類似画像検索への応用を目的とした 階層化属性付きグラフマッチングの高速化

北本 朝展　高木 幹雄
東京大学生産技術研究所

Abstract: This paper describes the fast matching of hierarchical attributed relational graphs with the aim of applying the method to similarity-based image retrieval. Best-first search A^* algorithm, admissible heuristic function, and maximum permissible cost are proposed to speed up the computation of graph matching. By means of these methods, the average computation time of graph matching speeds up about three times faster than that of using conventional methods, which speed-up reduces the retrieval time into only five seconds in the case of retrieving top 10 similar graphs from the database of 1027 graphs.

1　はじめに

本研究のテーマは、類似画像検索への応用を目的とした、階層化属性付き関係グラフのマッチングの高速化である。ここで類似画像検索とは、画像データベースの中から検索キーに類似した内容をもつ画像を検索する技術である。類似画像検索では、検索したい画像の内容に対応する何らかの検索規準をユーザがシステムに提示すると、システムは検索キーと蓄積データとの類似度を検索規準に基づいて 1 個ずつ計算し、類似度の大きい順に整列して出力する。しかし類似度の計算は一般に計算量が大きいため、画像データベースが巨大となるにしたがって類似度計算の高速化が大きな課題となる。

本研究では画像表現モデルとして、モデリング能力が強力であるグラフ構造を用いている。このグラフ構造間の類似度に相当するグラフマッチングコストの計算は、組合せ的爆発を引き起こす典型的な問題であり、この欠点が画像表現モデルとしてのグラフ構造の利用を阻んできた。そこで本研究はグラフマッチングの高速化を追究する。特に本研究の特徴は、解の最適性を犠牲にせずにグラフマッチングの高速化を達成したところにある。従来の手法では高速化の代償として解の質を準最適解に落とすアプローチが多いものの、応用分野によっては厳密な最適解を安定して得たい場合もあるからである。

本研究の高速化のポイントは、1) 最良優先探索 A^* アルゴリズム、2) 認容可能条件を満たすヒューリスティック関数、3) 最大許容コストの 3 点である。これらの高速化手法により、1027 個のグラフの中から上位 10 個のグラフを検索する検索時間が約 5 秒に短縮できた。

Fast Matching of Hierarchical Attributed Relational Graphs for an Application to Similarity-Based Image Retrieval
Asanobu KITAMOTO and Mikio TAKAGI
Institute of Industrial Science, University of Tokyo

2　グラフマッチング問題

2.1　階層化属性付き関係グラフ

本研究では、属性付き関係グラフ [1] を 2 層に階層化したモデル、すなわち階層化属性付き関係グラフ (Hierarchical Attributed Relational Graph) を画像表現モデルとして用いる。このモデルでは、ノード (node) は領域に関する情報、アーク (arc) は 2 領域の関係を表現する。またノードとアークのシンボルは記号的に画像を記述する一方、ノードとアークの属性は画像の数値的な情報を記述する。以下では階層的属性付き関係グラフ H のノード集合を N、アーク集合を $B \subseteq N \times N$ とする。またマッチングする 2 個の階層的属性付き関係グラフを H_1、H_2 と表記し、そのノード集合を N_1、N_2 とする。

2.2　グラフマッチング

次にグラフマッチングの基本的な概念を述べる。グラフマッチングとは、代替・削除・挿入という基本的な変形操作により、2 個のグラフを最小のコストで同型とするという、グラフ同型問題に帰着する [1]。このような 2 個のグラフのグラフマッチング問題は、2 個のグラフからそれぞれ任意に 1 個ずつ取り出した 2 個のノードのペア $(i, j) \in N_1 \times N_2$ を節点とした状態空間の中から、最小コストに対応する経路を探索する問題になる。この状態空間は図 1 のような木構造で表現される。 ここで節点の展開とは、ある状態を中心にしてそこから次に到達できる状態をすべて生成する処理のことを指す。すなわち状態 S のノードに対して、そこから次に到達できるすべての子ノード S' を生成する処理である。第 3 節で述べる探索方法の差異は、生成処理の方法の差異からくるものである。

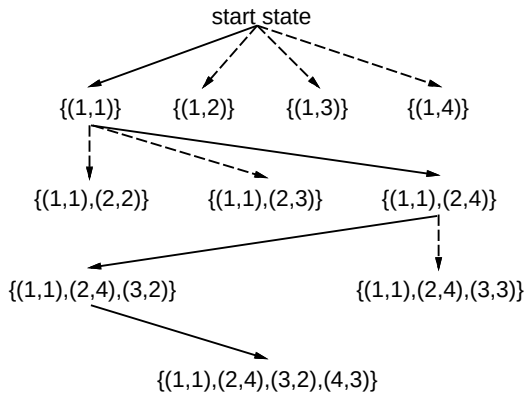

図 1: 4 個のノードを持つ 2 個のグラフをマッチングした場合の状態空間。実線が最小コストに対応した経路。

2.3 グラフマッチングコスト

まずマッチングコストは、シンボルコストと属性コストの和として定義する。そして個々のグラフ変形においてマッチングコストを計算し、最終的にはマッチングコストをすべて加算することで全体のグラフマッチングコストを計算できる。ここでは探索途中の節点 $(i,j) \in N_1 \times N_2$ で生じるマッチングコストについて説明する。

2.3.1 シンボルコスト

シンボルとはノードやアークの特徴を記号的に示すものであり、シンボル間の類似度を代替コストとして定義する。ここでシンボル α とシンボル β の代替コスト $\mathrm{subcost}(\alpha,\beta)$ は、二つのシンボル間で交換則が成立するとして以下のように定義する。

$$\mathrm{subcost}(\alpha,\beta) = \mathrm{subcost}(\beta,\alpha) = g_{\alpha\beta} \qquad (1)$$

さてここで、ノードの数が異なるグラフ間でもマッチングを可能とするために、特別なシンボルであるヌルシンボル— 大きさがゼロのノードやアーク — を導入する。ヌルシンボルを \$ と表記すると、シンボル α との代替コストを以下のように定義する。

$$\mathrm{subcost}(\$,\alpha) = \mathrm{subcost}(\alpha,\$) = g_{\alpha\$} \qquad (2)$$

2.3.2 属性コスト

まず属性としてサイズ属性と形状属性の 2 種類を考える。サイズ属性は要素の大きさを示すものであり、ノードの場合は対応する領域の面積、アークの場合は対応する関係の距離などに対応する。一方の形状属性は形状特徴などを数値的に表すものであり、ノードの場合は縦横比、アークの場合は水平方向に対する傾きなどがある。

次に属性コストの計算方法を述べる。以下ではノードの場合だけを取り上げるが、アークの計算方法も基本的に同様である。さて n 個の属性値を並べたベクトルを n 次元の属性ベクトルとしよう。節点 (i,j) でマッチングされる 2 つの n 次元属性ベクトル $\vec{x_i} = (x_1^i, \cdots, x_n^i)$、$\vec{y_j} = (y_1^j, \cdots, y_n^j)$ の間の距離 (マッチングコスト) は、

$$\mathrm{cost}(\vec{x_i}, \vec{y_j}) = \sum_{k=1}^{n} g_k \frac{p_k(x_k^i, y_k^j)}{s_k} \qquad (3)$$

と重み付き線形和として定義する。ここで重み係数 g_k は、人間が各属性を主観的に重視する程度を表現する定数である。一方の規格化定数 s_k は、各重み係数 g_k の値を同程度の大きさにするための補正項であり、通常は k 番目の属性の標準偏差とする。

さて重み係数が人間の主観を反映するものであることから、k 番目の属性に対応した距離関数 $p_k(x_k^i, y_k^j)(\geq 0)$ も 、属性値の単純な差分ではなく人間の知覚を反映した関数とするべきである。例えばサイズ属性では、$p(x,y) = |\log(x/y)| = |\log x - \log y|$ $(x, y \gg 1)$ のように倍率を基本にして距離関数を定義する。また、例えば主軸方向のように角度の性質を持った属性については、無向の場合は最大の角度差が $\pi/2$ であることを考慮すると、$p(x,y) = -||x-y| - \pi/2| + \pi/2$ $(-\pi/2 < x, y \leq \pi/2)$ という距離関数を定義すればよいだろう。

しかし先に導入したヌルシンボルは大きさが 0 の要素であるため、ヌルシンボルをもつノードやアークに対しては形状属性を定義することはできない。そこでヌルシンボルとのマッチングでは、形状属性コストの代わりに式 (2) の代替コストによってコストを定義する。

2.4 グラフマッチング高速化の基本方針

さてここでは今までの導入部を受けて、グラフマッチング高速化のための基本方針をまとめる。図 1 のような状態空間の探索では、節点を展開するごとに第 2.3 節のグラフマッチングコストを新たに計算しなければならない。しかしグラフマッチングコストの計算は大きな計算コストを要求するため、まず節点の展開回数をできるだけ少なくすることが第一の目標となる。次に、たとえ有望でない方向に探索が進んだとしても、できるだけ早めに探索経路を変更できれば節点の無駄な展開を抑えることができるだろう。そこで以下のような方針となる。

1. 最も望みのある節点から先に展開し、節点の無駄な展開をできるだけ抑える。
2. 現在の経路が有望でないとわかったら、できるだけ早い段階で枝刈りして別の経路を探索する。

さて 1 を実現するためには、有望な節点から順に展開できるように、展開可能な節点を有望さの程度に応じて整列させておけばよい。次に 2 を実現するためには、探索を続ける価値がある最大のコストをどの状態でも把握できるようにし、探索を続ける価値がなくなれば即座にマッチングを中断できる仕組みが必要となる。そこで以下では、1 に関連した高速化手法を第 3 節で、

表 1: 代表的な探索アルゴリズムの比較。最短経路を求める問題の場合。

	深さ優先探索	幅優先探索	最良優先探索
判断規準	開始節点から遠い節点を優先	開始節点から近い節点を優先	評価関数値が小さい節点を優先
長所	状態空間木の枝分かれが多いと効率がよい。	状態空間木の枝分かれが少ないと効率がよい。	優先順位付き待ち行列のおかげで展開する節点の数が少ない。
短所	状態空間木の枝分かれが少ないと効率が悪い。	状態空間木の枝分かれが多いと効率が悪い。	優先順位付き待ち行列の管理に余分な計算コストを必要とする。

1 と 2 に関連する高速化手法を第 4 節で、そして 2 に関連する高速化手法を第 5 節で述べる。

3 探索アルゴリズムの比較

一般的な探索アルゴリズムでは、展開済みの節点から到達できるがまだ展開していない節点 S の集合について何らかの評価関数 $f^*(S)$ を設け、その値が最大(または最小) の節点から展開していく。代表的な探索アルゴリズムには、深さ優先探索 (depth-first search)、幅優先探索 (breadth-first search)、最良優先探索 (best-first search) の 3 種類があり、その特徴を表 1 にまとめた。

ここでの問題は、最良優先探索とその他の手法との比較である。最良優先探索では、優先順位付き待ち行列を活用することで出発点からの距離を問わずに有望な節点から展開していくため、展開する節点の数を他の 2 手法に比べて削減することができる。しかし待ち行列を管理する必要があるため、この部分に余分な計算コストを必要とする。先にも述べたようにグラフマッチングの計算では節点展開の計算コストが大きいと予想されるため、展開の計算コストと待ち行列の管理コストとのトレードオフを実験を通して比較することを目指す。

4 認容可能条件を満たすヒューリスティック関数

次に評価関数 $f^*(S)$ について考察する。一般的に評価関数 $f^*(S)$ は、$f^*(S) = g(S) + h^*(S)$ という形式で表される。ここで $g(S)$ は開始節点 s から節点 S に至る経路のコストであり、$h^*(S)$ は節点 S から終了節点までの経路の最小コストの推定値である。もし $h^*(S)$ が真の値 $h(S)$ と全く同一ならば、最適経路上以外の節点を展開することなく最適解を発見できる。さて $g(S)$ の小さい順に S を展開する戦略が最良優先探索であるが、さらに探索効率を高めるためにヒューリスティック関数 $h^*(S)$ を活用する。特に、簡単な計算で真の値に近い推定が得られる関数が理想的である。このようなヒューリスティック探索の手法を応用することで、可能な解の数が組合せ的に爆発するのを防ぐことができる。

ここで $h^*(S)$ が認容可能条件 (admissibility condition) を満たせば、得られる解の最適性を保証しながら計算の高速化が可能となることが知られている。このようなアルゴリズムを A^* アルゴリズムという [2]。認容可能条件を満たすためには、$h^*(S)$ の推定値が真の値 $h(S)$ より小さいかまたは等しいことが常に成立しなければならない。そこで本研究では認容可能条件を満たす $h^*(S)$ を導出し、同時にその関数の計算が簡単であり実用的であることを示す。通常は問題領域の知識を用いてヒューリスティック関数を作るが [3]、本研究が提案するヒューリスティック関数は系統的に導くことができるものであるため、他の分野への応用も可能である。なお深さ優先探索と幅優先探索では、節点の展開順序の制御に $f^*(S)$ を使うことはないが、探索の枝刈りとして $f^*(S)$ の値を有効に活用することができる。

4.1 ノードのヒューリスティック関数

以下では節点 S までにマッチングされたノードの集合を M とする。すると節点 S の時点でマッチングされていないノード集合は $Y = N - M$ で表せる。

4.1.1 ノードの場合のシンボルコスト

まず Y_1 と Y_2 のシンボルの内訳を調べる。そして任意のシンボルの組合せ (α, β) に関して、Y_1 と Y_2 のノード数の差 $r_{\alpha\beta}$ を記録する。これらが将来の代替コストの原因となることから、このようなノード数の差を最小の代替コストで解消する方法を考える。

まず、任意のシンボルの組合せ (α, β) の中で最もシンボルコスト $g_{\alpha\beta}$ の値が小さい組合せに関するノード数の差 $r_{\alpha\beta}$ を調べ、$g_{\alpha\beta} \times r_{\alpha\beta}$ を $h^*(S)$ に加える。そしてシンボル α と β に関係するすべての組合せの差から $r_{\alpha\beta}$ の値を引く。この操作をすべての差が解消されるまで続けると、シンボルコストの $h^*(S)$ が得られる。

4.1.2 ノードの場合の属性コスト

式 (3) の距離関数 $p(x, y)$ は人間の知覚を反映した関数とするため、その関数形は一般に単純ではない。またヌルシンボルの導入により、属性コストについては統一的な扱いができない。したがってここでは、ヌルシンボルでも属性値が定義されており、かつ距離関数が $p(x, y) = |q(x) - q(y)|$ という差分の形式で表現されている属性に関するヒューリスティック関数を導出

する。この形式の距離関数は、サイズ属性に見られるように実用上重要な形式である。k 番目の属性がこの種類の属性であるとき、そのヒューリスティック関数 $h_k^*(S)$ は

$$h_k^*(S) = \sum_{\Psi} g_k \frac{p_k(x_k^i, y_k^j)}{s_k} \tag{4}$$

と定義できる。ここで $\Psi = \{(i,j)|(i,j) \in Y_1 \times Y_2\}$ である。ここで $h_k^*(S)$ が認容可能条件を満たすように、ノードの可能な組合せ集合 Ψ の中から最小の $h^*(S)$ を推定する。距離関数が $p_k(x_k^i, y_k^j) = |q(x_k^i) - q(y_k^j)|$ の形式の場合、式 (4) は以下のように変形できる。

$$\begin{aligned} h_k^*(S) &= \sum_{\Psi} g_k \frac{|q(x_k^i) - q(y_k^j)|}{s_k} \\ &\geq \left| \sum_{\Psi} \left(q(x_k^i) - q(y_k^j) \right) \right| \frac{g_k}{s_k} \\ &= \left| \sum_{i \in Y_1} q(x_k^i) - \sum_{j \in Y_2} q(y_k^j) \right| \frac{g_k}{s_k} \end{aligned} \tag{5}$$

ここで $|a| + |b| \geq |a + b|$ という関係を用いている。式 (5) は、真のマッチングコストよりも小さい推定値を常に与えるため認容可能条件を満たしている。しかも式 (5) は、まだマッチングされていないノード集合の属性値を個々に合計し、その差の絶対値を推定値とするという、単純で実用的な形式になっている。そこで式 (5) を認容可能条件を満たすヒューリスティック関数として提案する。

4.2 アークのヒューリスティック関数

アークについてもノードとほぼ同様の方法で $h^*(S)$ を計算できるので詳しくは述べない。しかしアークの場合は $h^*(S)$ の計算コストが大きい割にはあまりよい推定値が得られない。このため以後の実験ではアークのヒューリスティック関数は用いないこととする。

5 最大許容コスト

本研究は類似画像検索への応用を目的として、グラフマッチングの高速化を目指している。そこで類似検索の問題構造をうまく活用して高速化を図る。さてデータベースの N 個のグラフの中から、類似検索で上位 M 個のグラフを検索する問題を考えよう。そして既に検索キーとデータベース中の L 個 $(N > L \geq M)$ のグラフとのマッチングが終了しているとし、その時点での検索順位が上位 M 番目のグラフマッチングコストを C_M とする。さて $L+1$ 個目のグラフと検索キーとのマッチングでは、そのマッチングコストが C_M 以内かどうかだけに関心がある。もしマッチングコストが C_M を越えれば $L+1$ 個目のグラフは上位 M 位以内に検索されず、それならばマッチングを即座に中断すべきだろう。本論文ではこのコスト C_M を最大許容コストと呼ぶ。

最良優先探索の場合は、評価関数が $f^*(S) \leq C_M$ の状態、つまりマッチングを続ける価値がある状態しか優先順位付き待ち行列に登録しないという方法を使う (図 2 の 6 を参照)。すると、もし終了状態の一つが待ち行列内になければ、このマッチングの類似順位はたかだが M 位以下であると判定できるため、即座にマッチングを中断することができる。それに対して深さ優先探索と幅優先探索の場合は、探索経路の途中で $f^*(S) > C_M$ となった時点でこの情報を枝刈りに使うことができる。しかしそこでマッチングを中断することはできず、他の経路の可能性が尽きるまで探索し続けなければならない。

また C_M は検索が進むにつれて徐々に小さくなるが、特に検索初期段階で C_M を小さくできれば高速化の効果は大きい。N 個のグラフをマッチングする順序を記述したリストをマッチングリストと呼んでいるが、このリストの順序が実際は検索時間に大きな影響を与える。

6 実験および考察

6.1 類似画像検索の階層的モデル

著者らは現在、表 2 のような階層的モデルを提案し、主観的類似度に基づいた類似画像検索の研究をすすめている [4, 5, 6]。この階層的モデルの中で、本論文のグラフマッチングの高速化の研究は、第 5 レイヤ—2 画像間の類似度をグラフマッチングによって計算するレイヤ—に相当するものである。

6.2 実験の概要

今まで述べてきた、A^* アルゴリズム ・認容可能条件を満たすヒューリスティック関数 ・最大許容コスト、というすべての高速化手法を採り入れたグラフマッチングアルゴリズムを図 2 に示す。このように本論文で提案する図 2 のアルゴリズムは、グラフの階層性と高速化の観点から A^* アルゴリズムを改良したものである。また優先順位付き待ち行列には改良した 2 分探索木を用い、待ち行列の管理についても高速化のための改良をおこなった。

このアルゴリズムを用いて以下の実験をおこなう。1) 1027 枚の画像を階層化属性付き関係グラフに変換し画像データベースとする。2) その中から 1 個のグラフを検索キーとして選びだす。3) 検索キー自身を含めて 1027 個の中から上位 M 個を検索する時間を計測する。ここで計算時間としてはグラフマッチング部分のみに要する計算時間を測定し、入出力に要する時間などは含めていない。また 20 回の試行の平均時間を記録した。実験に用いたコンピュータは IBM SP-2 である。

本研究で用いた階層化属性付き関係グラフの平均的な構造は以下のようになる。ノードの平均個数 :上層 2.41 個、下層 6.32 個に対して、アークの平均個数 :上

Input: 階層化属性付き関係グラフ H_1 と H_2、および最大許容コスト C_M。

Output: H_1 と H_2 のグラフマッチングコスト (非類似度)、およびマッチングされたノードペアを記録した最適な経路。

Steps: 1. 出発状態 s を優先順位付き待ち行列 OPEN に登録する。
2. OPEN に関して
(a) 評価関数が最小の状態があればその状態を S としそれを OPEN から削除する。
(b) もし OPEN が空ならば失敗として 8 へ。
3. もし S が終了条件を満たすならば、S から s を指すポインタをたどることによって得られる解とともに成功として 8 へ。
4. 状態 S を展開し S のすべての子ノード S' を生成する。このとき
(a) もし S' が下位グラフを持っているならば、
i. もし下位グラフのマッチングが初めてであれば、下位グラフどうしのマッチングコスト $d(S')$ を再帰的に計算する。これが失敗ならば $d(S') = C_M$ とする。配列の S' に対応する要素に $d(S')$ を登録する。
ii. そうでなければ、配列の S' に対応する要素から過去のマッチングコスト $d(S')$ を読み出す。
(b) そうでなければ $d(S') = 0$ とする。
5. 状態 S' で新たに生じるコスト $c(S')$ を計算し $g(S') = g(S) + c(S') + d(S')$ とする。さらに状態 S' からゴールまでのコスト $h^*(S')$ を推定する。$f^*(S') = g(S') + h^*(S')$ を計算。
6. もし $f^*(S') \leq C_M$ ならばそれらを OPEN に登録する。そして OPEN を評価関数の値にしたがって再整列する。
7. 2 に戻る。
8. 上位グラフがあれば上位に戻る。なければ終了。

図 2: A^* アルゴリズム ・認容可能条件を満たすヒューリスティック関数 ・最大許容コストのすべての高速化手法を採り入れたグラフマッチングアルゴリズム。優先順位付き待ち行列 OPEN には、改良した 2 分探索木を用いている。

表 2: 類似画像検索システムの階層的モデル。本研究が扱う範囲は第 5 レイヤである。

	レイヤ	目的	規準	出力表現	方法
6	理解レイヤ	人間の主観を最適に反映する類似度空間の構成	例示画像と類似検索上位画像との適合度 ⇒ 最大	主観的な画像間類似度	教師付き / 強化学習 ・遺伝的アルゴリズム
5	**認識レイヤ**	**2 画像間の類似度の計算**	**グラフマッチングコスト ⇒ 最小**	**画像間類似度**	**グラフマッチング ・A^* アルゴリズム**
4	記述レイヤ	プリミティブ間の関係の構築	ゲシュタルト法則 ⇒ 最適	階層化属性付き関係グラフ	プレグナンツの法則
3	特徴レイヤ	画素集合の特徴をプリミティブを用いて表現	構成要素の単純さ + 物体への当てはまり ⇒ 最大	プリミティブ	形状分解 ・超楕円
2	統計レイヤ	物理世界の統計的性質に基づいた画素分類	最大尤度 + 自由パラメータ数 ⇒ 最小	混合密度	EM アルゴリズム ・情報量規準
1	観測レイヤ	物理世界を観測	測定誤差 ⇒ 最小	画像 (画素値配列)	受動型光学センサ

表 3: 1027 枚のグラフマッチングの計算時間 (20 回の試行の平均、単位は秒)。(a) 両方の高速化手法を用いた場合、(b) ヒューリスティック関数だけを用いる場合、(c) 最大許容コストだけを用いる場合、(d) 両方とも用いない場合。

	最良優先探索		幅優先探索		深さ優先探索	
	$M = 10$	$M = 1027$	$M = 10$	$M = 1027$	$M = 10$	$M = 1027$
(a)	**5.52**	**14.68**	**6.07**	**27.00**	**30.68**	**60.83**
(b)	**14.57**	**14.57**	**26.89**	**26.90**	**60.87**	**60.84**
(c)	**6.90**	**15.54**	**7.28**	**23.83**	**45.27**	**69.33**
(d)	**15.40**	**15.40**	**23.74**	**23.73**	**69.29**	**68.61**

層 1.42 個、下層 3.91 個。次にシンボルはノードもアークも各 1 個ずつ。また属性はノードの個数 :上層 1 個、下層 3 個に対して、アークの個数 :上層 1 個、下層 1 個である。次にシンボルと属性に関する重み係数の具体的な値は、主観的な類似度を反映するように最適化した重み係数を用いた [4]。この計算は先の階層的モデルの表 2 の第 6 レイヤに相当するので、本論文では計算方法については詳細に述べない。

6.3 実験の結果

3 種の探索方法を 4 通りの最適化について実験した。また検索枚数も $M = 5$ と $M = 1027$(全部) の 2 通りで実験した。その結果を表 3 にまとめる。

探索方法による比較 どの最適化手法に関しても、最良優先探索は他の探索方法を上回る効率を示している。これは、節点を展開するのに要する計算コストが、優先順位付き待ち行列の管理に必要な計算コストを上回ったことが原因である。一方、深さ優先探索の性能は他と比べてかなり悪い。この原因については 1) 深さ優先探索がグラフマッチング問題に適さない、2) プログラムの最適化が十分でない、という原因が考えられる。結論としては、このようなグラフマッチング問題に対して最良優先探索が有効であることを確認した。

認容可能条件を満たすヒューリスティック関数による効果 この関数による効果は表 3 を見る限りではそれほど顕著ではないが、(a) と (c) または (b) と (d) を比べればわかるように、最良優先探索では $M = 5$ の場合で約 20%の高速化を達成している。ただし他の探索方法では速度が低下している場合もある。これはヒューリスティック関数の推定値が、節点の展開回数を削減するためには大きな効果がなかったことを意味している。

最大許容コストによる効果 $M = 1027$ のように全部のデータに順位付けする類似検索の場合は、最大許容コストの効果がないことは明らかだろう。実際に表 3 の (c) と (d) を比較すると、$M = 1027$ の場合は検索時間は最大許容コストと無関係であるが、$M = 5$ の場合には検索時間に劇的な改善が見られる。ただしこの結果は、前もって最適に整列されたマッチングリストを用いて測定されたものである。つまり最大許容コストによる効果の上限を示す。一般的には異なる重み係数を用いて検索した前回の検索順位で整列したマッチングリストを用いるため、最大許容コストによる速度向上効果は実際にはもう少し割り引かなければならない。

7 まとめ

本研究で提案した高速化手法をすべて活用することにより、グラフマッチングの計算時間を、高速化を用いない場合に比べてわずか 35%にまで削減できた。また検索時間自体も 5 秒程度まで短縮することで、グラフ構造を活用した類似画像検索にも本手法が有効であることを示した。今後はデータベースのグラフの数をさらに増やし、「スケールアップ」の問題に対して本手法がどの程度有効かを検証することが大きな課題である。またさらに有効なヒューリスティック関数の導出によって、探索時間をさらに短縮することも今後の課題となる。

参考文献

[1] Bunke, H. and Allermann, G. **Inexact Graph Matching for Structural Pattern Recognition**. *Pattern Recognition Letters*, Vol. 1, pp. 245–253, 1983.

[2] Nilsson, N.J. **Principles of Artificial Intelligence**. Springer-Verlag, 1982.

[3] Stewart, B.S., Liaw, C.F., and White, C.C. **A Bibliography of Heuristic Search Research Through 1992**. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 24, No. 2, pp. 268–293, 1994.

[4] 北本朝展, 高木幹雄. **パイプライン型遺伝的アルゴリズムによる模擬育種法を用いた類似画像検索規準の学習**. **信学技報**, Vol. IE,, June 1996.

[5] Kitamoto, A. and Takagi, M. **Retrieval of Satellite Cloud Imagery Based on Subjective Similarity**. In *The 9th Scandinavian Conference on Image Analysis*, pp. 449–456, Uppsala, June 1995.

[6] 北本朝展, 高木幹雄. **ミクセルが存在する場合の混合密度推定**. **信学技報**, Vol. PRU95-202, pp. 33–40, 1996.