多次元分類：木構造分類とキーワード分類の複合的アプローチ

掛下哲郎[†] 原槙稔幸[††]

電子的に流通する様々な情報（エンティティ）の共有および活用を促進するためには，エンティティの分類および整理を円滑に行う必要がある．本論文では，このために多次元分類方式を提案する．提案方式では互いに独立した概念ごとに平衡した木構造を構成し，それらを組み合わせることで複数の分類基準に対応した情報整理を行う．本方式により，格納されているエンティティの概観や検索コストの低減が可能になる．また，木構造の管理コストも低い．さらに，同一のエンティティ集合に到達する検索経路を多数提供できるため，利用者の様々な検索要求に柔軟に対応できる．本論文では多次元分類方式のコスト評価モデルを作成し，モデル評価実験を行う．

Multidimensional Information Arrangement: A Hybrid Approach of Tree and Keyword Based Classifications

Tetsuro Kakeshita[†] and Toshiyuki Haramaki[††]

Classification and arrangement of digital information (entity) are essential to encourage entity sharing. We propose multidimensional information arrangement for this purpose. Corresponding to an independent concept, a balanced tree is organized. Entities are classified using multiple trees. Overview of the stored entities becomes possible and the retrieval cost is reduced as well as the management cost. The retrieval flexibility also becomes high since number of search paths are available to the same set of target entities. We develop a cost model for the proposed method and perform evaluation through experiment.

1. まえがき

各種デジタルメディアやインターネットの普及にともない，電子的に流通する情報が増大している．情報の洪水に押し流されることなくこれらの情報を十分に活用するためには，流通する情報の中から重要性の高いものを選択し，適切に分類整理することが不可欠である．このうち，情報の重要性を客観的に判断するためには各種の情報フィルタリング手法が提案されている[1)]．本論文では価値判断を済ませた情報に対する効果的な分類整理を行うために多次元分類方式を提案する．本方式は，複数の独立した木構造を用いてデータベースに格納された情報の一覧を様々な視点から可視化する手法でもある．

Halaszが提唱した次世代ハイパーメディアのための課題[2)]と，Papazoglouらが提唱した情報を分類する際の必要条件[3)]をまとめると以下の6項目があげられる．(1) データ分類項目を自由に定義できる．(2) データを随時追加できる．(3) 分類に使用する条件に一貫性がある．(4) 様々な形式のデータに対応する．(5) 検索の指針を利用者に提示できる．(6) 分類したデータを容易かつ柔軟に検索できる．これらの必要条件は利用者に対する利便性の提供に対応している．しかし，効果的な情報整理を行うためには，管理作業において自動化できない部分の低減が不可欠である．そこで本論文では，(7) 情報分類に要する管理コストが低いことを上記の必要条件に追加し，利用者レベルと管理者レベルでバランスのとれた情報分類方式を構築する．

以下，2章では既存の情報整理方式について上記の7つの条件に基づいて論じる．3章では多次元分類方式を定義する．多次元分類方式における検索コストおよび検索柔軟性の評価は4，5章で，管理コストの評価は6章でそれぞれ行う．以上の結果に基づいて，7章で多次元方式の適切な構成法を導き，8章では評価実験を行う．

† 佐賀大学理工学部知能情報システム学科
Department of Information Science, Faculty of Science and Engineering, Saga University
†† 大分大学工学部知能情報システム工学科
Department of Computer Science and Intelligent Systems, Faculty of Engineering, Oita University

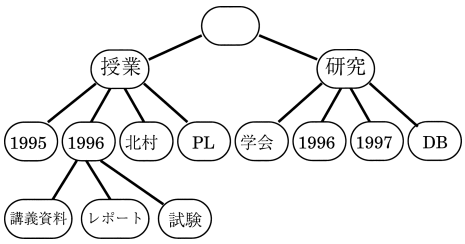

図 1 一貫性のない分類項目
Fig. 1 Inconsistent classification criteria.

2. 関連研究

階層構造を用いた分類は，ディレクトリ構造および Yahoo! Japan 等の WWW サーチエンジンとして広く使われている．この方式では単一の階層構造を用いるため，非常に単純かつ実現が容易である．しかし，**図 1** のように複数の分類項目を同一の階層構造上に配置した場合，階層上での分類の一貫性が保たれない場合が発生する（例：1996 年と北村）．その結果，必要とするデータを検索する際に後戻りが発生するため，利用者による検索の手間が増大する．また単一の階層構造を用いた場合には，データの検索経路は木構造の根から葉に向かう単一の経路に限定される．各利用者は，同一データを検索する際にも目的に応じた順序で条件指定を行い，中間結果を確認しながら検索を進める場合が多い．単一の階層構造はこのような要求には対応できない．複数の階層構造を用いれば上記の問題点を解決できる可能性はあるが，木構造の構成を系統的に行わなければ，利用者および管理者の双方に対して複雑さをかえって増大させる可能性が高い．

これに対して，キーワードを用いた分類（例：各種文献検索システム）は検索に対する自由度が高い．しかし，情報に付加されているキーワードと利用者が指定したキーワードの間で類似性の判断を必要とするため，情報検索および登録の際に検索洩れが発生する．キーワードが多義性を持つ場合には，この問題がさらに複雑化する．また，利用者には階層構造等がまったく提示されないため，データベースに登録されている情報の全体を概観できず，検索の指針となる情報も提示できない．

上記の問題点を緩和するために，多くの WWW 検索エンジンは階層構造とキーワード検索を組み合わせている．しかし，その多くは単一の階層構造を用いている．また，利用者による適切なキーワード指定を支援する機構は提供されていない．

植田らは情報を柔軟に分類するためにフレーム関係軸モデルを，情報を可視化する機構としてハイパーチャートをそれぞれ提案し，CastingNet でそれらを実装した[4]．このモデルでは情報（フレーム）と分類項目（関係軸）の間の対応を写像によって定義しているが，情報の追加・削除にともなう写像の保守コストに関する考察は行われていない．また，格納されている情報の全体構造を概観できないため，情報の検索指針を提示できない．

金田はキーワードを指定して行った検索結果に対して利用者が指定した軸を用いてソート（および絞り込み）して提示する手法を提案している[5]．これと同様に検索結果に対してクラスタリングを行ってから利用者に提示する方式も知られている[6),7]．しかし，これらの方式では利用者が検索キーを指定する際の支援機構が用意されていない．また，検索対象によらず検索コストを一定に保つための機構も含まれていない．したがって，本研究とは研究の方向性が異なる．

3. 多次元分類方式

分類の基本的原則は以下のように要約される[8]．

一貫性の原則 1 つの分野を 1 段階区分する区分原理はただ 1 つである．

排他性の原則 区分されたものは相互に排他的でなければならない．

充足性の原則 区分されたものは全体を覆わなければならない．

漸進の原則 区分は上位概念から下位概念へ順序だてて進み，飛躍してはならない．

2 章で述べた既存の方式における問題点を解決するために，本論文では上記の原則に従って多次元分類方式を定義する．

多次元分類方式は互いに独立した複数の木構造を用いて情報を分類する方式である．分類対象となる情報をエンティティと呼ぶ．エンティティの例としては，WWW ページ，PostScript/PDF ファイル，電子メール/ネットニュースの記事，URL，電子メールアドレス等がある．多次元分類では，分野，日付，作成者等のように互いに独立した概念を分類軸とし，これらを属性，属性の値を属性値と呼ぶ．同一属性の属性値間には is-a 関連（属性値 a の表す概念が属性値 b の表す概念を包含するならば b is-a a）または排他的関連（属性値 a，b の表す概念が共通部分を持たない）がなければならない．属性値間の is-a 関連を枝と見なすと，各属性について属性値を節点とする木構造を構成できる（**図 2**）．図 2 の属性「出典」および「作成者」には逆さの木を使用している．また図 2 は ‘言語理論’，‘DB’ 等のように複数の属性値（例：授業と研

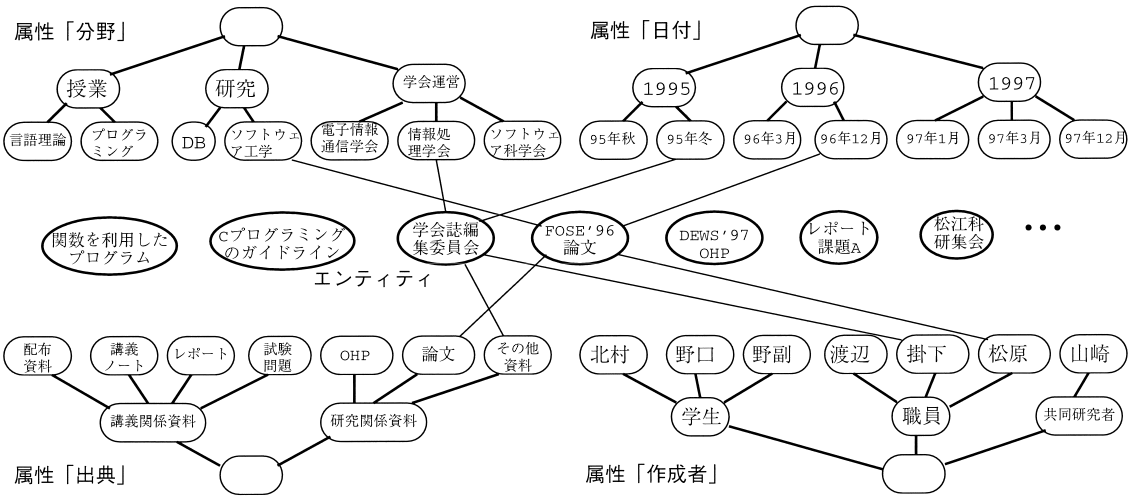

図 2 多次元分類方式
Fig. 2 Multidimensional information arrangement.

究）の子属性値となり得るものも含むが，親属性値を指定することでそれらの意味を限定する．各木構造において，根節点と葉節点をそれぞれ根属性値，葉属性値と呼ぶ．根属性値から葉属性値に至る経路に含まれる節点数の最大値を木構造の高さと呼ぶ．また，ある属性値 a に対して，根属性値から a に至る経路に含まれる節点数を a のレベルと呼ぶ．

エンティティは属性値と対応付ける（一般に多対多対応）ことで分類される．これにより，様々な形式のデータ分類に対応できる．また，属性値 a と対応するエンティティは a を汎化した属性値 a' とも対応すると定義する．エンティティの検索はいくつかの属性値を指定し，それらすべてと対応するエンティティを検索することで行われる．

以上の定義により，以下の特徴が得られる．

- 各属性の木構造は一貫性を持つ．
- 一貫した木構造を用いることで，情報の検索指針を利用者に提示できる．また，情報の概観が容易に行える．
- 複数の木構造を組み合わせて検索を行うことで，利用者の様々な検索要求に柔軟に対応できる．

4. 多次元分類における検索

本章では，多次元分類方式における検索モデルを示し，利用者が負担すべき検索の手間（検索コスト）を評価する．また，木構造に対する制約条件を定義する．これにより，検索対象エンティティによる検索コストのばらつきを最小限にする．

4.1 検索モデル

多次元分類における検索は，各属性について木構造の根属性値が選択された状態から開始する．利用者はいずれかの属性を選び，現在選択されている属性値の子または親属性値を新たに選択する．これに対応してシステムは各属性において選択された全属性値と対応するエンティティを検索して利用者に提示する．利用者は検索結果に基づいて属性値の選択を変更する．以上の検索作業は利用者が望むエンティティが発見されるまで繰り返される．属性値間の is-a 関連と排他的関連により，利用者は多くの場合に目的とする葉属性値まで後戻りすることなく到達できる．

現在選択されている属性値の子属性値を選択する作業は，候補エンティティの絞り込みに対応する．利用者に提示されているエンティティ数が多すぎる場合には，これによって検索結果を絞り込む．逆に，親属性値の選択は検索条件の緩和に対応する．これらを行う際に利用者は任意の属性を選べるため，検索における自由度が高い．

例 1 図 2 にエンティティと属性値の対応関係の一部を示す．属性「分野」,「日付」,「出典」,「作成者」において属性値 ‘ソフトウェア工学’, ‘1996’, ‘論文’, ‘松原’ をそれぞれ指定すると，‘FOSE’96 論文’ を含むエンティティ集合 S が検索される．S の要素数が多すぎる場合には，‘1996’ の代わりにその下位属性値 ‘96 年 12 月’ 等を指定すると S の部分集合が検索される．逆に，S 中に所望のものが存在しなかった場合には，属性値をより上位のものに変更する．たとえば，‘ソフトウェア工学’ を ‘研究’ に変更することで，検索され

表 1 コスト評価に用いる記号の定義
Table 1 Parameters for cost estimation.

記号	定 義
A_i	i 番目の属性
c_i	A_i の属性値が持つ子属性値数の最大値
h_i	A_i の木構造の高さ
K_i	A_i の葉属性値数
m_i	A_i の葉属性値に対応するエンティティ数の最大値
N	属性数
M	エンティティ数
R	各属性において葉属性値を 1 つずつ指定した場合に検索されるエンティティ数の最大値

るエンティティ集合は S を含む集合となる． □

多次元分類では，属性値の木構造を用いることで利用者が指定する属性値（キーワード）の範囲を限定する．これによって効率的な情報検索を行うための利用者支援機構を提供している．

4.2 検索コスト

多次元分類における検索は，属性 $A_1, \cdots, A_N$ において葉属性値 a_i $(i = 1, \cdots, N)$ を 1 つずつ指定し，指定されたすべての属性値に対応するたかだか R 個のエンティティを順に検索することで行われる．これを実行するために利用者が行う作業の手間を検索コスト SC と定義する．また，コスト評価を行う際に用いる記号をまとめて**表 1** に示す．

利用者が検索を行う際には，各属性値の木構造の各レベルにおいて，目的とする子属性値を指定する必要がある．多次元分類方式では，親子属性値間には is-a 関連が，兄弟属性値間には排他的関連がそれぞれ存在するため，子属性値の指定が曖昧さなく行える．したがって，正しい葉属性値を指定するために要する利用者の手間は，木構造の高さと各レベルにおける子属性値数の積に比例する．また，検索後に利用者が行うエンティティの選択コストは R に比例する．以上の考察から次式が導出できる☆．

$$SC = \sum_{i=1}^{N} c_i h_i + R$$

4.3 木構造に対する制約

検索コスト SC を均一化するためには，(1) 葉属性値のレベル，(2) 葉以外の属性値の子属性値数，(3) 葉属性値に対応するエンティティ数を各属性ごとに均等化すればよい．このうち (1) と (2) を均等化するために以下の制約条件を導入する．

(1) 属性 A_i の木構造は平衡している．
(2) 葉以外の属性値（中間属性値）が持つ子属性値数は $\lceil c_i/2 \rceil$ 以上 c_i 以下である．

一方，葉属性値に対応するエンティティ数を属性ごとに均等化することにより，各属性について同一レベルの属性値を選択した場合に検索されるエンティティ数が等しくなる．ここで，属性ごとに選択された属性値のレベルが異なっていてもよい．これにより，検索対象のエンティティによらず絞り込み効果が均等になるため，様々な検索ニーズに対して公平なサービスを提供できる．さらに，属性値の木構造に対する平衡条件と組み合わせることで，木構造が当該属性における属性値の分布を強く反映する．したがって，属性値の木構造を参照することでエンティティ全体を容易に概観できる．以下はそのための制約条件である．

(1) 属性 A_i の各葉属性値に対応するエンティティ数は m_i 以下である．
(2) 属性 A_i の葉属性値のうち兄弟関係にあるものは次のいずれかの条件を満足する．(2-a) それぞれの葉属性値に対応するエンティティ数は $m_i/2$ 以上である．(2-b) 葉属性値に対応するエンティティの総数は $(c_i - 1)m_i$ 以上である．

上記の制約は 6 章のアルゴリズムを適用することで守られる．なお，議論を単純化するために以下の制約を設ける．

(1) エンティティに対応する A_i の葉属性値は，各属性について 1 個である．中間属性値のみと対応しているエンティティは存在しない．

以下の議論は，本節で設けた制約に基づいて行う．

5. 検索柔軟性

多次元分類方式は，整理されたエンティティに対する複数利用者間での共有を想定している．この場合，同一のエンティティ集合を検索対象としていても，検索手順は利用者ごとに異なることが多い．たとえば，最初のエンティティ絞り込みを分野情報を用いて行う利用者と日付情報を用いて行う利用者が存在する．また，同一利用者であっても，検索対象エンティティによっては異なる検索手順を採用する場合もある．情報検索の際には，これらの要求に柔軟に応える必要がある．そのために検索柔軟性を定義する．

検索柔軟性は葉属性値 $a_1, a_2, \cdots, a_N$（a_i は属性 A_i の属性値）を指定してエンティティを検索するためにとりうる検索手順の数で定義する．検索手順は各 A_i の根から葉 a_i に至る属性値の系列に対応する．多次元分類方式では，各属性に含まれる属性値は先

☆ $c_i h_i$ と R の単位は異なるため比例定数が必要であるが，本論文の議論では比例定数は重要でないため省略して記す．他のコスト評価式についても個別に明示したうえで同様の省略を行う．

表 2 検索柔軟性の値
Table 2 Some values of retrieval flexibility.

h_i	N 3	4	5	6
2	90	2,520	1.13 e+5	7.48 e+6
3	1,680	3.69 e+5	1.68 e+8	1.37 e+11
4	34,650	6.31 e+7	3.06 e+11	3.25 e+15
5	7.57 e+5	1.17 e+10	6.23 e+14	8.88 e+19

祖から順に指定する必要があるが，異なる属性に含まれる属性値の指定順序は任意である．定義より a_i は $h_i - 1$ 個の先祖属性値を持つため，属性値指定は長さ $h_1 + h_2 + \cdots + h_N$ の属性値の列となる．この中で，A_1 の属性値に対する指定は，${}_{h_1+h_2+\cdots+h_N}C_{h_1}$ 通りだけ可能である．同様に，A_2 の属性値に対する指定は ${}_{h_2+\cdots+h_N}C_{h_2}$ 通りだけ可能である．以上の考察から，検索柔軟性は以下の式で表される．

$$\prod_{i=1}^{N-1} \begin{pmatrix} \sum_{j=i}^{N} h_j \\ h_i \end{pmatrix}$$

この式の値は属性数 N や属性値の木の高さ h_i の増加にともなって急速に増大する（**表 2** 参照）．c_i を小さくすることで，h_i を大きくできることにも注意されたい．このため，小規模な木構造を用いた場合でも十分高い検索柔軟性を提供できる．

6. 多次元分類におけるエンティティ管理

情報整理方式にはデータを随時変更できることが要求される．多次元分類における属性値の木構造には，4.3 節で導入した制約が課されている．これらの制約を満たしつつエンティティの追加削除を実現するためには，属性値の木構造を再構成する必要がある．そこで，B 木の再構成アルゴリズムを基本とした再構成アルゴリズムを提案する．これにより，属性値の木構造に対する再構成の判定と再構成範囲の決定が自動化できる．また，エンティティ登録および木構造の再構成に要するコスト（管理コスト）を評価する．

6.1 エンティティの登録とそのコスト

エンティティ e を登録するためには，各属性において属性値との対応を定義する必要がある．このために，木構造の根属性値から葉属性値に至る各レベルで最も概念の近い属性値を検索する．いずれかのレベルで検索された属性値が e に対応しない場合には，当該属性値の概念を拡大する必要が生じる．概念の近い属性値が複数存在する場合には，対応しているエンティティ数が最も少ないものを選択する．

e との対応関係を定義した結果，葉属性値 a に対応しているエンティティ数が m_i を超えた場合には，以下のアルゴリズムを用いて a を分割する．葉属性値に対応するエンティティ数の上限と，葉でない属性値が持つ子属性値数の上限が異なるため，B 木の再構成アルゴリズムと比較して複雑である．

(1) 属性値 a を 2 つの排他的な属性値 a_1，a_2 に分割する．ここで，a の概念は a_1，a_2 の概念を合併したものと一致しなければならない．a_1，a_2 はそれぞれ $m_i/2$ 個のエンティティと対応付ける．

(2) $m_i/2$ 個以下のエンティティと対応している a の兄弟属性値☆が存在するならば，それと a_1（または a_2）を合併する．

(3) a の親属性値 a_p の子属性値数が c_i 以下ならば終了する．

(4) a_p の持つ子属性値数が c_i を超えた場合，a_p を互いに排他的な属性値 a_{p1}，a_{p2} に分割する．a_{p1}，a_{p2} は，それぞれ $\lceil c_i/2 \rceil$ 個の子属性値を持たなければならない．

(5) 必要に応じてステップ (3)，(4) の分割処理を再帰的に繰り返す．

ステップ (2) は，兄弟関係にある葉属性値に対応するエンティティ総数が $(c_i - 1)m_i$ 以上の場合に必要となる．この条件のもとでは，$m_i/2$ 個以下のエンティティと対応する属性値は兄弟属性値中にたかだか 1 個しか存在しない．したがって，ステップ (1) および (2) により葉属性値に対応するエンティティ数に関する制約がつねに満たされる．また，ステップ (3) および (4) により木構造の平衡条件および子属性値数に関する制約が満たされる．

上記アルゴリズムの実行に際しては，属性値の分割および子属性値（またはエンティティ）との対応付けを管理者が行う必要があるが，分割すべき属性値の指示と木構造の再構成はコンピュータ側で行える．管理者が行うべき作業量を定量的に評価するために，管理コストはコンピュータによる自動化ができない部分について計量する．

e に対応する葉属性値を検索するコストは $\sum_{i=1}^{N} c_i h_i$ に比例する．また，属性 A_i において葉属性値の概念を拡張すると，属性値間の is-a 関連を守るために先祖属性値に対しても概念の拡張が必要になる．各属性値の概念拡張に要するコストは定数なので，A_i の属性値の概念を拡張するためのコストは h_i に比例する．以上より，属性値の概念を拡張するためのコスト

☆ A_i の木構造において a の兄弟節点と対応する属性値．

は $\sum_{i=1}^{N} h_i$ に比例するが，検索コストよりオーダーが小さいため，以下では考慮しない．これらをまとめると，属性値分割が発生しない場合のエンティティ登録コストは以下の式で表される．

$$\sum_{i=1}^{N} c_i h_i$$

木構造の再構成アルゴリズムを用いて A_i の葉属性値を分割するコストは，以下の式で評価できる．

$$m_i + c_i h_i$$

第1項は分割された属性値とエンティティの対応付けコストである（ステップ(1)）．第2項は属性 A_i の木構造に対する再構成コストである（ステップ(4)）．葉属性値の分割は，m_i 個のエンティティが追加されるたびにたかだか1回発生する．

葉属性値を分割する方法は ${}_{m_i}C_{m_i/2}$ 通りあるため，兄弟属性値との関係を十分考慮して将来の合併や分割が容易になる．これを行うためのコストは m_i に比例するが，葉属性値に対応するエンティティが多い場合には，その一部をランダムに選ぶことで分割候補の見当をつけられる．したがって，葉属性値の分割方針を決めるためのコストは m_i 以下になる．このため，葉属性値の分割コスト評価式では対応付けコストに含めている．

6.2 エンティティの削除とそのコスト

エンティティの削除にともなって，葉属性値に対応しているエンティティ数が制約条件を満たさなくなった場合には，以下のアルゴリズムに従って属性値を合併する．

(1) 兄弟関係にある葉属性値集合に対応するエンティティ総数が $(c_i - 1)m_i$ 以上ならば，当該属性値から対応関係を削除して終了する．
(2) 兄弟関係にある葉属性値集合に対応するエンティティ総数が $(c_i - 1)m_i$ 未満ならば，対応エンティティ数が最も少ない属性値 a を消去する．a と対応していたエンティティは，他の兄弟属性値を拡大して対応付ける．
(3) a_p の子属性値数が $\lceil c_i/2 \rceil$ 以上ならば終了する．
(4) a_p の子属性値数が $\lceil c_i/2 \rceil$ 未満かつ a_p の兄弟属性値が $\lceil c_i/2 \rceil$ 個の子属性値を追加保持可能ならば，a_p をその兄弟属性値と合併する．
(5) a_p の子属性値数が $\lceil c_i/2 \rceil$ 未満かつ a_p の兄弟属性値が $\lceil c_i/2 \rceil$ 個の子属性値を追加保持できないならば，兄弟属性値の持つ子属性値のいずれかを a_p に移動する．
(6) 必要に応じてステップ(3)〜(5)の合併処理を再帰的に繰り返す．

ステップ(1)，(2)により葉属性値に対応するエンティティ数に関する制約がつねに満足される．ステップ(2)の条件より，消去される葉属性値 a に対応していたエンティティ数は兄弟属性値に追加保持できるエンティティ数と等しい．ステップ(5)の条件が成立するとき，a_p の兄弟属性値はいずれも $\lceil c_i/2 \rceil + 1$ 個以上の属性値を持つ．したがって，いずれの兄弟属性値から子属性値を選択しても木構造の平衡条件は守られる．

このアルゴリズムは併合すべき属性値の指定と木構造の再構成を自動化できる．したがって，管理者は併合により得られた属性値の名前付けおよび子属性値（またはエンティティ）との対応付けだけを行えばよい．

通常のエンティティ削除時にはステップ(1)だけが実行される．これはコンピュータ側ですべて実行できるため，管理コストは0である．木構造の各レベルで最大 c_i 個の兄弟属性値間での合併が行われる可能性を考慮すると，属性 A_i における葉属性値の合併コストは以下の式で評価できる．

$$1 + c_i h_i$$

上式でも第1項はエンティティと属性値の対応付けコスト（ステップ(2)）であり，第2項は属性値の木構造の再構成コスト（ステップ(4)）となる．分割アルゴリズムでは，エンティティと属性値の対応付けを個別に行う必要があるが，合併の際には一括して対応付けが行えるためコストが異なる．また，属性値の併合が発生する頻度は，m_i に比例する．

属性値分割の際に兄弟属性値との関係を考慮しておくことで，属性値の合併を容易に行える．なお，属性値を分割する際に木構造から削除した属性値を保存しておくと，将来の属性値合併の際に参考にできる．同様に，属性値合併の際に元の属性値を保存しておくと，将来の属性値分割の際に参考にすることもできる．

本節のアルゴリズムではエンティティの削除にともなって属性値が削除されることがある．これに対しては過去に得られた属性値を削除しないという考え方もありうる．しかし，余分な情報を保持することで管理コストがより高くなる．また，これらの情報を保持しなくてもエンティティ集合の概観や絞り込みは可能である．

7. 多次元分類の適切な構成法

各属性において葉属性値を指定した場合に検索されるエンティティ数は，人手による検索が行える程度が

望ましい．認知心理学では，人間が一目で把握できる情報（チャンク）の数は 7 程度であることが知られている[9),10)]．このため，R にはたかだか 10 程度の値を用いる．

$K_i \approx c_i^{h_i-1}$ を考慮すると，属性 A_i の属性値総数は以下の式で評価できる．

$$\begin{aligned} &1 + c_i + c_i^2 + \cdots + c_i^{h_i-1} \\ &= \frac{1 - c_i^{h_i}}{1 - c_i} \approx \frac{1 - c_i K_i}{1 - c_i} \\ &= \left(1 + \frac{1}{c_i - 1}\right) K_i - \frac{1}{c_i - 1} \end{aligned}$$

属性値の木構造が平衡を保つためには，c_i の値は 3 以上でなければならない．$c_i \geq 3$ の範囲において，A_i の属性値数は K_i と $1.5K_i$ の間の値をとることが上式から分かる．一方，h_i は $1/\log c_i$ に比例するため，検索柔軟性の値は c_i が小さいほど大きくなる．葉属性値数を一定に保った場合，検索コストは c_i に対する増加関数になる．また，管理コストに対する c_i の寄与は小さい．したがって，$c_i = 3$ とすることで各コストのバランスが良好な状態に保たれる．

葉属性値に対応するエンティティ数に関する条件 (2-a) または (2-b) を満足する兄弟属性値集合について，葉属性値数 K_i に対して，各属性値に対応するエンティティ数の比率はそれぞれ $2/(K_i+1)$ および $2/K_i$ を超えない．これより R と K_i の間には以下の関係式が成立する．

$$R \geq M \times \prod_{i=1}^{N} \frac{2}{K_i} \tag{1}$$

式 (1) より，$M = 10^6$ 個程度のエンティティが存在した場合でも，$N = 5$ 個の属性を定義し，各属性には $K_i = 20$ 個程度の葉属性値を持たせることで条件が満足される．

属性 A_i の属性値数は K_i と $1.5K_i$ の間の値をとるため，属性値総数は $\sum_{i=1}^{N} K_i$ で評価できる．総加平均/相乗平均の公式と式 (1) より以下が成立する．

$$\sum_{i=1}^{N} K_i \geq N \sqrt[N]{\prod_{i=1}^{N} K_i} \geq 2N \sqrt[N]{\frac{M}{R}}$$

$N \sqrt[N]{M/R}$ の値は $N = \ln(M/R)$ で最小値をとる．したがって，属性数をこの限度内で増やすことで単一木構造を用いた場合（$N = 1$）と比較して属性値総数が減少する．以上により多次元分類方式では不要/人工的な分類を強いる可能性が低下する．

多次元分類を効果的に運用するためには属性の選択を適切に行うことも重要である．第 1 に属性値数が極端に少ないもの（例：性別）は不適切である．また，属性が互いに独立していることが必要である．そのために，属性間の相関係数ができるだけ小さくなるように属性集合を選択する必要がある．

上述した状況の下での管理コスト $m_i + c_i h_i$ について考える．$m_i = M/K_i$ の値は 5×10^4 程度になり，$\sqrt[N]{M^{N-1}}$ ($\sim M$) に比例して増大する．また，c_i の値が 3 であることを考慮すると，このときの h_i の値は 4 程度になる．これにより，管理コストにおいては葉属性値とエンティティの対応付けコスト（第 1 項）が主要な項になる．これに対して，木構造自身が比較的小さいこともあり，再構成コスト（第 2 項）は十分小さい．

一方，属性値の分割は m_i 個のエンティティが追加されるたびに生じるので，分割頻度は $\sqrt[N]{M^{N-1}}$ に比例して長期化する．

以上より，登録エンティティ数が増大するに従って属性値が分割される頻度は少なくなるが，属性値分割時のコストが大きくなるため，管理コストが集中する．

管理コスト $m_i + c_i h_i$ を分散するためには，葉属性値に対応するエンティティ数 m_i を一定値 L_{max} に制限する方法が考えられる．ただし，小さすぎる値を用いると属性値の木構造が頻繁に再構成されるため，総管理コストが増大する．また，大きすぎる値を用いた場合，登録エンティティ数が少ない段階では木構造が成長しないため，情報整理方式としての意味をなさない．そのため，M が比較的小さい段階では，m_i の値を 0 から L_{max} へ徐々に近付ける必要がある．

8. 評価実験

6 章で求めた管理コストは，分割される葉属性値に対応するエンティティ数に大きく依存する．しかし，属性値の木構造が成長するにつれて再構成頻度は少なくなるため，葉属性値に対応するエンティティ数のばらつきは大きくなる．また，適切な L_{max} の値を求めるためには，L_{max} と検索/管理コストの関係を調べる必要がある．そのために本章では，シミュレーションによる評価実験を行う．

評価実験では 10^6 個の属性値（整数型）をランダムに生成し，属性値の木構造に順次登録する．ここで，m_i の値を以下の式で定義する．なお，$N = 5$ としている．

$$m_i = min\left(L_{max}, \frac{M}{2\sqrt[N]{M/R}}\right) \tag{2}$$

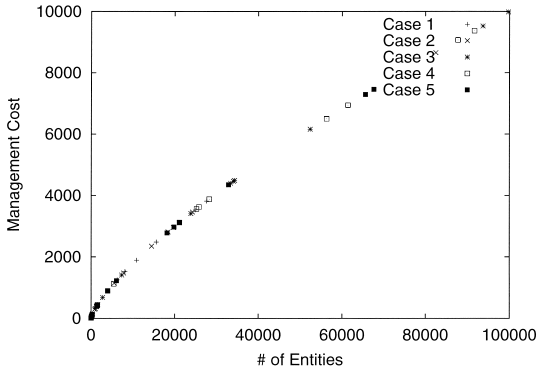

図 3 エンティティ数の増大にともなう管理コストの増加
Fig. 3 # of entities vs. management cost.

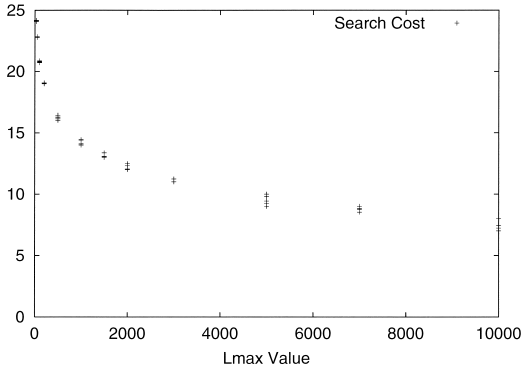

図 4 L_{max} による検索コストの変動
Fig. 4 L_{max} vs. search cost.

表 3 管理コスト別の再構成回数
Table 3 # of reorganization at each management cost.

管理コスト	再構成回数
0〜2000	36
2000〜4000	16
4000〜6000	4
6000〜8000	5
8000 以上	5

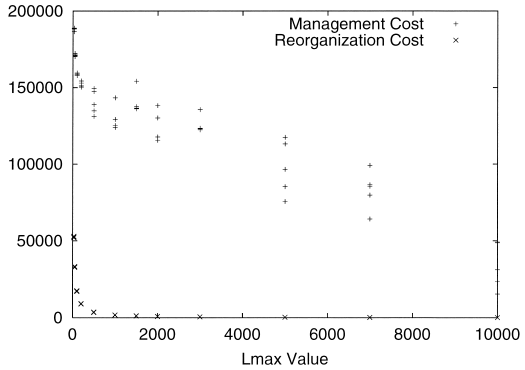

図 5 L_{max} による総管理コストの変動
Fig. 5 L_{max} vs. management cost.

$L_{max} = \infty$ の場合における登録エンティティ数と管理コストの関係を**図 3** に示す．これは，5 回のシミュレーション結果を示している．登録エンティティ数が増大するに従い，管理コストも増大する．**表 3** には管理コストごとの再構成回数を示す．管理コストが低いほど再構成回数が多い．以上より，登録エンティティ数が増えるに従って再構成頻度は少なくなるが，1 回あたりの再構成コストはむしろ増大することが分かる．

そこで，式 (2) において様々な L_{max} の値を用いて m_i の値を決定し，エンティティを登録する．木構造の再構成が必要になった時点で必要な管理コストを求める．また，エンティティ登録が完了した段階でランダムに発生したエンティティの検索コストを求める．

図 4 には L_{max} による検索コストの変動を示す．L_{max} の値が小さい場合には検索コストが増大する．これは，木構造が高くなるためである．**図 5** には L_{max} による管理コストおよび再構成コストの合計値の変動を示す．この値も L_{max} に対する減少関数である．L_{max} の値が小さい場合には，木構造が高くなり，再構成が発生しやすくなるためこの現象が生じる．なお，再構成コストが管理コストに占める割合は非常に小さい．すなわち，管理コストのほとんどは葉属性値とエンティティの対応付けに費やされる．

一方，**表 4** には様々な L_{max} の値に対する管理コストの分布を示す．L_{max} 値が小さい場合には管理コストの小さな再構成が頻繁に発生する．逆に，L_{max} を大きくすると再構成の発生頻度は低下するが，非常に管理コストの大きな再構成が発生する．これが，前章で議論した問題点である．なお，$L_{max} \geq 2000$ の場合における管理コスト 2000 以下の再構成の発生頻度はほぼ等しい．これは，小規模な再構成が木構造の比較的小さな段階で発生するためである．

表 4 管理コストの分布
Table 4 Distribution of management cost.

L_{max}	最瀕値（モード）		最大値	最小値
	値	相対度数		
30	34	99%	36	14
50	54	99%	56	14
100	104	99%	105	14
200	204	99%	205	14
500	504	98%	505	14
1000	1004	96%	1004	14
1500	1504	93%	1504	14
2000	2004	89%	2004	14
3000	3004	80%	3004	14
5000	5004	52%	5004	14
7000	7004	47%	7004	14

以上により，L_{max} の値は 1000〜2000 程度が望ましいと考えられる．

9. むすび

本論文では，各種のエンティティを柔軟に分類するために多次元分類方式を提案した．提案方式について各種のコスト評価を行い，効率的な構成法を導いた．本方式は独立した平衡木を用いることで検索コストの均等化を図り，利用者にエンティティ集合に対する検索指針を示すことができる．また，管理コストの分散にも配慮した．現在，我々は実用的な数のエンティティを登録したシステム構築を進めている．

多次元分類方式は，ハイパーリンクによる巡航操作を基礎とした方式と組み合わせることで，多数の利用者に対応した情報整理を効果的に行える．多次元分類では，大域的な検索および情報の概観を担当する．また，局所的な情報探索においてはエンティティ間での巡航操作を用いる．これにより，エンティティ間の関連に基づいた検索を組み込める．

多次元分類方式の管理コストをさらに低減するためには，葉属性値が分割された場合の属性値とエンティティの対応付けを自動化することが有効である．そのためには，エンティティのクラスタリング技術が有効と思われる．ただし，分類基準を属性が表現する概念と一致させる必要があるため，既存の手法をそのまま用いることはできない．この点に関する検討は今後の課題とする．

謝辞 著者の研究室を修了した北村繁広君（現 NEC 中部ソフトウェア）には，多次元分類方式の性質について種々の観点から検討していただきました．

参考文献

1) 森田昌宏，速見治夫：情報フィルタリングシステム—情報洪水への処方箋，情報処理，Vol.37, No.8, pp.751–758 (1996).
2) Halasz, F.: Reflections on NoteCards: Seven issues for the next generation of hypermedia Systems, *Comm. ACM*, Vol.31, No.7, pp.836–852 (1988).
3) Papazoglou, M.P. and Milliner, S.: Pro-active information elicitation in wide-area information networks, *Proc. Int. Symp. on Cooperative Database Systems for Advanced Applications* (*CODAS'96*), Kyoto Japan, pp.2–11 (1996).
4) 植田 学，増田佳弘，石飛康浩：CastingNet：情報の組織化と可視化ブラウジングのためのハイパーメディアシステム，コンピュータソフトウェア，Vol.12, No.4, pp.56–71 (1995).
5) Kanada, Y.: Axis-specified search: A fine-grained full-text search method for gathering and structuring excerpts, *Proc. ACM Conf. on Digital Libraries*, pp.108–117 (1998).
6) Morohashi, M. and Takeda, K.: Information outlining – filling the gap between visualization and navigation in digital libraries, *Proc. Int. Symp. on Research, Development and Practice in Digital Libraries*, pp.151–158, University of Library and Information Science (1995).
7) Takeda, K. and Nomiyama, H.: Information outlining and site outlining, *Proc. Int. Symp. on Research, Development and Practice in Digital Libraries*, pp.99–106, University of Library and Information Science (1997).
8) 長尾 真：知識と推論，岩波講座ソフトウェア科学 14，岩波書店 (1988).
9) Loftus, G.R. and Loftus, E.F.: *Human Memory: The Processing of Information*, Lawrence Erlbaum Associates (1976). 大村彰道（訳）：人間の記憶—認知心理学入門，東京大学出版会 (1980).
10) 御領 謙，菊地 正，江草浩幸：最新認知心理学への招待，新心理学ライブラリ 7，サイエンス社 (1993).

(平成 12 年 6 月 20 日受付)
(平成 12 年 9 月 27 日採録)

（担当編集委員 金森 吉成）

掛下 哲郎（正会員）

昭和 37 年生．昭和 59 年九州大学工学部情報工学科卒業．平成元年同大学院博士後期課程修了．工学博士．同年佐賀大学理工学部講師を経て，現在，助教授．データベースおよびソフトウェア工学の研究に従事．電子情報通信学会データ工学研究専門委員会幹事．ACM，日本ソフトウェア科学会等会員．

原槙 稔幸（正会員）

昭和 49 年生．平成 12 年佐賀大学理工学部知能情報システム学科卒業．同年大分大学工学部知能情報システム工学科技官．データベース，マルチメディア，分散システム等に興味を持つ．