サブグラフ問い合わせとスーパーグラフ問い合わせに対応した固有値を用いたグラフインデクス手法

宮奥　祥多† 　三井　良太† 　片山　薫†

† 首都大学東京 システムデザイン研究科
E-mail: †{miyaoku-shota,mitsui-ryota}@sd.ac.jp, ††kaoru@tmu.ac.jp

あらまし　大量のグラフデータを効率的に検索するため、さまざまな索引手法が提案されている．高橋らの提案した固有値を用いたグラフ索引手法 [13] では、クエリグラフを部分グラフとして含むグラフを検索するためのものであった．本論文では、さらにクエリグラフの部分グラフとなるグラフも検索することができる索引手法を提案する．また、接続行列を用いた新しい行列表現を用いることでインデクス性能の改善を行った．

キーワード　インデクス，データ構造，グラフデータ処理，半構造データ

An Indexing Method for Processing Subgraph and Supergraph Queries Using Eigenvalues

Shota MIYAOKU†, Ryota MITSUI†, and Kaoru KATAYAMA†

† Graduate School of System Design,Tokyo Metropolitan University　Asahigaoka 6–6, Hino-shi, Tokyo,191–0065 Japan
E-mail: †{miyaoku-shota,mitsui-ryota}@sd.ac.jp, ††kaoru@tmu.ac.jp

1. はじめに

近年，たんぱく質やＤＮＡ，化合物などのデータがグラフで表現されており，グラフは重要なデータモデルとなっている．あるグラフに他のグラフが含まれるかどうかを判定する問題は部部グラフ同型性判定問題とよばれ，コンピュータービジョンやグラフマイニング，画像の類似検索などに応用されている．部分グラフ同型性判定問題はＮＰ完全であることが知られており膨大な計算コストが必要となる．

本研究では一つのインデクスでクエリグラフのサブグラフ，スーパーグラフを検索するインデクス手法の提案をする．スーパーグラフとはクエリグラフを部分グラフとして含むグラフである．これまでに提案されてきたインデクス手法はクエリグラフのサブグラフもしくはスーパーグラフどちらか一方のみを検索するインデクス手法が多い [1], [4]〜[8], [10], [11]．高橋らの提案した固有値を用いたグラフ索引手法も同様にクエリグラフのスーパーグラフを検索するためのものである．その木構造であるインデクスのノード間を双方向リストで結ぶことで同じ索引を用いてサブグラフも検索することが可能となる．またグラフを行列で表現する際に，接続行列 N とその転置行列 N^t の積を用いることでを処理時間，出力される候補グラフ数，メモリ使用量の改善を図った．

本稿の構成は以下のようになっている．2 章ではグラフインデクスや部分グラフ同型性判定に関する研究について述べる．3 章では部分グラフ同型性判定，Interlace 定理とグラフの行列表現について述べる．4 章では提案手法であるインデクス手法とその探索，メンテナンスについて述べる．5 章で実験による提案手法の評価を行い，最後に 6 章でまとめと今後の課題について述べる．

2. 関連研究

グラフインデクスには，主にパスを用いたインデクスと頻出部分グラフを用いたインデクスがある．パスを用いたインデクスには，GraphGrep [7] や 1-index [5],A(k)-index [4]，D(k)-index [6] などがある．GraphGrep は頂点のラベルのパスを用いてハッシュテーブルを作成し，それを用いることでグラフインデクスを行う．1-index と A(k)-index はグラフの bisimilarity を用いてインデクスパスを生成する手法であり，D(k)-index はこの 2 つのインデクスを改良した手法である．

頻出部分グラフを用いる手法には gIndex [10]，FG-Index [1] などがある．gIndex は冗長なものを除去した頻出部分グラフを用いてインデクスを生成する．インデクスによって問い合わ

せを含む候補グラフを探索し，その後部分グラフ同型判定を行う．FG-Index も頻出部分グラフを用いてインデクスを生成する手法であるが，問い合わせが頻出部分グラフの集合に含まれる場合は部分グラフ同型判定が必要ないという利点がある．

また，GCoding [11] というインデクスは Intertlace 定理を用いている．グラフのある一つの頂点を任意に選び，その頂点に隣接する頂点でグラフをつくる．隣接する頂点数と隣接する頂点で作られたグラフの固有値を要素とする GCode と呼ばれるデータ集合を作成する．GCode から S-tree 構造を持つ GCode-Tree を作り Interlace 定理の条件式の一部を用いてグラフインデクスを行う．Ali S らの研究 [8] では有向非環グラフの隣接行列の固有値を用いて，M.Faith Demirci らの研究 [3] ではラプラシアン行列の固有値を用いて Shock Graph の部分グラフ検索のインデクスを作成し画像検索を行う．

部分グラフ同型判定問題に関する研究として，Ul l mann [9] の手法や VF2 [2] がある．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性判定を行う手法である．VF2 は 2 つのグラフの枝の間のマッピングの状態を更新していくことで 2 つのグラフの間の同型写像を作成する．VF2 は Ullmann の手法に比べて，使用するメモリ量が少ないという利点がある．

3. グラフの行列表現と Interlace 定理

本稿において，議論の対象はラベル付の連結無向グラフとする．以降，ラベル付の連結無向グラフをグラフと呼ぶこととする．

3.1 部分グラフ

グラフ g を $(V, E, L_v, L_e, F_v, F_l)$ と定義する．ここで $V = \{v_1, v_2, \ldots v_n\}$ は頂点の集合であり，$E = \{(v_i, v_j) | v_i, v_j \in V, i \neq j (i, j = 1, 2, \cdots n)\}$ は枝の集合である．また，L_v を頂点ラベルの集合，L_e を枝ラベルの集合とする．$F_v : V \to L_v$, $F_e : E \to L_e$ 各頂点と枝にラベルを割り当てる関数とする．

［定義 1］（部分グラフ）2 つのグラフ $g = (V, E, L_v, L_e, F_v, F_l)$ と $g' = (V', E', L'_v, L'_e, F'_v, F'_l)$ が与えられたとき，以下の条件を満たす単射 $f : V' \to V$ が存在すれば g' は g の部分グラフであるという．g' が g の部分グラフであることを $g' \subseteqq g$ と表現する．

（1） $(f(u), f(v)) \in E$.

（2） $F'_v(u) = F_v(f(u))$, $F'_v(v) = F_v(f(v))$.

（3） $F'_e(u, v) = F_e(f(u), f(v))$.

3.2 Interlace 定理

2 つの実数の列 $\alpha_1 \leqq \alpha_2 \leqq \cdots \leqq \alpha_n$ と $\theta_1 \leqq \theta_2 \leqq \cdots \leqq \theta_m$ について考える．$m < n$ とする．以下の式が成り立つとき，$\{\alpha_i\}$ は $\{\theta_j\}$ を interlace するという．

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

［定理 1］ 行列 A，B をそれぞれ n 次と m 次 $(m < m)$ の対称行列とし，それぞれの固有値を $\{\alpha_i\}(\alpha_1 < \alpha_2 < \cdots < \alpha_n)$, $\{\beta_i\}(\beta_1 < \beta_2 < \cdots < \beta_m)$ と書く．この時 $S^T S = I$ を満たす

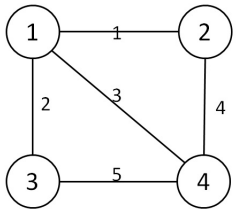

図 1 Graph g

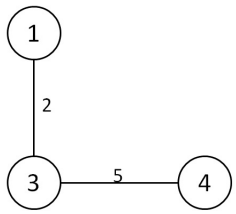

図 2 Graph g'

ある S について $B = S^T AS$ を満たせば B の固有値は A の固有値を interlace する．

2 つの行列 A と B の固有値が式 (1) を満たしたとしても，A と B は必ずしも $B = S^T AS$ を満たすとは限らない．しかし，式 (1) が満たされていない場合は，$B = S^T AS$ を満たす行列 S は存在しないことがわかる．

3.3 接続行列を用いたグラフの表現

グラフ $g = (V, E, L_v, L_e, F_v, F_l)$ が与えられ，g の頂点の集合を $V = \{v_1, \ldots, v_n\}$ とする．これまでわれわれが用いてきたグラフの行列表現 R を以下に示す．

$$R := \begin{pmatrix} P & N \\ N^t & Q \end{pmatrix}$$

P, Q は，それぞれ頂点ラベル，枝ラベルを持つ対角行列である．N は接続行列を表す．この行列表現の次数は $|V + E|$ である．行列の次数を下げることでインデクス構築時間とメモリ使用量の削減するために，接続行列の転置を用いた以下のグラフの行列表現 M を用いる．接続行列は対称行列ではない．しかし，Interlace 定理を使用する場合，行列は対称行列である必要があるので接続行列の転置と接続行列の積を用いる．

$$M = N^t N$$

この行列 M は次数 $|E|$ の対称行列となる．

行列 M で用いる接続行列 N を以下のように定義する．N^v は頂点ラベル，N^e は枝ラベルを用いた接続行列であり，添え字 i, j は行列の i 行 j 列 の要素を表す．

$$N^v_{ij} := \begin{cases} F_v(v_i) & ((v_i v_j) \in E) \\ 0 & (\text{それ以外}) \end{cases}$$

$$N^e_{ij} := \begin{cases} F_e((v_i, v_j)) & ((v_i v_j) \in E) \\ 0 & (\text{それ以外}) \end{cases}$$

グラフ g とその部分グラフを g' としその行列表現を M_g, M'_g とすると，必ず M'_g は M_g の部分行列になる．よって 2 つの行列の固有値が Interlace 定理の条件式を満たすかどうかを調べることで部分グラフで無いかを判定することができる．以後 g' の固有値と g の固有値が Interlace 定理の条件式を満たすことを $g' \subseteqq_i g$ と表す．

［例 1］ 図 1 にグラフ g を示す．図 1 の頂点ラベルを用いた

接続行列 N_g^v，g の接続行列 N_g^v の転置を用いた行列 M_g^v,g' の接続行列の転置を用いた行列 $M_{g'}^v$ は以下のようになる．

$$M_g^v = \begin{bmatrix} 5 & 1 & 1 & 4 & 0 \\ 1 & 10 & 1 & 0 & 9 \\ 1 & 1 & 17 & 16 & 16 \\ 4 & 0 & 16 & 20 & 16 \\ 0 & 9 & 16 & 16 & 25 \end{bmatrix} \quad M_{g'}^v = \begin{bmatrix} 10 & 9 \\ 9 & 25 \end{bmatrix}$$

$$N_g^v = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 0 & 0 & 2 & 0 \\ 0 & 3 & 0 & 0 & 3 \\ 0 & 0 & 4 & 4 & 4 \end{bmatrix} \qquad N_{g'}^v = \begin{bmatrix} 1 & 0 \\ 3 & 3 \\ 0 & 4 \end{bmatrix}$$

行列 S を以下のように定めると $M_{g'}^v = S^T M_g^v S$ となる．g'は g の部分グラフであるが確かに提案した行列表現が Interlace 定理を使える形になっていることが分かる．

$$S = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$$

4. Interlace 定理を用いたインデクス手法

インデクスを用いたグラフ問い合わせは以下の 2 つの段階を経て実行される．

（1） グラフデータベース D に含まれる各グラフの固有値を計算し， インデクスを生成する．

（2） 問い合わせ処理を 2 段階に分けて行う．

（a） 問い合わせ q と (1) で生成したインデクスを用いて，候補グラフの集合 C_q を得る．候補グラフとは，クエリグラフを部分グラフとして含む可能性のあるグラフのことである．

（b） C_q に含まれているすべてのグラフについて，実際に q を含んでいるかを検証する．この処理で，グラフ問い合わせの正解のグラフ集合 D_q を得る．

4.1 Interlace 定理による固有値の大小関係

以下の命題を用いてインデクスを構築する．2 つの固有値列 $\alpha = \{\alpha_1 \leqq \alpha_2 \leqq \cdots \leqq \alpha_n\}$，$\beta = \{\beta_1 \leqq \beta_2 \leqq \cdots \leqq \beta_m\}(n > m)$ が与えられたとする． β が α を Interlace するとする．つまり以下の不等式が成り立つ．

$$\alpha_i \leqq \beta_i \leqq \alpha_{i+(n-m)} (i = 1, 2, \cdots, m)$$

［命題 1］ 固有値列 $\gamma = \{\gamma_1 \leqq \gamma_2 \leqq \cdots \leqq \gamma_l\}(l < m)$ が，γ が β を Interlace するならば，γ は α を Interlace する．
つまり $\beta_i \leqq \gamma_i \leqq \beta_{i+(l-n)}$ ならば $\alpha_i \leqq \gamma_i \leqq \alpha_{i+(l-m)}$ が成り立つ． □

［命題 2］ 固有値列 $\gamma = \{\gamma_1 \leqq \gamma_2 \leqq \cdots \leqq \gamma_l\}(l > n)$ が，α が γ を Interlace するならば，β は α を Interlace する．
つまり $\gamma_i \leqq \alpha_i \leqq \gamma_{i+(l-n)}$ ならば $\gamma_i \leqq \beta_i \leqq \gamma_{i+(l-m)}$ が成り立つ． □

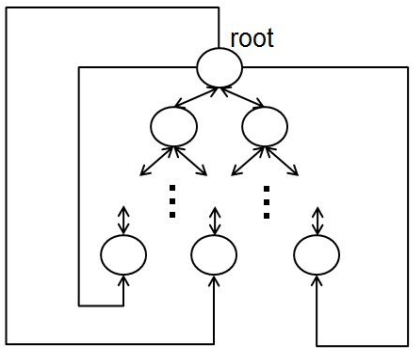

図 3 インデクスの構造

命題 2 も命題 1 [13] と同様に証明できる．
[証明] $\alpha_i \leqq \beta_i$ なので, 以下の不等式が成り立つ．

$$\gamma_i \leqq \beta_i \tag{2}$$

命題の仮定より $\gamma_{i+(n-m)} \leqq \alpha_{i+(n-m)} \leqq \gamma_{(i+(l-m)+(n-m))}$, $\gamma_{(i+(l-m)+(n-m))} = \gamma_{i+(l-m)}$, $\beta_i \leqq \alpha_{i+(n-m)}$ なので以下の不等式が成り立つ．

$$\beta_i \leqq \gamma_{i+(l-m)} \tag{3}$$

(2)，(3) より $\gamma \leqq \beta_i \leqq \gamma_{i+(l-m)}$ が成り立つ． ■

4.2 インデクスの構築

インデクスの構築ではに Interlace 定理によるグラフの固有値の大小関係を利用する．しかし，Interlace 定理を利用して部分グラフ同型性判定を行う際，グラフサイズの差が大きくなると部分グラフでない候補グラフが多くなってしまう．そこで宮奥らは枝ラベルによるグラフ分割を用いたインデクス手法を提案した [12]．本稿でもこの手法を用いる．グラフ g の枝ラベル i を持つ分割グラフ g^i，グラフ g' の枝ラベル i を持つ分割グラフを g'^i とする．$g^i \not\subseteq_i g'^i$ のとき g は g' の部分グラフでないことが分かる．分割グラフを用いたインターレース定理の利用は， まず全てのラベル i 分割グラフをすべての i について $g^i \leqq_i g'^i$ が成り立ち，元のグラフ g と g' について $g \leqq_i g'$ が成り立つとき $g \leqq_{ia} g'$ と表す．2 つのグラフ g, g' について $g \leqq_{ia} g'$ ならば g' は g の子ノードとして接続する．分割グラフは元のグラフが格納されているノードに一緒に格納する．最後に root からすべての葉ノードを親ノードとして接続する．これはインデクスの辿りかたによってサブグラフ，スーパーグラフの問い合わせに対応するためである．詳しくは後述する．インデクスの構築例を図 4 に示す．以下では，インデクスのあるノード p に格納されているグラフを g_p，子ノード集合を $childNode(p)$，ある子ノードを $childNode(p)_i$，親ノード集合を $parentNode(p)$，ある親ノードを $parentNode(p)_i$ と表す．インデクスの構築アルゴリズム $ConstructIndex$ を Algorithm1，Algorithm2 に示す．
［例 2］ 表 2 に示す固有値を持つグラフ $g1, g2, g3$ を用いてインデクスを構築する．ただし簡略化のため分割したグラフの固有値は用いず, 元のグラフの固有値のみを用いてインデクスを構築する．表 2 のグラフについて Interlace 定理の条件式を調べ

Algorithm 1 ConstructIndex(D)

Input: グラフデータベース D

Output: インデクス

1: インデクスの root ノードを生成
2: **for** each $g \in D$ **do**
3: CreateNode($root$,g)
4: **end for**

Algorithm 2 CreateNode(nod,g)

Input: インデクスのあるノード nod
インデクスに挿入するグラフ g

1: **if** nod が子ノードを持たない **then**
2: g のノードを nod の子ノードとする.
3: **else**
4: **for** $i = 1, 2, \cdots |childNode(nod)|$ **do**
5: **if** $g_{childNode(nod)_i} \subseteq_{ia} g$ **then**
6: g のノードを nod の子ノード かつ,
$childNode(g)_i$ の親ノードとする.
7: **else if** $g \subseteq_{ia} g_{childNode(nod)_i}$ **then**
8: CreateNode($childNode(nod)_i, g$)
9: **end if**
10: **end for**
11: g のノードを nod の子ノードとする.
12: **end if**
13: 子ノードを持たないすべてのノードを root の親ノードとする.

ると,$g2 \subseteq_i g1$.$g3 \subseteq_i g1$. $g3 \not\subseteq_i g2$ なので $g2$,$g3$ はそれぞれ $g1$ の子ノードとなる. 最後に $root$ から逆にインデクスを辿れるようすべての子ノードを持たないノードを root の親ノードとする. 構築したインデクスを図 4 に示す.

表 1 グラフとその固有値

g	eig(g,1)	eig(g,2)	eig(g,3)	eig(g,4)	eig(g,5)	eig(g,6)
g1	−3	−1	1	4	10	
g2	−2	0	3	8		
g3	0	2	9			
q	−2	−1	0	1	2	9

4.3 サブグラフ問い合わせ

命題 1 よりグラフデータベース D に含まれるグラフ g と問い合わせ q が，$q \not\subseteq_{ia} g$ であるならば，D に含まれる $g' \subseteq_{ia} g$ であるすべてのグラフ g' は問い合わせとの間で Interlace 定理を満たさない．このことからインデクスを辿ることですべてのグラフについて Interlace 定理の条件式を判定せずに部分グラフでないグラフの除去を行うことができる．問い合わせ q についてグラフ問い合わせを行う場合インデクスの root を始点として子ノードの方向に探索する．インデクスのあるノード p に格納された g とクエリグラフ q が $q \subseteq_{ia} g$ かどうかを判定する．$q \not\subseteq_{ia} g$ であれば g から root までのすべてのグラフが候補グラフではない．サブグラフ問い合わせアルゴリズム ProcSubgraphQuery を Algorithm3 に示す.

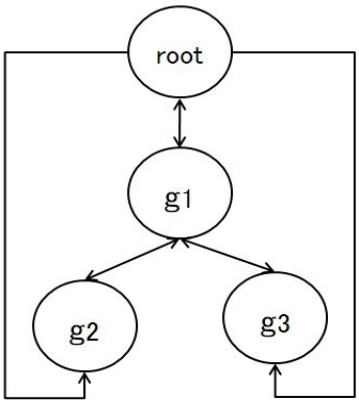

図 4 表 2 のグラフのインデクス

Algorithm 3 ProcSubgraphQuery(nod,q)

Input: インデクスのノード nod (初期値は $root$)
クエリグラフ q

Output: 候補グラフ集合 C_q

1: **for** $i = 1, 2, \cdots |childNode(nod)|$ **do**
2: **if** $q \subseteq_{ia} g_{childNode(nod)_i}$ **then**
3: nod に格納されているグラフ g_{nod} を候補グラフ集合 C_q に加える.
4: ProcSubgraphQuery($childNode(nod)_i$,q)
5: **end if**
6: **end for**

4.4 スーパーグラフ問い合わせ

命題 2 よりグラフデータベース D に含まれるグラフ g と問い合わせ q が，$g \not\subseteq_{ia} q$ であるならば，D に含まれる $g \subseteq_{ia} g'$ であるすべてのグラフ g' は問い合わせとの間で Interlace 定理を満たさない．このことから親が子を Interlace する関係のインデクスを辿ればスーパーグラフの問い合わせを行うことができる．4.2 で述べたインデクスは子が親を Interlace する関係で構築されている．つまりサブグラフ問い合わせとは逆にインデクスを辿ることでスーパーグラフの問い合わせを行うことができる．スーパーグラフ問い合わせアルゴリズムを Algotithm4 に示す.

Algorithm 4 ProcSupergraphQuery(nod,q)

Input: インデクスのノード nod (初期値は $root$)
クエリグラフ q

Output: 候補グラフ集合 C_q

1: **for** $i = 1, 2, \cdots |parentNode(nod)|$ **do**
2: **if** $g_{parentNode(nod)_i} \subseteq_{ia} q$ **then**
3: nod に格納されているグラフ g_{nod} を候補グラフ集合 C_q に加える.
4: ProcSupergraphQuery($parentNode(nod)_i$,q)
5: **end if**
6: **end for**

［例 3］ 図 4 のインデクスに対して, 表 2 のグラフ q のスーパーグラフ問い合わせを行う．$g2 \supsetneqq_i q$ なので $g2$ から $root$ のノードまでのすべてのノードはクエリのスーパーグラフでないことが分かる．したがって $g1$ については Interlace 定理の条件式を判定しない．次に $g3$ と q について Interlace 定理の条件式を判定する．$g3 \subseteqq_i q$ であるので $g3$ を C_q に加える．さらに $parentNode(g3)$ を探索する．$g1 \supsetneqq_i q$ であり, すべての $parentNode(root)$ の探索が終了したので $C_q = \{g3\}$ を出力する．今回 $g1$ が候補でないことが分かっていたにも関わらず,$g3$ が候補であったため $g1$ に対して Interlace 定理の条件式の判定を行った．これは実装の際, 候補でないことが分かっているノードにフラグを立てておき, そのフラグを見ることで回避できる．

4.5 グラフの挿入と削除

グラフを挿入する場合は，構築時と同様に Algoritm2 を用いてグラフを挿入する．既に格納されているグラフを削除する場合，まず目的のグラフが格納されているノードを探索する．目的のグラフを含むノード p について，p を削除し，$parentNode(p)$ と $childNode(p)$ を接続する．

Algorithm 5 DeleteNode(nod,d)

Input: インデクスのノード nod (初期値は $root$)
削除するグラフ d

1: **for** $i = 1, 2, \cdots |childNode(nod)|$ **do**
2: **if** $g_{childNode(nod)_i}$ が d である **then**
3: $nod \leftarrow childNode(nod)_i$
4: **if** $childNode(nod)$ が存在する **then**
5: $parentNode(nod)$ を $childNode(nod)$ の親ノードとする
6: **end if**
7: nod を削除
8: **end if**
9: DeleteNode($childNode(nod)_i, d$)
10: **end for**

5. 実験と評価

提案したインデクスの性能を評価し，提案手法が有効な条件を検証する．実験環境は Core i7-860 2.80GHz 8.0GB RAM 上で行う．OS は Windows 7 Professional 64bit を用いた．実験データとして人口データをランダムに生成し使用する．

5.1 スーパーグラフ問い合わせ

5.1.1 グラフの平均頂点数を変化させる実験

データベースは 1000 個のグラフデータベースとし，グラフの頂点数を変化させその候補グラフ数と処理時間について検証する．平均頂点数を 10 から 35 まで変化させ，平均枝数はその平均頂点数の張りうる枝の半分の枝数とした．図 5 に平均頂点数と処理時間の関係を，図 6 に平均頂点数と候補グラフ数の関係を示す．処理時間はグラフサイズが大きくなるにつれ増えている．

5.1.2 ラベル数を変化させる実験

データベースのグラフ数 1000 個, グラフの平均頂点数は 20 で変化させず, ラベル数を 2 から 20 まで変化させて実験を行った．ラベル数の変化と候補グラフ数の関係を図 7 に示す．スーパーグラフ問い合わせにおいてラベルを増やすことでフィルタリング性能が向上していることがわかる．

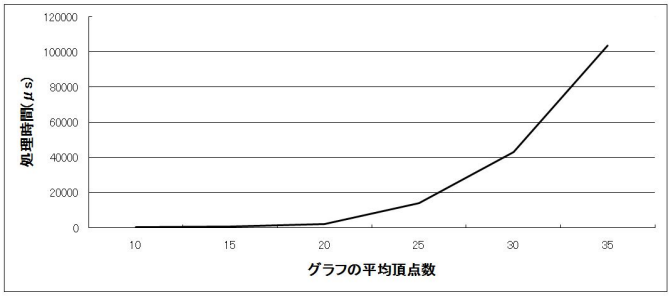

図 5 平均頂点数と処理時間

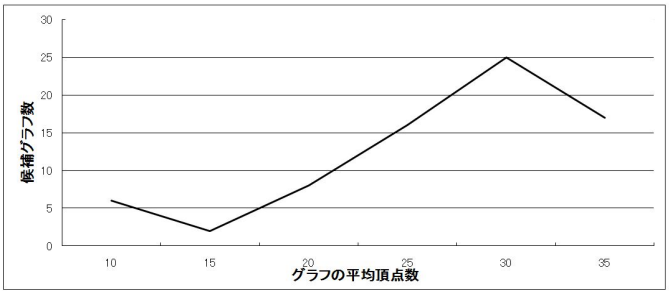

図 6 平均頂点数と候補グラフ数

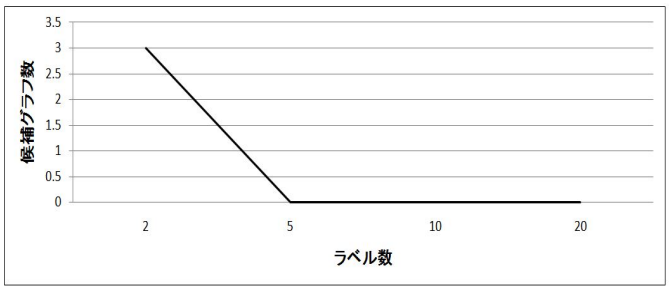

図 7 ラベル数と候補グラフ数

5.2 サブグラフ問い合わせ

実験の比較対象として宮奥ら [12] が提案したインデクスを使用する．

5.2.1 グラフの平均頂点数を変化させる実験

宮奥らの手法で用いられていた行列表現ではグラフサイズが大きくなるにつれて行列の次数大きくなってしまう問題があった．提案手法で用いる行列表現は従来のものに比べて行列の次数を小さく抑えることができる．そこでデータベースは 1000 個のグラフデータベースとし，グラフの頂点数を変化させその候補グラフ数と処理時間について検証する．平均頂点数を 10 から 35 まで変化させ，平均枝数はその平均頂点数の張りうる枝の半分の枝数とした．クエリグラフはデータベースのグラフの平均頂点数を持つグラフと平均頂点数の半分の頂点を持つグラフを用いた．

（1） クエリグラフに平均頂点数の頂点を持つグラフを用いた実験

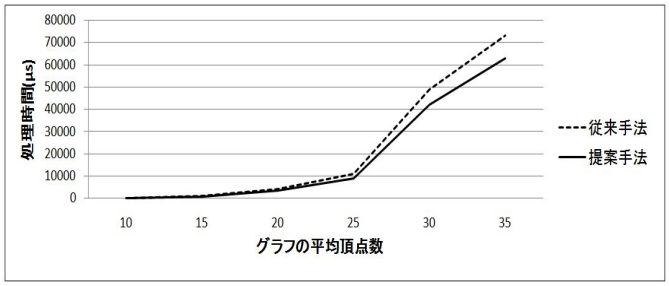

図 8　平均頂点数と処理時間

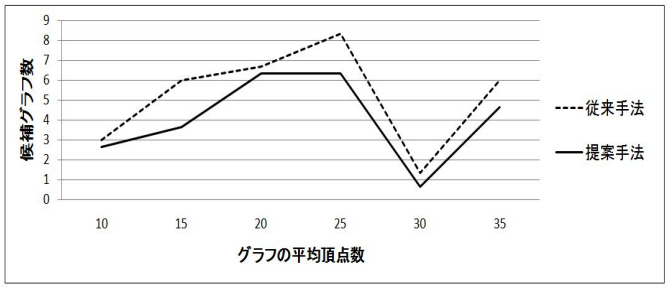

図 9　平均頂点数と候補グラフ数

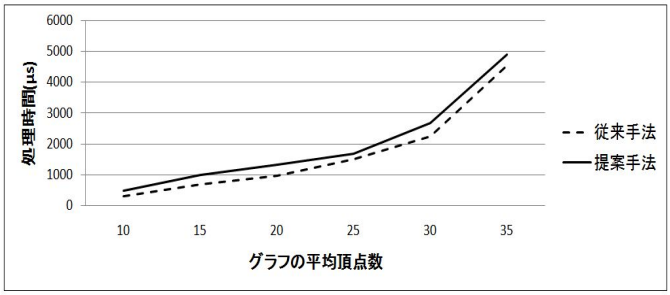

図 10　平均頂点数と処理時間

グラフデータベースの平均頂点数の変化と処理時間の関係を図 8 に示す．データベースのグラフの平均頂点数の変化と候補グラフ数の関係を図 9 に示す． 図 9 から従来手法に比べ若干ではあるがフィルタリング性能が向上している．しかし図 8 から扱うグラフが大きくなるにつれて提案手法が高速に処理出来ていることがわかる．先に述べた通り，提案手法ではグラフサイズの増加による行列の次数の増加が従来に比べ緩やかなためである．

（2） **クエリグラフに平均頂点数の半分の頂点を持つグラフを用いた実験**

グラフデータベースの平均頂点数の変化と処理時間の関係を図 10 に示す．データベースのグラフの平均頂点数の変化と候補グラフ数の関係を図 11 に示す． 図 11 からデータベースのグラフとクエリグラフの大きさの差が大きい条件では提案手法が従来手法に比べフィルタリング性能で劣っていることが分かる．また図 10 から処理時間の点でも提案手法は従来に比べ劣っている．これは提案手法がフィルタリング性能が劣っているので，インデクスを深く探索しなければならないためである．

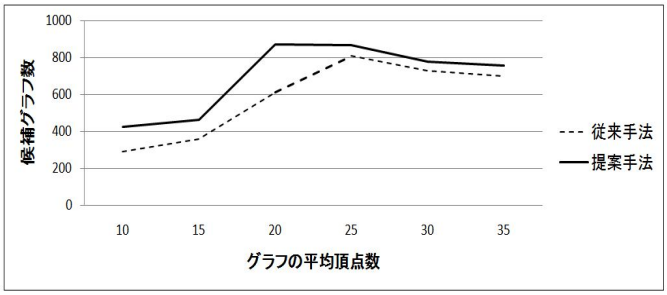

図 11　平均頂点数と候補グラフ数

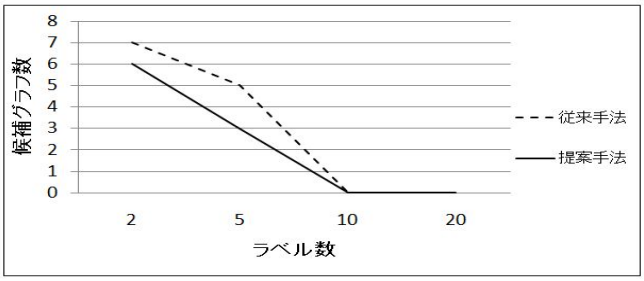

図 12　ラベル数と候補グラフ数

5. 2. 2　ラベル数を変化させる実験

さらにデータベースのグラフ数 1000 個, グラフの平均頂点数は 20 で変化させず, ラベル数を 2 から 20 まで変化させて実験を行った．ラベル数の変化と候補グラフ数の関係を図 12 に示す．従来手法では扱うラベル数が増えていくにつれフィルタリング性能が向上する. 提案手法でも同様にラベルを増やすことでフィルタリング性能が向上していることがわかる．また提案手法ではインデクスの一つのノード当たりの固有値が少ないので，インデクスに用いるメモリ使用量が少ない利点がある．

6. ま と め

本稿ではグラフデータベースから問い合わせのサブグラフ, スーパーグラフの候補を効率的に求めるためのインデクスを提案した．実験の結果から新しい行列表現を用いた提案手法が従来よりもグラフ除去性能が高い場合があることを確認した．グラフ分割を利用した改良では従来手法よりメモリ使用量が大きくなってしまう．情報を増やさずにより高い判定精度を出すために，固有値に関する定理や条件式を利用することでメモリ使用量を増やすことなく判定精度や処理時間を向上させることができると期待される．

文　献

[1] James Cheng, Yiping Ke, Wilfred Ng, and An Lu. Fg-index: towards verification-free query processing on graph databases. In *SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pp. 857–872, New York, NY, USA, 2007. ACM.

[2] L. P. Cordella, P. Foggia, C. Sansone, and M. Vento. An improved algorithm for matching large graphs. In *Proc. of the 3rd IAPR TC-15 Workshop on Graph-based Representations in Pattern Recognition*, pp. 149–159, Ischia, Italy, May 2001.

[3] M. Fatih Demirci, Reinier H. van Leuken, and Remco C.

Veltkamp. Indexing through laplacian spectra, June 2008.

[4] Raghav Kaushik, Pradeep Shenoy, Philip Bohannon, and Ehud Gudes. Exploiting local similarity for indexing paths in graph-structured data. In *18th International Conference on Data Engineering (ICDE'02)*, pp. 129–140, 2002.

[5] Tova Milo and Dan Suciu. Index structures for path expressions. *Lecture Notes in Computer Science*, Vol. 1540, pp. 277–295, 1999.

[6] Chen Qun, Andrew Lim, and Kian Win Ong. D(k)-index: An adaptive structural summary for graph-structured data. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, San Diego, California, USA, June 9-12, 2003*, pp. 134–144, 2003.

[7] Dennis Shasha, Jason T.L.Wang, and Rosalba Giugno. Algorithmics and applications of tree and graph searching. In *Proceedings of the Twenty-first ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 3-5, Madison, Wisconsin,USA*. ACM, 2002.

[8] Ali Shokoufandeh, Diego Macrini, Sven J. Dickinson, Kaleem Siddiqi, and Steven W. Zucker. Indexing hierarchical structures using graph spectra, 2005.

[9] J. R. Ullmann. An algorithm for subgraph isomorphism. *J. Assoc. Comput. Mach*, Vol. 23, pp. 31–42, 1976.

[10] Xifeng Yan, Philip S. Yu, and Jiawei Han. Graph indexing: A frequent structure based approach. In *SIGMOD '04: Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, 2004.

[11] Lei Zou, Lei Chen, Jeffrey Xu Yu, and Yansheng Lu. A novel spectral coding in a large graph database, 5 2008.

[12] 宮奥祥多, 片山薫. ラベルによるグラフ分割を用いた固有値に基づくグラフ索引手法. 第 2 回データ工学と情報マネジメントに関するフォーラム, 兵庫,2010,3, 3 2010.

[13] 高橋俊介, 片山薫. グラフ索引のための interlace 定理によるグラフの包含関係を考慮したデータ構造の提案. 情報処理学会第 2 回データ工学と情報マネジメントに関するフォーラム", 宮崎,2008,3, 3 2008.