.

4 問合せ処理

4.1 問合せ処理の概要

問合せ処理のステップは以下のようになる.

1. ユーザから与えられた OQL の問合せをモノイド内包表記表現に変換する. OQL から内包表記への変換規則は, [Feg94, FM95] では述べられていないが容易に与えることができる. OQL の代表的な構文に対する変換規則を, 表示的意味論にしたがって図 1に示す. TE[[...]] は式に関する一般的な変換, TD[[...]] はドメインに関する変換を表している. TO[[...]] は OQL の演算子をモノイド内包表記の対応する演算子に変換する.
2. モノイド内包表記表現を正規化する. モノイド内包表記の正規化は, 式の簡単化, 問合せ処理における中間的なデータ構造の削減, および問合せの最適化を容易化するために行なわれ, [Feg94, FM95] では正規化のために 6 個の変換規則が提案されている. ここではそれらのうち特に重要である二つのみを示す.

$$\mathcal{M}\{e \mid \bar{q}, v \leftarrow \mathcal{N}\{e' \mid \bar{r}\}, \bar{s}\} \Rightarrow \mathcal{M}\{e \mid \bar{q}, \bar{r}, v \hookleftarrow e', \bar{s}\}$$
$$\mathcal{M}\{e \mid \bar{q}, \mathrm{some}\{pred \mid \bar{r}\}, \bar{s}\} \Rightarrow \mathcal{M}\{e \mid \bar{q}, \bar{r}, pred, \bar{s}\}$$

ここで, $v \hookleftarrow e$ は式 e の変数 v への代入を表す. また, $\bar{q}$, $\bar{r}$, $\bar{s}$ は限定子の 0 個以上の並びを表す. 前者は入れ子状の内包表記表現を平坦化するための規則である. 後者は存在限量 (some) を消去し平坦化するための規則である. これにより, 正規化されたモノイド内包表記表現には存在限量は含まれないことになる.

本稿では [Feg94, FM95] の 6 個の正規化規則に加え, 全称限量に関する以下の変換規則を付け加える.

$$\mathrm{all}\{pred \mid \bar{q}\} \Rightarrow \mathrm{all}\{true \mid \bar{q}, pred\}$$

$pred$ は条件および述語, またはそれらを論理演算子により組み合わせたものである. この正規化により, 5 節で述べる集合値検索条件の抽出を容易にする.

3. 正規化されたモノイド内包表現について最適化のための変換規則を適用し, 問合せ実行の候補 (一般に複数個) を生成する. 後述するように, 適用する規則を論理レベルと物理レベルの二つのグループに分け, 段階をふんで適用する.
4. ステップ 3 で得られたモノイド内包表現のそれぞれをモノイド代数表現に変換し, モノイド代数における最適化の規則 (可能であれば) を適用する.
5. ステップ 4 で得られたモノイド代数についてコスト評価を行ない最適な候補を選び出し実行する.

[Feg94, FM95] では上のステップ 3 に対応する最適化は行なわず, モノイド内包表現を直接モノイド代数に変換し最適化する方式が提案されていた. しかし, そのようなアプローチでは, 内包表記に基づく最適化手法に関するこれまでの研究成果 [PS94, Tri89] を有効利用できない. また, 内包表記はモノイド代数などの実行レベルの代数に比べより抽象的であり, 最適化の見通しが立てやすいという利点も存在する. よって, 内包表記を対象とした最適化は最適化の効率の面でも有効であると考えられる.

TE[[select distinct E_1 from E_2 where E_3]]
⇒ set{TE[[E_1]] | TD[[E_2]], TE[[E_3]]}
TE[[select E_1 from E_2 where E_3]]
⇒ bag{TE[[E_1]] | TD[[E_2]], TE[[E_3]]}
TE[[E_1 and E_2]] ⇒ TE[[E_1]], TE[[E_2]]
TE[[E_1 in E_2]] ⇒ TE[[E_1]] ∈ TE[[E_2]]
TE[[E_1 ϕ E_2]] ⇒ TE[[E_1]] TO[[ϕ]] TE[[E_2]]
TE[[exists I in E_1: E_2]]
⇒ some{TE[[E_2]] | I ← TE[[E_1]]}
TE[[forall I in E_1: E_2]]
⇒ all{TE[[E_2]] | I ← TE[[E_1]]}
TE[[E_1.E_2]] ⇒ TE[[E_1]].TE[[E_2]]
TE[[(E)]] ⇒ TE[[E]]
TE[[E]] ⇒ E (E が定数名, 識別子のとき)
TD[[D_1, D_2]] ⇒ TD[[D_1]], TD[[D_2]]
TD[[I in E]] ⇒ I ← TE[[E]]

図 1: OQL からモノイド内包表記への変換規則

4.2 問合せ最適化のための一般的規則

4.1節で述べた問合せ処理のステップ3では, 与えられた問合せに対しモノイド内包表記において成立する等価性に基づく変換規則を適用し, 問合せ実行プランの候補を求める. 本研究では, 等価性に基づく変換規則を, 論理レベルと索引などを考慮する物理レベルの二つのグループに分けて考え, 適用の順番も論理レベルを先にする. まず, 論理レベルの規則として, 内包表記に基づく問合せ処理においてすでに一般的に知られている等価性 [PS94, Tri89] を示し, 次いで経路式を分解/統合するための規則を示す. 最後に, 物理的な規則の例として, 索引の導入に関する規則を示す.

論理レベルの規則 以下の規則において, f, gはフィルタ式を表している. また, $FV(e)$ により, 式 e 中の自由変数の集合を表している.

gen/1 ($FV(x) \cap FV(v) = FV(y) \cap FV(u) = \emptyset$ のとき)
$\mathcal{M}\{e \mid \bar{q}, x \leftarrow u, y \leftarrow v, \bar{r}\}$
$\Leftrightarrow \mathcal{M}\{e \mid \bar{q}, y \leftarrow v, x \leftarrow u, \bar{r}\}$

gen/2 ($FV(x) \cap FV(f) = \emptyset$ のとき)
$\mathcal{M}\{e \mid \bar{q}, x \leftarrow u, f, \bar{r}\} \Leftrightarrow \mathcal{M}\{e \mid \bar{q}, f, x \leftarrow u, \bar{r}\}$

gen/3 $\mathcal{M}\{e \mid \bar{q}, f, g, \bar{r}\} \Leftrightarrow \mathcal{M}\{e \mid \bar{q}, g, f, \bar{r}\}$

gen/4 $\mathcal{M}\{e \mid \bar{q}, f, g, \bar{r}\} \Leftrightarrow \mathcal{M}\{e \mid \bar{q}, f \wedge g, \bar{r}\}$

gen/1 から **gen/3** は, 一般に *qualifier interchange* と呼ばれる規則である. **gen/1** は, 生成式の入れ換えを表しており, 結合演算の順序の入れ換えに相当する. **gen/2** は, 右向きに使用すればフィルタ式を先に実行することになり, *filter promotion* と呼ばれる. これは, リレーショナルデータベースにおいて選択演算を先に実行するよう式を変換することに相当する. **gen/3** はフィルタ式の入れ換え, **gen/4** はフィルタ式の組合せ (右向き)/分解 (左向き) である.

本研究では一般に入れ子構造を持つオブジェクト指向データベースを対象とするため, 最適化の処理において経路式 (path expression) を扱う必要が生じる. そのため, 以下に経路式の分解のための規則を示す. p, p', p'' がそれぞれ長さ1以上であるような経路 $p.p'.p''$ が存在するとする. また, 経路 $p.p'$ のエクステント $ext(p.p')$ が存在するものとする. ただし, 経路 $p.p'$ のエクステントとは, 経路 $p.p'$ の末端の属性が単純値をとる場合にはそのドメイン, コレクション値をとる場合にはコレクションの要素のドメインとする. このとき, 経路式を分解する規則は以下のように与えられる.

path/1 (経路 $p.p'$ が単純値属性をとるとき)
$\mathcal{M}\{e \mid \bar{q}, u \leftarrow p, \bar{r}\}$
$\Rightarrow \mathcal{M}\{e[v/u.p'] \mid \bar{q}, u \leftarrow p, v \leftarrow ext(u.p'), v = u.p', \bar{r}[v/u.p']\}$

path/2 (経路 $p.p'$ がコレクション値をとるとき)
$\mathcal{M}\{e \mid \bar{q}, u \leftarrow p, \bar{r}\}$
$\Rightarrow \mathcal{M}\{e[v/u.p'] \mid \bar{q}, u \leftarrow p, v \leftarrow ext(u.p'), w \leftarrow u.p', v = w, \bar{r}[v/u.p']\}$

ただし, $e[v/u.p']$, $\bar{r}[v/u.p']$ は, それぞれ e, $\bar{r}$ 中に現れる経路 $u.p'$ を v で置き換えたものを表す.

逆に, 経路式を組み合わせる規則は以下のように与えられる. ただし, $ext(p.p') = s$ が成り立つものとする.

path/3 (経路 $p.p'$ が単純値属性をとるとき)
$\mathcal{M}\{e \mid \bar{q}, u \leftarrow p, v \leftarrow s, v = u.p', \bar{r}\}$
$\Rightarrow \mathcal{M}\{e[u.p'/v] \mid \bar{q}, u \leftarrow p, \bar{r}[u.p'/v]\}$

path/4 (経路 $p.p'$ がコレクション値をとるとき)
$\mathcal{M}\{e \mid \bar{q}, u \leftarrow p, v \leftarrow s, w \leftarrow u.p', v = w, \bar{r}\}$
$\Rightarrow \mathcal{M}\{e \mid \bar{q}, u \leftarrow p, \bar{r}\}$

これらの規則は, [FV95] において folding/unfolding と呼ばれている変換に相当する.

物理レベルの規則 内包表記に基づく問合せ処理において, 索引を導入し検索の効率化をはかるための変換規則についてもすでに提案がなされており, その規則は一般に *index introduction* と呼ばれている [Tri89]. ただし, 従来の研究では, リレーショナルデータベースのようにフラットな構造のデータベース上の索引が考慮の対象であった. 本研究では, 一般に入れ子構造を有するオブジェクト指向データベースを対象としているため, 入れ子オブジェクトのための索引手法 [Ber93] を有効利用することを考え, 規則を一般化する.

index/1 (a が定数であり, 経路 $p.v$ 上に二次索引 I が存在するとき)
$\mathcal{M}\{e \mid \bar{q}, x \leftarrow p, x.v = a, \bar{r}\}$
$\Rightarrow \mathcal{M}\{e \mid \bar{q}, x \leftarrow Index(I, equal, a), \bar{r}\}$

$Index(I, equal, a)$ は, 経路 $p.v$ のインスタンスのうち末端の値が a であるものについて, その経路のルートオブジェクトのオブジェクト識別子のコレクションを返す.

$Index(\ldots)$ は一種のテンプレートであり, 実際に実装されている索引手法にしたがって具体化される. たとえば, パスインデックス (path index) [Ber93] が実装されている場合には, 規則は以下のように具体化できる.

pathidx/1 (a が定数であり, 経路 $p.v$ 上にパスインデックス I が存在するとき)
$\mathcal{M}\{e \mid \bar{q}, x \leftarrow p, x.v = a, \bar{r}\}$
$\Rightarrow \mathcal{M}\{e \mid \bar{q}, x \leftarrow PathIdx(I, equal, a), \bar{r}\}$

5 集合値検索の最適化

5.1 変換規則

集合値検索条件の抽出 本研究では集合値索引の効率的な利用を目標としているが, 正規化されたモノイド内包表記表現の中には集合値検索は明示的には現れていない. よって, まず, パターンマッチに基づいて内包表記表現を書き換えることにより, 集合値検索条件を抽出することが必要である.

まず, メンバシップに関する検索条件 $T \ni q$を導くための規則を示す.

membership/1 (a が定数, かつ $FV(\bar{r}) \not\ni v$のとき)
$\mathcal{M}\{e_1 \mid \bar{q}, v \leftarrow e_2, v = a, \bar{r}\} \Rightarrow \mathcal{M}\{e_1 \mid \bar{q}, a \in e_2, \bar{r}\}$
membership/2 (a が定数のとき)
$\mathcal{M}\{e_1 \mid \bar{q}, u \leftarrow p, u.v = a, \bar{r}\}$
$\Rightarrow \mathcal{M}\{e_1 \mid \bar{q}, u \leftarrow p, a \in p.v, \bar{r}\}$

前者の規則は変数 vが残りの限定子に含まれない場合に有効であり, $T \ni q$の条件をくくり出すために用いられる. 後者の規則は後述する集合値索引の導入において利用されるもので, p は経路を意味している.

次に, 全称限量 (all) から集合値検索条件を抽出するための規則を示す.

all/1 $\mathrm{all}\{true \mid v \leftarrow y, v \in x\} \Rightarrow x \supseteq y$
all/2 ($ext(p.v)$ が存在するとき)
$\mathcal{M}\{e \mid \bar{q}, u \leftarrow p, \mathrm{all}\{true \mid x \leftarrow u.v, \bar{f}\}, \bar{r}\}$
$\Rightarrow \mathcal{M}\{e \mid \bar{q}, u \leftarrow p, u.v \subseteq \mathrm{set}\{x \mid x \leftarrow ext(p.v), \bar{f}\}, \bar{r}\}$
all/3
$\mathcal{M}\{e \mid \bar{q}, u \leftarrow p, \mathrm{all}\{true \mid x \leftarrow ext(p.v), x \in u.v, \bar{f}\}, \bar{r}\}$
$\Rightarrow \mathcal{M}\{e \mid \bar{q}, u \leftarrow p, \mathrm{set}\{x \mid x \leftarrow ext(p.v), \bar{f}\} \subseteq u.v, \bar{r}\}$

all/1 は, メンバシップに関する検索条件 $T \ni q$と全称限量の組合せから集合の包含関係 ($\supseteq$) に基づく検索条件を導き出すための規則である. **all/2** は, 入れ子構造の内部に対する全称限量を集合値検索 $T \subseteq Q$ (is-subset) に変換する規則である. ただし, $ext(p.v)$ は経路 $p.v$に対するエクステントを表している. **all/3** は全称限定を集合値検索条件 $T \supseteq Q$ (has-subset) に変換する規則である. **all/2, all/3** の規則において, $\bar{f}$は 0 個以上のフィルタ式を表す.

最後に, 集合値検索条件 $T \sqcap Q$ を導くための規則を示す.

intersect/1 $\mathcal{M}\{e \mid \bar{q}, u \leftarrow p, x \leftarrow s, x \in u.v, \bar{r}\}$
$\Rightarrow \mathcal{M}\{e \mid \bar{q}, u \leftarrow p, u.v \sqcap s, \bar{r}\}$

集合値索引の導入 次に, 集合値索引導入のための規則を示す. これらの規則は 4.2 節で述べた物理レベルの規則に相当する. いずれも, 経路 $p.v$上に集合値索引が存在するときにのみ適用できる.

setidx/1 $\mathcal{M}\{e_1 \mid \bar{q}, x \leftarrow p, e_2 \in x.v, \bar{r}\}$
$\Rightarrow \mathcal{M}\{e_1 \mid \bar{q}, y \leftarrowtail e_2, x \leftarrow SetIdx(I, has\text{-}member, y), \bar{r}\}$
setidx/2 $\mathcal{M}\{e_1 \mid \bar{q}, x \leftarrow p, e_2 \subseteq x.v, \bar{r}\}$
$\Rightarrow \mathcal{M}\{e_1 \mid \bar{q}, y \leftarrowtail e_2, x \leftarrow SetIdx(I, has\text{-}subset, y), \bar{r}\}$
setidx/3 $\mathcal{M}\{e_1 \mid \bar{q}, x \leftarrow p, x.v \subseteq e_2, \bar{r}\}$
$\Rightarrow \mathcal{M}\{e_1 \mid \bar{q}, y \leftarrowtail e_2, x \leftarrow SetIdx(I, is\text{-}subset, y), \bar{r}\}$
setidx/4 $\mathcal{M}\{e_1 \mid \bar{q}, x \leftarrow p, x.v \sqcap e_2, \bar{r}\}$
$\Rightarrow \mathcal{M}\{e_1 \mid \bar{q}, y \leftarrowtail e_2,$
$x \leftarrow SetIdx(I, has\text{-}intersection, y), \bar{r}\}$

ただし, 上の規則すべてにおいて, yは既存の変数と衝突しない新しい変数であるものとする. 4.2 節における議論と同様, これらの集合値索引導入のための規則は, 集合値索引を抽象化したものであり, 具体的な変換規則の内容は集合値索引の実装方式によって異なる. 集合値索引の実装手法として集合シグネチャファイルを用いた場合, 上の規則は以下のように具体化できる.

ssigfile/1 $\mathcal{M}\{e_1 \mid \bar{q}, x \leftarrow p, e_2 \in x.v, \bar{r}\}$
$\Rightarrow \mathcal{M}\{e_1 \mid \bar{q}, y \leftarrowtail e_2,$
$x \leftarrow SSigFile(I, has\text{-}subset, y),$
$y \in x.v, \bar{r}\}$
ssigfile/2 $\mathcal{M}\{e_1 \mid \bar{q}, x \leftarrow p, e_2 \subseteq x.v, \bar{r}\}$
$\Rightarrow \mathcal{M}\{e_1 \mid \bar{q}, y \leftarrowtail e_2,$
$x \leftarrow SSigFile(I, has\text{-}subset, y),$
$y \subseteq x.v, \bar{r}\}$
ssigfile/3 $\mathcal{M}\{e_1 \mid \bar{q}, x \leftarrow p, x.v \subseteq e_2, \bar{r}\}$
$\Rightarrow \mathcal{M}\{e_1 \mid \bar{q}, y \leftarrowtail e_2,$
$x \leftarrow SSigFile(I, is\text{-}subset, y),$
$x.v \subseteq y, \bar{r}\}$
ssigfile/4 $\mathcal{M}\{e_1 \mid \bar{q}, x \leftarrow p, x.v \sqcap e_2, \bar{r}\}$
$\Rightarrow \mathcal{M}\{e_1 \mid \bar{q}, y \leftarrowtail e_2,$
$x \leftarrow SSigFile(I, has\text{-}intersection, y),$
$x.v \sqcap y, \bar{r}\}$

いずれの規則も, 経路 $p.v$上に集合シグネチャファイルが存在する場合にのみ適用できる. **ssigfile/1** はメンバシップに関する集合値検索 $T \ni q$の変換であるが, 集合シグネチャファイルでは$T \ni q$を単に$T \supseteq Q$ の特殊な場合として扱うため, $T \supseteq Q$ の問合せとして扱われている. いずれの場合もフォルスドロップレゾリューションに相当する条件が追加される.

5.2 問合せの変換例

前節で述べた変換規則の有効性を示すため, 具体的な問合せとその変換例を示す. 問合せ `Q1` は, 受講している授業がすべてデータベースの分野に属している学生の名前を求める OQL の問合せである.

```
Q1
select distinct s.name
from s in Students
where forall c in s.cources: (c.category = "DB")
```

この問合せは図 1に示した規則により直接的に以下のモノイド内包表記表現に変換できる.

$$\mathrm{set}\{s.name \mid s \leftarrow Students, \\ \mathrm{all}\{c.category = DB \mid c \leftarrow s.cources\}\}$$

正規化により

$$\mathrm{set}\{s.name \mid s \leftarrow Students, \\ \mathrm{all}\{true \mid c \leftarrow s.cources, \\ c.category = DB\}\}$$

が得られる. これに対して **all/2** を適用することにより,

$$\mathrm{set}\{s.name \mid s \leftarrow Students, \\ s.cources \subseteq \mathrm{set}\{c \mid c \leftarrow Cources, \\ c.category = DB\}\}$$

が得られる.

問合せ Q2 は, データベースの分野の授業をすべて受講している学生の名前を求める問合せである.

```
Q2
select distinct s.name
from s in Students
where forall c in (select c
                   from Cources
                   where c.category = DB)
      : (c in s.cources)
```

この問合せは, 以下のモノイド内包表記表現に直接的に変換できる.

$$\mathrm{set}\{s.name \mid s \leftarrow Students, \\ \mathrm{all}\{c \in s.cources \mid \\ c \leftarrow \mathrm{set}\{c \mid c \leftarrow Cources, c.category = DB\}\}\}$$

これに正規化の規則を適用することにより,

$$\mathrm{set}\{s.name \mid s \leftarrow Students, \\ \mathrm{all}\{true \mid c \leftarrow Cources, c.category = DB, \\ c \leftarrow c, c \in s.cources\}\}$$

が得られる. $c \leftarrow c$ の変数名のつけかえは自明であるので, これを消去して

$$\mathrm{set}\{s.name \mid s \leftarrow Students, \\ \mathrm{all}\{true \mid c \leftarrow Cources, c.category = DB, \\ c \in s.cources\}\}$$

となる. **gen/3** によりフィルタ式を入れ換えた後 **all/3** を適用することにより,

$$\mathrm{set}\{s.name \mid s \leftarrow Students, \\ \mathrm{set}\{c \mid c \leftarrow Cources, \\ c.category = DB\} \subseteq s.cources\}$$

が得られる.

以上は論理レベルの変換例である. 物理レベルの変換例を示すため, 経路 Students.cources 上に集合値索引 I が存在し, 経路 Cources.category 上にパスインデックス (この場合は単なる B-木と考えられる) I' が存在する場合を考える. 問合せ Q1 に対する変換結果に **pathidx/1** を適用することにより,

$$\mathrm{set}\{s.name \mid s \leftarrow Students, \\ s.cources \subseteq set\{c \mid c \leftarrow PathIdx(I', equal, DB)\}\}$$

が得られ, **setsig/3** を適用することで

$$\mathrm{set}\{s.name \mid y \leftarrow set\{c \mid c \leftarrow PathIdx(I', equal, DB)\}, \\ s \leftarrow SSigFile(I, is\text{-}subset, y), \\ s.cources \subseteq y\}$$

が得られる. $PathIdx(\ldots)$ の値が集合であるなら, さらに

$$\mathrm{set}\{s.name \mid y \leftarrow PathIdx(I', equal, DB), \\ s \leftarrow SSigFile(I, is\text{-}subset, y), \\ s.cources \subseteq y\}$$

と変型できる. これが最適な問合せ実行プランであるかどうかはさらに下位レベルでのコスト評価が必要となるが, 一つの候補が得られたことになる.

6 まとめと今後の課題

本稿では, オブジェクト指向データベース標準 ODMG-93 の問合せ言語 OQL を対象に, 内包表記に基づく問合せの最適化について議論した. 特に, 集合値索引を有効利用するため, 集合値検索条件の抽出と集合値索引を用いた問合せへの変換規則について述べた.

今回は集合シグネチャファイルを, 与えられた集合値検索条件に基づいて (フォルスドロップを含んだ) 検索結果を返す索引として抽象的に扱ったが, 実際にはその実装方式により性能や特性が大きく異なる [IKO93, 石川 95b, IK95]. [石川 95a] において試みた, 集合シグネチャの物理的な構成のモノイド代数による記述を統合化し, より物理的な処理の統合化をはかることが一つの課題となる.

もう一つの課題として, 最適化のための変換規則の追加と, それらの効果的な制御方法が挙げられる. 変換規則の種類とその適用順に関する議論は [PS94] で多少なされているが, 最適化処理の効率化のためにはそれらの制御方式は重要である. その他の課題としては, コスト評価の枠組の開発などが挙げられる.

謝辞

日頃から御支援いただいている奈良先端科学技術大学院大学 マルチメディア統合システム講座 (植村研究室) の皆様に感謝いたします. また, 御助言等いただいた筑波大学 北川博之助教授に御礼申し上げます.

参考文献

[Ber93] E. Bertino: "A Survey of Indexing Techniques for Object-Oriented Database Management Systems", in J. C. Freytag, D. Maier, and G. Vossen eds., *Query Processing for Advanced Database Systems*, chapter 13, pp. 383–418, Morgan Kaufmann, San Mateo, California, 1993.

[BLST94] P. Buneman, L. Libkin, D. Suciu, and V. Tannen: "Comprehension Syntax", *ACM SIGMOD Record*, 23(1):87–96, Mar. 1994.

[CAD+95] R. G. G. Cattell, T. Atwood, J. Duhl, G. Ferran, M. Loomis, and D. Wade eds.: オブジェクト・データベース標準: ODMG-93 Release 1.1, 共立出版, 1995.

[CT94] D. K. C. Chan and P. W. Trinder: "Object Comprehensions: A Query Notation for Object-Oriented Databases", in D. S. Bowers ed., *Proc. 12th British National Conf. on Databases (BNCOD 12)*, Vol. 826 of *Lecture Notes in Computer Science*, pp. 55–72, Guildford, UK, July 1994, Springer-Verlag.

[Feg94] L. Fegaras: "A Uniform Calculus for Collection Types", Technical Report 94-030, Department of Computer Science and Engineering, Oregon Graduate of Science & Technology, Portland, OR, 1994.

[FM95] L. Fegaras and D. Maier: "Towards an Effective Calculus for Object Query Languages", in *Proc. of ACM SIGMOD*, pp. 47–58, San Jose, CA, May 1995.

[FV95] D. Florescu and P. Valduriez: "Rule-Based Query Rewriting in the Flora Optimizer", *IEICE Trans. on Information Systems*, E78-D(11):1412–1423, Nov. 1995.

[HNP95] J. M. Hellerstein, J. F. Naughton, and A. Pfeffer: "Generalized Search Trees for Database Systems", in *Proc. of Very Large Data Bases*, pp. 562–573, Zurich, Switzerland, Sept. 1995.

[IK94] Y. Ishikawa and H. Kitagawa: "Analysis of Indexing Schemes to Support Set Retrieval of Nested Objects", in *International Symposium on Advanced Database Technologies and Their Integration (ADTI'94)*, pp. 55–62, Nara, Japan, Oct. 1994.

[IK95] Y. Ishikawa and H. Kitagawa: "Design and Performance Analysis of Indexing Schemes for Set Retrieval of nested Objects", *IEICE Trans. on Information Systems*, E78-D(11):1424–1432, Nov. 1995.

[IKO93] Y. Ishikawa, H. Kitagawa, and N. Ohbo: "Evaluation of Signature Files as Set Access Facilities in OODBs", in *Proc. ACM SIGMOD Intl. Conf. on Management of Data*, pp. 247–256, May 1993.

[PS94] A. Poulovassilis and C. Small: "Investigation of Algebraic Query Optimisation Techniques for Database Programming Languages", in *Proc. 20th VLDB Conf.*, pp. 415–426, Santiago, Chile, Sept. 1994.

[Tri89] P. Trinder: *A Functional Database*, PhD thesis, University of Oxford, Oxford, England, Dec. 1989, (available as Technical Monograph PRG-82, Programming Research Group, 8-11 Keble Road, Oxford OX1 3QD, England, or, Technical Report CSC 90/R10 Computer Science Department, Glasgow University, Glasgow, Scotland.

[Wad87] P. Wadler: "List Comprehensions", in S. L. P. Jones ed., *The Implementation of Functional Programming Languages*, chapter 7, pp. 127–138, Prentice-Hall, 1987.

[石川 95a] 石川, 北川: "集合シグネチャファイルによるコレクションオブジェクトの問合せ処理", 情報処理学会研究報告, 95(65):9–16, July 1995, IPSJ-DBS-104-2.

[石川 95b] 石川, 北川, 大保: "シグネチャファイルによる集合値検索のコスト評価", 情報処理学会論文誌, 36(2):383–395, Feb. 1995.