

関数型データベース検索に適したデータ構造とそのオペレータの実装

4R-10

舛谷和幸 富井尚志 有澤博

横浜国立大学 工学部 電子情報工学科

1 はじめに

データベースで扱う情報は、従来の文字、数字などのテキスト情報に加えて、画像、映像、音声などさまざまな表現形態を持つマルチメディア情報へとその範囲を広げている。

それらマルチメディア情報の大きな特徴は、基本的に時間軸上に連なった列であり、しかもそれらが階層構造を持っている、という点にある。著者らの研究室ではこのような情報を統合的にモデリングできる AIS データモデル [1] を採用した DBMS を試作している。この DBMS では検索言語は、検索結果としてオブジェクト式と呼ばれる階層構造を生成する。本稿は、それら階層構造を持った列情報を計算機上で表現し、さらに効果的に基本的操作ができる ADT を提案する。

2 検索結果の構造

AIS-DBMS で採用された関数型 DB 検索言語 Multi-Media Query Language:MMQL[2] は、DBMS 内の平坦な構造の中から検索結果としてオブジェクト式と呼ばれる階層構造をもった列を生成する。この構造の例を図に示すと図 1 の様になる。オブジェクト式

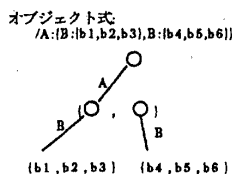


図 1: オブジェクト式の例

の構造を BNF によって表したものを図 2 に示す。

ここで、各属性の要素の構造をその属性の基本形 (cardinality) とよぶ。さらに BNF によって生成された木構造が階層構造を持つために列について以下の

```

<オブジェクト式> ::= <原子> | <組> | <属性>
<組> ::= <L bracket> <属性> { <comma> <属性> } <R bracket>
<属性> ::= <属性名> <colon> <原子>
           | <属性名> <colon> <原子>
           | <属性名> <colon> <属性>
           | <属性名> <colon> <列>
<列> ::= <L brace> <オブジェクト式>
         { <comma> <オブジェクト式> } <R brace>
<原子> ::= 文字列 | 数値 | 主体識別子
         | 画像データ | その他のマルチメディアデータ
<属性名> ::= 文字列
<L bracket> ::= "[" <R bracket> ::= "]"
<L brace> ::= "{" <R brace> ::= "}"
<colon> ::= ":" <comma> ::= ","
    
```

図 2: オブジェクト式の BNF による定義

ような規則を加える。

- 列 (p_1, p_2, \dots, p_n) について、すべての p_i が原子であれば、その列は階層的である。
- 列 (p_1, p_2, \dots, p_n) について、 p_i が属性であるか組であるとき、すべての p_i について同一の属性名を持つ属性はすべて基本形が同じであり、かつその属性の要素すべてを含む列もまた階層的であれば、その列は階層的である。
- オブジェクト式において、すべての列は階層的でなければならない。

3 計算機上での情報表現

以上のような特質を持つオブジェクト式の効率的な実装は、AIS-DBMS においてキーとなる技術である。オブジェクト式が検索式の途中結果であり、かつ、検索結果自体であるからである。

また、オブジェクト式自身は階層をもった列という特徴を持っており、これを効率的に表現・操作できるように、工夫されていなければならない。

ここでは、これらの特徴に着目して実装の基本単位となる Object Expression Primitive List:OEPL を導入する。

3.1 OEPL

OEPL は、ある属性を持った列を記憶装置にリストの形式によって蓄える。これは、オブジェクト式が「属性」という集合を演算の基とするための仕様である。OEPL は、大きく分けて、属性の名前など、注目属性固有の情報を格納し、属性間のつながりも管理する TAG と呼ばれる部分と、実際の属性の要素を格納する ALIST と呼ばれる部分とに分けられる。階層の情報を蓄えるため、OEPL による情報管理は

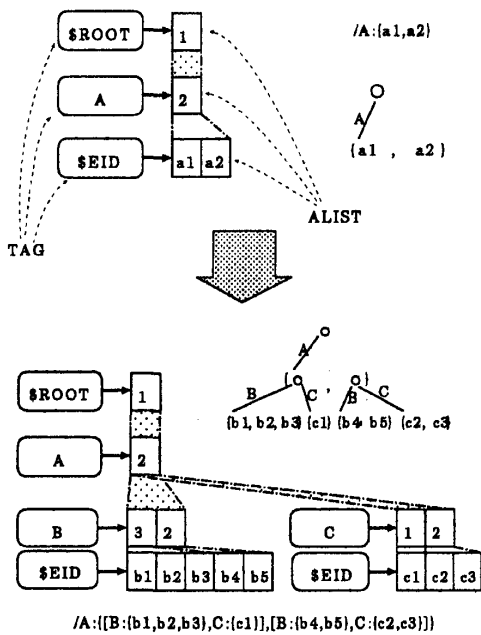


図 3: OEPL とそのオペレーション例

基本的に OEPL を一つの属性に対して二つ配し、下位に位置する OEPL に実際の属性の要素の情報を格納し、上位に位置する OEPL にはそれぞれの属性の要素の「重み」と呼ばれる情報を格納する。

ここで、簡単な例を参照して説明する。今、このような構造をオブジェクト式で表すと図 3 のようになる。

この構造は、属性 A が 2 つの要素をもち、それぞれが属性 B に 3 つと 2 つ、属性 C に 1 つと 2 つの要素を持っていることを表している。この、一つの属性の要素が自分の内部にいくつの要素を持っているか、を表すのが「重み」とよばれる情報であり、重みが属性 A と B、A と C のように上下に連なるこ

とにより、木を構成する。そして、重みは、一つの属性の要素として列をなしている。

このようにして、属性と属性、また、属性の要素同士の階層的なつながりを、TAG,ALIST および「重み」を用いることで、階層構造を持つ列の表現が可能となる。しかも重みはいわば論理的なポインタであり、属性間の階層の情報は TAG にのみ保存して各属性の要素間には物理的なポインタを一切張らない。

このため、属性の追加など、OEPL に対する基本操作は下位の属性のに対するポインタの付けかえなどをほとんど行なわない、効率良い実現が可能である。

また、一つの属性 A の要素がさらに複数の属性 R_n を持つ (組、と呼ぶ) ことも、各 R_n に対応した TAG を設けるという操作で表現できる。さらに組を生成する際、 $i, j \in n$ とすると、 R_i と R_j の要素の間には関連がないため、各 R_i と R_j の生成の過程では並列化が可能である。

以上のことから、OEPL は簡素な基本要素でオブジェクト式を表現し、検索中に行なわれるオブジェクト式の木の枝の生長、退化等の操作を効率良く行なえる構造、であるといえる。

4 むすび

本稿では、階層構造を持った列であるオブジェクト式を非常に簡素な基本要素で表現でき、その操作も効率良く行なえる OEPL 構造を提案した。

現在、この構造を基にして検索言語 MMQL の実装が行なわれている。

参考文献

[1] H. Arisawa, T. Tomii, H. Yui, H. Ishikawa : "Datamodel and architecture of multimedia database for engineering applications.", IEICE Trans. Inf. & Syst., vol.E78-D, No.11, pp.1362-1368, Nov. 1995.

[2] 富井尚志, 有澤博: "マルチメディアデータベースにおける映像モデリングと操作言語", 電子情報通信学会論文誌, Vol.79-E, No.4, April 1996.