4E1-6in

併合後の探索コストを考慮した木構造のクラスタリング手法

Tree Clustering to Lower User's Search Costs

井元 麻衣子*1 Maiko Imoto　数原 良彦*1 Yoshihiko Suhara　鷲崎 誠司*1 Seiji Susaki

*1日本電信電話株式会社 NTT サービスエボリューション研究所
NTT Service Evolution Laboratories, NTT Corporation

In this paper, we aim to create an easy-to-search category hierarchy by combining multiple category hierarchies. We regard a category hierarchy as a tree structure; nodes equal categories so this problem is cast as a tree clustering task. We propose a tree clustering method that constructs a tree structure with minimal search cost. To apply conventional agglomerative hierarchical clustering methods to the method, we first must define the inter-tree distance. Our definition is based on two scores that model the user's search cost of the tree structure. The scores reveal the strength of parent-child relationships and the association between nodes. Our approach merges trees based on the distance in the same manner as Ward's method. Compared to baseline methods, our method can build a better tree structure with respect to the overall ranking against five aspects.

1. はじめに

情報検索のひとつの方法として，システムが提示する選択肢をユーザが選択するディレクトリ検索がある．ディレクトリ検索では，ユーザは検索キーワードを入力する必要がなく，ユーザが具体的な検索キーワードを生成できない曖昧な検索要求を持つ場合にも，システムから提示された選択肢を選択することで検索対象を絞り込むことができる．現在，goo カテゴリー検索*1，Yahoo!カテゴリ*2などのディレクトリ検索サービスがある．このような検索サービスでは，階層構造はサービスごとに人手によって構築されている．このような複数の階層構造を併合することで，階層構造に配置する検索対象の増減に伴う階層構造の再構築の際に，不要なカテゴリの削減，適切なカテゴリの追加が容易になり，階層構造の分類精度を向上させることができる．また，それによってユーザの検索時間の削減が可能となる．階層構造の一部はカテゴリを中間ノード，アイテムを葉ノードとみなした木構造で表現することができ，階層構造の併合は木構造の併合によって実現できる．そこで本研究では，探索の観点において効率のよい木構造の併合を実現するために，ユーザが葉ノードのアイテムまでたどりつくときの木構造の探索のしにくさ (探索コスト) を計算可能な方法で定義し，併合後の探索コストが最小となる木構造の併合を行う階層的クラスタリング手法を提案する．

本稿の構成を示す．2 章で関連研究について述べる．3 章で提案手法，4 章で実験について述べ，5 章で本稿をまとめる．

2. 関連研究

階層的クラスタリング手法とは，N 個のノード集合が与えられたとき，1 個のノードだけを含む N 個のクラスタがある初期状態から，ノード n_1 とノード n_2 の間の距離 $d(n_1 \cdot n_2)$ からクラスタ間の距離 $d(C_1 \cdot C_2)$ を計算し，距離が最も近い 2 つのクラスタを併合し，この処理をクラスタが 1 個になるまでくり返すことで木構造を構築する手法である．クラスタ C_1 とクラスタ C_2 の距離関数 $d(C_1 \cdot C_2)$ の選び方によって最短距離法，最長距離法，群平均法，ウォード法などがある [Adami 03, Punera 05]．木構造をクラスタとみなし，木構造間に距離関数を定義することで，木構造の併合に階層的クラスタリング手法を適用することが可能である．

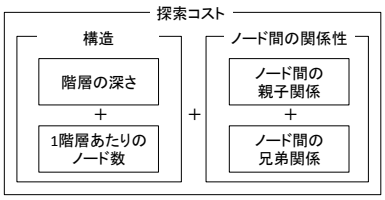

図 1: 探索コストを構成する 2 つの観点と 4 つの要素

3. 提案手法

本章では，階層構造を木構造とみなした際にノード間の関係性に基づくノード間の距離を定義する．次に，それを用いてユーザが根ノードから葉ノードのアイテムまでたどりつくコストを探索コストとして定義し，探索コストの算出方法を説明する．そして，探索コストに基づいて距離関数を定義し，木構造集合を併合して 1 つの木構造を構築するクラスタリング手法を提案する．ここで，本研究で対象とする木構造はノードのラベルをカテゴリとみなし，子ノードを区別しない無順序木とする．

3.1 探索コストの定義

本研究では，図 1 に示すとおり，木構造 T の探索コストを**構造に由来する探索コスト** $search\_cost\_structure(T)$ と**ノード間の関係性に由来する探索コスト** $search\_cost\_relation(T)$ によって定義する．具体的には

$$search\_cost(T) = \beta \times search\_cost\_structure(T) + (1-\beta) \times search\_cost\_relation(T) \quad (1)$$

として算出する．ここで，β は 2 つの探索コストの寄与率を表し，$0 \leq \beta \leq 1$ とする．以下，2 つのコストの算出方法について述べる．

3.1.1 構造に由来する探索コスト

T の構造に由来する探索コストは，T の階層の深さ，1 階層あたりのノード数に基づいて定義する．本研究では，既存のディレクトリ検索サービスで用いられている階層構造の多くが階層の深さを 3 から 4 に設計していることから，理想的な階層の深さは 3 か 4 であるとする．また，探索の観点において木構造の 1 階層あたりのノード数は 4〜8 が適しているという知見

連絡先: imoto.maiko@lab.ntt.co.jp

*1 http://category.goo.ne.jp/ (2013 年 2 月現在)

*2 http://dir.yahoo.co.jp/ (2013 年 2 月現在)

[Macgregor 86] に基づき，理想的な 1 階層あたりのノード数は 4〜8 とする．ただし，T に含まれるノード数が $3^4 = 81$ より少ないか $4^8 = 65536$ より多い場合には，2 つの条件を満たす木構造を構築できない．このような場合には与えられたノード数において構築可能な木構造の中で $search\_cost\_structure(T)$ が最小となる T が構造の観点で最も探索しやすい木構造であるとする．$search\_cost\_structure(T)$ は T の階層の深さに由来する探索コスト $depth\_cost(T)$ と 1 階層あたりのノード数に由来する探索コスト $sibling\_cost(T)$ を用いて

$$search\_cost\_structure(T) = depth\_cost(T) + sibling\_cost(T)$$

によって算出する．$depth\_cost(T)$ は

$$depth\_cost(T) = \frac{1}{|L|} \sum_{l \in L} leaf\_depth\_cost(l)$$

によって算出する．ここで，集合 L は T の葉ノード集合を表す．また，$|L|$ は集合 L の要素数を表す．$leaf\_depth\_cost(l)$ は葉ノード l の深さが 3 か 4 であれば最小の値をとり，それ以外のときは葉ノードの深さに応じて大きい値をとる．このとき，葉ノードの深さが小さいときには葉ノードの深さの 1 の差がユーザに与えるコストの増加量は大きく，葉ノードの深さが大きいときには葉ノードの深さの 1 の差がユーザに与えるコストの増加量は小さいものとして，対数関数を用いてコストを設計する．すなわち

$$leaf\_depth\_cost(l) = \begin{cases} 1 & (l = 3, 4 \text{ のとき}) \\ \log(l) + 1 & (\text{それ以外}) \end{cases}$$

によって算出する．$sibling\_cost(T)$ は

$$sibling\_cost(T) = \frac{1}{|K|} \sum_{k \in K} node\_num\_cost(k)$$

によって算出する．ここで，集合 K は T の葉ノード以外のノード集合を表す．また，$|K|$ は集合 K の要素数を表す．$node\_num\_cost(k)$ は k の子ノード数 s が 4〜8 であれば最小の値をとり，それ以外のときは s の大きさに応じて大きい値をとる．このとき，$node\_num\_cost(k)$ は $leaf\_depth\_cost(l)$ と同様の性質を持たせる．つまり，$node\_num\_cost(k)$ は

$$node\_num\_cost(k) = \begin{cases} 1 & (4 \leq s \leq 8 \text{ のとき}) \\ \log(s) + 1 & (\text{それ以外}) \end{cases}$$

によって算出する．

3.1.2 ノード間の関係性に由来する探索コスト

T のノード間の関係性に由来する探索コストは，T におけるノード間の上位下位関係と兄弟関係を考慮したノードの配置に基づいて定義する．ユーザが選択したノードの子ノード集合として適切でないノードがユーザに提示される場合や，ユーザに提示されるノード集合がユーザにとって比較しづらい場合，ユーザは上位階層に戻ってノードの再選択を行うなどの手戻りが発生してしまう．このとき，高さが 2 以上異なるノード間の上位下位関係はユーザの探索のしやすさに影響を与えないものとして，本研究では高さ 1 の部分木におけるノード間の関係性に着目する．具体的にはノード間の親子関係と兄弟関係に着目し，親子関係が適切に設定されているか，兄弟ノードに同一の親ノードをもつことが適切なノード群が配置されているかに基づいて高さ 1 の部分木に対して部分木スコアを算出し，それを用いて $search\_cost\_relation(T)$ を定義する．$search\_cost\_relation(T)$ は以下のステップによって算出する．

1. T の葉ノード以外のノード集合の任意の 2 つのノードの組合せに対して，親子スコアと兄弟スコアを算出する．
2. 親子スコアと兄弟スコアを更新する．
3. T を高さ 1 の部分木に分割する ．
4. 高さ 1 の部分木それぞれについて部分木スコアを算出する．
5. 部分木スコアを用いて $search\_cost\_relation(T)$ を算出する．

親子スコア $Oscore(p, c)$ はノード p とノード c における親子関係の強さを示す推定値である．兄弟スコア $Kscore(c1, c2)$ はノード $c1$ とノード $c2$ における兄弟関係の強さを示す推定値である．また，部分木スコア $Dscore(p, C_p)$ は高さ 1 の部分木の親ノード p，p の子ノード集合 C_p における親子関係の強さと兄弟関係の強さを示す推定値である．

まず，T の葉ノード以外のノード集合の任意の 2 つのノードの組合せに対して，親子スコアと兄弟スコアを算出する．これらは既存の木構造における 2 つのノードの配置関係に基づいて算出する．$Oscore(p, c)$ は p と c との上位下位関係における距離が短ければ短いほど p と c における親子関係は強いとして大きくする．つまり

$$Oscore(p, c) = \begin{cases} \frac{1}{n+1} & (p\text{が } c \text{ の上位語であるとき}) \\ -\frac{1}{n+1} & (p \text{ が } c \text{ の下位語であるとき}) \\ 0 & (\text{それ以外}) \end{cases}$$

によって算出する．ここで，上位語とは木構造をノードから根ノードまで下位階層に戻ることなくたどるときに通過する中間ノードと根ノードのことである．下位語とは木構造をノードから葉ノードまで上位階層に戻ることなくたどるときに通過する中間ノードのことである．また，2 つのノードの間に含まれる中間ノードとは，$node1$ が $node2$ の上位語であるとき，$node1$ の下位語であり，かつ $node2$ の上位語である中間ノードのことである．n は p と c の間に含まれる中間ノードの個数を表す．すなわち，n が小さいほど p と c の上位下位関係における距離は短い．$Kscore(c1, c2)$ は，$c1$ と $c2$ が同一の親ノードをもつ兄弟ノードとして適切であれば高く，そうでなければ低くする．つまり

$$Kscore(c1, c2) = \begin{cases} \frac{1}{m+1} & (c1 \text{ と } c2 \text{ に共通の上位語が} \\ & \quad \text{存在し，かつ，} m \leq t \text{ のとき}) \\ 0 & (\text{それ以外}) \end{cases}$$

によって算出する．ここで，2 つのノード $node1$，$node2$ の共通の上位語とは，各ノードの上位語集合をとり，2 つの集合において共通の要素のうち最も高さが小さい中間ノードであり，かつ，各ノードとその中間ノードの間に含まれる中間ノードの個数が一致する中間ノードである．m は $c1$ または $c2$ と共通の上位語の間に含まれる中間ノードの個数を表す．また，t は 0 以上の整数であり，人手で設定する．

T の葉ノード以外のノードの任意の 2 つの組合せ全てに対して $Oscore(p, c)$ と $Kscore(c1, c2)$ を算出したのちに，$Oscore(p, c1) = 1$，$Kscore(c1, c2) = 1$ となる 2 つのノードの組合せ p，$c2$ を抽出する．p と $c1$ が親子関係であり，かつ $c1$ と $c2$ が兄弟関係であれば，p と $c2$ は親子関係である可能性が高い．この性質を利用して，すでに算出した $Oscore(p, c2)$ が 1 でなければ，$Oscore(p, c2) = 1$ に更新する．また，$Oscore(p, c1) = 1$，$Oscore(p, c2) = 1$ となる 2 つのノードの組合せ $c1$，$c2$ を抽出する．p と $c1$，p と $c2$ がそれぞれ親子関係であれば，$c1$ と $c2$ は兄弟関係である可能性が高い．この性質を利用して，すでに算出した $Kscore(c1, c2)$ が 1 でなければ，$Kscore(c1, c2) = 1$ に更新する．

次に，T を高さ 1 の部分木に分割する．中間ノード数が n 個の T は，根ノードと中間ノードをそれぞれ親ノードとする高さ 1 の部分木 $(n + 1)$ 個に分割する．

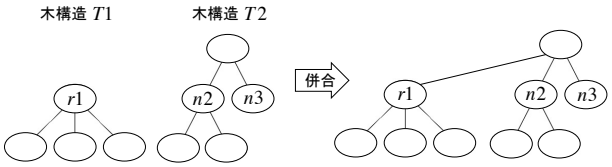

図 2: $r1$ と $n2$ の併合イメージ図

そして，高さ 1 の部分木それぞれについて部分木スコアを

$$Dscore(p, C_p) = \alpha \frac{1}{|C_p|} \sum_{c \in C_p} Oscore(p, c) + (1-\alpha) \frac{1}{\gamma} \sum_{(c1,c2) \in C_p} Kscore(c1, c2) \quad (2)$$

によって算出する．ここで，$|C_p|$ は集合 C_p の要素数を表す．また，γ は集合 C_p の任意の 2 つの要素の組合せ数を表し，$\gamma = \begin{pmatrix} |C_p| \\ 2 \end{pmatrix}$ である．α は 2 つのスコアの寄与率を表し，$0 \leq \alpha \leq 1$ とする．

最後に $search\_cost\_relation(T)$ を

$$search\_cost\_relation(T) = \sum_{k \in K} \frac{1}{Dscore(k, C_k)}$$

によって算出する．ここで，K の要素 k における子ノード集合を C_k とする．

3.2 木構造のクラスタリング

本研究では，複数の木構造を入力として併合後の $search\_cost(T)$ を考慮した木構造のクラスタリング手法を提案する．提案手法は，以下のステップによって行う．

1. 木構造集合において，異なる木構造に含まれる根ノード r と根ノード以外のノード n の組合せすべてに対して損失コスト $cost(r, n)$ を算出する．
2. $cost(r, n)$ が最小である 2 つのノードが兄弟ノードになるように 2 つの木構造を併合する．
3. 木構造集合の要素数が 1 であれば処理を終了し，そうでなければ 1. に戻る．

2 つの木構造 $T1$，$T2$ について，$T1$ の根ノード $r1$ と $T2$ の根ノード以外のノード $n2$ に対して $r1$ と $n2$ を併合したときの損失コスト $cost(r1, n2)$ の算出方法について説明する．$r1$ と $n2$ を併合した木構造は図 2 にようになる．併合する 2 つのノードを決定する際に，併合時の 2 つのノード間の関係性のみに基づく最短距離法や最長距離法による木構造の併合では併合後の木構造の探索コストを考慮しないため，探索コストが大きい木構造が構築されてしまう可能性がある．したがって，提案手法では Ward 法に基づいて $cost(r1, n2)$ を 2 つの木構造間の距離とする階層的クラスタリングを行う．損失コスト $cost(r1, n2)$ は $T1$ と $T2$ の探索コストの和と $r1$ と $n2$ を併合した木構造 $T3$ の探索コストの差，つまり

$$cost(r1, n2) = search\_cost(T3) - (search\_cost(T1) + search\_cost(T2))$$

によって算出する．$cost(r1, n2) > 0$ のとき，$cost(r1, n2)$ が大きければ大きいほど $T3$ の探索コストは大きく，$cost(r1, n2) < 0$ のとき，$T3$ の探索コストは小さい．したがって，$cost(r1, n2)$ が最小となる $r1$ と $n2$ を併合することで，探索コスト最小の $T3$ を構築することができる．ここで，図 2 における $r1$ と $n2$ を併合した木構造と $r1$ と $n3$ を併合した木構造は同一であるため，$cost(r1, n3)$ は算出する必要がない．

4. 実験

本稿で定義した探索コストを最小にする木構造集合の併合クラスタリング手法によって構築した木構造がユーザにとって探索しやすいかを被験者評価に基づいて検証した．

4.1 データセット

既存のディレクトリ検索サービスのうち，goo カテゴリー検索，Yahoo!カテゴリ，All About コンテンツテーマ[*3]で利用されている階層構造に含まれるカテゴリをノードとみなして，1 階層から 2 個以上のカテゴリを抽出し，62 個のカテゴリを実験に用いた．62 個のカテゴリの任意の 2 つの組合せ (1891 通り) における親子スコアと兄弟スコアは，これらの階層構造を用いて $t = 4$ として算出した．2 つのカテゴリの配置関係が階層構造によって異なる場合には，算出したスコアのうち最大値をスコアとして用いた．実験では，まず 62 個のカテゴリから 21 個，26 個，27 個，36 個，49 個のカテゴリを抽出し，それぞれカテゴリ集合を生成した．ただし，各カテゴリ集合の各カテゴリには兄弟スコアが 1 となるカテゴリが 1 つ以上存在するようにした．そして，カテゴリ集合に対して部分木スコアが最大となる，木構造の最小単位である高さ 1 の木構造を 5 個，6 個，6 個，7 個，11 個構築し，これらの木構造集合をそれぞれ入力として 5 つの木構造を構築した．

4.2 実験条件

以下の 4 つの手法で構築した木構造を評価対象とした．

提案手法 1 (2) 式において $\alpha = 0.8$ とする提案手法 (MA1).
提案手法 2 (2) 式において $\alpha = 0.2$ とする提案手法 (MA2).
比較手法 1 最短距離法に基づく手法 (NNM).
比較手法 2 群平均法に基づく手法 (GAM).

ここで，併合前の 2 つの木構造を $BT1$，$BT2$ とし，$BT1$ の根ノードを r，$BT2$ の根ノード以外のノードを n とする．また，集合 S は n の兄弟ノードを要素とする集合を表し，$m = |S|$ とする．比較手法 1，2 では，r と集合 K をそれぞれ 1 つのクラスタとして，兄弟スコアをノード間の距離として用いた．また，本実験では全ての手法において (1) 式は $\beta = 0.5$ とした．

4.3 評価

兄弟スコアの算出方法の評価 (評価 1) と木構造の探索のしやすさの評価 (評価 2) を行った．

4.3.1 評価 1：兄弟スコアの算出方法

3 人の被験者に 2 つのカテゴリ間の関係を判定してもらい，兄弟関係であると判定された 2 つのカテゴリの組を無作為に 125 組抽出した．そして，125 組を提案手法によって算出した兄弟スコアごとに分け，その割合を用いて考察した．

実験結果を表 1 に示す．数値は 2 つのカテゴリの組合せ 125 組を提案手法で算出した兄弟スコアで分類したときの組合せの個数を示している．括弧内の数字は組合せの割合を表し，例えば 0.10 は 125 組の 10%の 2 つのカテゴリの組合せに対して提案手法で兄弟スコアとして 1 を算出していることを意味する．

実験結果より，3 人の被験者が兄弟関係であると判定した 2 つのカテゴリの組合せのうち，10%に対しては最も高い兄弟スコアである 1，54%に対しては 2 番目に高い兄弟スコアである 0.5 を付与することができており，125 組の 64%に対しては提案手法によって高い兄弟スコアを付与することができていることが分かった．これは，ユーザの主観と一致する 2 つのカテゴリの関係を推定できていると解釈することができる．一方で，36%のカテゴリの組合せに対しては既存の階層構造において叔父叔母の関係であるため兄弟スコアを 0 と算出していたこと

*3 http://allabout.co.jp/ (2013 年 2 月現在)

表 1: 兄弟スコアの評価結果

Kscore($c1$, $c2$)=1	Kscore($c1$, $c2$)=0.5	Kscore($c1$, $c2$)=0
12 (0.10)	68 (0.54)	45 (0.36)

が分かった．ここで，ノード $n1$ と $n2$ が叔父叔母の関係であるとは，ノード $n3$ が 2 つのノードの上位語であり，$n3$ と $n1$ (または $n2$) の間に含まれる中間ノードの個数が 0，かつ，$n3$ と $n2$ (または $n1$) の間に含まれる中間ノードの個数が 1 である階層構造における $n1$ と $n2$ の配置関係をいう．既存の階層構造において叔父叔母の関係にある 2 つのカテゴリに対して高スコアな兄弟スコアを算出することを今後検討することで，2 つのカテゴリの関係の推定精度向上が可能となる．

4.3.2 評価 2：木構造の探索のしやすさ

木構造の構造に由来する探索のしやすさとノード間の関係性に由来する探索のしやすさの観点から手法の評価を行った．

木構造の構造に由来する探索のしやすさは階層の深さ (depth)，1 階層のノード数 (sibling) の 2 つの観点から評価した．木構造ごとに葉ノードの深さの平均 AD を算出し，5 つの木構造の AD の平均を階層の深さに対する評価値とした．また，木構造ごとに 1 階層のノード数が 4〜8 である割合 PS を算出し，5 つの木構造の PS の平均を 1 階層のノード数に対する評価値とした．

ノード間の関係性に由来する探索のしやすさは [Chuang 04] の評価方法の一部を用いて

親子関係 (oyako) p と $c1$ は親子カテゴリとして適切か，
兄弟関係 (kyodai) C_p の各要素は兄弟カテゴリとして適切か，
兄弟ノードの区別 (distinction) C_p の各要素は選択肢として区別しやすいか，

の 3 つの観点から 3 人の被験者による主観評価を行った．ここで，$c1$ は p の子ノードであり，木構造の併合によって p の子ノードになったノードである．また，C_p は併合後の p の子ノード集合である．

親子関係の評価は p と $c1$ の関係を判定してもらった．兄弟関係の評価は C_p の任意の 2 つのカテゴリの組合せについて，それらの関係を判定してもらった．兄弟ノードの区別の評価は，C_p の任意の 2 つのカテゴリの組合せについて，それらが選択肢として区別しやすいかを判定してもらい，区別しやすいと判定した割合に基づいて C_p の各要素が選択肢として区別しやすいかを 0 点〜5 点の 6 段階で評価してもらった．

木構造の観点に対する評価値は，親子関係の評価は 3 人の被験者が親子関係であると判定した 2 つのカテゴリの組合せの割合，兄弟関係の評価は兄弟関係であると判定した 2 つのカテゴリの組合せの割合，兄弟ノードの区別の評価は 4 点以上と判定したノード集合の割合とした．観点ごとの評価値は 5 つの木構造の評価値の平均とした．つまり，評価値が高い手法ほどその観点に対する評価が高い．

観点ごとの評価値と総合評価結果を表 2 に示す．括弧内の数字は観点に対する手法の順位を表す．階層の深さの評価は，入力の木構造集合のノード数が少ない場合でも木構造の階層の深さは 3〜4 が望ましいとして，評価値が 3〜4 に近い手法が上位になるように順位をつけた．それ以外は評価値の降順に順位をつけた．また，総合順位は 5 つの観点の順位の和の昇順に順位をつけた．

実験結果において，5 つの観点を考慮して併合する 2 つのノードを決定する MA1，MA2 は著しく評価値が低い観点がないことが分かった．ノード間の親子関係を重視して併合する 2 つのノードを決定する MA1 は，木構造が縦方向に併合されやすくノード数が少ない木構造においても階層の深さが大きくなった．ノード間の兄弟関係のみに基づいて併合する 2 つのノードを決定する NNM は，ノード間の関係性に由来する探索のしやすさの評価値は高いが構造に由来する探索のしやすさの評価値は低い結果となった．これらのことから，木構造の探索コストの算出方法は併合する 2 つのノードを決定する際に重視する観点との対応が取れていると解釈することができる．

表 2: 5 つの観点の評価結果

method	depth	sibling	oyako	kyodai	distinction	総合順位
MA1	3.02 (1)	0.54 (1)	0.28 (2)	0.47 (2)	0.53 (3)	1
MA2	2.86 (3)	0.54 (1)	0.21 (3)	0.43 (3)	0.57 (2)	3
NNM	2.26 (4)	0.45 (4)	0.29 (1)	0.60 (1)	0.66 (1)	2
GAM	2.95 (2)	0.50 (3)	0.18 (4)	0.33 (4)	0.46 (4)	4

また，4 つの手法を 5 つの観点で総合的に評価すると MA1 が最も高い評価結果となり，併合後の探索コストを最小にする木構造がユーザにとって探索しやすい木構造であると解釈することができる．併合する 2 つのノードをノード間の兄弟関係のみに基づいて決定する NNM では 1 つのノード n の子ノードになるように併合される木構造が多く，そのノードの子ノード数だけが大きく，階層の深さは小さくなる傾向があった．これにより，木構造の併合によって子ノード数が変化しない階層が多いため，1 階層あたりのノード数が少ない階層が多くなり sibling の評価値が低かった．また，NNM はノード間の関係性に由来する探索のしやすさの評価は木構造全体で見ると高いが，n を親ノードとする高さ 1 の部分木に対するノード間の関係性に由来する探索のしやすさの評価は著しく低かった．これらから，入力の木構造集合のノード数が少ない場合においても階層の深さは 3〜4 が望ましく，1 階層あたりのノード数が多くなるとユーザにとって探索しにくいと言える．今回の実験では，α の値を大きく設定した MA1 によって構築した木構造の方が MA2 によって構築した木構造よりもユーザにとって探索しやすいという実験結果になったが，前述したように兄弟関係を正しく推定できていない 2 つのカテゴリの組合せもあるため，α の値の設定方法は今後再検討する必要がある．

5. まとめと今後の課題

本稿では，ユーザにとって探索しやすい木構造の特徴について検討し，ユーザの探索にかかる探索コストを定義した．そして，探索コストが小さい木構造を構築するクラスタリング手法を提案した．実験の結果，併合後の探索コストを考慮した提案手法によってユーザにとって探索しやすい木構造を構築可能であることを確認した．今後は，2 つのカテゴリ間における兄弟関係の推定精度向上に向けた再検討を行う．また，ユーザにとって探索しやすい木構造の特徴をさらに検討し，探索コストの算出式や 2 つの寄与率 α，β について再検討する．

参考文献

[Adami 03] G. Adami, P. Avesani and D. Sona: Bootstrapping for Hierarchical Document Classification, CIKM2003, pp.295—302 (2003)

[Punera 05] K. Punera, S. Rajan and J. Ghosh: Automatically Learning Document Taxonomies for Hierarchical Classification, WWW2005, pp.1010—1011 (2005)

[Macgregor 86] J. Macgregor, E. Lee and N. Lam: Optimizing the Structure of Database Menu Indexes: A Decision Model of Menu Search, Human Factors: The Journal of the Human Factors and Ergonomics Society, Vol. 28, No. 4, pp.387—399 (1986)

[Chuang 04] S.-L. Chuang and L.-F. Chien: A practical web-based approach to generating topic hierarchy for text segments, CIKM2004, pp. 127—136 (2004)