集合の階層を保持するデータベース

3F−2

山崎康雄† 古川哲也‡ 島崎眞昭‡
(†九州大学工学部 ‡九州大学大型計算機センター)

1. はじめに

大量の実験データや収集データをデータベース化し解析や分析を行っていく上でデータを分類する作業は不可欠である.その支援のためにはデータベース管理システムが(1)集合それ自体,(2)解析/分析作業の過程で生じた集合間の包含関係という2つの情報を扱えなくてはならない.本稿ではデータの集合を互いに共通集合を持たない部分集合に分割するという分類方法を考えこの結果生じる(1),(2)の情報をデータベース上に蓄え矛盾なく更新を行う方法を示す.

2. データの分類

与えられたデータの集合をある性質を基に分類する.個々のデータをオブジェクトとする.n個のオブジェクト$o_1, o_2, \cdots, o_n$からなる集合の名前をsとするときsの値を$i(s) = \{o_1, o_2, \cdots, o_n\}$で表す.集合$s$は与えられた値により外延的に定義される場合と条件式を満足するオブジェクトの集合として内包的に定義される場合がある.$C_s(o)$を条件C_sを満たすオブジェクトoに対してのみ真となる述語とする.sが条件C_sによって内包的に定義されているとき$^{\forall}o{\in}i(s)\ \ C_s(o)$である.集合$s$が外延的に定義されているとき$C_s = \perp$(未定義)と表す.外延的に定義された集合および内包的に定義された集合をそれぞれ外延的集合,内包的集合と呼ぶ.

集合sを互いに共通集合を持たないm個の部分集合$s_1, s_2, \cdots, s_m$に分けるときsの分割の定義とその満たすべき制約を次のようにする.

【定義 1】 分割
集合sの分割は以下の2つの制約を満たすようなsの部分集合の名前集合$\{s_1, s_2, \cdots, s_m\}$である.集合$s$を分割における(集合$s_i$の)親,集合$s_i$を(集合$s$の)子と呼ぶ$(i = 1, 2, \cdots, m)$.

【制約 1】 包含制約 分割におけるすべての子は親の部分集合である. $^{\forall}s_i{\in}p(s)\ \ i(s_i) \subseteq i(s)$. □

【制約 2】 排他制約 分割中のどの2つの子も互いに共通集合を持たない. $^{\forall}s_i{\in}p(s)\ \ i(s_i) \cap (s)i(s_j) = \emptyset$. □

更に子を再帰的に分割することによって階層的な分類が行える.この結果生じたすべての集合の名前の集合を$\boldsymbol{S}$,すべてのオブジェクトの集合を$\boldsymbol{O}$とする.データベースがこれらの情報を蓄えていることは以下の4つの関数を求めることができるということで表される.

- 集合sの値を求める関数$i(s)$. $^{\forall}s{\in}\boldsymbol{S}\ i(s) \subseteq \boldsymbol{O}$.
- オブジェクトoの属する集合を求めるiの逆関数$\bar{i}(o)$. $\bar{i}(o) = \{s \mid o{\in}i(s)\}$. $^{\forall}o{\in}\boldsymbol{O}\ \bar{i}(o) \neq \emptyset$.
- 集合sの分割(子)を求める求める関数$p(s)$.集合sが分割を持たないとき$p(s) = \emptyset$とする.
- 集合の親を求めるpの逆関数$\bar{p}(s)$. $\bar{p}(s) = s'\ \ (s{\in}p(s'))$. $^{\exists}u{\in}\boldsymbol{S}\ i(u) = \boldsymbol{O}\ \ \bar{p}(u) = \emptyset$.

また集合の子孫および先祖を次のように定義する.

【定義 2】 子孫・先祖
集合sの子孫および先祖をそれぞれ以下のような関数$p^+(s)$, $\bar{p}^+(s)$で表す.

$p^+(s) = p(s) \cup_{\forall s_i{\in}p(s)} p^+(s_i)$.
$\bar{p}^+(s) = \bar{p}(s) \cup \bar{p}^+(\bar{p}(s))$.

また$p^*(s) = \{s\} \cup p^+(s)$, $\bar{p}^*(s) = \{s\} \cup \bar{p}^+(s)$とする.

3. 更新操作

データベース上に記憶した階層中の集合への更新操作を考える.集合sの分割$p(s)$に対する子s_iの挿入$ins(s, s_i)$および削除$del(s, s_i)$と集合sに対するオブジェクトoの挿入$ins(s, o)$および削除$del(s, o)$という4つの更新操作を考える.これらの操作を行うとき包含制約および排他制約をみたすように関数p, $\bar{p}$, i, $\bar{i}$を次のように変更しなければならない.

《$ins(s, s_i)$》条件: $s{\in}\boldsymbol{S}\ \ \wedge\ \ s_i{\notin}\boldsymbol{S}$.
$p(s)$にs_iを追加し$C_{s_i} \neq \perp$のとき$i(s_i)$を作成する.

```
ins(s, s_i): begin
               p(s) ← p(s) ∪ {s_i}
               i(s_i) ← {o | o∈i(s), C_{s_i}(o)}
               forall o∈i(s_i) do
                 ī(o) ← ī(o) ∪ {s_i}
             end.
```

《$del(s, s_i)$》条件: $s{\in}\boldsymbol{S}\ \ \wedge\ \ s_i{\in}p(s)$.
$p(s)$と$\bar{i}(o)$からs_iを削除し集合sの子孫に対しても同様の操作を行う.

```
del(s, s_i): begin
               forall s_j∈p(s_i) do
                 del(s_i, s_j)
               p(s) ← p(s) − {s_i}
               forall o∈i(s_i) do
                 ī(o) ← ī(o) − {s_i}
               S ← S − {s_i}
             end.
```

《$ins(s, o)$》条件: $s{\in}\boldsymbol{S}\ \ \wedge\ \ ^{\forall}s'{\in}\bar{p}^*(s) C_{s'}{\neq}\perp C_{s'}(o)\ \wedge \bar{i}(o){\subseteq}\bar{p}^+(s)\ \wedge\ ^{\forall}s'{\in}\bar{p}^+(s)\,^{\forall}s_i{\in}p(s')-\bar{p}^*(s)\ C_{s_i}{\neq}\perp\neg C_{s_i}(o)$.
一般に包含制約により$^{\forall}s{\in}\boldsymbol{S}\ \ ^{\forall}s_i{\in}p(s)\ \ ^{\forall}o{\in}i(s)\ \ C_{s_i}{\neq}\perp\ C_{s_i}(o)\ \ o{\in}i(s_i)$.集合中のオブジェクトが子の条件を満たしていれば子にも挿入されなければならない.オブジェクトoを集合sに挿入するときoが挿入されるべき最も深いsの子孫を求める$bot(s, o)$を次のように定義する.

$$bot(s, o) = \begin{cases} s & \neg\ ^{\exists}s_i{\in}p(s)\ C_{s_i}{\neq}\perp\ \wedge\ C_{x_i}(o) \\ bot(s_i) & ^{\exists}s_i{\in}p(s)\ C_{s_i}{\neq}\perp\ \wedge\ C_{x_i}(o) \end{cases}$$

Databases Maintaining Set Hierarchies
Yasuo YAMASAKI, Tetsuya FURUKAWA, Masaaki SHIMASAKI
Kyushu Univ.

$bot(s,o)$ の先祖の i に o を挿入する.

$ins(s,o)$: begin
forall $s_i \in \bar{p}^*(bot(s,o))$ do
$i(s_i) \leftarrow i(s_i) \cup \{o\}$
$\bar{i}(o) \leftarrow \bar{i}(o) \cup \{s_i\}$
end.

《$del(s,o)$》条件： $s \in S \ \wedge \ o \in i(s)$.

内包的な集合からオブジェクトを削除する場合には親からも削除しなければならない. 集合 s の祖先の中で一番高い外延的な集合を求める $top(s)$ を $top(s) = f(u)$ とする. ただし $i(u) = O$ とし関数 f は次の様に定義する.

$$f(x) = \begin{cases} x & C_{next(s)} = \perp \\ f(next(x)) & C_{next(s)} \neq \perp \end{cases}$$

ただし $next(x) = p(x) \cup \bar{p}^*(s)$ とする. $top(s)$ の子孫の i から o を削除する.

$del(s,o)$: begin
forall $s_i \in \bar{i}(o) - \bar{p}^*(top(s))$ do
$i(s_i) \leftarrow i(s_i) - \{o\}$
$\bar{i}(o) \leftarrow \bar{i}(o) - \{s_i\}$
if $\bar{i}(o) = \emptyset$ then
$O \leftarrow O - \{o\}$
end.

4. 関数の実現

階層を表す情報すなわち 4 つの関数 $p, \bar{p}, i, \bar{i}$ の実現 (検索の実現) を考える. 逆関数 $\bar{p}, \bar{i}$ はそれぞれ p, i を用いて $\bar{p}(s) = \{s' \mid s \in p(s')\}$, $\bar{i}(o) = \{s \mid o \in i(s), s \in S\}$ で実現できるので p, i の実現のみを示す.

[$p(s)$] 索引 $P_s = \{(s, s_i) \mid s_i \in p(s), \ s \in S\}$ の記憶で実現でき検索は $p(s) = \{s_i \mid (s, s_i) \in P_s\}$ となる.

[$i(s)$] $i(s)$ の実現には内包的な方法と外延的な方法がある. $int(i(s))$ を $i(s)$ が内包的な実現法をとっているときのみに真となる述語とする.

〈外延的な実現法・索引記憶〉集合 s の子の i を (i) 利用しないものと (ii) するものがある. これらの実現法は集合 s の定義法に関わらず用いることが出来る.

i) 索引 $I_s = \{(s,o) \mid o \in i(s)\}$ を記憶し 検索は $i(s) = \{o \mid (s,o) \in I_s\}$ となる.

ii) 集合 s の子の部分集合 $r(s) \subseteq p(s)$ の i の実現により一部を代表させる. ただし $i(s)$ の実現を $r(s)$ の i に依存させることになるので $r(s)$ の i の実現が $i(s)$ に依存していてはならない. $r(s)$ は $R_s = \{(s, s_i) \mid s_i \in r(s)\}$ の記憶で実現でき検索は $r(s) = \{s_i \mid (s, s_i) \in R_s\}$ となる. $i(s)$ は索引 $I_s = \{(s,o) \mid o \in i(s) - i(s_i), s_i \in r(s)\}$ の記憶および $r(s)$ の i により実現でき検索は $i(s) = \{o \mid (s,o) \in I_s\} \cup \{o \mid o \in i(s_i), \ s_i \in r(s)\}$ となる. $r(s)$ を集合 s の代表と呼ぶ.

〈内包的な実現法・条件式の記憶〉内包的な実現方法は集合 s の条件式 C_s を記憶しておき検索時に $i(s)$ を作り出すもので検索は $i(s) = \{o \mid C_s(o), \ o \in i(\bar{p}(s))\}$ となる. また

$$e(s) = \begin{cases} s & \neg \ int(i(s)) \\ e(\bar{p}(s)) & int(i(s)). \end{cases}$$

$$C^{(s)} = \begin{cases} true & \neg \ int(i(s)) \\ C_s \wedge C^{(\bar{p}(s))} & int(i(s)). \end{cases}$$

とするとき $i(s) = \{o \mid C^{(s)}(o), \ o \in i(e(s))\}$ あるいは $i(s) = \{o \mid C^{(s)}(o), \ o \in I_{e(s)}\}$ のようにも表せる. この実現法を用いることが出来るのは s が内包的集合である場合のみである.

$R_s \subseteq P_s$ なので $(P_s - R_s)$ と R_s の記憶で P_s と R_s を表せる. あるいは $R = \bigcup_{s \in S} R_s$ とすると $(P - R)$ と R の記憶で P と R を表せる. また $i(s)$ が内包的な実現方法を用いている場合で集合 s の子の多くが外延的な i の実現方法をとっているときにはこれらで代表される外延的な実現方法に置き換えた方が効率的な実現が期待できる.

5. 更新処理の実現

実現した関数上での更新処理を考える. 更新操作 $ins(s,o)$, $del(s,o)$, $ins(s,s_i)$, $del(s,s_i)$ は関数 $p, \bar{p}, i, \bar{i}$ への変更であり $\bar{p}, \bar{i}$ の実現はそれぞれ p, i に依っているので $p(s)$ に対する集合 s_i の挿入 $ins(p(s), s_i) : p(s) \leftarrow p(s) \cup \{s_i\}$ および削除 $del(p(s), s_i) : p(s) \leftarrow p(s) - \{s_i\}$ と $i(s)$ に対するオブジェクト o の挿入 $ins(i(s), o) : i(s) \leftarrow i(s) \cup \{o\}$ および削除 $del(i(s), o) : i(s) \leftarrow i(s) - \{o\}$ という 4 つの操作が実現できれば更新操作は実現できる.

《$ins(p(s), s_i)$》 :
実現は $P_s \leftarrow P_s \cup \{(s, s_i)\}$ となる. 更に集合 s_i が新しく s_i の親 s を代表するとしたときに s の代表 $r(s)$ および実現 I_s を更新しなければならない. すなわち $\neg int(i(s)) \ \wedge \ s_i \in r(s)$ のとき $r(s) \leftarrow r(s) \cup \{s_i\}$ および $I_s \leftarrow I_s - i(s_i)$.

《$del(p(s), s_i)$》 :
実現は $P_s \leftarrow P_s - \{(s, s_i)\}$ となる. 更に集合 s_i が s_i の親 s を代表していたときは s の代表 $r(s)$ および実現 I_s を更新しなければならない. すなわち $\neg int(i(s_i)) \ \wedge \ s_i \in r(s)$ のとき $r(s) \leftarrow r(s) - \{s_i\}$ および $I_s \leftarrow I_s \cup i(s_i)$.

《$ins(i(s), o)$》 :
$i(s)$ が内包的に実現されているときはなにもしない. 外延的に実現されているとき通常はオブジェクト o を記憶する. $I_s \leftarrow I_s \cup \{(s,o)\}$. ただしオブジェクト o が集合 s の代表中に含まれているときは逆に削除する. すなわち $^{\exists} s_i \in r(s) \ o \in i(s_i)$ のとき $I_s \leftarrow I_s - \{(s,o)\}$. ただし $i(s_i)$ は更新処理後のものとする. 《$del(i(s), o)$》 :
$i(s)$ が内包的に実現されているときはなにもしない. 外延的に実現されているときオブジェクト o を削除する. $I_s \leftarrow I_s - \{(s,o)\}$. 更に集合 s が s の i が親の i を代表しているときは親にオブジェクト o を記憶させる. すなわち $^{\exists} s \in r(\bar{p}(s))$ のとき $I_{\bar{p}(s)} \leftarrow I_{\bar{p}(s)} \cup \{(\bar{p}(s), o)\}$.

6. むすび

データベースでの集合の記憶および更新時の一貫性管理について議論した. 集合の記憶方法にはいくつかの方法があったが検索効率と更新効率を考慮した記憶方法の組合せの決定方法が必要である. また 1 つの集合に対して複数の分割を行いたい場合も考えられるので複数の階層の扱いも考える必要がある.