社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2007－DBS－143（81）
2007／7／4

Interlace定理に基づく多分木を用いたグラフの索引手法

高橋 俊介† 片山 薫††

† 首都大学東京システムデザイン研究科 〒191-0065 東京都日野市旭が丘6-6
E-mail: †takahasi-siyunsuke@ed.tmu.ac.jp, ††kaoru@comp.metro-u.ac.jp

あらまし グラフデータベース内の大量のグラフ集合から，与えられた入力グラフを含むようなグラフを効率的に発見することは重要であり，グラフマイニングなどに利用されている．しかし，部分グラフ同型判定問題はNP完全であり多くの計算コストを必要とする．このため，部分グラフ同型判定を実施する前に，予め対象のグラフ集合から部分グラフを含まないグラフを除いておくことが有効である．本稿では，固有値に基づく多分木インデクスを利用してグラフを除去する手法を提案する．

キーワード グラフ，固有値，Interlace定理，索引

An Indexing Method for Graph Using Multi-way Tree Based on the Interlace Theorem

Shunsuke TAKAHASHI† and Kaoru KATAYAMA††

† Graduate School of System Design,Tokyo Metropolitan University Asahigaoka 6–6, Hino-shi, Tokyo, 191–0065 Japan
E-mail: †takahasi-siyunsuke@ed.tmu.ac.jp, ††kaoru@comp.metro-u.ac.jp

Abstract It is important to find out the graphs which include the provided input graph efficiently from the graph set in graph database. However,the Subgraph Isomorphism problem is NP-complete and needs a lot of calculation costs. So it is effective to remove the the graphs not containing input graph from the graph set before proceeding it. We propose the method which removes the graphs using Multi-way Tree index Based on Eigenvalues.

Key words Graph,Eigenvalue,Interlace Theorem,indexing

1. はじめに

近年，情報社会の発展とともに大量のデータを扱う問題が増えてきている．我々はその中で，グラフと呼ばれるデータ構造に着目し研究を行っている．グラフデータベースはパターン認識やコンピュータビジョン[5]，類似検索[2]などの情報学の多くの分野で利用されており，データベース中のグラフデータの中から検索対象を効率よく発見するための研究が行われている．しかし，2つのグラフの間の包含関係を調べる問題は部分グラフ同型判定問題と呼ばれていて，NP完全に属する問題であることが知られている．よって，大量のデータを扱う場合には膨大な計算コストが必要とされるため，部分グラフ同型判定の前に効率的にグラフを除くことで計算コストを減らすことが有効であると考えられている．大量のグラフ集合から，問い合わせ（入力グラフ）を部分グラフとして含まないグラフを検索することを考える．グラフは隣接行列と呼ばれる正方行列で表現することができる．その行列の固有値と，行列の包含関係を調べるための定理にInterlace定理[4]がある．この定理を用いることであるグラフが他のグラフの部分グラフでないかを判定することが可能である．

以前，我々はグラフの固有値を基にした完全2分木を用いてグラフを索引する手法を提案した．[9] 完全2分木にはグラフ集合の固有値の中央値を蓄えておく．問い合わせ（入力グラフ）が与えられた時，入力グラフの固有値をもとに木を走査することにより，グラフ集合から入力グラフを部分グラフとして含まないグラフを削除することができる．

本稿では，2分木を用いた索引の応用として多分木をインデクスとするグラフの索引手法を提案し，実験を行うことで提案手法の性能を評価する．さらに，Interlace定理の誤りを防ぐため，木に蓄えられた値の幅を固有値の誤差が超えないように計算の精度を決定する．また，固有値の比較を逆にすることで逆の問題，すなわち入力グラフに含まれているグラフを見つける問題にも対応することができることを確認した．

2. 関連研究

部分グラフ同型判定問題に関する研究として，VF2[7]や

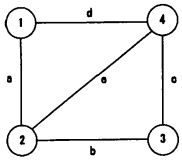

図1 グラフG

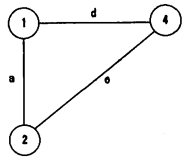

図2 グラフH

Ullmann [6] による手法がある．VF2はグラフ同型判定，部分グラフ同型判定の両方を行うことができ，使用するメモリ量が少ないため大規模グラフに適用できるという利点を持っている．Ullmannの手法はバックトラックと呼ばれる手続きを用いて部分グラフ同型判定を行う手法である．

グラフの索引に関する研究として，Yanら [3] のgIndexと呼ばれる手法がある．これはグラフ集合から頻出する部分を検索し，それらを用いてグラフに索引をつける．また，グラフ構造のデータだけでなくXMLにも適用できるKaushikら [8] の手法は，2つの頂点のBisimilarityを利用してインデクスパスを生成する．

3. Interlace定理とグラフ

3.1 Interlace定理

定理[Interlace Theorem [4]] S を $S^TS = I$ を満たす $n \times m$ の実数行列であるとする．対称行列 A とその n 個の固有値 $\{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ $(\alpha_1 \leqq \alpha_2 \leqq \ldots \leqq \alpha_n)$ について，行列 $B = S^TAS$ を考える．B は m 個の固有値 $\{\lambda_1, \lambda_2, \ldots, \lambda_m\}(\lambda_1 \leqq \lambda_2 \leqq \ldots \leqq \lambda_m)$ を持つ対称行列である．このとき，これらの固有値について以下の関係式が成り立つ．

$$\alpha_i \leqq \lambda_i \leqq \alpha_{i+(n-m)} (i = 1, \ldots, m) \tag{1}$$

この定理は逆は成り立たない．すなわち，2つの行列 A と B が関係式 (1) を満たしたとしても，必ずしも A と B が $B = S^TAS$ を満たすとは限らない．しかし，関係式 (1) を満たさない場合は，A と B は必ず $B = S^TAS$ を満たさない．

3.2 グラフとその行列表現

グラフは $G = (V, E)$ で定義される．V は頂点の集合であり，E は枝の集合である．本稿ではラベル付きの無向グラフを考える．

定義 $G = (V, E)$ $(V = \{v_1, v_2, \ldots, v_m\}, E = \{v_iv_j | v_i, v_j \in V\})$ とする．G の隣接行列 K とは各成分 K_{ij} が以下の性質を満たす正方行列である．$l(v_k)$，$l(v_iv_j)$ はラベルを表している．

$$K_{ij} := \begin{cases} l(v_k) & (i = j = k) \\ l(v_iv_j) & (i \neq j, v_iv_j \in E) \\ 0 & (i \neq j, v_iv_j \notin E) \end{cases} \tag{2}$$

[例 1] 図1にグラフ G を示す．G の隣接行列 M_G は式 (3) で表される．

$$M_G = \begin{bmatrix} 1 & a & 0 & d \\ a & 2 & b & e \\ 0 & b & 3 & c \\ d & e & c & 4 \end{bmatrix} \tag{3}$$

```
Procedure SequentialProcess(G,A)
1:for i ← 1 to n
2:   do dif ← length[G_i] - m
3:   if dif < 0
4:     then flag ← 1
5:     else flag ← 0
6:     for j ← 1 to m
7:       do if G_i[j] > A[j] or A[j] > G_i[j + dif]
8:         then    flag ← 1
9:   if flag = 0
10:    then G_i は候補グラフである
11:    else G_i は候補グラフではない
```

図3 アルゴリズム SequentialProcess

また，図2に G の誘導部分グラフ H を示す．グラフ H の隣接行列 M_H は式 (4) で表わされる．

$$M_H = \begin{bmatrix} 1 & a & d \\ a & 2 & e \\ d & e & 4 \end{bmatrix} \tag{4}$$

行列 M_G と M_H はある行列 S について $M_H = S^TM_GS$ となりInterlace定理を満たす．この関係より，一方のグラフが他方のグラフの誘導部分グラフではないかを判定することが可能である．一般の部分グラフの場合には，グラフの接続行列を利用した行列表現を用いることにより誘導部分グラフの場合と同様に処理することが可能である．[1]

4. 提案手法

4.1 逐次処理

以下では，グラフ集合に含まれるグラフの隣接行列とその固有値が与えられているものとして話を進める．

逐次処理とは，n 個のグラフ $G = \{G_1, G_2, \ldots, G_n\}$ について，m 個の固有値を持つ入力グラフ A とInterlace定理を判定する処理のことである．ただし，A の固有値は A が入力された後に計算して求める．ここで，A の i 番目の固有値を $A[i]$，G の i 番目のグラフ G_i の j 番目の固有値を $G_i[j]$ と表すこととする．Interlace定理を判定する前処理として，グラフ G_i の固有値数から A の固有値数を引いた値 dif を計算する．dif が負値になる場合，Interlace定理を判定する前に G_i が A を含まないことがわかる．なぜなら，A より頂点数が少ないグラフに A が含まれることはないからである．逐次処理のアルゴリズムを図3に示す．図3において $length[K]$ はグラフ K の固有値数を表している．

4.2 提案手法の原理

提案手法ではInterlace定理の関係式 (1) の左側の不等号を利用する（式 (5)）．

$$\alpha_k \leqq \lambda_k \qquad (k = 1, \ldots, m) \tag{5}$$

n個のグラフの集合 $G = \{G_1, G_2, \ldots, G_n\}$ と頂点数が m 個の入力グラフ A が与えられたとする．ただし，G に含まれるグラフはそれぞれ少なくとも m 個の頂点を持つものとする．

固有値を用いたグラフの除去
1:k 番目の固有値について $l-1$ 個の分割値 b_j を決定し，グラフを l 個の部分集合に分割しておく
2: $A[k] < b_1$ の場合，b_1 以上の固有値を持つ G の部分集合を除去する
3: $b_{j-1} \leqq A[k] < b_j (2 \leqq j \leqq l-1)$ の場合，b_j 以上の固有値を持つ G の部分集合を除去する
4: $A[k] \geqq b_{l-1}$ 場合，グラフは除去しない．

図 4　提案手法の原理

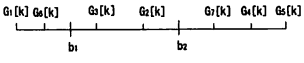

図 5　グラフの固有値のマッピング

A の固有値をすべて計算し，その中から小さい順にいくつかを選ぶ．G に含まれるグラフの固有値と関係式 (5) を利用することによりから A を含まないグラフを除去することを考える．そこために，グラフ集合を固有値の値によっていくつかの部分集合に分割する．いま，k 番目の固有値を用いて G を l 個の大きさが等しい部分集合に分割することを考える．G に含まれているグラフの k 番目の固有値について，グラフを l 等分するために $l-1$ 個の分割値を決め，それらを $b_1, b_2, \ldots, b_{l-1}$ ($b_1 \leqq b_2 \leqq \ldots \leqq b_{l-1}$) とする．これらの分割値に従って，グラフを l 個の大きさが等しい部分集合に分割しておく．例えば，$n=6$，$l=3$ の場合，2 番目と 3 番目の固有値の間の値を b_1，4 番目と 5 番目の固有値の間の値を b_2 として，グラフを大きさが 2 である 3 つの部分集合に分割する．入力グラフの固有値 $A[k]$ と $b_1, b_2, \ldots, b_{l-1}$ を比較し，その結果に応じて A を含まないグラフを G の中から除去する．$b_1, b_2, \ldots, b_{l-1}$ と A の固有値 $A[k]$ の比較は図 4 の手順に沿って行う．

この比較により，少ない数値比較によって G からグラフを除去することができる．提案手法では，この比較を 1 番目の固有値について行ってグラフを除去し，残った G の部分集合それぞれについて 2 番目の固有値を用いてグラフを除去する，というように再帰的にグラフの除去を行う．この処理を何番目の固有値まで実行するかを予めパラメータ *depth* として与えておく．これは，後に述べる多分木インデクスの木の深さに等しい．A の *depth* 個の固有値について分割値との比較を実行し，残ったグラフについて改めて Interlace 定理を順に判定していく．

[例 2]　7 個のグラフからなるグラフ集合 G の k 番目の固有値を 3 つに分割することを考える．グラフの固有値は図 5 のように左から昇順に数直線にマッピングされているとする．2 つの分割値を b_1, b_2 とし，それらによってグラフは $\{G_1, G_6\}$，$\{G_3, G_2\}$，$\{G_7, G_4, G_5\}$ の 3 つの部分集合に分割される．

まず，b_1 より小さい固有値 $A[k]$ が入力される様子を図 6 に示す．この場合は b_1 以上の固有値はすべて除去の対象となるので，$\{G_2, G_3\}, \{G_4, G_5, G_7\}$ を候補から除去する．次に，b_1 以上で b_2 未満の固有値 $A[k]$ が入力される様子を図 7 に示す．この場合は b_2 以上の値の固有値が除去の対象となるので，$\{G_4, G_5, G_7\}$

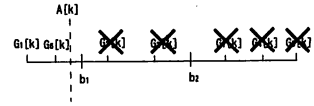

図 6　比較処理の様子 (入力グラフの固有値が b_1 未満の場合)

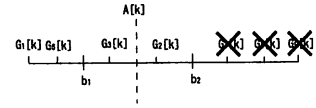

図 7　比較処理の様子 (入力グラフの固有値が 2 つの分割値の間の値である場合)

を候補から除去する．最後に，b_2 以上の固有値 $A[k]$ が入力された場合は，分割値との比較で固有値の大小は判定できないのでグラフは除去しない．

4.3　データ構造の構築

提案手法の問い合わせを実現するために，事前にデータ構造を構築してグラフに索引をつける．データ構造は完全 l 分木 ($l>1$) の形で実現する．

n 個のグラフ集合を $G = \{G_1, G_2, \ldots, G_n\}$ とする．まず，データ構造に必要なノードを考える．完全 l 分木の深さは固有値比較回数のパラメータ *depth* に等しい．深さが *depth* である完全 l 分木に必要なノード数 N は以下の式で求められる．

$$N = \sum_{k=0}^{depth} l^k \tag{6}$$

l 分木の葉のノードとにはグラフの部分集合を，葉でないノードにはあるグラフ集合の $l-1$ 個の分割値 $b_1, b_2, \ldots, b_{l-1}$ を蓄える．G に含まれる全てのグラフの 1 番目の固有値の値から，分割値 $b_1, b_2, \ldots, b_{l-1}$ を求めてそれらの値を木の根に蓄える．すべてのグラフの 1 番目の固有値について，b_1 未満の固有値，b_1 以上 b_2 未満の固有値のようにグラフを l 個に分割する．分割したグラフの集合を l 個の子ノードに送る．このとき，固有値が小さい集合ほど左側のノードに送ることとする．

次に，送られたそれぞれのノードで G の部分集合と 2 番目の固有値について根と同様の処理を行う．これによってグラフの l 個の部分集合はさらにそれぞれ l 個に分けられて，それぞれの子ノードに送られる．これを *depth* 回だけ繰り返す．結果として，木の葉のリスト $list_g$ にはそれぞれほぼ同じ数の G の部分集合が蓄えられることになる．

[例 3]　深さが 2 である 3 分木にグラフを格納することを考える．グラフ集合を $G = \{G_1, G_2, \ldots, G_9\}$ とする．ただし，簡単のためにすべてのグラフは同じ固有値数 $k(>depth)$ であるとし，さらに $G_1[i] < G_2[i] < \ldots < G_9[i] (1 \leqq i \leqq k)$ が成り立つとする．

必要なノード数は式 (6) より 13 となる．図 8 に 1 段目の処理の様子を示す．ただし，図 8 のグラフは添え字の番号のみで表している．G の 1 番目の固有値の分割値 b_1, b_2 により G は G_{11}, G_{12}, G_{13} に分けられて次段に送られる．2 段目の各ノードも同様に 2 つの分割値を設定することで G_{11}, G_{12}, G_{13} をそ

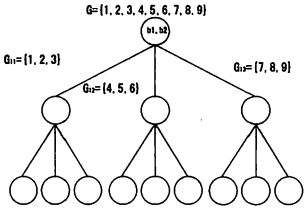

図 8　3 分木の構築 (1 段目)

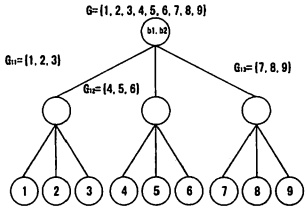

図 9　3 分木の構築 (2 段目)

れぞれ 3 つの部分集合に分割する．以上の操作によって，G は 9 個の大きさがほぼ等しい部分集合に分けられる．3 段目は葉ノードなので，送られてきた部分集合を $list_g$ に格納して終了となる．以上の様子を図 9 に示す．

4.4　問い合わせの実行

実際に完全 l 分木を用いて問い合わせを行う．この節では 4.3 節で示した 3 分木の例を用いて問い合わせを実行する．3 分木のあるノード T に接続している 3 つの子ノードを左から順に $left[T], middle[T], right[T]$ とする．

まず，与えられた入力グラフの固有値を計算する．固有値を計算する際，2 分法を用いて任意の精度で計算を行う．このとき，木のあるノードに蓄えられた複数の分割値の幅を固有値の誤差が超えないように精度を決定する．これは分割値との比較の際に，固有値の誤差によって Interlace 定理の判定が誤ってしまうのを防ぐためである．

完全 3 分木の k 段目において蓄えられている分割値 b_1, b_2 と入力グラフ A の k 番目の固有値を比較する．

- $A[k] < b_1$ のとき，子ノード $left[T]$ を走査する．
- $b_1 \leqq A[k] < b_2$ のとき，子ノード $left[T], middle[T]$ を走査する．
- $b_2 \leqq A[k]$ のとき，子ノード $left[T], middle[T], right[T]$ を走査する．

これを葉に到達するまで繰り返し，到達した葉に蓄えられているグラフ集合を全て報告する．報告された全てのグラフについて逐次的に Interlace 定理を判定する．

特別な場合として，木の段数よりグラフ集合のあるグラフの固有値が少ない場合は到達した最後のノードに $list_s$ を持たせてグラフを蓄えるようにする．木の段数より固有値が少ない A が与えられた場合は，走査が止まったノードの $list_s$ に蓄えられたグラフと，そのノードより下の葉に含まれるすべてのグラフについて Interlace 定理を判定する．

また，逆の問題としてグラフ集合の中から入力グラフの部分グラフにならないグラフを除去する場合は，分割値との比較の際の不等号を逆にすることで除去することが可能である．その後式 (1) を判定する際には，入力グラフとグラフ集合のグラフを逆にして判定すればよい．

Procedure ScanTree(T,A,n)	
1:	if $n \geqq$ length[A]
2:	then SequentialProcess($list_s[T]$,A)
3:	LinearInquiry($list_m[T]$,A)
4:	ScanTree($right[T]$,A,n+1)
5:	ScanTree($middle[T]$,A,n+1)
5:	ScanTree($left[T]$,A,n+1)
6:	if $n = depth$
7:	then SequentialProcess($list_g[T]$,A)
8:	else
9:	then if $b_2 \geqq A[n]$
10:	then ScanTree($right[T]$,A,n+1)
11:	else if $b_1 \geqq A[n]$
12:	then ScanTree($middle[T]$,A,n+1)
13:	ScanTree($left[T]$,A,n+1)

図 10　アルゴリズム ScanTree(3 分木の場合)

完全 3 分木の走査アルゴリズムを図 10 に示す．図 10 の T は木のノードであり，初期値は木の根である．また，n は木の深さあり，初期値は根の深さと同じ 0 である．

5.　実験と評価

5.1　実験内容と環境

提案したインデクスを用いた検索方法について以下の項目を評価するための実験を行った．各実験において，多分木は 2 分木から 6 分木までを使用した．

- 頂点数の大小と走査時間，除去率の関係
- 候補グラフ数と走査時間，除去率の関係
- 木の深さとグラフの除去率の関係

ここで，除去率とはグラフ数全体に対するインデクスで除去するグラフ数の割合のことであり，走査時間とは入力グラフの固有値計算を除く処理時間のことである．また，候補グラフ数は Interlace 定理の判定を行った結果部分グラフの候補として残ったグラフの数とする．

グラフはグラフ数，頂点と枝の平均数，頂点と枝のラベル数，頂点数と枝数の広がりを設定してランダムに生成した．今回，枝のラベル数を変化させた場合についても実験を行った．枝ラベルは 10 個の場合を基準とし，半分の 5 個の場合のデータを生成した．

5.2　実験結果と考察

5.2.1　頂点数の大小と走査時間，除去率の関係

グラフ集合の平均頂点数を 100 で一定として，グラフ数を変化させたときの走査時間と除去率について考える．表 1 は木の深さを 4 として，入力グラフの頂点数が 50，100，130 のときにグラフ数を 1000 個から 1000000 個まで変化させたときの平均の除去率を示している．グラフ集合の平均頂点数に対して入力グラフの頂点数が小さいほど部分グラフになる可能性は高く，頂点数が大きいほど可能性は小さい．このことは表 1 の結果か

表 1　入力グラフの頂点数と木の分岐数の変化と除去率

	入力グラフの頂点数		
	50個	100個	130個
2分木	12.50%	37.40%	75.00%
3分木	18.20%	42.10%	79.40%
4分木	17.20%	42.60%	82.80%
5分木	36.73%	46.00%	86.81%
6分木	18.98%	44.30%	86.95%

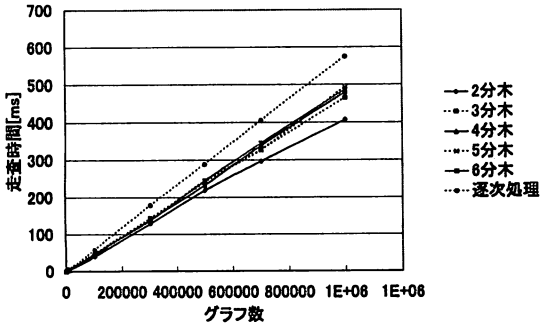

図 11　グラフ数の変化と走査時間（入力グラフ：頂点数 50，グラフ集合：平均頂点数 100，木の深さ：4）

らも読み取ることができる．また，表 1 からは木の分岐が増えるほど除去率が大きくなってはいない場合があることがわかる．頂点数 130 の場合は分岐数が大きいほど除去率は大きいが，頂点数 50，100 の場合はそうはなっていない．これは，2 分木と 4 分木では共通の分割値が存在するが，今回用いた他の多分木は分割値は分岐によって全く異なることが原因であると考えられる．このため，分岐数を増やしても逆に除去できるグラフ数を減らしてしまう可能性がある．

次に，走査時間について考える．入力グラフの頂点数と走査時間の関係は，頂点数が大きいほど逐次処理よりも提案手法が有効となることがわかり，以前の提案手法 [9] とほぼ同じ傾向となった．結果の例として，頂点数が 50 の場合を図 11 に示す．図 11 より，すべての多分木が逐次処理より短い時間で処理できていることがわかる．また，グラフ数が大きいほど提案手法が有効であることも確認できた．頂点数が 100，130 になると，逐次処理と提案手法の差はより大きくなる．

5.2.2　候補グラフ数と除去率の関係

グラフ集合のグラフ数を 100000 で一定にしたまま平均頂点数を変化させることで，候補グラフ数と除去率，走査時間の関係を考える．図 12 に入力グラフの頂点数が 50，木の深さ 4 のときの平均頂点数と除去率の関係を示す．図 12 において，平均頂点数が 50 の時の候補グラフ数が 5550，平均頂点数 200 の時が 49645 であった．図 12 から，どの木においても候補グラフが増えるほど除去率は小さくなっていることがわかる．ただし，どの平均頂点数においても分岐数が大きいほど除去率が高いという結果にはなっていない．しかし，平均頂点数 200 の場合に着目すると 2 分木では全くグラフを除去できていないが他の木では若干ではあるがグラフを除去できていることがわかる．

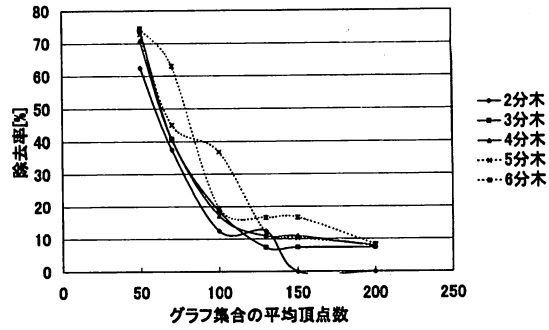

図 12　平均頂点数の変化と除去率（入力グラフ：頂点数 50，グラフ集合：グラフ 100000 個，木の深さ：4）

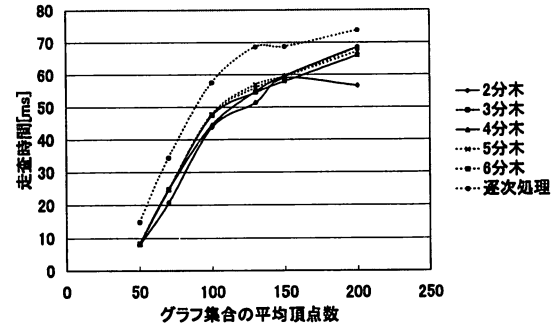

図 13　平均頂点数の変化と走査時間（入力グラフ：頂点数 50，グラフ集合：グラフ 100000 個，木の深さ：4）

候補グラフが全体の約半分を占める場合でも，提案手法を用いることでグラフを除去できることが確認できた．

次に，走査時間についての結果を図 13 に示す．条件は図 12 の場合と同じである．図 13 より，すべての多分木が逐次処理より短い時間で処理できていることがわかる．これは，候補グラフが大きくなっても，提案手法でグラフを除去することにより Interlace 定理の判定回数を減らし，走査時間を減らせているためであると考えられる．この結果からも，提案手法の有効性を確認することができた．

5.2.3　木の深さと除去率の関係

同じ入力グラフとグラフ集合について，木の深さを変化させた場合に除去率がどのように変化するかを考える．候補グラフが少ない場合と多い場合について実験を行った．

- 入力グラフ：頂点数 130，グラフ集合：平均頂点数 100 のグラフ 100000 個，候補グラフ数 27

結果を図 14 に示す．多分木はそれぞれ 100000 個のグラフを細かく分けられる限界まで深さを変化させた．図 14 より，すべての多分木において深さが大きくなるほど除去率が大きくなっていることがわかる．ただし，多分岐になればなるほど深さが限界になるのが早いため，最終的な除去率は 2 分木が最も大きくなっている．このことから，候補グラフが少なくグラフ集合の大きさが既知であるなら 2 分木で最も大きい深さで処理することが一番効率がよいと考えられる．ただし，候補グラフの数

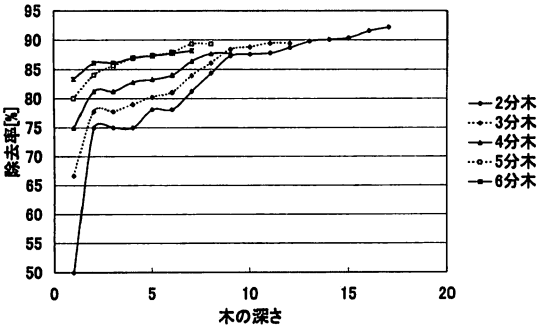

図 14　木の深さの変化と除去率（入力グラフ：頂点数 130，グラフ集合：平均頂点数 100 のグラフ 100000 個，候補グラフ数：27）

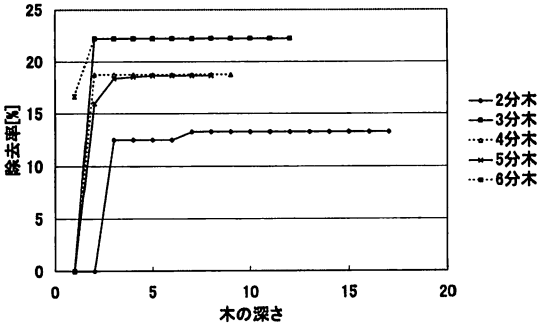

図 15　木の深さの変化と除去率（入力グラフ：頂点数 70，グラフ集合：平均頂点数 200 のグラフ 100000 個，候補グラフ数：56075）

は入力グラフによって変化するので，候補グラフが少ないかを予め判断することは容易ではない．

- 入力グラフ：頂点数 70，グラフ集合：平均頂点数 200 のグラフ 100000 個，候補グラフ数：56075

結果を図 15 に示す．図 15 では分岐数が 6，3，4，5，2 の順で除去率が高くなっているが，いずれの場合もすぐに一定値になってしまっている．この結果から，候補グラフが多い場合は多分木による効率的なグラフの除去ができず，除去率はすぐに飽和してしまうことがわかる．よって，候補グラフが多い場合には木の深さはそれほど重要な要素ではないと言える．

5.2.4　枝のラベル数の変化と除去率との関係

最後に，枝のラベル数と除去率との関係について考える．表 1 の条件と同じ条件で枝ラベルを半分の 5 個にした場合の除去率の測定を行い，その結果を表 2 に示す．表 2 を見ると，頂点数 50 と 100 の場合は除去率は大きくなり，130 の場合は除去率が小さくなっていることがわかる．グラフデータはランダムに生成している．しかし，ひとつの頂点数に対してはどの多分木も同じ傾向であることから，原因はデータのランダム性ではなく他にあると考えられる．考えられる原因のひとつとして，Interlace 定理の判定はグラフの隣接行列の固有値を用いるので，枝ラベルがグラフの固有値に与える影響によって除去率が変化したと考えられる．

表 2　入力グラフの頂点数と木の分岐数の変化と除去率

	入力グラフの頂点数		
	50個	100個	130個
2分木	25.00%	56.26%	68.76%
3分木	22.45%	67.71%	72.66%
4分木	27.37%	64.91%	70.53%
5分木	38.61%	67.64%	72.53%
6分木	28.47%	69.82%	73.82%

6.　まとめと今後の課題

本稿では，部分グラフ同型判定の前処理として判定対象となるグラフを減らすために，固有値に基づく多分木のデータ構造を用いたグラフの索引手法を提案した．また，実験を行うことで提案手法が有効である場合を確認することができた．木の分岐数と除去率の関係がデータによって異なることや，候補グラフ数の大小に応じて有効な処理が異なることも確認することができた．

今後の課題としては，より多くの種類のグラフデータによる実験とその結果の解析が挙げられる．5.2.4 節で枝ラベルが固有値に影響する可能性が考えられたので，その原因を追究したいと考えている．それに加えて，頂点のラベルや枝数，頂点や枝の広がりなどの変化によって除去率や候補グラフ数がどのように変化するのかも検証していきたい．また，実データに対する提案手法の有効性の検証や，さらに効率のよりグラフの手法の考案についても行っていきたいと考えている．

謝辞　本研究の一部は，(独) 日本学術振興会科学研究費補助金基盤研究 (C)(課題番号:19500089) による．

文　献

[1] 長屋 未来, 片山 薫, 石川 博, "大規模グラフを対象とした部分グラフ同型判定における *Interlace* 定理の利用", 電子情報通信学会 第 17 回データ工学ワークショップ DEWS2006,2006,3.

[2] Xifeng Yan,Philip S.Yu and Jiawei Han, *"Substructure Similarity Search in Graph Databases"*, Proceedings of ACM-SIGMOD International Conference on Management of Data,2005.

[3] Xifeng Yan,Philip S.Yu and Jiawei Han, *"Graph Indexing: A Frequent Structure-based Approach"*, Proceedings of ACM-SIGMOD International Conference on Management of Data,2004.

[4] Willem H.Haemers, *"Interlacing Eigenvalues and Graphs"*, Linear Algebra and its Applications,"226-228",593-616,1995.

[5] Olga Veksler, *"Efficient Graph-Based Energy Minimization Methods In Computer Vision"*, Cornell University,1999.

[6] , J. R. Ullmann, *"An Algorithm for Subgraph Isomorphism"*, Journal of the ACM (JACM),23,1,1976,31-42.

[7] Luigi P. Cordella,Pasquale Foggia,Carlo Sansone and Mario Vento, *"An Inproved Algorithm for Matching Large Graphs"*, Proceedings of the 3rd IAPR TC-15 Workshop on Graph-based Representations in Pattern Recognition,Ischia,2001,149-159.

[8] Raghav Kaushik,Pradeep Shenoy,Philip Bohannon and Ehud Gudes, *"Exploiting Local Similarity for Indexing Paths in Graph-Structured Data"*, 18th International Conference on Data Engineering (ICDE'02),2002,129-140.

[9] 高橋 俊介, 片山 薫, 固有値を用いたグラフの索引手法の提案とその実験的評価, データベースと Web 情報システムに関するシンポジウム (DBWeb2006),2006,11.