データの多重分類階層の構成

古川哲也

九州大学経済学部

データの多様化と増大にともない，大量のデータを整理し簡単に利用できるようにすることの重要性が高まっている。データを階層的に分類することでこのような要求に対応することができる。本稿では，データを複数の視点から階層的に分類する多重分類階層の構成について議論している。分類に用いるデータの性質ごとに階層を作り，それを合成することで，利用目的にあった分類階層を得る。また，分類階層を維持する上での望ましい性質を示し，それを満たさないときの対処法を提案している。分類階層を多重化することにより，その多様性や不均一性に対応でき，再構成も容易となる。

Organizing Multiple Classification Hierarchies of Data

Tetsuya Furukawa

Dept. Economic Eng., Kyushu Univ.

It becomes more important to arrange increasing data. Classification hierarchies can cope with this requirement. In this paper, we discuss how to organize multiple classification hierarchies, which classify data from multiple points of view. A hierarchies are created for each category of data and combined to generate a suitable classification hierarchy. Desirable properties to maintain hierarchies are also shown. The multiplication manages variety and unbalance of hierarchies and allows us to rebuild them easily.

1. はじめに

近年の高性能化とダウンサイジングにともない，コンピュータの利用は様々な分野へ拡大している。流通するデータも多様化しており，収集した大量のデータをコンピュータで管理し利用できるようにしなければならない。本稿では，収集したデータを整理し，複数の視点から階層的に分類する多重分類階層の構成について議論する。

大量のデータから必要なデータを取り出し利用できるようにするためには，データを分類しておくことが有用である。分類によってできたデータ集合の数が多すぎると，必要なデータを取り出す際の候補が多くなり利用しづらいものとなる。逆にデータ集合の数が少ないと，１つの集合に含まれるデータ数が多くなり，取り出したデータからさらに必要なデータを選ぶ処理を行わなければならない。すなわち，分類の粒度が問題となる。分類を階層的に行うことで，必要なデータ集合を取り出すことが容易となる。

１つの分類階層で複数の性質を用いて分類すると，どの性質を階層のどのレベルの分類に用いるかで適用順序ができる。階層における分類規準の順序が固定したものであれば，利用目的によっては使いづらいものとなる。しかし，考え得る順序の階層をすべて準備するのは現実的ではない。分類の規準となる性質ごとに階層を作り，実際に利用する階層はそれらを合成して作成することにより柔軟な分類階層の利用が可能となる。

階層的にデータの分類を行うときには，次の点を考慮しなければならない。

（１）分類の多様性　分類の定義方法は様々であり，階層中の同一レベルでも異なる手法で分類される場合もある。

（２）階層の不均一性　階層構造は，すべてのデータに対して同レベルで定義されるとは限らない。一部のデータに対してのみ階層が定義される場合もある。

（３）複数階層の支援　分類の規準はデータの利用目的によって異なるため，それぞれの目的に合ったデータの階層が必要である。すなわち，複数の分類階層を提供する必要がある。

（４）階層の再構成　データの挿入や削除，データに対する視点の変化に対応して階層構造を変更できなければならない。

データの階層に関する研究には，データの持つ属性値やデータ間の対応関係を用いてデータを階層的に出力したり，階層構造のビューを構築するものがある。非正規関係による利用者インタフェースの研究もこの種のものである[4, 6]。また，オブジェクト指向データベースは汎化階層を直接表現できる[3]。このような方法は，スキーマに基づいて階層構造を作り出すものである。しかし，スキーマに基づく階層構造の構成では，階層の多様性や不均一性に対応することはできない。データベースで集合自体を扱う研究にも様々なものがある[2, 7]が，値を集合値にまで拡張する研究がほとんどであり，集合の階層構造をデータベースで支援しようとするものは知られていない。データのクラスタリングに関する研究は，データベースのアクセスを高速化するための物理的な構成に関するものがほとんどである[9]。構造化していないデータベースに関する研究では，オブジェクトの対応関係や視点の多様性を用いて階層構造を形成している[1, 5]。

様々な手法で定義された階層を組み合せることにより複数の分類階層を柔軟に構成できる。個々の階層をデータと分離し独立なものとすることで階層の多様性や不均一性に対応し，その合成により利用目的に合った複数の分類階層を提供する。そのために，本稿ではデータの分類により階層がどのように生成されるのかについて議論し，分類階層を記憶する上での階層の望ましい性質を示す。さらに，実際に階層を記憶する方法とそれらを利用する方法を検討し，階層の合成法について議論する。

2. 分類による階層の生成

対象となるデータをオブジェクト o，分類によって生成されたオブジェクトの集合をグループ g とする。グループ g を構成するオブジェクトの集合を $m(g) = \{o_1, o_2, \cdots, o_n\}$ で表し，グループ集合 $\boldsymbol{g}$ に対して $m(\boldsymbol{g}) = \bigcup_{g \in \boldsymbol{g}} m(g)$ とする。また，すべてのオブジェクトからなる集合をグループ g_U とする。

［例１］　オブジェクトはいくつかの性質を持っており，それを用いて分類する。ある利用者は，g_U をまず g_a と g_b に分け，さらに，g_a を g_{a1} と g_{a23} に，g_b を g_{b1}, g_{b2}, g_{b3} に分けたとする（図１(a)）。ここで，g_{a23} は，g_{a2} と g_{a3} の和集合である。一方，オブジェクトを g_1, g_2, g_3 に分け，次に g_1 を

g_{a1} と g_{b1} に分け，g_2 と g_3 をまとめて g_{23} として扱う方がよいとする利用者がいるかもしれない（図1 (b)）。このように，利用者が必要とする階層構造では，分類に用いる性質の順序や階層のレベルの違いが生じる。□

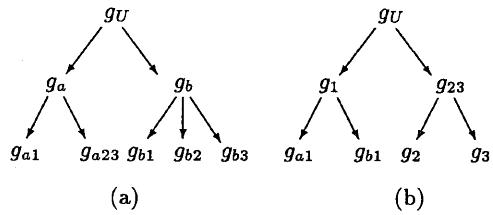

図1　データの分類

オブジェクト集合 $\boldsymbol{O}$ に対し，オブジェクトを分類してグループを作成する。グループ g を定義する方法には，次のものがある。

［外延的グループ］　オブジェクト集合 $\boldsymbol{O}$ に対し，要素を列挙して g を定義する。$m(g) = \{o_1, o_2, \cdots, o_n\}$ $(o_i \in \boldsymbol{O})$ である。

［内包的グループ］　オブジェクト集合 $\boldsymbol{O}$ に対し，要素の満たすべき性質によって g を定義する。$m(g) = \{o \mid P_g(o), o \in \boldsymbol{O}\}$ である。ここで，P_g は g に含まれるオブジェクトが満たす性質を表す述語である。

グループ g のオブジェクト集合はさらに分類される。このようにして生成された階層をグループの分割とよぶ。

［定義 1］　グループの集合 D_g=$\{g_1, g_2, \cdots, g_m\}$ で，g_i $(1 \leq i \leq m)$ が $m(g)$ に対して定義されているとき，D_g は g の分割であるといい，g を分割の親，g_iを分割の子という。□

［例 2］　すべてのオブジェクトからなるグループ g_U を3つのグループ g_1, g_2, g_3 に分類する。$\{g_1, g_2, g_3\}$ は，g_U の分割である（図2 (a)）。g_U は他の性質によって $\{g_a, g_b\}$ にも分割される（図2 (b)）。$\{g_1, g_2, g_3\}$ と $\{g_a, g_b\}$ が独立であれば，$g_1 \cap g_a \neq \emptyset$, $g_1 \cap g_b \neq \emptyset$ などとなる。□

逆に，すでに存在する複数のグループをまとめて1つのグループとする場合がある。グループに含まれるオブジェクトの性質により定義されていることから，内包的グループの一種とみなすことができる。

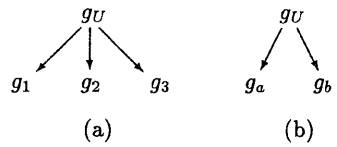

図2　グループの分割

［定義 2］　グループ g が $\{g_1, g_2, \cdots, g_m\}$ の和集合として定義されている，すなわち $m(g) = \bigcup_{i=1}^{m} m(g_i)$ であるとき，g は $S_g = \{g_1, g_2, \cdots, g_m\}$ の統合であるといい，g を統合の親，g_i を統合の子という。□

g' が g の分割 $\{g_1, g_2, \cdots, g_m\}$ の部分集合 $\{g_1, g_2, \ldots, g_k\}$ $(k < m)$ の統合であれば，$\{g', g_{k+1}, \cdots, g_m\}$ も g の分割である。

［例 3］　図2 (a) の分割で，g_2 と g_3 の分割を必要としない利用者がいたとする。$\{g_2, g_3\}$ の統合 g_{23} を導入する（図3）と，$\{g_1, g_{23}\}$ も g_U の分割となる。□

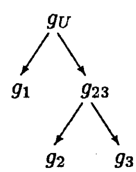

図3　グループの統合

グループは分割と統合を複数回適用して生成される。分割と統合により，グループの包含関係の階層ができる。g は g_i の親，または g_i の祖先 g' の親のとき g_i の祖先，g_i を g の子孫であるという。また，すべてのグループの集合を G とする。

3.　階層の完全性と単純性

分割における g と D_g や統合における g と S_g の関係は，グループの階層を維持管理するための重要な性質である。

［定義 3］　g の子グループの集合 $\boldsymbol{g} = \{g_1, g_2, \cdots, g_m\}$ は，$\bigcup_{i=1}^{m} m(g_i) = m(g)$ のとき g に対して完全であるといい，$\bigcup_{i=1}^{m} m(g_i) \subset m(g)$

のとき g に対して不完全であるという。また，$\boldsymbol{g}$ は $m(g_i)\cap m(g_j)=\emptyset\ (i\neq j)$ のとき単純であるという。 □

一般には，不完全な階層や単純でない階層が要求される場合がある。そのような階層を実現するために，論理的な構造と物理的な構造を分離する。

不完全な階層に対しては，どの子グループにも含まれないオブジェクトからなるグループを導入して対処する。

［定義 4］ g の子グループの集合 $\boldsymbol{g} = \{g_1, g_2, \cdots, g_m\}$ が不完全であるとき，$m(g_c) = m(g) - m(\boldsymbol{g})$ となるグループ g_c を g の $\boldsymbol{g}$ に対する補完という。 □

不完全であるように作成された階層であっても，補完を加えることによって完全な階層とすることができる。

グループの分割は，不完全であるものも存在するが，統合による親グループの生成では，子グループ集合は親グループに対して完全である。分割，統合ともに，親グループに対する子グループ集合は，単純でないものが存在し得る。

統合によるグループ g' がグループ g の分割 $D_g = \{g_1, g_2, \cdots, g_m\}$ の部分集合 $S_{g'}$ に対して定義されたとき，グループ集合 $D'_g = (D_g - S_{g'})\cup\{g'\}$ も g の分割となる。グループ g' が複数の分割の部分集合の和集合 $S_{g'} = \bigcup_{i=1}^{k} D'_{g_i}\ (D'_{g_i} \subseteq D_{g_i})$ の統合であり，分割の親グループで共通集合を持つものが存在するとき，すなわち，ある i, j に対し $m(g_i)\cap m(g_j) \neq \emptyset$ であるとき，g' は単純とはならないことがある。

単純でない階層に対しては，共通集合と差集合に分割した子グループ集合とすることで単純な階層とすることができる。

［定義 5］ g の子グループの集合 $\boldsymbol{g} = \{g_1, g_2, \cdots, g_m\}$ が単純でないとき，$m(g_i) \cap m(g_j) \neq \emptyset$ となる子グループ g_i, g_j を要素が $m(g_i)\cap m(g_j)$, $m(g_i) - m(g_j)$, $m(g_j) - m(g_i)$ となるグループで置き換える操作を繰り返し，単純な階層となったものを $\boldsymbol{g}$ の単純化という。 □

不完全な階層と同様に，作成された階層が単純でなければ，単純化することで容易な管理が可能となる。

共通集合が若干である場合は物理的に単純化することで対処できるが，共通集合が多い場合は，一般には分割が独立した概念を複合して行われている場合が多い。すなわち，共通集合を持たない $\boldsymbol{g}$ の最大部分集合を子グループ集合とする階層を複数作るべきである。

同様に，ある分割 D_g 中のグループ g_i の分割 D_{g_i}が g_i の定義とは独立であれば，この分割は g_i に対して行うべきではない。D_{g_i} と同様の規準による分割を D_g 中の他のグループに対しても行うことができるためで，この規準による分割は，D_g の親グループ g に対して行う方がよい。

［例 4］ グループ g の分割 $D_g = \{g_1, g_2\}$ で，g_1, g_2 を D_g とは独立した規準に従ってさらに分割したとする。その分割を $D_{g_1} = \{g_{1a}, g_{1b}\}$, $D_{g_2} = \{g_{2a}, g_{2b}\}$ とすると，g 以下の階層は図 4 (a) のようになる。D_{g_1} と D_{g_2} は同じ規準でそれぞれ g_1 と g_2 を分割しているので，g の分割 $D'_g = \{g_a = g_{1a}\cup g_{2a}, g_b = g_{1b}\cup g_{2b}\}$ を作ることもできる。D'_g の各グループに対して D_g と同様の規準で分割を行えば，$D_{g_a} = \{g_{1a}, g_{2a}\}$, $D_{g_b} = \{g_{1b}, g_{2b}\}$ となる（図 4 (b)）。2 つの分割を独立に行えるならば，g を親とする 2 つの独立した階層が得られる（図 4 (c)）。図 4 (a), (b) のどちらの階層も，図 4 (c) の 2 つの階層から作り出すことができる。

同様に，図 1 の階層も図 2 (a) 又は図 3 と図 2 (b) の 2 つの階層から作り出すことができる。 □

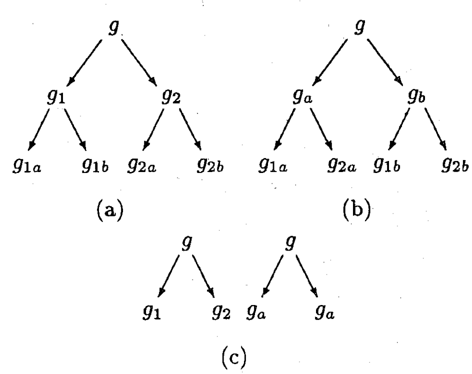

図 4 独立した階層の表現

図4の階層は，グループの和集合や共通集合を求めることによりどれからも他の2つの階層を作ることができる。しかし，D_g と D'_g は独立した規準の分割であるならば，(c) を記憶しておくべきである。(a) や (b) のような独立した規準による分割の合成による階層は，5節で示すように複合階層とする。

4. グループの記憶と管理

生成されたグループをどのように記憶し，記憶されたグループをどのように検索するかについて考察する。グループの表現には次の方法がある。

(1) データの分割による表現

属するグループごとにデータを分割する。関係モデルでは関係の横切りに，オブジェクト指向モデルではクラスの汎化階層に対応する。この方法では，独立した n 個の集合に対しては $O(2^n)$ 個の関係またはクラスが必要である，集合の作成，削除でスキーマの変更が必要である等の問題が生じる。例1の階層に対しては，g_{a1}, g_{a2}, g_{a3}, g_{b1}, g_{b2}, g_{b3} を作成する。

(2) 属性値による表現

オブジェクトの属性値として属するグループ名又はグループの識別子を持たせる。ある視点からオブジェクトがグループ $g_1, g_2, \cdots, g_m$ に分類されていれば，1つの属性でこれらのグループを表すことができる。例1の階層に対しては，{1, 2, 3} と {a, b} を取り得る値とする2つの属性が加えられる。しかし，独立した n 個のグループに対しては n 個の属性が必要であり，集合の作成，削除時にはスキーマの変更が必要となる。また，階層が不均一であればオブジェクトごとに必要な属性が異なることになる。利用目的によってはグループの作成や削除の可能性があり，実用的な方法とはいえず，固定的な階層のみに有効である。

(3) 索引による表現

グループ g について，g に含まれるオブジェクトの索引を作る。必要な情報はグループとそれに含まれるオブジェクトの対応関係のみである。関係モデルでは，グループ名とオブジェクトの識別子を属性とする関係を作る等によっても実現できる。

(4) 条件式による表現

グループが何らかの規則，例えば述語 P を真とするオブジェクトとして定義されていれば，P を記憶することによりグループを記憶することができる。グループに属するオブジェクトを求めるには，P による選択演算を行う。(2) も選択演算でグループに属するオブジェクトを求めることになるが，(2) ではスキーマとして集合が表現されているのに対し，この方法は集合とその条件式のみを記憶している点で異なる。

オブジェクトの追加や削除によってグループに含まれるオブジェクトの数が変わってくる。そのような場合には，著しくオブジェクトが増加したグループを分割したり，オブジェクトが減少したグループと他のグループとの統合が行われる。また，新たな概念の発生などによっても階層構造の再構成が必要となる。したがって，スキーマとしてグループの階層を記憶する (1), (2) の方法ではなく，データとは独立にグループの階層を記憶する (3), (4) の方法が適している。

分割や統合によって定義されたグループの階層を記憶する方法，およびそのグループに属するオブジェクトを求める方法として，次の3種類を考える。記憶する階層は単純かつ完全であるものとする。

［定義6］ グループ g のデータベース中での実現を $I(g)$ で，g の検索を $R(g)$ で表す。

索引法： $I(g) = \{(g, o) \mid o \in m(g)\}$

$R(g) = \{o \mid (g, o) \in I(g)\}$

述語法： $I(g) = (P_g, g')$

$R(g) = \{o \mid P_g(o), o \in R(g')\}$

間接法： $I(g) = \{(g, g') \mid g' \in \boldsymbol{g}\}$

$R(g) = \{o \mid o \in R(g_i), (g, g_i) \in I(g)\}$

（$\boldsymbol{g}$ は g の子グループ集合）

すべてのグループの実現を $I(G) = \{I(g) \mid g \in G\}$ で表す。 □

グループの実現は実際に記憶される情報を，検索は記憶された情報を用いてグループの要素を求める方法を示している。

一般に，外延的グループは索引法，内包的グループは述語法，統合によるグループの生成は間接法に対応するが，索引法はグループの生成過程によらずすべてのグループに用いることができるし，間接法も子グループを持つグループに適用できる。直接参照することが多いグループに対しては，述語法よりも索引法による実現の方が適している場合がある。子グループが索引法で実現されていれば，記憶効率の点から間接法による実現の

方がよいこともある。一方，索引法によって実現した場合には，保守の問題が生じる。

グループの生成過程に従って実現法を決めれば，すでに存在するグループを用いて新たなグループが生成されるので，グループの検索ができないような実現にはならない。しかし，間接法を統合以外に適用すると，グループの検索ができなくなる場合がある。述語法によるグループの実現は祖先のグループを参照するのに対し，間接法によるグループの実現は子孫のグループを参照する。参照に閉路ができると，あるグループのオブジェクト集合を求めることができない。

［定義 7］ G の実現 $I(G)$ における従属グラフは，G のグループと 1 対 1 に対応する節点の集合と，$R(g_i)$ による $R(g_j)$ の参照を表す g_i に対応する節点から g_j に対応する節点に向かう有向枝の集合からなる有向グラフである。 □

［補題 1］ G の実現 $I(G)$ における従属グラフが非巡回であれば，任意のグループ g に対し $R(g)$ を求めることができる。 □

記憶する階層は，作成された階層を完全かつ単純となるように変更したものとする。それを実際に利用する階層構造に関連づける。記憶する階層を物理階層，利用する階層を論理階層として区別する。論理階層の各グループは，必ず物理階層のいくつかのグループの和集合となるように関連づけることができる。

5. 複合階層の構成

論理階層を合成して利用目的に合った階層を構成する。データの分割方法が複数ある場合，目的に合った順序で階層構造を生成する。

定義されたグループによる階層をグループ階層とする。

［定義 8］ グループ集合 $\boldsymbol{g}$ に対し，$\boldsymbol{g}$ が g の分割，または g が $\boldsymbol{g}$ の統合のとき，$(g,\boldsymbol{g})$ を原子階層，g を原子階層の親，$\boldsymbol{g}$ の要素を原子階層の子という。原子階層の集合 $H=\{(g_i,\boldsymbol{g}_i) \mid 1 \le i \le k\}$ は，その親がすべて異なり $(g_i \neq g_j\ (i \neq j))$，$H$ 中の他の原子階層の子とはならない親が唯一であるとき，グループ階層であるという。

H 中の原子階層の親グループとはならない子グループを H の子グループといい，H の子グループの集合を $leaf(H)$ で表す。他の原子階層の子グループとはならない親グループを H の根グループといい，$root(H)$ で表す。 □

［例 5］ 論理階層を図 2 (b) および図 3 とすると，原子階層は $h_1(g_U,\{g_1,g_2,g_3\})$, $h_2(g_U,\{g_a,g_b\})$, $h_3(g_U,\{g_1,g_{23}\})$, $h_4(g_{23},\{g_2,g_3\})$ である。グループ階層は，$H_1=\{h_1\}$, $H_2=\{h_2\}$, $H_3=\{h_3\}$, $H_4=\{h_4\}$, $H_5=\{h_1,h_3\}$ となる。 □

複合階層は，作業環境に応じた順序で複数個のグループ階層を組み合せて定義する。グループ階層の合成を次のように定義する。

［定義 9］ グループ g_0 によるグループ g の制限 g_0*g は，$m(g')=m(g_0)\cap m(g)$ となるグループ g' である。グループ集合 $\boldsymbol{g}$ に対し，$g_0*\boldsymbol{g}=\{g_0*g \mid g\in\boldsymbol{g}\}$ とする。原子階層 $(g,\boldsymbol{g})$ の制限 $g_0*(g,\boldsymbol{g})$ は，階層 $(g',\boldsymbol{g}')$ $(g'=g_0*g, \boldsymbol{g}'=g_0*\boldsymbol{g})$ である。

グループ g_0 によるグループ階層 H の制限 g_0*H は，グループ階層 $\{(g,\boldsymbol{g}) \mid (g,\boldsymbol{g})=g_0*(g',\boldsymbol{g}')(g'\neq root(H)),(g,\boldsymbol{g})=(g_0,g_0*\boldsymbol{g}')(g'=root(H)),(g',\boldsymbol{g}')\in H\}$ である。

グループ階層 H_1,H_2 の合成 H_1*H_2 は，$H_1\cup\bigcup_{g\in leaf(H_1)} g*H_2$ である。 □

複合階層は，複数個のグループ階層の合成により構成される。定義より明からなように，グループ階層の合成演算は結合則が成り立つ，すなわち $(H_1*H_2)*H_3=H_1*(H_2*H_3)$ である。したがって，複合階層の定義では，グループ階層の順序のみが必要であり，合成演算の適用順序は問題とならない。

［定義 10］ 複合階層 $\mathcal{H}$ は，階層を制限するグループ g_0 とグループ階層 H_i $(1\le i\le n)$ の列 $(g_0,H_1,H_2,\cdots,H_n)$ で定義される制限された原子階層の集合 $\{(g,\boldsymbol{g}) \mid (g,\boldsymbol{g})\in g_0*H_1*H_2*\cdots*H_n\}$ である。 □

複合階層は仮想的なグループ階層なので，すでに定義された他の複合階層を用いたの再帰的な定義も可能となる。また，複合階層の定義で，階層を制限するグループを全オブジェクト集合 g_U とすれば，複合階層は，H_1 の親グループのオブジェクト集合に対する階層構造となる。

グループ階層 H_1 に対し，$leaf(H_1)$ に含まれ

るすべてのグループに他のグループ階層 H_2 を適用するのではなく，その一部のグループのみを H_2 の構造としたい場合がある。H_2を適用するグループを特定するため，どのグループに H_2 を適用するのかを記述するように，複合階層の定義を拡張するのは軽微な定義の変更である。

［例 6］ 図 1 (a) の階層構造は，複合階層 $H_2(g_{23} * H_3, g_b * H_1)$ として定義できる。また，図 1 (b) の階層構造は，$H_5(g_1 * H_2)$ となる。 □

複合階層 $\mathcal{H} = (g_0, H_1, H_2, \cdots, H_n)$ で，グループ階層 H_i $(1 \leq i \leq n)$ の親グループを g_i とする。$\mathcal{H}$ 中のグループ g は，それがグループ階層 H_i 中のグループ g' を制限することによって生成されたものであれば，それに属するオブジェクト集合を $R(g) = g_0 \cap R(g_1) \cap R(g_2) \cap \cdots \cap R(g_{i-1}) \cap R(g)$ で求めることができる。

複合階層中のグループ g へのオブジェクトの挿入と削除は，g が H_i 中のグループ g' を制限したものあれば，g' に対するオブジェクトの挿入と削除となる。

このように，グループ階層はオブジェクトが実際に分類された階層構造であり，複合階層は分類の仮想的な構造である。

6. むすび

オブジェクトとして表されたデータ集合に対して，全体を階層構造で表す多重分類階層の構成について検討した。複合階層は，独立に生成されたグループ階層を組み合せることによって定義され，同じオブジェクト集合に対して利用に適合した異なる階層を構築することができる。

親グループが子グループの和集合に一致する完全性や，子グループ間に共通集合がない単純性を満たさない階層は，完全かつ単純となるように変換し，グループ階層に用いる論理階層と対応を取ることで利用可能となる。また，多重分類階層を作成する対象のデータはどのようなものでもよく，データの形式を問わない。

階層構造の定義により，グループ間の包含関係が規定される。また，グループの分割が完全分割であれば，グループ間には共通集合がないという制約が生じる。独立に定義された階層でも，なんらかの関係がある場合もある。このような制約はグループの実現を保守する際に積極的に利用すべきものであり，階層構造の健全性を保持する上で考慮しなければならないものである。

複合階層においても，任意のグループ階層の列が定義として許されるわけではない。定義によっては以降のグループ階層が意味をなさないものもある。複合階層によるデータの検索を考えると，複合階層の定義を考慮して階層構造の実現法を決める必要がある。

謝辞 本研究の一部は文部省科学研究費補助金特定領域研究「高度データベース」の援助を受けている。

参考文献

[1] Buneman, P., Davidson, S., and Hillebrand, G., "A Query Language and Optimization Techniques for Unstructured Data," *Proc. ACM SIGMOD Int'l Conf. on Management of Data*, pp. 505-516, June 1996.

[2] d'Amore, F. and Liberatore, V., "The List Update Problem and the Retrieval of Sets, " *Proc. Third Scandinavian Workshop Algorithm Theory*, pp. 178-192, 1992 (*Lecture Notes in Computer Science*, Vol. 621, Springer-Verlag).

[3] den Bussche, J. V. and Gucht, D. V., "A Hierarchy of Faithful Set Creation in Pure OODB's," *Proc. Int'l Conf. on Database Theory*, pp. 326-340, 1992 *Lecture Notes in Computer Science*, Vol. 646, Springer-Verlag).

[4] Levene, M., "The Nested Universal Relational Database Model," *Lecture Notes in Computer Science*, Vol. 595, Springer-Verlag, 1992.

[5] 森下淳也, 上島紳一, 大月一弘, 杉山武司, "階層構造グラフを用いた半構造化データの段階的構造化手法の提案", 情報処理学会研究報告, 96-DBS-111-2, 平成 9 年 1 月.

[6] Roth, M., Korth, H., and Silberschats, A., "Extending Algebra and Calculus for Nested Relational Databases," *ACM Trans. Database Syst.*, Vol. 13, No. 4, pp. 389-417, April 1988.

[7] Rundensteiner, E. A. and Bic, L., "Set Operations in Object-Based Data Models," *IEEE*

Trans. Knowledge and Data Eng., Vol. 4, No. 4, pp. 382-398, August 1992.

[8] 山崎康雄，古川哲也，島崎眞昭，“データベースを用いたデータ解析のための集合操作”，情報処理学会研究報告，93-DBS-94-5，平成5年7月．

[9] Zhang, T., Ramakrishnan, R., and Linvny, M., “BRICH: An Efficient Data Clustering Method for Very Large Databases,” *Proc. ACM SIGMOD Int'l Conf. on Management of Data*, pp. 103-114, June 1996.